Contribution à l'étude de l'Éducation algorithmique chez les apprenants du secondaire à l'aide d'analyses statistiques implicatives selon MGK

Bruno Bakys Ralahady

ECOLE DOCTORALE

PROBLEMATIQUES DE L'ÉDUCATION ET DIDACTIQUES DES DISCIPLINES

PE2Di

THÈSE

pour obtenir le grade de docteur de l'Université d'Antananarivo

Spécialité "Didactique de l'informatique"

Contribution à l'étude de l'Education algorithmique chez les apprenants du secondaire à l'aide d'analyses statistiques implicatives selon M_{GK}

présentée et soutenue publiquement par

RALAHADY BRUNO BAKYS

le 22 Avril 2022

Membres du jury :

Judith RAZAFIMBELO RAHOLDINA,	Professeure Titulaire Emérite. Université d'Antananarivo, Madagascar	President
Dominique TOURNÈS,	Professeur des universités. Université de La Réunion	Rapporteur
Jean-Claude REGNIER,	Professeur Emérite. Université Lyon-2, France	Rapporteur
Thomas MAHATODY,	Maître de conférences HDR. Université de Fianarantsoa, Madagascar	Examinateur
Angelo Fulgence RAHERINIRINA,	Maître de conférences HDR. Université de Fianarantsoa, Madagascar	Examinateur
André TOTOHASINA,	Professeur Titulaire. Université d'Antsiranana, Madagascar	Directeur
Jean Claude LABERCHE,	Professeur Emérite. Ton Duc Thang University, Ho Chi Minh, Vietnam	Co-Directeur

PE2Di

Ecole Normale Supérieure d'Antananarivo

Complexe scolaire Ampefiloha, Antananarivo

THESE

ECOLE DOCTORALE

PROBLEMATIQUES DE L'ÉDUCATION ET DIDACTIQUES DES DISCIPLINES

PE2Di

THÈSE

pour obtenir le grade de docteur de l'Université d'Antananarivo

Spécialité "Didactique de l'informatique"

Contribution à l'étude de l'Education algorithmique chez les apprenants du secondaire à l'aide d'analyses statistiques implicatives selon M_{GK}

présentée et soutenue publiquement par

RALAHADY BRUNO BAKYS

le 22 Avril 2022

Membres du jury :

Judith RAZAFIMBELO RAHOLDINA,	Professeure Titulaire Emérite. Université d'Antananarivo, Madagascar	President
Dominique TOURNÈS,	Professeur des universités. Université de La Réunion	Rapporteur
Jean-Claude REGNIER,	Professeur Emérite. Université Lyon-2, France	Rapporteur
Thomas MAHATODY,	Maître de conférences HDR. Université de Fianarantsoa, Madagascar	Examinateur
Angelo Fulgence RAHERINIRINA,	Maître de conférences HDR. Université de Fianarantsoa, Madagascar	Examinateur
André TOTOHASINA,	Professeur Titulaire. Université d'Antsiranana, Madagascar	Directeur
Jean Claude LABERCHE,	Professeur Emérite. Ton Duc Thang University, Ho Chi Minh, Vietnam	Co-Directeur

PE2Di

Ecole Normale Supérieure d'Antananarivo

Complexe scolaire Ampefiloha, Antananarivo

THESE

Remerciements

La réalisation de cette these a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner toute ma gratitude.

Je voudrais tout d'abord adresser toute ma reconnaissance à Madame Judith RAZAFIMBELO RAHOLDINA, Professeure Titulaire Emérite, Directrice de l'Ecole Doctorale Thématique : « Problématiques de l'Education et Didactiques des Disciplines » de l'Université d'Antananarivo, d'avoir accepté ma candidature en tant que doctorant.

Je désire aussi remercier mon Directeur Monsieur TOTOHASINA André, Professeur titulaire des Universités, pour ses meilleures collaborations et soutiens moraux durant la préparation de cette thèse.

Je tiens à remercier spécialement à mon co-Directeur Monsieur Jean Claude LABERCHE, Professeur Emérite, pour avoir relu et corrigé ma thèse. Ses conseils de rédaction ont été très précieux.

Je remercie également à mon ancien co-Directeur Monsieur Jean SIMON, Maître de conférences, qui m'a fourni les outils nécessaires en didactique.

J'adresse mes sincères remerciements à Monsieur Dominique TOURNES, Professeur des universités et Monsieur Jean-Claude REGNIER, Professeur Emérite, d'avoir accepté d'être rapporteurs de mes travaux. Je profite de l'occasion pour leur adresser mes sincères respects et leur exprimer ma profonde gratitude pour leurs commentaires et jugements très pertinents sur ma thèse, tant sur le fond que sur la forme.

J'exprime mes profondes reconnaissances aux membres du Jury pour leurs critiques et remarques.

Je n'oublie jamais les enseignants des collèges Notre-Dame de Lourdes et Sainte Thérèse et à tous leurs personnels avec qui j'ai eu beaucoup des idées pour soutenir le corps de mes recherches.

Mes remerciements vont aussi à tous mes élèves aussi, étudiants et étudiantes pour leur amitié et l'ambiance véritablement chaleureuse.

Je remercie mes très chers parents, qui ont toujours été là pour moi. Je remercie mes frères et sœurs, pour leurs soutiens et leurs encouragements.

Enfin, je tiens à remercier particulièrement ma femme RAMIARINTSOA Béatrice Eva pour ses compréhensions, ses soutiens amoureux et ses encouragements durant la préparation de cette thèse.

Les mots manquent aux émotions.

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations

ACM	Association of Computing Machinery
ARFF	Attribute-Relation File Format
ASCII	American Standard Code for Information Interchange
ASI	Analyse Statistique Implicative
ASI-MGK	Analyse Statistique Implicative selon M_{GK}
B2i	Brevet informatique et Internet
BEPC	Brevet d'Études du Premier Cycle
B.O.E.N.	Bulletin officiel de l'éducation nationale
C2i.	Certificat Informatique et Internet
CD	Compact Disc
CD-ROM	Compact Disc - Read Only Memory
CEPE	Certificat d'Etudes Primaires Élémentaire
CHIC	Hierarchical Implicative and Cohesive Classification
CNRS	Centre National de la Recherche Scientifique
CRINFP	Centres Régionaux de l'Institut National de Formation Pédagogique
CRLF	Carriage Return Line Feed
CSTA	Computer Science Teachers Association
CSV	Comma-separated values
D.I.	Didactique de l'Informatique
DHP	Direct Hashing and Pruning
DIC	Dynamic Itemset Counting
DIDEC	Direction Diocesien de l'Education Catholique
DLL	Dynamic Link Library
DREN	Direction Régionale de l'Education Nationale
DVD	Digital Versatile Disc
ECD	Extraction de Connaissances à partir des Données
EDMI	Education et Didactique des Mathématiques et de l'Informatique
ENSET	Ecole Normale Supérieure pour l'Enseignement Technique
EPI	Enseignements Pratiques Interdisciplinaires
EPS	Éducation Physique et Sportive
ETL	Extraction Transfer Loading
FORTRAN	FORmula TRANslating
FRAM	Fikambanan'ny Ray Aman-drenin'ny Mpianatra

INFP	Institut National de Formation Pédagogique
ICAR	Interactions, Corpus, Learning and Representation
IREMI	Instituts de Recherche sur l'Enseignement des Mathématiques et de l'Informatique
ISN	Informatique et Sciences du Numérique
LCP	Logique de Conception des Programmes
MGK	Mesures de Guillaume Kentchaff
MIT	Massachusetts Institute of Technology
MLJ	Machine Learning in JAVA
OCDE	Organisation for Economic Co-operation and Development
OS	Operating System
P21	Partnership for 21st-Century Skills
PAO	Publication Assisté par Ordinateur
PC	Personal Computer
PE2Di	Problématiques de l'Education et Didactiques des Disciplines
PFÉQ	Programme de Formation de l'École Québécoise
PGCD	Plus Grand Commun Diviseur
PISA	Programme for International Student Assessment
PPCM	Plus Petit Commun Multiple
PréAO	Présentation Assisté par Ordinateur
RA	Regle d'Association
RAM	Radom Access Memory
SGBD	Système de Gestion de Base de Données
SIA	Société d'Informatique Appliquée
SSDM	Semantically Similar DataMining Algorithm
TIC	Technologies d'Information et de Communication
TICE	Technologies de l'Information et de la Communication pour l'Enseignement
TID	Transaction Identity
UMR	Unité Mixte de Recherche
UNESCO	United Nations Educational, Scientific and Cultural Organization

Introduction générale

Le sujet de cette thèse, proposé par TOTOHASINA André, Professeur à l'université d'Antsiranana concerne la définition des programmes en informatique, tels qu'ils devraient être enseignés dans les classes des collèges de Madagascar dans les années à venir.

Partout dans le monde de l'éducation, l'informatique est une science relativement récente puisqu'en France les premiers enseignements de l'informatique au lycée datent des années 1960. Dans la formation générale, le rapport Simon, remis en 1980 au président Valery Giscard d'Estaing recommandait d'enseigner l'informatique dès la classe de quatrième. Puis de 2012 à 2016, les programmes scolaires du premier degré et du second degré ont progressivement introduit des notions d'informatique. Dans le premier degré, cet enseignement est dispensé par les professeurs des écoles. Dans le second degré, il n'y a pas de professeurs dont l'informatique soit la discipline principale et cet enseignement est dispensé par des professeurs d'autres disciplines. Seul l'enseignement supérieur à des maîtres de Conférences et des professeurs des Universités. Depuis la rentrée 2016, le programme de cinquième, quatrième et troisième comporte un enseignement d'informatique dans le cadre des mathématiques et de la technologie.

Les élèves s'initient à la programmation dans une démarche de projet. Ils apprennent à écrire, mettre au point et exécuter un programme. Ils découvrent la notion de variable et d'affectation, de séquence d'instructions, de boucle, d'instruction conditionnelle, de déclenchement d'une action par un événement (Bulletin officiel spécial n°11 du 26 novembre 2015 N, Arrêté du 9-11-2015 J.O. du 24-11-2015 MENESR - DGESCO MAF 1) Le langage le plus utilisé est Scratch.

Lorsque l'on considère un programme d'informatique comprenant des notions de base, il apparaît que l'enseignement de l'informatique est beaucoup plus complexe qu'il n'y paraît à première vue. Ceci est probablement dû au fait qu'il n'y a pas d'enseignant spécialisé dans la matière, et qu'en plus un vocabulaire très particulier et parfois rebutant, est introduit. C'est le cas de la notion d'algorithme dont l'appropriation par les élèves peut être conditionnée par de nombreux facteurs allant de la persuasion de leurs enseignants ou encore par leur formation antérieure. De même, on peut penser que le problème très prosaïque de la disponibilité du matériel informatique peut influencer l'enseignement de l'informatique.

Nous serons alors amenés à concevoir un protocole d'expérimentation, à réaliser des initiations selon ce dernier, à collecter des données, traiter et analyser ces dernières avec des outils statistiques appropriées dont l'analyse implicative selon M_{GK}, entre autres. Notons que l'extraction des connaissances à partir de cet outil statistique figure parmi les problématiques de recherche de l'équipe d'accueil EDMI de notre école doctorale PE2Di.

Par ailleurs, dans la littérature sur l'extraction des connaissances à partir des données volumineuses, la recherche des règles d'association intéressantes est un thème privilégié. Les compor-

tements d'apprenants et ceux de l'enseignant des mathématiques constitueront majoritairement ainsi nos variables statistiques dont il convient d'étudier et d'identifier les liens implicatifs, traductibles en termes de règles d'association, voire en liens causaux.

Le problème de l'utilité et de la pertinence des règles d'association extraites est alors primordial, car dans la plupart des cas, les jeux de données réels conduisent à plusieurs milliers voire plusieurs millions de règles d'association dont la mesure de confiance est élevée, et parmi lesquelles se trouvent de nombreuses règles redondantes. Des algorithmes, des mesures des règles ont apporté déjà des solutions au problème, mais ils produisent encore une trop grande masse de règles, sélectionnant certaines règles sans intérêt et ignorant des règles intéressantes. La qualification des règles pose plusieurs questions légitime, ainsi : la règle est-elle le fruit du hasard ? Son évaluation est-elle significativement supérieure au seuil ? Serait-elle toujours valide si les données n'avaient pas été exactement ce qu'elles sont ou encore si le seuil d'acceptation avait été augmenté, même légèrement ?

Pour mettre de l'ordre dans tous ces facteurs qui semblent se poser partout, nous avons décidé de considérer le problème qui nous est posé sous la forme d'un système en utilisant le schéma systémique développé par nous à partir de celui proposé par The Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES-2019) dans les études des écosystèmes (Figure 1).

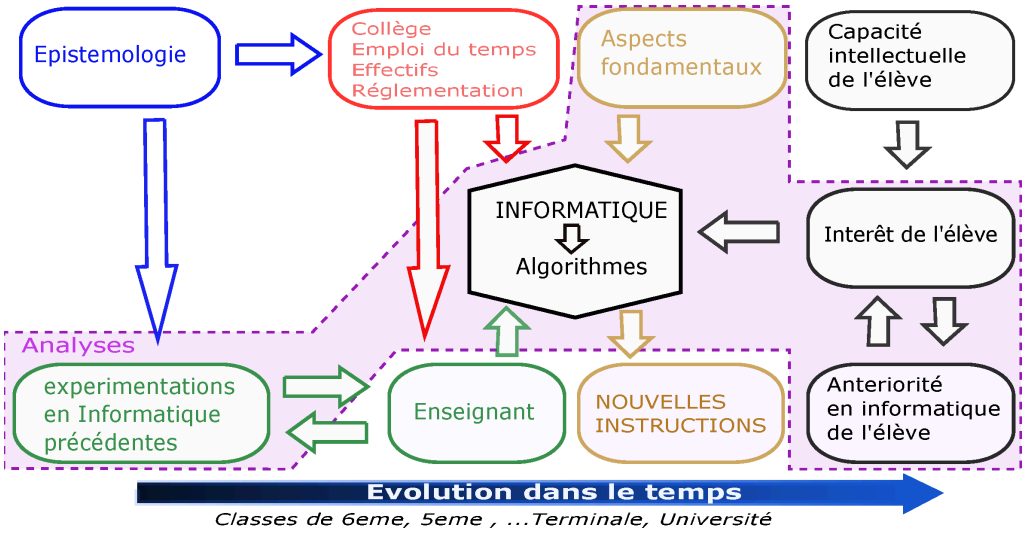


FIGURE 1 – Le systeme « Enseignement de l'informatique au collège » avec les quatre grands groupes de facteurs qui agissent et interagissent sur lui.

L'aspect épistémologique est en bleu, les aspects réglementaires sont en rouge, les aspects fondamentaux de l'informatique en marron, les aspects variables de l'enseignant en vert, les aspects personnels de l'élève en noir, les flèches indiquent les grandes interactions.

En relation avec les facteurs qui agissent et interagissent le champ d'analyse qui est le nôtre dans ce travail est délimité par le polygone en violet. Ce schéma simplifie grandement notre démarche, qui se déroulera en six parties :

Dans le premier et deuxième chapitre, nous presenterons les théories des regles d'associacions avec ses prorpiétés mathématiques et certaines algorithmes extractions. Nous continuerons par la presentation et la caracterisation de notre outil d'analyse ASI – MGK.

Dans le troisième chapitre nous ferons l'étude épistémologique de l'algorithme et le panorama de son enseignement au collège. Nous verrons ainsi quels sont les grands concepts informatiques définis par la résolution de l'OCDE qui doivent être enseignés et les moyens préconisés pour y parvenir.

Dans le quatrième chapitre, nous discuterons de « l'état des lieux » de l'enseignement de l'informatique à Madagascar au collège. On observera le programme existant fourni par le ministère de l'Éducation nationale malgache, les horaires prévus pour enseigner chaque discipline, les ressources humaines, la capacité intellectuelle et le matériel disponible pouvant accueillir l'enseignement de l'informatique au collège.

Puis nous nous rapprocherons des enseignants en charge de cet enseignement et leur manière de l'aborder en fonction de leur specialisation d'origine. Ces enseignants eux-mêmes, travaillent dans des structures qui leur imposent des contraintes spécifiques.

De plus, cet enseignement s'adresse à des élèves qui ont eux-mêmes des aprioris sur l'informatique en raison de leur contexte familial ou sociétal qui peuvent les amener à être plus ou moins réceptifs à cet enseignement. L'inventaire étant fait nous présenterons l'expérience que nous avons prévue pour répondre à la question qui nous a été posée. Nous décrirons les sites expérimentaux et les méthodes utilisées tant au niveau pédagogique incluant le logiciel Scratch, qu'au niveau du contrôle de l'acquisition des connaissances que ce soit de manière automatique ou scientifique avec l'utilisation de l'ASI – MGK.

Dans le cinquième chapitre nous présenterons les résultats obtenus lors de ces expériences, dont certaines sont très originales. Et nous discuterons ces résultats et présenterons des propositions pour améliorer l'enseignement de l'informatique au collège.

Enfin dans le sixième chapitre, nous presenterons les applications de notre outil ASI-MGK dans d'autres domaines.

Première partie

Extraction de connaissances à partir des données (ECD).

Chapitre 1

Règles d'association.

Sommaire

Si notre objectif de recherche fondamentale est l'Analyse Statistique Implicative M_{GK} que nous utiliserons comme méthode et outil d'analyse lors de nos expériences pédagogiques avec les élèves et les enseignants de collège, il est indispensable cependant de présenter d'abord les fondements théoriques de l'informatique ; Ainsi nous présenterons brièvement la fouille de données, le fondement mathématique de la théorie de règles d'association et les algorithmes utilisés pour l'extraction des règles valides.

Mais plutôt que de reprendre des données déjà bien connues et même largement décrites, ne serait ce que par plusieurs membres de notre laboratoire de recherche nous avons choisi d'y donner un aspect pédagogique afin que nos collègues des collèges puissent y trouver des éléments pour construire leurs cours.

Nous leur proposons de partir d'un exemple très simple connu de tous. Les élèves sont devant un clavier d'ordinateur et souhaitent écrire sur leur écran l'article défini pluriel les.

Que fait-il ?

Il met en marche l'ordinateur et choisit le programme qui va lui permettre d'avoir accès à toutes les lettres, les signes, les chiffres qui sont quelque part dans l'ordinateur.

1. Il choisit une partie de ce qui est en réserve dans l'ordinateur : ici ce sont les lettres.
2. Il tape d'abord sur le l, puis sur le et enfin les.
3. Celles-ci apparaissent sur l'écran, mais de manière ordonnée pour former le mot les.

En processus informatique on peut présenter ces étapes de la manière suivante :

1. base de données	$\Longrightarrow$	2. fouille	$\Longrightarrow$	3.extraction	$\Longrightarrow$	4.classification.
zlqse		rN e,4,K,l, L, ç,s,G		e,l,s		les

La base de données est constituée : - chiffres(0 3 1 ⋯) - signes +-/ :! ⋯

- lettres (z l q s e ⋯) - ponctuation. ; : ⋯

Mais le passage d'une étape à une autre ne peut se faire spontanément, puisqu'il est devant une machine, il est nécessaire qu'il y ait des instructions, c'est le rôle des algorithmes qui sont des suites d'opération mathématique programmées. Il est alors possible de donner une première approche du système informatique (Figure 1.1).

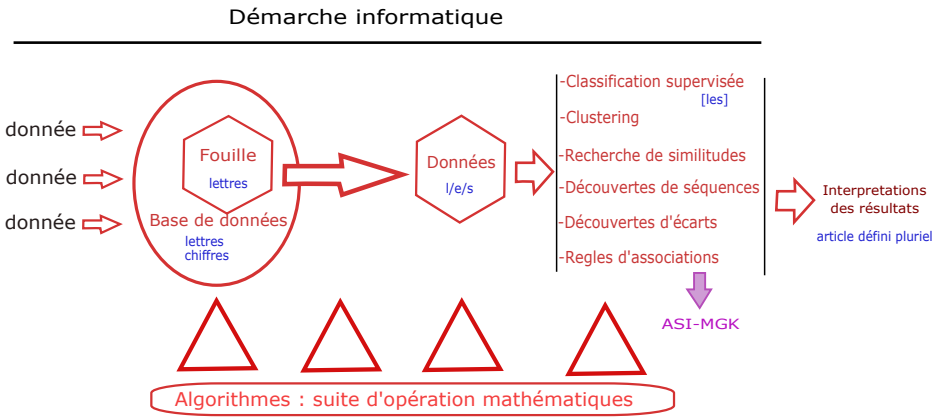


FIGURE 1.1 – Le système informatique tel qu'il peut apparaitre en premier abord quand on souhaite écrire le nom « les » sur l'écran d'un ordinateur.

La figure 1.1 nous permet d'appréhender le système informatique dans sa globalité, le détail des étapes et l'extraction des données à partir des données sont beaucoup plus compliquées comme l'a illustré Bemarisika (2016) (Figure 1.2).

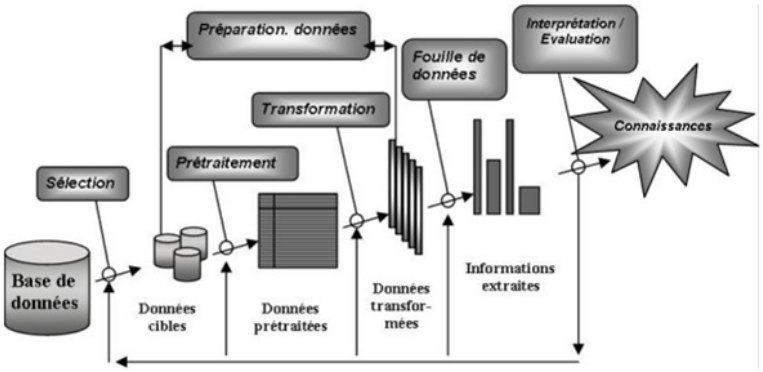


FIGURE 1.2 – Les étapes du processus de l'ECD selon Bemarisika (2015).

L'auteur reprend dans sa figure, le processus d'extraction de connaissances à partir de bases de données décrites précédemment par (Fayyad et al., 1996a) comme un processus composé de plusieurs étapes. Il s'agit de :

1. La fouille des données qui aboutit à sélectionner des sous-ensembles de données intéressantes.
2. Le prétraitement des données pour éliminer, voire corriger des données manquantes ou aberrantes.
3. La transformation des données pour qu'elles puissent subir les opérations mathématiques des algorithmes.
4. L'observation des résultats observés.
5. L'interprétation de ces résultats débouchant sur de nouvelles connaissances.

Ce sont ces points qui seront détaillés maintenant en essayant d'insister sur l'aspect didactique de chacun.

1.1 ECD et théorie des règles d'association.

1.1.1 Introduction

L'extraction de connaissances à partir des données (ECD) est un axe de recherche très actif, qui a vu le jour au cours des trois dernières décennies (Fayyad et al., 1996a). La progression de la technologie du matériel informatique dans la même période nous a amené à une puissance accrue des ordinateurs et à de nouveaux des équipements de collecte des données. Cette technologie a donné un grand coup de pouce à l'industrie de bases de données et de l'information et a permis la gestion de transactions, à la recherche d'informations et à l'analyse de données de très grand volume. La croissance rapide et l'énorme quantité de données collectées et stockées dans les bases de données ont excédé notre habilité humaine. Pour faire face à ce problème, de nombreux travaux ont dû trouver de nouvelles méthodes de gestion des données permettant de traiter ces volumes de manière plus rapide.

La fouille de données est l'étape centrale du processus de l'extraction de connaissances consistant à découvrir de nouveaux modèles au sein de grandes quantités de données. Avant de développer l'ECD, il est nécessaire de rappeler ce qu'est une fouille de données.

1.1.2 Fouille de données.

La fouille de donnée est l'exploration et l'analyse de grandes quantités de données afin d'y découvrir des motifs et de faire émerger, par des méthodes algorithmiques, des tendances ou des schémas à partir d'un grand volume de données prétraitées. La fouille de données tire son nom à la locution anglaise *data mining* qui est un domaine nouveau qui trouve sa source dans les années 1980. Elle consiste à rechercher et extraire de l'information utile et inconnue à partir de gros volumes de données stockées dans des bases ou des entrepôts de données (Preux, 2011).

Avant de développer la fouille de données, il s'avère utile de définir tout d'abord le terme *donnée*.

Définition 1 (Les données) *D'un point de vue informatique, une donnée est un élément contenant une information. Ces informations de différentes natures peuvent prendre différents formats : données numériques, données binaires, données textuelles, ···*

Définition 2 (La base de données) *Une base de données désigne alors un* ***ensemble d'informations stockées*** *sur un système informatique. Une base de données répond à de nombreuses problématiques dont le stockage efficace, le tri et la sélection des données. On discerne donc deux organisations distinctes :* ***l'organisation logique****, qui désigne le modèle sémantique selon lequel les données sont stockées et* ***l'organisation physique****, qui désigne la manière dont les données sont organisées sur le disque dur (Basque, 2005).*

Définition 3 (la fouille de données) *La fouille de données est un processus d'extraction d'informations recevables, compréhensibles, afin de distinguer des relations et des motifs préalablement inconnus dans les données afin de nous aider dans la prise des décisions (Friedman, 1997).*

Définition 4 *La fouille de données est un ensemble de méthodes utilisées dans le processus d'extraction de connaissances afin de distinguer des relations et des motifs préalablement inconnus dans les données (Cios, K.J., Pedrycz, W. and Swiniarski, R.W, 1998).*

Définition 5 *Sur le plan théorique, la fouille de données est une étape dans le processus d'extraction et de connaissances, qui consiste en l'application d'algorithmes de découverte et d'analyse de données qui, avec des limites computationnelles acceptables, produit une certaine énumération de motifs (ou modèles) à partir des données (Fayyad et al., 1996a).*

Définition 6 *La fouille de données est un processus inductif, itératif et interactif dont l'objectif est la découverte de modèles de données valides, utiles et compréhensibles dans de larges bases de données (Talbi, 2000)*

Définition 7 *La fouille de données est un ensemble de techniques d'exploration de données permettant d'extraire d'une base de données des connaissances sous la forme de modèles de description afin de décrire le comportement actuel des données et/ou prédire le comportement futur des données (ESPINASSE, 2008).*

Pour la suite, nous employons de façon indifférente la locution fouille de données et *data mining*.

1.1.3 Différentes tâches effectuées lors des fouilles de données (*Data Mining*).

Le *Data Mining* désigne en réalité un ensemble de traitements très différents, menant à la découverte de connaissances variées. Ces traitements sont :

- La **classification supervisée** (tâche de prédiction) : affecter une classe à chaque instance, chaque classe associée à un concept ou comportement spécifique à identifier.
- Le **clustering** (ou classification non supervisée) (tâche descriptive) : identifier des groupes d'instances.
- La **découverte de règles d'associations** (tâche descriptive) : rechercher des implications entre attributs, ou entre classes d' attributs.
- La **découverte de séquences** : similaire à la recherche de règles d'association avec insertion de la notion de temps.
- La **détection de déviation / la détection d'écart** : identifier des valeurs exceptionnelles.
- La **recherche de similitudes** : identifier des séquences communes entre instances (domaine de la bio-informatique).

La fouille de données est une appellation qui regroupe donc plusieurs techniques très différentes les unes des autres. Le domaine est vaste ; nous avons choisi de porter plus particulièrement notre attention sur la recherche de règles d'association.

Après avoir fouillé les données, il va falloir extraire les connaissances souhaitées à partir des données. C'est l'objet du paragraphe suivant.

1.1.4 Extraction de Connaissances à partir de Données (ECD).

Les quantités de données recueillies dans les bases de données doublent tous les neuf mois, soit deux fois plus vite que la non officielle loi de Moore. Si l'on se réfère à des applications scientifiques, ce sont des giga-octets de données qui sont ainsi collectées et stockées. Dans des entreprises commerciales (assurances, la grande distribution ou encore dans le domaine bancaire), dans le domaine des recherches médicales, de nombreuses données sont collectées, mais ne sont pas nécessairement exploitées par la suite. Grâce aux techniques d'extraction de connaissances, les bases de données volumineuses sont devenues des sources riches et fiables pour la génération et validation de connaissances. La littérature sur l'extraction de connaissances à partir des données volumineuses délivre une variabilité des méthodes d'approche, notamment dans l'extraction des règles d'association. L'ECD est un domaine de recherche en plein essor visant à exploiter rapidement les grandes quantités de données (Goethals & Zaki, 2003).

Ceci nous amène à compléter les définitions déjà données au ğ précédent par deux nouvelles ;

Définition 8 *L'ECD désigne le processus non trivial d'extraction des informations implicites, de structures inconnues, valides et potentiellement utiles ou exploitables dans des bases de données (Piatetsky-Shapiro, 1991a; Fayyad et al., 1996b).*

Définition 9 *Le terme ECD est communément confondu avec la fouille de données. Il vient de la traduction de l'expression anglaise "knowledge discovery in databases (KDD)". Ce terme a été introduit*

par Piatetsky-Shapiro (1991b); Frawley et al. (1992).

L'extraction des règles d'association est un processus itératif [1] et interactif constitué de plusieurs phases allant de la sélection et la préparation des données jusqu'a l'interprétation des résultats, en passant par la phase de recherche des connaissances : le data mining. La plupart des approches proposées pour l'extraction des itemsets fréquents reposent sur les quatre phases suivantes :

a- Préparation des données. Cette phase consiste à sélectionner les données (attributs et objets) de la base de données utiles à l'extraction des règles d'association et transformer ces données en un contexte d'extraction. Ce contexte, ou jeu de données, est un triplet $\mathcal{B} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ dans lequel $\mathcal{O}$ est un ensemble d'objets, $\mathcal{I}$ est un ensemble d'attributs, également appelés items, et $\mathcal{R}$ est une relation binaire entre $\mathcal{O}$ et $\mathcal{I}$.

b- Extraction des ensembles fréquents d'attributs. Cette phase consiste à extraire du contexte tous les ensembles d'attributs binaires appelés itemsets, qui sont fréquents dans le contexte $\mathcal{B}$. Un itemset l est fréquent si son support, qui correspond au nombre d'objets du contexte qui « contiennent » l, est supérieur ou égal au seuil minimal de support minsupport défini par l'utilisateur.

c- Génération des règles d'association. Durant cette phase, les itemsets fréquents extraits durant la phase précédente sont utilisés afin de générer les règles d'association qui sont des implications entre deux itemsets fréquents $l_1, l_2 \in \mathrm{F}$ tels que $l_1 \subset l_2$, de la forme $r : l_1 \rightarrow (l_2\ l_1)$. Afin de limiter l'extraction aux règles d'association les plus informatives, seules celles qui possèdent une valeur supérieure ou égale à la valeur critique calculée à partir du seuil confiance défini par l'utilisateur sont générées.

d- Interprétation des résultats. Cette phase consiste en la visualisation par l'utilisateur des règles d'association extraites du contexte et leur interprétation afin d'en déduire des connaissances utiles pour l'amélioration de l'activité concernée. Le nombre important de règles d'association extraites en général impose le développement d'outils de classification des règles, de sélection par l'utilisateur de sous-ensembles de règles, et de leur visualisation sous une forme intelligible.

1.2 Fondement mathématique de la théorie de règles d'association.

Cette section présente les notions mathématiques qui seront utiles dans la théorie de la fouille de règles d'association. Nous présenterons les notions de base sur l'algèbre de Boole, les structures de treillis, les familles de Moore et la correspondance de Gallois.

1. Qui est fait ou répète plusieurs fois. En informatique, se dit d'un processus de calcul ou d'une structure de programme qui met en œuvre des séquences d'instructions répétées plusieurs fois. https://www.larousse.fr/dictionnaires/francais/itératif/44575

1.2.1 Algèbre de Boole.

Georges Boole (mathématicien anglais, 1815-1864) a introduit une algèbre à deux valeurs pour étudier la logique. L'algèbre de Boole constitue une structure algébrique de premier plan de par sa simplicité, mais aussi de par ses domaines d'applications pratiques et variées. Divers problèmes, tels que les problèmes de fouille de règles d'association, les problèmes de conception et de test des circuits logiques, peuvent être vus comme une séquence d'opérations sur des fonctions booléennes. Nous donnons dans la suite les définitions de base en algèbre de Boole.

Définition 10 ***Algèbre de Boole.*** *On dit que l'ensemble* B *muni des opérateurs* $\wedge$, $\vee$ *et* $\neg$ *a une structure d'algèbre de Boole s'il vérifie les axiomes suivants :*

- $x \vee 0 = x$, $x \wedge 1 = x$
- *Commutativité :* $x \wedge y = y \wedge x$, $x \vee y = y \vee x$
- *Associativité :* $x \wedge (y \wedge z) = (x \wedge y) \wedge z$, $x \vee (y \vee z) = (x \vee y) \vee z$
- *distributivité :* $x \wedge (y \vee z) = (x \wedge y) \vee (y \wedge z)$, $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
- *complementation :* $x \vee \neg x = 1$ $x \wedge \neg x = 0$

Théorème 1 $\forall (x, y, z) \in \mathrm{B}^3$, *on a :*

- *Idempotence :* $x \wedge x = x$, $x \vee x = x$
- *Absorption :* $x \wedge (y \vee x) = x$, $x \vee (y \wedge z) = x$
- *Loi de Morgan :* $\neg(x \vee y) = \neg x \wedge \neg y$, $\neg(x \wedge y) = \neg x \vee \neg y$
- *Eléments neutres :* $x \vee 1 = x$, $x \wedge 0 = 0$

Définition 11 ***Variable logique.*** *Une variable logique est une variable qui peut prendre deux états :* ***vrai*** *ou* ***faux****.*

Il est aussi possible d'associer le *chiffre 1* à la valeur *vrai* et le *chiffre 0* à la valeur *faux*. Dans ce cas, une telle variable est appelée variable *booléenne*.

Définition 12 ***Variable booléenne.*** *On appelle variable booléenne toute variable prenant sa valeur dans doubleton* $\mathrm{B} = \{0, 1\}$.

On munit B des opérations suivantes :

- L'opération binaire de disjonction notée $\vee : \mathrm{B} \times \mathrm{B} \longrightarrow \mathrm{B} : (x, y) \longmapsto x \vee y$
- L'opération binaire de conjonction notée $\wedge : \mathrm{B} \times \mathrm{B} \longrightarrow \mathrm{B} : (x, y) \longmapsto x \wedge y$
- L'opération unaire de complémentation notée $\neg : \mathrm{B} \longrightarrow \mathrm{B} : x \longmapsto \neg x$

Définition 13 ***Relation binaire.*** *Une relation binaire entre deux ensembles* X *et* Y*, notés* $\mathcal{R}$*, est un ensemble des couples* (x, y) *tels que* $x \in \mathrm{X}$, $y \in \mathrm{Y}$ *: tout couple* $(x, y) \in \mathrm{XY}$ *élément de cet ensemble se note alors* $x\mathcal{R}y$*. Si* $\mathrm{X} = \mathrm{Y}$*, nous parlons alors d'une relation binaire sur l'ensemble* X*.*

Définition 14 ***Relation d'ordre.*** *Une relation binaire* $\mathcal{R}$ *sur un ensemble* X*, est appelée une relation d'ordre, si pour tous* $x, y, z \in \mathrm{X}$ *elle vérifie les trois conditions suivantes :*

- *Réfléxivité :* $x\mathcal{R}x$
- *Antisymétrie :* $x\mathcal{R}y$ *et* $y\mathcal{R}x \Rightarrow x = y$
- *Transitivité :* $x\mathcal{R}y$ *et* $y\mathcal{R}z \Rightarrow x\mathcal{R}z$

Une représentation graphique d'une relation d'ordre $\mathcal{R}$ *est appelé le diagramme de Hasse de* $\mathcal{R}$*.*

1.2.2 Structures de treillis.

Dans la littérature, il existe deux définitions équivalentes de treillis : une concernant la relation d'ordre et l'autre algébrique.

Définition 15 *D'après Dedekind (1900), un treillis est un ensemble* T *muni de deux lois internes habituellement notées* $\vee$ *et* $\wedge$*, pour tous* x, y, z *vérifiant :*

- *Associativité :* $x \wedge (y \wedge z) = (x \wedge y) \wedge z$, $x \vee (y \vee z) = (x \vee y) \vee z$
- *Commutativité :* $x \wedge y = y \wedge x$, $x \vee y = y \vee x$
- *Idempotence :* $x \wedge x = x$, $x \vee x = x$
- *Absorption :* $x \wedge (y \vee x) = x$, $x \vee (y \wedge z) = x$

Définition 16 *Considérons un ensemble ordonné* T*. On dit que* T *un* ***inf-demi-treillis (resp. sup-demi-treillis)*** *si pour tout couple* $(x, y) \in T \times T$ *l'infimum* $x \wedge y$ *(resp. le supremum* $x \vee y$*) existe. Un treillis est un ensemble ordonné qui est à la fois inf-demi-treillis et sup-demi-treillis.*

Théorème 2 *Un sup-demi-treillis (resp. inf-demi-treillis) ayant un plus petit élément (resp. plus grand) est un treillis (Monjardet, 2003).*

Théorème 3 ***Théorème de Stone (Isomorphisme de treillis)*** *Le treillis* $(\mathcal{P}(\mathcal{I}), \subseteq)$*, où* $\mathcal{I}$ *est un ensmble de cardinal n est isomorphe au treillis* $(\mathrm{B}^n, \leq)$

Si on considère un ensemble d'items $\mathcal{I}$ et une base de transactions $\mathcal{D}$, on définit alors dans notre cadre la fonction Γ qui s'écrit :

$\Gamma : \mathcal{P}(\mathcal{I}) \to \mathrm{B}^n \times \mathrm{N} : \mathrm{X} \longmapsto ((b_1, b_2, \ldots, b_n), \text{fréquence}(\mathrm{X}))$ tel que $b_i = 1$ ssi $x_i \in \mathrm{X}$, 0 sinon.

Définition 17 *On appelle* ***sous-treillis*** *d'un treillis* T *tout sous-ensemble non vide* Q *de* T *tel que pour tout* $a, b \in \mathrm{Q}$*,* $a \vee b$ *et* $a \wedge b$ *appartiennent à* Q*.*

Définition 18 *Un treillis* T *est dit complet s'il est à la fois inf-demi-treillis complet et sup-demi-treillis complet.*

Remarque 1 *Dans un treillis fini, on peut toujours définir une relation de couverture. Donc, tout treillis fini peut être représenté par un diagramme de Hasse.*

Définition 19 *Un élément* $e \in \mathrm{T}$ *est un sup-irréductible, si et seulement si e couvre un seul élément. Alors un treillis* T *est dit atomistique (resp. coatomistique), si tous les sup-irréductibles (resp. inf-irréductible) sont atomes (resp. coatomes).*

Définition 20 *Un treillis* T *est dit modulaire, si pour tous* $x, y, z \in \mathrm{T}$ *tels que* $x \leq z$*, on a :* $x \vee (y \wedge z) = (x \vee y) \wedge z$*. Un treillis* T *est dit distributif si pour tous* $x, y, z \in \mathrm{T}$*,* $x \vee (y \wedge z) = (x \wedge y) \vee (x \wedge y)$*. Un treillis booléen est un treillis* T *qui est la fois distributif et atomistique.*

1.2.3 Famille de Moore.

Une famille de Moore appelée aussi Systèmes de Fermeture permettant d'obtenir qu'elles sont criptomorphe à d'autres notion telles que : les Opérateurs de Fermeture, Systèmes implificatif, etc.(Domenach, 2002). Nous allons voir la relation entre la correspondance réciproque et les familles de Moore et Opérateurs de Fermetures, et les ou la famille(s) de Moore et Systèmes implicatifs.

Définition 21 *Soit* E *un ensemble. Une famille de Moore sur* E *est une partie* $\mathcal{F}$ *de l'ensemble* $\mathcal{P}(E)$ *de parties de* E *vérifiant les deux premières conditions suivantes :*
C1 : $E \in \mathcal{F}$
C2 : $\mathcal{F}' \subseteq \mathcal{F} \Rightarrow \cap\mathcal{F}' \in \mathcal{F}$
C3 : $F_1, F_2 \in \mathcal{F}' \subseteq F_1 \cap F_2 \in \mathcal{F}$

Si $\mathcal{F}$ est un ensemble fini, donc E l'est aussi, la condition C2 peut être remplacée par C3. Les éléments de $\mathcal{F}$ sont appelés aussi les *fermés* de $\mathcal{F}$. Dans la suite, nous ne considérons que les familles de Moore finies.

Exemple 1 *Soit* $E = \{A, B, C, D, G\}$.

Alors $\mathcal{F} = \{\varnothing, A, B, C, D, DE, BCD, ABCDG\}$ est une famille de Moore sur l'ensemble E. Ici, les ensembles finis sont notés comme des mots. Par exemple "AG" désigne la partie $\{A, G\}$.

Exemple 2 *Soit* E *un ensemble et,* A *et* B *deux parties de* E. *Alors, la famille* $\mathcal{F}_{A,B}$*, de sous ensemble de* E*, définie par :* $\mathcal{F}_{A,B} = \{X \subseteq E \quad ou \quad B \subseteq X\}$*, est une famille de Moore. En particulier :*

- *Pour* $A = \varnothing$, $\mathcal{F}_{\varnothing,B} = \{X \subseteq E : B \subseteq X\}$
- *Pour* $B = \{i\}$, $\mathcal{F}_{A,\{i\}} = \{X \subseteq E : A \nsubseteq X \text{ ou } i \in X\}$ *est noté* $\mathcal{F}_{A,i}$

Remarque 2 *Notons que* $\mathcal{F}_{A,B} = \mathcal{F}_{A,B-A}$*. En effet, si* $X \in \mathcal{F}_{A,B}$ *alors il est clair que* $X \in \mathcal{F}_{A,B-A}$*. Réciproquement, si* $X \notin \mathcal{F}_{A,B-A}$*, i.e.* $A \subseteq B$ *et* $B \nsubseteq X$*, alors* $A \subseteq X$ *et* $B - A \nsubseteq X$*, i.e.* $X \notin \mathcal{F}_{A,B-A}$*. Par ailleurs, d'après la définition de* $\mathcal{F}_{A,B}$*, si* F *est un fermé qui contient* A *alors* F *contient aussi* B*, en d'autres termes* A *implique* B*. Ainsi, l'ensemble* $\mathcal{F}_{A,B}$ *sera dit* ***famille de Moore implicative****.*

1.2.4 Correspondances de Galois associées à une relation binaire.

Ce sous paragraphe présente les notions de correspondances de Galois. On va ignorer les expressions qui éloignent la théorie de la fouille de données.

Définition 22 *Soient* $(E, \leq), (F, \leq)$ *deux ensemble ordonnées et* $f : E \to F$ *et* $g : F \to E$ *deux applications. Le couple d'applications* (f, g) *sera dit correspondance de Galois entre* E *et* F *si, pour tous* $x, x' \in E$, $y, y' \in F$*, les trois conditions suivantes sont vérifiées :*
C4 : $x \leq x'$ *implique* $f(x) \geq f(x')$ *(antitonie) ;*
C5 : $y \leq y'$ *implique* $g(x) \geq g(x')$ *(antitonie) ;*
C6 : $x \leq gof(x)$ *et* $y \geq fog(y)$ *(antitonie) ;*

L'application f (resp. g) est appelée *application galoisienne* de E vers F (resp. de F vers E).

Proposition 1 *Soient* $(E,\leq),(F,\leq)$ *deux ensembles ordonnés et* $f : E \to F$ *et* $g : F \to E$ *deux applications. Le couple* (f,g) *est une correspondance de Galois si et seulement si :*

C7 : Pour tout $(x,y) \in E \times F, x \leq g(y) \Leftrightarrow y \leq f(x)$

Proposition 2 *Soit* (f,g) *une correspondance de Galois. On a les égalités suivantes :* $f = fogof$ *et* $g = gofog$. *Ainsi, une correspondance de Galois* (f,g) *sera dit un couple* ***involutif****.*

Proposition 3 *Soit* (f,g) *une correspondance de Galois entre deux ensembles* E *et* F. *Notons* $\phi = gof$ *et* $\phi' = fog$. *Les deux applications* ϕ *et* ϕ' *sont de fermetures respectivement sur* E *et* F.

Proposition 4 *Une application* $f : E \to F$ *(resp.* $g : F \to E$*) est résiduée (resp. résiduelle) si et seulement si* f *(resp.* g*) est une application galoisienne de* E *dans* F^d *(resp.* F^d *vers* E*), où* F^d *est le dual de* F.

Donc, la notion d'applications résiduelle et celle de correspondance de Galois sont équivalentes.

Soient E et F deux ensembles finis et $\mathcal{R}$ une relation binaire de E vers F. Définissons deux fonctions $f_\mathcal{R}$ et $g_\mathcal{R}$ de la façon suivante :

$f_\mathcal{R} : \mathcal{P}(E) \to \mathcal{P}(F) : X \longmapsto f_\mathcal{R}(X) = \bigcap_{x \in X}\{y \in F : x\mathcal{R}y\} = \{y \in F : \text{pour tout} \quad x \in X, x\mathcal{R}y\}.$

$g_\mathcal{R} : \mathcal{P}(F) \to \mathcal{P}(E) : Y \longmapsto g_\mathcal{R}(Y) = \bigcap_{y \in Y}\{x \in E : x\mathcal{R}y\} = \{x \in E : \text{pour tout} \quad y \in Y, x\mathcal{R}y\}$

Théorème 4 *(Ganter & Wille, 1999) Le couple* $(f_\mathcal{R}, g_\mathcal{R})$ *est une correspondance de Galois. Réciproquement, si* (f,g) *est une correspondance de Galois entre les ensembles* $\mathcal{P}(E)$ *et* $\mathcal{P}(F)$ *alors* $\mathcal{R}_{f,g} = \{(x,y) \in E \times F : x \in g(\{y\})\} = \{(x,y) \in E \times F : y \in f(\{x\})\}$ *est une relation binaire de* E *vers* F. *Par ailleurs, nous avons les égalités suivantes :* $f_{\mathcal{R}_{(f,g)}} = f$, $g_{\mathcal{R}_{(f,g)}} = g$ *et* $\mathcal{R}_{(f_\mathcal{R}, g_\mathcal{R})} = \mathcal{R}$

Corollaire 1 *Les applications* $\phi = g_\mathcal{R} o f_\mathcal{R}$ *et* $\phi' = f_\mathcal{R} o g_\mathcal{R}$ *sont des opérateurs de fermetures respectivement sur* $\mathcal{P}(E)$ *et* $\mathcal{P}(F)$.

Terminologies

- tout $X \in \mathcal{P}(E)$ tel que $g_\mathcal{R} o f_\mathcal{R}(X) = X$ est un fermé de X
- tout $Y \in \mathcal{P}(F)$ tel que $f_\mathcal{R} o g_\mathcal{R}(Y) = Y$ est un fermé de Y

Ce type de correspondance de Galois joue un rôle important en analyse formelle de concepts (AFC). Cette notion d'AFC fournit un cadre théorique fondamental pour la fouille des règles d'association d'un contexte binaire. En AFC, l'ensemble E désigne un ensemble fini d'entités et F un ensemble fini d'attributs ou variables.

1.2.5 Contexte d'extraction de règles d'association.

Nous définissons dans ce paragraphe les principaux concepts liés à la tâche d'extraction des règles d'association.

Définition 23 Transaction. *Soit* $\mathcal{D} = \{t_1, \ldots, t_n\}$ *un ensemble de n transactions (une succession d'items exprimée selon un ordre donné et prédéfini). Supposons que* $\mathcal{D}$*, soit binaire, c'est-à-dire, qu'on peut décrire chaque transaction* T *au moyen d'un ensemble fini d'attributs* $\mathcal{I} = \{i_1, \ldots, i_m\}$*, où* $i_k = items, k = \overline{1,m}$. *A chaque transaction* T*, on associe un identifiant TID (Transaction IDentifier).*

Une transaction T *est un ensemble d'items tel que* T *est inclus dans* $\mathcal{I}$. *Dans une base de données de vente, une transaction consiste à un ensemble d'articles achetés par un client particulier, appelés items ou attributs (Salleb, 2003; Feno, 2007).*

Définition 24 *Un contexte binaire est un triplet* $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *dans lequel* $\mathcal{O}$ *et* $\mathcal{I}$ *sont, respectivement, des ensembles finis de transactions (ou objets) et d'items (ou attributs), et* $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$ *est une relation binaire entre les transactions et les items. Un couple* $(o, i) \in \mathcal{R}$ *dénote le fait que la transaction* $o \in \mathcal{O}$ *contient l'item* $i \in \mathcal{I}$.

On suppose que les données $\mathcal{D}$ à explorer sont binaires, i.e. qu'on peut décrire chaque transaction au moyen d'un ensemble fini d'items $\mathcal{I} = \{i_1, \ldots, i_m\}$, également appelés attributs. Chaque transaction t sera donc un sous-ensemble de $\mathcal{I}$. De plus, on associe à chaque transaction un identifiant $\texttt{TID} : \mathcal{O} = \{o_1, \ldots, o_n\}$, c'est-à-dire $\forall (o, i) \in \mathcal{O} \times \mathcal{I}$ tel que $o[i] = 1$ si l'objet i est présent dans o et $o[i] = 0$ sinon. Le motif X est un sous-ensemble d'items de $\mathcal{I}$: $X \subseteq \mathcal{I}$. Dans un contexte d'extraction, les transactions sont généralement dénotées par des nombres et les items par des lettres. Le tableau 1.2 ci-dessous présente un exemple de contexte binaire ayant quatre items $\{A, B, C, D\}$ et cinq transactions $\{1, 2, 3, 4, 5\}$.

Définition 25 Base de données ou contexte formel. *Un contexte d'extraction est un triplet* $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *dans lequel* $\mathcal{O}$ *et* $\mathcal{I}$ *sont, respectivement, des ensembles finis d'objets et d'items (ou attributs), et* $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$ *est une relation binaire entre les objets et les items. Un couple* $(o, i) \in \mathcal{R}$ *dénote le fait que l'objet* $o \in \mathcal{O}$ *contient l'item* $i \in \mathcal{I}$.

Une base de données est constituée d'un ensemble fini d'objets $\mathcal{O}$, d'un ensemble fini d'attributs $\mathcal{I}$, et d'une relation $\mathcal{R}$ entre ces deux ensembles. Cette relation peut prendre diverses formes suivant le type de base que l'on étudie. Un contexte d'extraction de règles d'association $\mathcal{D}$ constitué de cinq objets, chacun identifié par son TID, et quatre items est représenté dans la table 1.1. Le tableau 1.2 représente cette même base mais dans un contexte binaire.

TID	Attributs
o_1	A C
o_2	B C D
o_3	D
o_4	A C
o_5	A C D

TABLEAU 1.1 – Contexte d'extraction

Définition 26 Tidset. *On appelle Tidset tout sous-ensemble d'identificateurs de transactions* TID *de* $\mathcal{I}$.

Pour simplifier, on écrira un Tidset sans les accolades et sans les virgules séparant les éléments de l'ensemble.

Exemple 3 *Le Tidset* $\{1, 5\}$, *noté* 15, *représente des identificateurs des transactions* 1 *et* 5 *d'une base de données.*

TID	A	B	C	D
o_1	1	0	1	0
o_2	0	1	1	1
o_3	0	0	0	1
o_4	1	0	1	0
o_5	1	0	1	1

TABLEAU 1.2 – Représentation en mode binaire : $\mathcal{O} = \{o_1, o_2, o_3, o_4, o_5\}$ et $\mathcal{I} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$

Définition 27 Base transactionnelle. *Une base transactionnelle $\mathcal{D}$ est un ensemble de couples formés d'un identificateur de transaction* TID *et de la transaction t proprement dite, telle que :*

$$\mathcal{D} = \{(t, \mathrm{X}_t) | t \in \mathrm{T}, \mathrm{X}_t \subseteq \mathcal{I}\}$$

Exemple 4 *Le tableau 1.2 ci-dessus représente une base transactionnelle de* 5 *transactions sur* 4 *items, où $\mathcal{O}$ joue le rôle de l'ensemble fini de transactions* T*. Pour simplifier l'écriture, on note, dans ce cas, les o_k par k.*

$$\mathcal{I} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$$
$$\mathrm{T} = \{1, 2, 3, 4, 5\}$$
$$\mathcal{D} = \{(1, \mathrm{AC}), (2, \mathrm{BCD}), (3, \mathrm{D}), (4, \mathrm{AC}), (5, \mathrm{ACD})\}$$

Définition 28 Item. *Le terme item, traduction anglaise de article, objet, attribut appartenant à un ensemble fini d'éléments distincts $\mathcal{I} = \{x_1, x_2, \ldots, x_m.\}$, a pour origine de bases de données de vente (Agrawal et al., 1993).*

Exemple 5 *Dans le tableau 1.2, l'ensemble fini d'éléments $\mathcal{I} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$ contient quatre items* A, B, C, D.

Définition 29 Itemset. *Un itemset est un ensemble d'items, tel que : $\mathrm{X} \subseteq \mathcal{I}$, où $\mathcal{I}$ un ensemble fini d'attributs (ou items). La conjoction **Ordinateur** $\wedge$ **Scanner** $\wedge$ **Imprimante** indique un itemset composé des trois items **Ordinateur, Scanner, Imprimante**.*

Définition 30 k-Itemset. *Un k-itemset est un sous-ensemble d'items, tel que : $\mathrm{X} \subseteq \mathcal{I}$ de taille k. La conjoction **Ordinateur** $\wedge$ **Scanner** $\wedge$ **Imprimante** est 3-itemset.*

Définition 31 Motif. *Un motif est un sous-ensemble de $\mathcal{I}$. On dit qu'un motif $\mathcal{I}$ est inclus dans l'objet o (ou que o contient $\mathcal{I}$) si i et o sont en relation : $\forall i \in \mathcal{I}, (o, i) \in \mathcal{R}$. Un motif de taille k est noté k-motif. Les motifs sont aussi appelés ensembles d'items (ou itemsets) dans la litterature anglo-saxonne.*

Définition 32 Image d'un motif, image d'un ensemble d'objet. *Soit f la fonction qui fait correspondre à un motif, l'ensemble des objets qui le contiennent tel que :*

$$\mathrm{O} \longmapsto f(\mathrm{O}) = \{i \in \mathcal{I} | \forall o \in \mathrm{O}, (o, i) \in \mathcal{R}\}$$

Soit g sa fonction duale, définie pour un ensemble d'objet :

$$\mathrm{I} \longmapsto g(\mathrm{I}) = \{o \in \mathcal{O} | \forall i \in \mathrm{I}, (o,i) \in \mathcal{R}\}$$

Les fonctions f et g définissent la connexion de Galois d'une relation binaire. Les opérateurs associés $\varphi = gof$ et $\varphi' = fog$ sont des opérateurs de fermeture de Galois (Birkhoff, 1967; Barbut & Monjardet, 1970; Ganter & Wille, 1999).

Définition 33 Correspondance de Galois dans un contexte d'extraction. *Soit $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ un contexte d'extraction. Dans un contexte d'extraction, les objets sont dénotés par des nombres et les attributs par des lettres. Nous définissons deux fonctions f et g suivantes, récapitulant les liens entre les sous-ensembles d'objets et sous-ensembles d'attributs induits par $\mathcal{R}$.*

$$\mathcal{P}(\mathcal{O}) \to \mathcal{P}(\mathcal{I}) : \mathrm{O} \longmapsto f(\mathrm{O}) = \{i \in \mathcal{I} | \forall o \in \mathrm{O}, o\mathcal{R}i\}$$

$$\mathcal{P}(\mathcal{I}) \to \mathcal{P}(\mathcal{O}) : \mathrm{I} \longmapsto g(\mathrm{I}) = \{o \in \mathcal{O} | \forall i \in \mathrm{I}, o\mathcal{R}i\}$$

Le couple (f,g) est une correspondance de Galois entre l'ensemble des parties de $\mathcal{O}$ et l'ensemble des parties de $\mathcal{I}$ (Barbut & Monjardet, 1970; Ganter & Wille, 1999).

Définition 34 Fermeture de correspondance de Galois dans un contexte d'extraction. *Soit* $\mathrm{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *un contexte d'extraction, f et g les opérateurs de connexion de Galois définis dans la précédente définition. Alors, les opérateurs $h = fog$ et $h' = gof$ sont des opérateurs de fermeture de la correspondance de Galois (Ganter & Wille, 1999).*

Définition 35 Opérateur de fermeture disjonctive *Soit $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ un contexte d'extraction. l'opérateur de fermeture disjonctive, noté φ_d, est définie comme suit (Hamrouni T. et al., 2007) :*

$$\varphi_d : \mathcal{P}(\mathcal{I}) \to \mathcal{P}(\mathcal{I}) : \mathrm{I} \longmapsto \varphi_d(\mathrm{I}) = \{i \in \mathcal{I} | (\forall o \in \mathrm{O})((o,i) \in \mathcal{R}) \Rightarrow ((\exists i_{\mathcal{I}} \in \mathrm{I})((o, i_{\mathcal{I}}) \in \mathcal{R}))\}$$

Autrement dit, la fermeture disjonctive d'un itemset I est égale à l'ensemble des items qui apparaissent *uniquement* dans les transactions qui contiennent au moins un item de I.

Définition 36 Générateur minimal. *(Bastide et al., 2000) Un motif $g \subseteq \mathrm{I}$ est dit générateur minimal d'un motif fermé f, si et seulement si $f_c(g) = f$ et il n'existe aucun motif $g_1 \subset g$ tel que $f_c(g_1) = f$.*

Exemple 6 *Considérons le contexte ullustré dans le tableau 1.2. Dans ce cas, $\{\mathrm{A}\}$ est l'ensemble minimal d'items communs aux transactions $\{o_1, o_4, o_5\}$, i.e. $f_c(\mathrm{A}) = \mathrm{AC}$, donc* A *est un générateur minimal de* AC.

Afin d'évaluer l'intérêt d'un motif, plusieurs mesures sont utilisées dont les plus connues sont présentées à travers la définition suivante.

Définition 37 Support d'un itemset. *(Hamrouni et al., 2005)Soit $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ un contexte d'extraction et $\mathrm{I} \subseteq \mathcal{I}$ un itemset. Nous distinguons trois types de supports relatifs à* I *:*

1. $Supp(\wedge \mathrm{I}) = |\{o \in \mathrm{O} | \forall i \in \mathrm{I}, (o,i) \in \mathcal{R}\}|$
2. $Supp(\vee \mathrm{I}) = |\{o \in \mathrm{O} | \exists i \in \mathrm{I}, (o,i) \in \mathcal{R}\}|$

3. $Supp(\neg \mathrm{I}) = |\{o \in \mathrm{O} | \forall i \in \mathrm{I}, (o,i) \notin \mathcal{R}\}|$

Expliquons sémantiquement ces supports :

1. $Supp(\wedge \mathrm{I})$ correspond au support conjoctif de I, i.e. au nombre de transactions qui contiennent tous les items de I.
2. $Supp(\vee \mathrm{I})$ correspond au support disjonctif de I, i.e. au nombre de transactions qui contiennent au moins un item de I.
3. $Supp(\neg \mathrm{I})$ correspond au support négatif de I, i.e. au nombre de transactions qui ne contiennent aucun item de I.

Exemple 7 *Considérons le contexte ullustré par le tableau 1.2. Nous avons* $Supp(\wedge(\mathrm{AC})) = |\{o_1, o_4, o_5\}| = 3$, $Supp(\vee(\mathrm{AC})) = |\{o_1, o_2, o_4, o_5\}| = 4$ *et* $Supp(\neg(\mathrm{AC})) = |\{o_3\}| = 1$,

Définition 38 *On appelle* ***support*** *d'un itemset* X, *noté* $Supp(\mathrm{X}')$, *le rapport entre le nombre de transactions contenant* X *et le nombre total de transactions de* $\mathcal{D}$ *(Vaillant, 2006).*

$$Supp(\mathrm{X}') = \frac{|\{\mathrm{T} \in \mathcal{D} : \mathrm{X} \subseteq \mathrm{T}\}|}{|\mathcal{D}|} = \mathcal{P}(\mathrm{X}')$$

Définition 39 *Le support d'un itemset est la fréquence d'apparition de cet itemset dans les données (Vaillant, 2006). Il est le rapport de la cardinalité de l'ensemble des transactions qui contiennent tous les attributs de* X *par la cardinalité de l'ensemble de toutes les transactions.*

Au besoin des notions introduites dans la suite de ce mémoire, on va créer une autre table enregistrant une base de données $\mathcal{D}$ pour un ensemble de 6 transactions $\{t_1, \ldots, t_6\}$ décrit par 5 items $\{i_1, \ldots, i_5\}$.

TID	Attributs
t_1	i_1, i_3 i_4
t_2	i_2, i_3, i_5
t_3	i_1, i_2, i_3, i_5
t_4	i_2, i_5
t_5	i_1, i_2, i_3, i_5
t_6	i_2, i_3, i_5

TABLEAU 1.3 – Base de données $\mathcal{D}$, $\mathrm{T} = \{t_1, t_2, t_3, t_4, t_5, t_6\}$ et $\mathrm{X} = \{i_1, i_2, i_3, i_4, i_5\}$

Exemple 8 *Dans les données du tableau 1.3 listées ci-dessus, on a :*

$$Supp(i_1) = \frac{|\{t_1, t_3, t_5\}|}{6} = 3/6 = 50\%$$
$$Supp(i_2, i_3) = \frac{|\{t_2, t_3, t_5, t_6\}|}{6} = 4/6 = 67\%$$

Cette définition du support est une définition absolue. Il possède une variante relative, obtenue en normalisant le support par rapport à la base, c'est-à-dire en divisant sa valeur par le nombre de transactions contenues dans la base de données. On se ramène ainsi à une fréquence

d'apparition du motif dans la base. Par la suite, nous ferons preuve d'un abus de langage en parlant de probabilité plutôt que de fréquence. L'analogie dans ce domaine est simple, et permet d'introduire des notions classiques du calcul des probabilités.

Propriété 1 $Supp(X \cup Y) \leq min(Supp(X'), Supp(Y)) \leq max(Supp(X'), Supp(Y)) \leq Supp(X \cap Y)$

Preuve En effet, plus un itemset est grand, plus le nombre de transactions le contenant est faible, d'où une diminution de son support. L'union de deux itemsets est l'intersection des transactions les contenant. Cette vérification est évidente, en utilisant la table 1.3, on a :

$$Supp(i_1, i_2, i_3) = \frac{|\{t_2, t_5\}|}{6} = 33\% \leq min(Supp(i_1) = 50\%, Supp(i_2, i_3) = 67\%) = 50\%$$

Définition 40 Itemset fréquent. *Le cadre mathématique est classique et formalise celui défini par Agrawal et al. (1993); Agrawal & Srikant (1994a). On dit qu'un itemset* X *est fréquent lorsque son support est supérieur ou égal à un seuil de référence* $minsupp \in [0,1]$*, fixé à priori par l'utilisateur. Donc un itemset n'ayant pas le support suffisant est dit non fréquent. Autrement dit, dans une base de données de transactions, un seuil de référence* $minsupp$ *s'appelle aussi une valeur de support minimal (minsupp). Un motif* X *est dit fréquent si :*

$$Supp(X') \geq minsupp.$$

Exemple 9 *Si on choisit, dans la table 1.3, le seuil minimal de support* $minsupp = 50\%$*, on obtient que* $\{i_1\}$ *et* $\{i_2, i_3\}$ *sont fréquents, mais non pas que* $\{i_1, i_2, i_3\}$*. En revanche, en choisissant un seuil à* 30%*, les trois itemsets sont tous fréquents.*

Propriété 2 *Etant donné* $minsupp$*, un seuil de référence fixé par l'utilisateur.*

- *Tout sous-ensemble d'un itemset fréquent est fréquent ;*
- *Tout sur-ensemble d'un itemset non fréquent est non fréquent.*

Preuve. Soit $minsupp$ un seuil de référence fixé. Or par définition, tout itemset de support supérieur ou égal à $minsupp$ est fréquent. Soit *X'* un sous-ensemble d'un itemset frequent *X*. D'après la propriété 1, on a : $X' \subseteq X \Rightarrow Supp(X') \geq Supp(X')$. X étant fréquent, $Supp(X') \geq minsupp$. Ainsi, $Supp(X') \geq minsupp$, et X' est donc fréquent.

De par cette même propriété, Y' étant un sur-ensemble d'un itemset *Y* non fréquent, on déduit que : $Y \subseteq Y' \Rightarrow Supp(Y) \geq Supp(Y')$. Y étant non fréquent, $minsupp > Supp(Y)$. Il devient alors $minsupp > Supp(Y')$ et Y' est donc non fréquent.

Définition 41 Motif fréquent maximal. *Un motif* X *est dit fréquent maximal s'il est fréquent et que tous ses sur-ensembles sont non fréquents. Formellement, l'ensemble* $\mathcal{F}_m$ *des motifs fréquents maximaux d'un contexte* $\mathcal{D}$ *est défini par :*

$$\mathcal{F}_m = \{X \in \mathcal{I}; \forall Y \supset X, Y \notin \mathcal{I}\}$$

Définition 42 Motif fréquent fermé. *Un motif fermé est un ensemble maximal de motifs communs à un ensemble d'objets. Un motif* $X \subseteq \mathcal{I}$ *est dit* φ *– fermé (ou tout simplement fermé) si* $\varphi(X) = X$*. Il*

est dit φ *– fréquent fermé s'il est à la fois* φ *– fréquent et fermé. Formellement, l'ensemble des motifs fréquents fermés d'un contexte* $\mathcal{D}$ *est défini par :*

$$\mathcal{F}_f = \{X \in \mathcal{I} : \varphi(X) = X \text{ et } supP(X') \geq minsup\}$$

Remarque 3 *Pour un motif* $X \subseteq \mathcal{I}$*, l'image* $\varphi(X)$ *sera appelée la fermeture de X. Elle correspond au plus petit fermé qui contient* X.

Définition 43 Fermeture conjonctive d'un motif. *(Pasquier et al., 2005) La fermeture conjonctif d'un motif* $I \subseteq \mathcal{I}$*, noté* $f_c(I)$*, est égal à* $max \subseteq \{I' \subseteq \mathcal{I} | (I' \subseteq I) \text{ et } (Supp(\wedge I) = Supp(\vee I))\}$

Exemple 10 *Considérons le contexte du tableau 1.2. Dans ce cas,* $\{AC\}$ *est l'ensemble maximal d'items communs de transactions* $\{o_1, o_4, o_5\}$*, i.e.* $f_c(AC) = AC$*, donc* AC *est un motif fermé conjonctif.*

Définition 44 Fermeture disjonctive d'un motif. *La fermeture disjonctive d'un motif* $I \subseteq \mathcal{I}$*, notée* $f_d(I)$*, est égal au* $max \subseteq \{I' \subseteq \mathcal{I} | (I' \subseteq I) \text{ et } (Supp(\wedge I) = Supp(\vee I))\}$ *Hamrouni T. (2009)*

Définition 45 Motif essentiel. *Soient* $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *un contexte d'extraction et un motif* $I \subseteq \mathcal{I}$*.* I *est un motif essentiel si et seulement si* $Supp(\vee I))\} > max\{Supp(\vee I \setminus \{i\}) | i \in I\}$ *(Casali et al., 2005).*

Exemple 11 *Soit le contexte défini dans le tableau 1.2, nous remarquons que le motif* AC *n'est pas essentiel car,* $Supp(\vee(AC)) = Supp(\vee A) = 4$

Définition 46 Bordure positive. *Étant donné un contexte* $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *et un seuil minimum de support* ***minsupp****, on appelle bordure positive* $\mathcal{B}^+$ *l'ensemble des itemsets peu fréquents dont tous les sur-ensembles sont fréquents. Ces itemsets correspondent aux itemsets fréquents maximaux.*

$$\mathcal{B}^+ = \{X \in \mathcal{I} | SupP(X') \geq minsupp, \forall X' \supseteq X, Supp(X') \geq minsupp\}$$

Définition 47 Bordure négative. *Étant donné un contexte* $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *et un seuil minimum de support* ***minsupp****, on appelle bordure négative* $\mathcal{B}^-$ *l'ensemble des itemsets peu fréquents dont tous les sous-ensembles sont fréquents. Ces itemsets correspondent aux itemsets peu fréquents maximaux.*

$$\mathcal{B}^- = \{X \in \mathcal{I} | SupP(X') < minsupp, \forall X' \subseteq X, Supp(X') \geq minsupp\}$$

Définition 48 Règle d'association. *Une règle d'association est une implication conditionnelle de la forme* $X \rightarrow Y$*, où* X *et* Y *sont des ensembles d'attributs disjoints* $(X, Y \subseteq \mathcal{I}$ *et* $X \cap Y = \varnothing)$*. La partie gauche de la règle* X *est appelée la prémisse et la partie droite* Y *est appelée la conclusion de la règle. Dans un contexte formel, les règles d'association sont des règles qui sont extraites d'une base de données transactionnelles et qui décrivent des associations entre certains éléments.*

Définition 49 Règle d'association dans un cadre formel. *Dans un contexte formel, les règles d'association sont des règles qui sont extraites d'une base de données transactionnelles et qui décrivent des associations entre certains éléments.*

Définition 50 *On définit* ***le support d'une règle d'association*** *comme étant le support de l'itemset* $X \cup Y$*, i.e. la proportion de transactions contenant à la fois* X *et* Y *:*

$$Supp(X \to Y) = Supp(X \cup Y) = \frac{|\{T \in \mathcal{D} : X \subseteq T et\, Y \subseteq T\}|}{|\mathcal{D}|}$$

Il correspond à la proportion d'entités contenant à la fois la prémisse et le conséquent de la règle.

Définition 51 Support et Confiance d'une règle. *On définit également* ***la confiance d'une règle associative*** $X \to Y$*, notée* $Conf(X \to Y)$*, comme étant le rapport entre le support de l'itemset* $X \cup Y$ *et de celui de* X *:*

$$Conf(X \to Y) = \frac{Supp(X \cup Y)}{Supp(X')} = \mathcal{P}_{X'}(Y')$$

Il s'agit de la proportion d'entités contenant le conséquent parmi celles qui contiennent la prémisse.

Exemple 12 *De la règle* $\{i_2, i_5\} \to \{i_3\}$*, le support de l'ensemble* $\{i_2, i_3, i_5\}$ *étant égal à* 4 *et le nombre total de transactions étant égal à* 6*, donc le support de la règle est* $4/6 \approx 67\%$.

La confiance est obtenue en divisant le support de l'itemst $\{i_2, i_3, i_5\}$ *par le support de l'itemset* $\{i_2, i_5\}$*. Comme il y a* 5 *transactions contenant* $\{i_2, i_5\}$*, la confiance de cette règle est donc* $4/5 = 80\%$.

Définition 52 Règle valide au sens support-confiance. *D'après Agrawal et al. (1993), en fixant au préalable un minimum de seuil de support et un minimum de seuil de confiance, respectivement noté "****minsupp****" et "****minconf****", une règle d'association* $X \to Y$ *est valide au sens support-confiance si :*

$$Supp(X \to Y) \geq minsupp$$
$$Conf(X \to Y) \geq minconf$$

Exemple 13 *Si on fixe un seuil minimum de support* $minsupp = 50\%$ *et un seuil minimum de confiance* $minconf = 70\%$*, la règle* $\{i_2, i_5\} \to \{i_3\}$*, est valide au sens support-confiance, car* $Supp(\{i_2, i_5\} \to \{i_3\}) = 67\% \geq 50\%$ *et* $Conf(\{i_2, i_5\} \to \{i_3\}) = 80\% \geq 70\%$

Il se trouve que support confiance n'est pas à l'abri d'un inconvénient. En effet, selon Totohasina (2008), confiance n'est pas implicative, puis avec (supp, conf) seul, il arrive que beaucoup de règles redondantes soient validés.

1.2.6 Construction de M_{GK}

Présentons maintenant la construction de la mesure M_{GK}

1. Si X défavorise Y, on a : $P_{X'}(Y') < P(Y')$ soit $P_{X'}(Y') - P(Y') < 0$. Dans le cas de l'indépendance entre X et Y, on a : $P_{X'}(Y') - P(Y') = 0$. Par ailleurs, $0 \leq P_{X'}(Y') \Rightarrow -P(Y') \leq P_{X'}(Y') - P(Y')$. L'égalité est atteinte au cas de l'incompatibilité entre X et Y. En considérant l'indépendance comme situation limite de dépendance négative. On peut alors écrire :

$$-P(Y') \leq P_{X'}(Y') - P(Y') \leq 0, \text{si} \quad X \quad \text{défavorise} \quad Y$$

soit

$$-1 \leq \frac{P_{X'}(Y') - P(Y')}{P(Y')} \leq 0, \text{si} \quad X \quad \text{défavorise} \quad Y$$

2. Si X favorise Y, on a : $P_{X'}(Y') > P(Y')$ soit $P_{X'}(Y') - P(Y') > 0$. Dans le cas de l'indépendance entre X et Y, on a : $P_{X'}(Y') - P(Y') = 0$. Par ailleurs, $P_{X'}(Y') \leq 1 \Rightarrow P_{X'}(Y') - P(Y') \leq 1 - P(Y')$. L'égalité est atteinte au cas de l'implication logique entre X et Y. En considérant l'indépendance comme situation limite de dépendance positive, on peut écrire :

$$0 \leq P_{X'}(Y') - P(Y') \leq 1 - P(Y'), \text{si} \quad X \quad \text{favorise} \quad Y$$

soit

$$0 \leq \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')} \leq 1, \text{si} \quad X \quad \text{favorise} \quad Y$$

La définition de cette mesure s'obtient, de la manière suivante, à partir de ces deux propriétés :

Définition 53 *La mesure* M_{GK} *est définie par :*

$$M_{GK}(X \to Y) = \begin{cases} \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')}, & \textit{if } X \textit{ favors } Y \\ \frac{P_{X'}(Y') - P(Y')}{P(Y')}, & \textit{if } X \textit{ disfavors } Y \end{cases}$$

1.2.7 Propriétés mathématiques de la mesure M_{GK}

Nous présentons dans cette sous-section les principales propriétés de la mesure M_{GK}. Dans un contexte $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$, considérons deux motifs X et Y tels que $P(X') \neq 0, P(Y') \neq 0, P(X') \neq 1, P(Y') \neq 1$. Avant d'établir les propriétés de la mesure M_{GK}, nous rappelons les propriétés liant les dépendances positives et négatives entre deux motifs. Le lemme ci-après montre que la notion de dépendance positive (resp. négative) entre deux motifs est une relation symétrique.

Lemme 1 X *favorise* Y *(resp. défavorise* Y*), si et seulement si,* Y *favorise* X *(resp. défavorise* X*).*

Démonstration 1

$$\begin{aligned} X \textit{favorise } Y &\Leftrightarrow P_{X'}(Y') > P(Y') \\ &\Leftrightarrow P(X' \cap Y') > P(X')P(Y') \\ &\Leftrightarrow P_{Y'}(X') > P(X') \\ &\Leftrightarrow Y \textit{ favorise } X \end{aligned}$$

■

$$\begin{aligned} X \textit{défavorise } Y &\Leftrightarrow P_{X'}(Y') < P(Y') \\ &\Leftrightarrow P(X' \cap Y') < P(X')P(Y') \\ &\Leftrightarrow P_{Y'}(X') < P(X') \\ &\Leftrightarrow Y \textit{ défavorise } X \end{aligned}$$

■

Les résultats de la proposition suivante résultent de la définition de M_{GK}.

Proposition 5 ***Situations de référence.*** *Pour tous motifs* X *et* Y*, on a :*

- X *et* Y *sont incompatibles, si et seulement si* $M_{GK}(X \to Y) = -1$
- X *défavorise* Y*, si et seulement si* $-1 \leq M_{GK}(X \to Y) \leq 0$
- X *et* Y *sont indépendant, si et seulement si* $M_{GK}(X \to Y) = 0$
- X *favorise* Y*, si et seulement si* $0 < M_{GK}(X \to Y) \leq 1$
- X *implique logiquement* Y*, si et seulement si* $M_{GK}(X \to Y) = 1$

Ces propriétés expriment le fait que M_{GK} prend ses valeurs sur l'intervalle $[-1;1]$ tout en re étant les situations de références telles que l'incompatibilité, la dépendance négative, l'indépendance, la dépendance positive et l'implication logique entre la prémisse et le conséquent d'une règle.

Proposition 6 ***Situations de référence à l'équilibre.*** *(Totohasina et al., 2004, 2005). À l'équilibre c'est à dire lorsque* $n_{XY} = n_{\overline{XY}}$*, on a alors :*

$$M_{GK}(X \to Y) \approx \frac{1}{2}$$

On peut ainsi assimiler que la mesure M_{GK} est une mesure de déviation d'équilibre pour $n = |\mathcal{O}|$ suffisamment grand, i.e., pour un grand volume de données.

Démonstration 2 *Notons qu'une règle pourrait être intéressante si la prémisse favorise le conséquent. Ainsi, considérons une règle d'association* $X \to Y$ *telle que* X *favorise* Y*. À l'équilibre* $|X \cap Y| = |X \cap \overline{Y}|$*, on a donc :*

$$\begin{aligned} M_{GK}(X \to Y) &= \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')} \\ &= \frac{\frac{1}{2} - \frac{|Y|}{n}}{1 - \frac{|Y|}{n}} \end{aligned}$$

■

Dans la plupart de cas n est suffisamment grand, l'expression ci-dessus peut donc approximée comme suit.

$$M_{GK}(X \to Y) = \frac{1}{2} + o(\frac{1}{n})$$

Proposition 7 X *et* Y *deux motifs.*
(1) Si X *favorise* Y*, alors on a :*

$$M_{GK}(Y \to X) = \frac{1 - P(Y')}{1 - P(X')} \frac{P(X')}{P(Y')} M_{GK}(X \to Y) \tag{1.1}$$

Ainsi est non symetrique.

(2) Si X *défavorise* Y*, alors on a :*

$$M_{GK}(Y \to X) = M_{GK}(X \to Y)?? \tag{1.2}$$

Ainsi est non symetrique.

Démonstration 3 *(1) Si* X *favorise* Y, *alors on a :*

$$\begin{aligned} M_{GK}(Y \to X) &= \frac{P_{Y'}(X') - P(X')}{1 - P(X')} \\ &= \frac{P(X \cap Y) - P(X')P(Y')}{P(Y')(1 - P(X'))} \\ &= \frac{P(X')}{P(Y')} \frac{1 - P(Y')}{1 - P(X')} \frac{P(X \cap Y) - P(X')P(Y')}{P(Y')(1 - P(X'))} \\ &= \frac{P(X')}{P(Y')} \frac{1 - P(Y')}{1 - P(X')} M_{GK}(X \to Y) \end{aligned}$$

(2) Si X *défavorise* Y, *alors on a :* ■

$$\begin{aligned} M_{GK}(Y \to X) &= \frac{P_{Y'}(X') - P(X')}{P(X')} \\ &= \frac{P(X' \cap Y') - P(X')P(Y')}{P(X')P(Y')} \\ &= \frac{P_{X'}(Y') - P(Y')}{P(Y')} \\ &= M_{GK}(X \to Y) \end{aligned}$$

■

Corollaire 2 *La proposition 3 nous montre que la mesure* M_{GK} *est favorablement non symétrique.*

Proposition 8 *(1) Si* X *favorise* Y, *nous avons la relation d'équivalence des deux règles contraposées.*

$$M_{GK}(\overline{Y} \to \overline{X}) = M_{GK}(X \to Y) \tag{1.3}$$

Ainsi, la composante favorisante a la qualité d'être implicative.

(2) Si X *défavorise* Y, *on a :*

$$M_{GK}(\overline{Y} \to \overline{X}) = \frac{P(X')P(Y')}{P(\overline{X'})P(\overline{Y'})} M_{GK}(X \to Y) \tag{1.4}$$

Démonstration 4 *(1) Cas où* X *favorise* Y, *alors on a :*

$$\begin{aligned} M_{GK}(\overline{Y} \to \overline{X}) &= \frac{P_{\overline{Y'}}(\overline{X'}) - P(\overline{X'})}{1 - P(\overline{X'})} \\ &= \frac{-P_{\overline{Y'}}(X') + P(X')}{P(X')} \\ &= \frac{-P(X' \cap \overline{Y'}) + P(X')P(\overline{Y'})}{P(X')P(\overline{Y'})} \\ &= \frac{-P(X')[1 - P_{X'}(Y')] + P(X')[1 - P(Y')]}{P(X')[1 - P(Y')]} \\ &= \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')} \\ &= M_{GK}(X \to Y) \ \textit{CQFD} \end{aligned}$$

■

(2) Pour le cas où X *défavorise* Y*, on a :*

$$\begin{aligned}
M_{GK}(\overline{Y} \to \overline{X}) &= \frac{P_{\overline{Y'}}(\overline{X'}) - P(\overline{X'})}{P(\overline{X'})} \\
&= \frac{-P_{\overline{Y'}}(X') + P(X')}{P(\overline{X'})} \\
&= \frac{-P(X' \cap \overline{Y'}) + P(X')P(\overline{Y'})}{P(\overline{X'})P(\overline{Y'})} \\
&= \frac{-P(X')[1 - P_{X'}(Y')] + P(X')[1 - P(Y')]}{P(\overline{X'})P(\overline{Y'})} \\
&= \frac{P(X')P(Y')}{P(\overline{X'})P(\overline{Y'})} \frac{P_{X'}(Y') - P(Y')}{P(Y')} \\
&= \frac{P(X')P(Y')}{P(\overline{X'})P(\overline{Y'})} M_{GK}(X \to Y) \ \textit{CQFD}
\end{aligned}$$

■

Corollaire 3 M_{GK} *est une mesure implicative dans la zone d'attraction. [C'est la conséquence de la relation (4.1)].*

La mesure M_{GK} s'avère ainsi fort intéressante pour mesurer la dépendance positive entre deux motifs : soit entre motifs positifs, soit entre motifs positifs et négatifs. En effet, par le Lemme 2, on peut toujours se ramener au cas où l'un des motifs (prémisse ou conséquent) favorise l'autre. Donc, dans la pratique, nous retenons tout simplement que M_{GK} est implicative en adoptant la composante favorisante M^f_{GK}comme l'unique composante active pour valider ou non une règle d'association.

Proposition 9 *Soient* X *et* Y *deux motifs positifs, on a :*

$$M_{GK}(X \to \overline{Y}) = -M_{GK}(X \to Y) \tag{1.5}$$

Démonstration 5 *(1) Cas où* X *favorise* Y *:*

$$\begin{aligned}
M_{GK}(X \to \overline{Y}) &= \frac{P_{X'}(\overline{Y'}) - P(\overline{Y'})}{1 - P(\overline{Y'})} \\
&= \frac{1 - P_{X'}(Y') - 1 + P(Y')}{P(Y')} \\
&= -\frac{P_{X'}(Y') - P(Y')}{P(Y')} \\
&= -M_{GK}(X \to Y) \ \textit{CQFD}
\end{aligned}$$

(2) Cas où X *défavorise* Y*, on a :* ■

$$\begin{aligned}
M_{GK}(X \to \overline{Y}) &= \frac{P_{X'}(\overline{Y'}) - P(\overline{Y'})}{P(\overline{Y'})} \\
&= \frac{1 - P_{X'}(Y') - 1 + P(Y')}{1 - P(Y')} \\
&= -\frac{P_{X'}(Y') - P(Y')}{1 - P(Y')} \\
&= -M_{GK}(X \to Y) \ \textit{CQFD}
\end{aligned}$$

■

Le corollaire ci-dessous est une conséquence de la proposition précédente. La mesure M_{GK} permet de calculer les règles négatives valides à partir de règles positives correspondantes.

Corollaire 4 *Soient* $\alpha \in [0,1]$, X *et* Y *deux motifs positifs tels que* X *défavorise* Y, *on obtient la relation suivante :*

$$-1 \leq M_{GK}(X \to Y) \leq -\alpha \Leftrightarrow \alpha \leq M_{GK}(X \to \overline{Y}) < 1$$

Les relations entre les règles négatives à droite et les règles négatives à gauche sont données par la proposition suivante.

Proposition 10 *(1) Si* X *défavorise* Y *(donc,* X *favorise* $\overline{Y}$ *et aussi* $\overline{X}$ *favorise* Y*), on a :*

$$M_{GK}(\overline{X} \to Y) = \frac{P(X')}{P(\overline{X'})}\frac{P(\overline{Y'})}{P(Y')}M_{GK}(X \to \overline{Y}) \tag{1.6}$$

(2) Si X *favorise* Y *(donc,* X *défavorise* $\overline{Y}$ *et aussi* $\overline{X}$ *défavorise* Y*), on a :*

$$M_{GK}(\overline{X} \to Y) = \frac{P(X')P(Y')}{P(\overline{X'})P(\overline{Y'})}M_{GK}(X \to \overline{Y}) \tag{1.7}$$

Démonstration 6 *(1)* X *défavorise* Y *on a :*

$$\begin{aligned}
M_{GK}(\overline{X} \to Y) &= \frac{P_{\overline{X'}}(Y') - P(Y')}{P(Y')} \\
&= \frac{P(\overline{X'} \cap Y') - P(\overline{X'})P(Y')}{P(\overline{X'})P(Y')} \\
&= \frac{-P(X' \cap Y') + P(X')P(Y')}{P(\overline{X'})P(Y')} \\
&= -\frac{P(X')[P_{X'}(Y') - P(Y')]}{P(\overline{X'})P(Y')} \\
&= -\frac{P(X')}{P(\overline{X'})}\frac{P(\overline{Y'})}{P(Y')}\frac{P_{X'}(Y') - P(Y')}{P(\overline{Y'})} \\
&= \frac{P(X')}{P(\overline{X'})}\frac{P(\overline{Y'})}{P(Y')}M_{GK}(X \to \overline{Y}) \ \textit{CQFD}
\end{aligned}$$

■

(2) X *favorise* Y*, on a :*

$$\begin{aligned}
M_{GK}(\overline{X} \to Y) &= \frac{P_{\overline{X'}}(Y') - P(Y')}{1 - P(Y')} \\
&= \frac{P(\overline{X'} \cap Y') - P(\overline{X'})P(Y')}{P(\overline{X'})P(\overline{Y'})} \\
&= \frac{P(Y')[1 - P_{Y'}(X')] - P(\overline{X'})P(Y')}{P(\overline{X'})P(\overline{Y'})} \\
&= -\frac{P(X')P(Y')}{P(\overline{X'})P(\overline{Y'})}\frac{P_{X'}(Y') - P(Y')}{P(Y')} \\
&= \frac{P(X')P(Y')}{P(\overline{X'})P(\overline{Y'})}M_{GK}(X \to \overline{Y}) \ \textit{CQFD}
\end{aligned}$$

■

Proposition 11 *Soient* X *et* Y *deux motifs d'un contexte* $\mathcal{D}$.
(i) Si X *favorise* Y, *on a :*

$$M_{GK}(\overline{X} \to \overline{Y}) = \frac{P(X')}{P(\overline{X'})} \frac{P(\overline{Y'})}{P(Y')} M_{GK}(\overline{Y} \to \overline{X}) \tag{1.8}$$

(ii) Si X *défavorise* Y, *on a :*

$$M_{GK}(\overline{X} \to \overline{Y}) = M_{GK}(\overline{Y} \to \overline{X}) \tag{1.9}$$

Démonstration 7 *(i)* X *favorise* Y, *donc on a :*

$$\begin{aligned} M_{GK}(\overline{X} \to \overline{Y}) &= M_{GK}(Y \to X) \\ &= \frac{P(\overline{X'})}{P(X')} \frac{P(\overline{Y'})}{P(Y')} M_{GK}(\overline{Y} \to \overline{X}) \ \textit{CQFD} \end{aligned}$$

(ii) X *défavorise* Y, *donc on a :*

$$\begin{aligned} M_{GK}(\overline{X} \to \overline{Y}) &= \frac{P(X')}{P(\overline{X'})} \frac{P(Y')}{P(\overline{Y'})} M_{GK}(Y \to X) \\ &= \frac{P(X')}{P(\overline{X'})} \frac{P(Y')}{P(\overline{Y'})} M_{GK}(X \to Y) \\ &= M_{GK}(\overline{Y} \to \overline{X}) \ \textit{CQFD} \end{aligned}$$

■

■

Proposition 12 *Soient* X *et* Y *deux motifs d'un contexte* $\mathcal{D}$.
(i) Si X *favorise* Y, *alors :*

$$M_{GK}(Y \to \overline{X}) = -\frac{P(\overline{Y'})}{P(Y')} M_{GK}(X \to Y) \tag{1.10}$$

(ii) Si X *défavorise* Y, *alors on a :*

$$M_{GK}(Y \to \overline{X}) = -\frac{P(X')}{P(\overline{X'})} M_{GK}(X \to Y) \tag{1.11}$$

Démonstration 8 *(i)* X *favorise* Y, *donc on a :*

$$\begin{aligned} M_{GK}(Y \to \overline{X}) &= \frac{P_{Y'}(\overline{X'}) - P(\overline{X'})}{1 - P(\overline{X'})} \\ &= \frac{-P_{Y'}(X') + P(X')}{P(X')} \\ &= -\frac{P(\overline{X'})}{P(X')} \frac{P_{Y'}(X') - P(X')}{1 - P(X')} \\ &= -\frac{P(\overline{X'})}{P(X')} M_{GK}(Y \to X) \\ &= -\frac{P(\overline{Y'})}{P(Y')} M_{GK}(X \to Y) \ \textit{CQFD} \end{aligned}$$

■

(ii) X *défavorise* Y, *donc on a :*

$$\begin{aligned}
M_{GK}(Y \to \overline{X}) &= \frac{P_{Y'}(\overline{X'}) - P(\overline{X'})}{P(\overline{X'})} \\
&= \frac{-P_{Y'}(X') + P(X')}{P(\overline{X'})} \\
&= -\frac{P(X')}{P(\overline{X'})} \frac{P_{Y'}(X') - P(X')}{P(X')} \\
&= -\frac{P(X')}{P(\overline{X'})} M_{GK}(Y \to X) \\
&= -\frac{P(X')}{P(\overline{X'})} M_{GK}(X \to Y) \ \textit{CQFD}
\end{aligned}$$

■

Proposition 13 *Soient* X *et* Y *deux motifs d'un contexte* $\mathcal{D}$.
(i) Si X *favorise* Y, *alors on a :*

$$M_{GK}(\overline{Y} \to X) = -\frac{P(X')}{P(\overline{X'})} M_{GK}(X \to Y) \tag{1.12}$$

(ii) Si X *défavorise* Y, *alors on a :*

$$M_{GK}(\overline{Y} \to X) = -\frac{P(Y')}{P(\overline{Y'})} M_{GK}(X \to Y) \tag{1.13}$$

Terminologie : Toute RA de type $Y \to \overline{X}$ est dite une règle négative à gauche (RA-ND)

Démonstration 9 *(i)* X *favorise* Y, *alors on a :*

$$\begin{aligned}
M_{GK}(\overline{Y} \to X) &= \frac{P_{\overline{Y'}}(X') - P(X')}{1 - P(X')} \\
&= \frac{P(X' \cap \overline{Y'}) - P(X')P(\overline{Y'})}{P(\overline{X'})P(\overline{Y'})} \\
&= -\frac{P(X')}{P(\overline{X'})} \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')} \\
&= -\frac{P(X')}{P(\overline{X'})} M_{GK}(X \to Y) \ \textit{CQFD}
\end{aligned}$$

■

Terminologie : Toute RA *de type* $\overline{Y} \to X$ *est dite une règle négative à gauche (*RA-NG*) tandisque* $Y \to X$ *est une RA*

(ii) X *défavorise* Y *donc :*

$$\begin{aligned}
M_{GK}(\overline{Y} \to X) &= \frac{P_{\overline{Y'}}(X') - P(X')}{P(X')} \\
&= \frac{P(X' \cap \overline{Y'}) - P(X')P(\overline{Y'})}{P(X')P(\overline{Y'})} \\
&= \frac{-P_{X'}(Y') + P(Y')}{P(\overline{Y'})} \\
&= -\frac{P(Y')}{P(\overline{Y'})} \frac{P_{X'}(Y') - P(Y')}{P(Y')} \\
&= -\frac{P(Y')}{P(\overline{Y'})} M_{GK}(X \to Y) \ \textit{CQFD}
\end{aligned}$$

■

Proposition 14 *Soient* X *et* Y *deux motifs d'un contexte* $\mathcal{D}$.
(i) Si X *favorise* Y, *alors on a :*

$$M_{GK}(\overline{Y} \to X) = \frac{P(X')}{P(\overline{X'})} M_{GK}(X \to \overline{Y}) \tag{1.14}$$

(ii) Si X *défavorise* Y, *alors on a :*

$$M_{GK}(\overline{Y} \to X) = \frac{P(Y')}{P(\overline{Y'})} M_{GK}(X \to \overline{Y}) \tag{1.15}$$

Démonstration 10 *(i)* X *favorise* Y, *on a :*

$$\begin{aligned} M_{GK}(\overline{Y} \to X) &= -\frac{P(X')}{P(\overline{X'})} M_{GK}(X \to Y) \\ &= \frac{P(X')}{P(\overline{X'})} M_{GK}(X \to \overline{Y}) \ \textit{CQFD} \end{aligned}$$

(ii) X *défavorise* Y *donc on a :*

$$\begin{aligned} M_{GK}(\overline{Y} \to X) &= -\frac{P(Y')}{P(\overline{Y'})} M_{GK}(X \to Y) \\ &= \frac{P(Y')}{P(\overline{Y'})} M_{GK}(X \to \overline{Y}) \ \textit{CQFD} \end{aligned}$$

■

Les liens entre les valeurs de M_{GK} pour les règles d'association qui se trouvent sur une chaîne du treillis de motifs sont données par la proposition suivante :

Proposition 15 ***Multiplicativité.***
(i) Si $X \subseteq Y \subseteq Z$, *alors on a :*

$$M_{GK}(X \to Z) = M_{GK}(X \to Y) M_{GK}(Y \to Z) \tag{1.16}$$

(ii) Si $X_1 \subseteq X_2 \subseteq \cdots \subseteq X_i \subseteq X_{i+1} \cdots \subseteq X_p$, *alors on a :*

$$M_{GK}(X_1 \to X_p) = \prod_{i=1}^{p-1} M_{GK}(X_i \to X_{i+1}) \tag{1.17}$$

Démonstration 11 *Il suffit de démontrer la relation (i), car (ii) est la généralisation de (i).*
En effet, les inclusions $X \subseteq Y \subseteq Z$ *implique* $Z' \subseteq Y' \subseteq X'$ *ce qui fait* $P_{X'}(Y') = \frac{P(X' \cap Y')}{P(X')} = \frac{P(Y')}{P(X')} \geq P(Y')$.
Par ailleurs, les trois motifs X, Y, Z *se favorisent deux à deux, on a :*

$$\begin{aligned} M_{GK}(X \to Y) &= \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')} \\ &= \frac{P(Y') - P(X')P(Y')}{P(X')(1 - P(Y'))} \\ &= \frac{P(Y')(1 - P(X'))}{P(X')(1 - P(Y'))} \end{aligned}$$

■

Par suite, on a :

$$\begin{aligned} M_{GK}(X \to Y) M_{GK}(Y \to Z) &= \frac{P(Y')(1-P(X'))}{P(X')(1-P(Y'))} \frac{P(Z')(1-P(Y'))}{P(Y')(1-P(Z'))} \\ &= \frac{P(Z')(1-P(X'))}{P(X')(1-P(Z'))} \\ &= \frac{P_{X'}(Z') - P(Z')}{1-P(Z')} \\ &= M_{GK}(X \to Z) \end{aligned}$$

■

Remarque 4 *Grâce à cette propriété multiplicative, l'utilisation de la mesure* M_{GK} *permet l'élaguer des branches de transitivité sur un diagramme de Hasse partiel ou dans un graphe sous-treillis a priori dense. Ce qui permet d'améliorer la lisibilité d'un graphe implicatif.*

Le corollaire suivant est une conséquence de la Proposition 11 ci-dessus.

Corollaire 5 *Soient* $X_1, X_2, \cdots, X_i, X_{i+1}, \cdots, X_p$ *des motifs tels que* $X_1 \subseteq X_2 \subseteq \cdots \subseteq X_i \subseteq X_{i+1} \cdots \subseteq X_p$.
(i) Si $X_1 \to X_p$ *est* (M_{GK}, α)*-valide alors* $\forall\, i, j \in \{1, \cdots, p\}$*, avec* $i < j$*,* $X_i \to X_j$ *est* (M_{GK}, α)*-valide.*
(ii) S'il existe $i, j \in \{1, \cdots, p\}$ *tels que* $X_i \to X_j$ *est non* (M_{GK}, α)*-valide alors* $\forall\, l, k \in \{1, \cdots, p\}$ *tels que* $l \leq i$ *et* $j \leq k$*, la règle* $X_l \to X_k$ *est non* (M_{GK}, α)*-valide.*

La proposition ci-dessous montre que la mesure M_{GK} permet de sélectionner moins de règles que la mesure Confiance pour un même seuil de validité.

Proposition 16 X *et* Y *deux motifs d'un contexte* $\mathcal{D}$*. On a :*

$$M_{GK}(X \to Y) \leq Conf(X \to Y)$$

Démonstration 12 *C'est essentiellement la composante favorisante qui va guider la sémantique de* M_{GK}.

$$\begin{aligned} M_{GK}(X \to Y) - Conf(X \to Y) &= \frac{P_{X'}(Y') - P(Y')}{1-P(Y')} - P_{X'}(Y') \\ &= \frac{P_{X'}(Y') - P(Y') - P_{X'}(Y') + P(Y')P_{X'}(Y')}{1-P(Y')} \\ &= \frac{P(Y')[P_{X'}(Y') - 1]}{1-P(Y')} \\ &\leq 0 \end{aligned}$$

■

Nous présentons les méthodologies de visualisation interactive des règles d'association, conçues pour faciliter la tâche de l'utilisateur confronté à de grands ensembles de règles. Elles sont fondées sur :

- des principes de visualisation d'information pour la construction de représentations visuelles efficaces (cf 2.8.1) ;
- des principes cognitifs de traitement.

A la sortie des algorithmes de fouille de données, les ensembles de règles d'association sont souvent de simples listes textuelles. Chaque règle consiste en un itemset qui constitue la prémisse, un itemset qui constitue la conclusion, et les valeurs numériques du support et de la mesure d'intérêt de règle (voir section 3). Les trois principales approches pour le post-traitement des règles d'association sont les suivantes :

1. évaluer, ordonner, et filtrer les règles avec des indices autres que le support et la confiance ;
2. organiser une exploration interactive des règles pour l'utilisateur ;
3. représenter les règles sous forme graphique.

1.3 Algorithmes d'extraction de règles d'association

Dans les traitements classiques de bases de données, on s'intéresse à des bases de données B composées d'un ensemble d'instances (ou enregistrements) et d'attributs en général continus. Dans le problème de recherche de règles d'association, on suppose la base de données discrétisée de manière à travailler sur une base D composée d'un ensemble fini d'instances et d'un ensemble fini d'attributs discrets (ou nominaux) appelés items. La recherche de règles d'association consiste à produire des règles de dépendance, des relations entre les items d'une base D, et ceci afin de prédire l'occurrence d'autres items. Chaque instance d'une base B associe une valeur à chacun des attributs continus (et est donc un ensemble de valeurs continues).

Dans l'ensemble des travaux existants, l'extraction de règles d'association est décomposée en deux sous-problèmes :

– la recherche des ensembles fréquents d'items (i.e. les itemsets fréquents),
– la génération des règles d'association à partir de ces itemsets fréquents.

1.3.1 Recherche des ensembles fréquents d'items.

En 1994, Agrawal & Srikant (1994a) ont proposé deux algorithmes pour calculer les itemsets fréquents. Ils ont été fondateurs dans le domaine de la fouille de données algorithmique. Les itemsets sont préalablement agencés selon un ordre lexicographique. Le premier, Apriori, consiste, en incrémentant successivement l'ordre k, dans un premier temps, à construire, à partir de cet ordre k, les itemsets d'ordre $k+1$, puis, dans un second temps, à ne retenir que les $(k+1)$-itemsets fréquents. Cela se fait en testant chaque $(k+1)$-itemset avec toutes les transactions de la base. Le second algorithme, AprioriTid, améliore les performances d'Apriori, car les $(k+1)$-itemsets ne sont plus testés avec toutes les transactions de la base, mais avec un sous-ensemble de celle-ci, qui diminue au fur et à mesure que des transactions ne sont plus concernées par les itemsets. Ce second algorithme ne nécessite de lire la base de données qu'une seule fois, mais il est nécessaire qu'elle tienne intégralement en mémoire.

La complexité d'Apriori est O($nm2m$), où n est le nombre de transactions, et m le nombre d'attributs. Elle montre que la durée d'exécution de l'algorithme croît linéairement avec le nombre de transactions, et exponentiellement avec le nombre d'attributs. Pour calculer les $(k+1)$-itemsets à partir des k-itemsets, les propriétés suivantes, issues de l'antimonotonicité du support sont utilisées :

– Si un itemset est fréquent, alors tous ses sous-sets le sont également ;
– Si un itemset n'est pas fréquent, alors ses supersets ne le sont pas également.

L'antimonotonicité du support vient de ce que sa relation d'ordre est inversée par rapport à l'inclusion des itemsets entre eux. En effet, le support d'un k-itemset est supérieur ou égal au support d'un $(k+1)$-itemset le contenant. Ce $(k+1)$-itemset est appelé superset du k-itemset, qui est un de ses sous-sets. La littérature parle également d'itemset plus général au sujet d'un superset, et d'itemset plus spécifique pour un sous-set. Ainsi, alors que la valeur de k augmente, le support des itemsets décroît.
Pour construire un $(k+1)$-itemset, une combinaison est réalisée entre deux k-itemsets qui ne diffèrent que d'un seul élément. Par exemple, le 2-itemset $\{a,b\}$ est la combinaison des 1-itemsets $\{a\}$ et $\{b\}$. Le 4-itemset $\{a,b,c,d\}$ est la combinaison des 3-itemsets $\{a,b,c\}$ et $\{a,b,d\}$, mais également des 3-itemsets $\{a,b,d\}$ et $\{b,c,d\}$, ou, $\{a,b,d\}$ et $\{a,c,d\}$, etc. Cette méthode de construction assure, grâce à l'antimonotonicité, que tous les k-itemsets fréquents seront extraits étant donné un support minimum et éventuellement une valeur maximale de k.

1.3.2 Caractéristiques des itemsets.

En fonction de leur support, les itemsets sont dotés de caractéristiques :

– Un itemset est dit *clos* ou *fermé* (Pasquier et al., 1999a) si aucun de ses supersets n'a de support identique au sien. Cela signifie que tous ses supersets ont un support inférieur ;
– Un itemset est dit *maximal* si aucun de ses supersets n'est fréquent ;
– Un itemset est dit *générateur* si tous ses sous-sets ont un support supérieur.

À partir de ces définitions Bayardo et al. (1999) a introduit la notion de bordure (Border) pour caractériser la séparation entre les itemsets fréquents et les itemsets non fréquents :

– Une bordure positive est l'ensemble des itemsets fréquents maximaux, c'est-à-dire qu'ils n'ont pas de supersets fréquents ;
– Une bordure négative est l'ensemble des itemsets non fréquents, dont les sous-sets de premier ordre inférieur sont des itemsets fréquents.

La bordure sépare donc l'ensemble des itemsets en deux sous-ensembles, celui des itemsets fréquents, et celui des itemsets non fréquents. Elle est exploitée par des algorithmes de recherche de règles d'association.

1.3.3 Quelques algorithmes d'amélioration de la recherche d'itemsets fréquents.

Plusieurs algorithmes proposent d'améliorer la recherche d'itemsets fréquents. Ils répondent à des besoins de performance, ou de rapidité d'exécution, et prennent également en compte, par exemple, la possibilité de pouvoir charger intégralement la base en mémoire ou non, ou la possibilité de traiter de très longs itemsets, comme c'est le cas dans le domaine de la biologie. Nous en donnons une courte liste qui est loin d'être exhaustive.

Le FP-Growth (Han et al., 2000) Jiawei Han s'appuie sur une structure de données compactes appelée FP-tree (Frequent Pattern Tree) créée à partir des itemsets fréquents et des transactions. Le FP-Tree est constitué d'une racine nulle, à partir de laquelle sont structurées les transactions

sous la forme d'un arbre dont les noeuds correspondent aux 1-itemsets. Par ailleurs, un tableau indexe ces derniers en les triant par leur support, à partir desquels sont reliés, sous la forme d'une liste chaînée, les noeuds correspondant à chaque instance de l'item. Le parcours de l'arborescence permet ensuite de trouver les itemsets fréquents.

La Partition. Proposé par Savasere et al(1995), la base étant divisée en partitions pouvant tenir en mémoire, dans une première phase, les itemsets fréquents sont calculés dans chacune d'elles. A l'issue, ils sont fusionnés pour élaborer des supersets potentiellement fréquents. Dans une seconde phase, les supports de ces itemsets sont calculés.
Ce calcul est effectué de manière simple en utilisant les tidlists. Une tidlist est un ensemble constitué d'un itemset et de l'ensemble des identifiants uniques des transactions qui contiennent cet itemset. Pour calculer le support de la réunion de deux itemsets, il suffit de considérer l'intersection des ensembles de transactions contenus dans leurs tidlists respectives. Le cardinal de cette intersection est alors le support de cette réunion.

L'Eclat. Mohammed J. Zaki utilise également les tidlists. La recherche des itemsets fréquents est réalisée en profondeur et s'arrête dès qu'il n'y a plus de k-itemsets satisfaisant le support minimum. Cette méthode est très rapide, mais elle nécessite une capacité mémoire qui peut être grande, en fonction du nombre de transactions (Zaki et al., 1997).

L'Echantillonnage de la base (Sampling). Toivonen (1996) propose de considérer un échantillon aléatoire de la base, afin d'en extraire un sous-ensemble d'itemsets fréquents et sa bordure négative. Pour cela, il prend en compte un support inférieur au support minimum défini par l'utilisateur. Le support des itemsets du sous-ensemble est ensuite calculé, ainsi que celui des itemsets de la bordure. Si la bordure ne contient pas d'itemsets fréquents, alors l'ensemble des itemsets fréquents est correct et complet. Sinon, il faut reprendre le traitement depuis le début, afin d'obtenir tous les itemsets fréquents.

Le DIC (Dynamic Itemset Counting). Brin et al. (1997b) avaient pour objectif de diminuer le nombre de passes dans la base de donnée, par comparaison avec l'Apriori qui nécessite de balayer la base pour chaque ordre d'itemsets. Pour cela, elle est divisée en plusieurs parties séparées par un point de contrôle. Durant chaque passe, à chaque point de contrôle, les $k-$itemsets dont le support est supérieur au support minimum, sont utilisés pour construire les $(k+1)-$itemsets.
Le support de ces derniers commence alors à être calculé durant cette même phase. Ainsi, il n'est plus nécessaire de parcourir autant de fois la base que dans l'Apriori. Cet algorithme demande une capacité mémoire suffisante pour être capable de traiter des itemsets de différents ordres simultanément.

Le Close et l'A-Close. Nicolas Pasquier montrent que les meilleures règles d'association sont extraites à partir des itemsets présents sur la bordure. Pour l'obtenir, Pasquier et al. ont proposé les algorithmes Close (Pasquier et al., 1999b) puis A-Close (Pasquier et al., 1999b) qui permettent de trouver, de manière itérative, les itemsets fréquents clos. Pour cela, ils s'appuient sur leurs itemsets générateurs. À partir de cet ensemble, il est alors possible de déduire tous les itemsets fréquents.

Le MaxMiner. De la même manière que l'algorithme précédent, MaxMiner cherche les itemsets clos fréquents et s'intéresse aux itemsets maximaux (Bayardo, 1998). À partir de cet ensemble, il est aisé de déduire tous les itemsets fréquents. Durant son exécution, le MaxMiner cherche à trouver le plus tôt possible des longs itemsets fréquents pour réduire ensuite son champ de recherche. En effet, quand un long itemset maximum est découvert, alors il est inutile de poursuivre le processus avec ses sous-items, car, d'après le principe d'antimonotonicité, tous ses sous-items sont fréquents. D'autres algorithmes exploitant l'ensemble des itemsets maximaux ont été étudiés comme, MaxEclat (Zaki et al., 1997) ou MaxClique (Zaki et al., 1997).

Le DHP (Direct Hashing and Pruning). Proposé par Park et al. (1995), DHP est une légère modification de l'Apriori qui utilise des tables de hachage pour diminuer le nombre d'itemsets candidats générés. Cet algorithme propose également une diminution de la taille de la base de données, au fur et à mesure des itérations.
Lors des deux premières itérations (gestion des 1−itemsets et des 2−itemsets), on utilise une table de hachage sur les numéros des 2−itemsets. A chacune des cases de la table de hachage, on associe un compteur. À chaque fois qu'une instance contient l'un des itemsets associés (par la fonction de hachage) à une case de la table de hachage, on incrémente le compteur de cette case. On garde les itemsets associés aux cases dont le compteur est suffisant.

À partir de la deuxième itération, on effectue un pruning des instances de la base. Pour chaque instance, si le nombre de bits de cette instance est supérieur à k, l'instance peut être utile à la keme itération, sinon, on la supprime.

L'AprioriTid. Proposé par Agrawal & Srikant (1994b). Cet algorithme utilise des listes d'itemsets associées aux identifiants des objets en relation avec les itemsets, afin de diminuer progressivement la taille du jeu de données. Il permet de diminuer progressivement la taille de la base, dans le but de la stocker en mémoire et de ne plus réaliser d'opérations d'entrée-sortie, après le premier balayage.

À partir de la deuxième itération, la procédure AprioriGen est utilisée pour générer les itemsets candidats de l'itération suivante dans un ensemble $\bar{C}_k$. Chaque élément de l'ensemble $\bar{C}_k$ est un couple (ID, c_k) dans lequel c_k est la liste des k-itemsets candidats contenus dans l'instance d'identifiant ID. Le support d'un itemset candidat c_k correspond au nombre d'apparitions de c_k dans $\bar{C}_k$.

L'AprioriHybrid. Proposé par Agrawal & Srikant (1994b). AprioriHybrid constitue un mélange d Aprioris et AprioriTid en fonction de l'itération à laquelle on se trouve.

Le Sampling. Proposé par Toivonen (1996). Cet algorithme effectue la génération des itemsets sur un échantillon de la base. Leurs supports sont ensuite calculés sur la base entière. Au fur et à mesure du parcours de la base, on affine les résultats (la valeur des supports). On supprime des itemsets qui se révèlent non fréquents. Cet algorithme suppose de disposer d'un échantillon représentatif de la base !

Le SSDM. Avec un autre type d'approche, Escovar et al. (2005) a proposé le Semantically Similar Data Mining Algorithm (SSDM) qui utilise de la logique floue pour extraire des itemsets fréquents. Pour ce faire, l'utilisateur doit préalablement remplir une table de similarité indiquant les items faisant partie de mêmes catégories. Par ailleurs, en plus du support, il est nécessaire de fixer un minimum de similarité qui permet à l'algorithme de choisir si deux items sont similaires ou non.

1.4 Conclusion partielle.

Il semble que l'utilisation de la mesure de qualité M_{GK} permette de s'affranchir des problèmes de méthodes d'extraction des règles d'association à l'aide de la mesure de qualité Confiance. Il convient également de noter que les études des propriétés mathématiques de la mesure M_{GK} montrent qu'elle est plus sélective que la mesure de la qualité de confiance qui n'est autre que la probabilité conditionnelle.

Des études sur l'algorithme d'exploration de règles ont été trouvées dans la littérature. Nous avons mis en évidence une nouvelle lumière sur l'ensemble d'algorithmes du motif. En effet, nous avons utilisé l'algorithme à priori que nous utilisons pour extraire l'ensemble des motifs fréquents.

Nous avons ensuite proposé des algorithmes d'extraction valables pour des règles d'association au sens de la mesure de qualité M_{GK} sur la base d'un seuil critique.

Dans le chapitre chap. 2, nous utiliserons ces théories pour définir et concevoir une application que nous appellerons ASI – MGK. Logiciel d'analyse statistique qui met en œuvre toutes les propriétés mathématiques de la mesure M_{GK}.

Chapitre 2

Contribution : conception de l'ASI – MGK.

Sommaire

Après avoir présenté les théories sur les règles d'association dans le chapitre précédent, ce chapitre continuera sur sa mise en œuvre, il s'agit de la conception d'un outil d'analyse statistique basé sur une mesure d'intérêt de la règle d'association M_{GK}.

2.1 Développement de L'ASI – MGK.

2.1.1 Analyse statistique implicative (ASI).

L'analyse statistique implicative (ASI.) est une méthode non symétrique d'analyse de données croisant des sujets ou des objets avec des variables de tous types : booléen, numérique, modal, vectoriel, séquentiel, intervalles.

Fondée dans les années 1980 par Régis Gras, professeur émérite à l'université de Nantes, équipe DUKE du Laboratoire d'informatique de Nantes Atlantique, elle continue à se développer sous l'impulsion de cet éminent chercheur et celle de chercheurs de son équipe, Jean-Claude Régnier entre autres, professeur à l'UMR 5191 ICAR, Université Lumière Lyon-II.

Visant l'extraction des règles d'association implicatives, la méthodologie consiste à attribuer une valeur numérique, une mesure entre 0 et 1, à des règles R du type ; « *si la variable a est observée, alors généralement la variable b l'est aussi* » ou dans les cas plus généraux : « *si la variable a prend une certaine valeur alors la variable b prend généralement une valeur supérieure ; recherche en didactique de discipline, en psychplogie, sociologie, diagnostic médical etc* » (Gras, 2015).

Actuellement, on dispose de deux approches ASI remarquables : l'ASI selon l'implication statistique de Gras et l'ASI basée sur M_{GK} dite ASI – MGK. À travers la structuration de l'ensemble des règles, l'utilisateur peut accéder à des relations de type causal et prédictif pondérées par ces indices. « *L'ASI n'est pas une méthode mais un cadre théorique, large, dans lequel se traitent des problèmes modernes de l'extraction des connaissances à partir des données. C'est une théorie générale dans le domaine de la causalité parce qu'elle répond à des faiblesses d'autres théories, elle apporte un outillage formel et des méthodes pratiques de résolution de problèmes. Ses applications sont multiples...* » (Gras, 2014).

Deux logiciels existent, l'un appelé CHIC (acronyme de Hiérarchical Implicative and Cohesive Classification), il est développé par Raphaël Couturier, professeur d'université à l'Université de Franche-Comté et l'autre appelé ASI – MGK, développé par notre équipe de recherche à l'Université d'Antsiranana. Ils sont évolués selon les extensions de la théorie qui sont fonctionnelles pour traiter tous les problèmes numériques et graphiques nécessaires à l'utilisation de l'ASI.

Grâce à ces outils informatiques, de nombreuses applications dans différents domaines (*pédagogie, psychologie, sociologie, bioinformatique, agro-business, histoire de l'art, etc.*) ont montré la pertinence du modèle probabiliste choisi et, par conséquent, la richesse des informations obtenues par le traitement des variables. Ils ont notamment permis de mettre en évidence des pépites de connaissances que d'autres méthodes n'ont pas extraites et d'exprimer des relations causales affectées par une intensité.

2.1.2 Spécification de l'ASI – MGK.

Il y a plusieurs manières de concevoir les logiciels, dans notre cas nous proposons un logiciel qui permet de mener des travaux dirigés dans le cadre de l'enseignement du *data mining* :

a Nous nous attachons au fond et non pas à la forme, dans ce cas qu'importe l'interface du logiciel et son mode opératoire, l'essentiel est de pouvoir mettre en œuvre les méthodes de fouille de données.

b Il faut néanmoins que l'interface utilisateur soit simple pour que les étudiants puissent se familiariser avec les standards.

c Il est nécessaire également que l'utilisation du logiciel ne requiert pas des compétences particulières lors de sa mise en oeuvre. Cela concerne surtout la préparation des données (possibilité d'importer simplement des fichiers en provenance de tableurs par exemple) et la définition des traitements (il ne doit pas être nécessaire d'apprendre un langage de script particulier pour définir des séries de traitements).

A partir de ces éléments, nous avons défini un cahier des charges du logiciel que nous voulons construire.

1. Installation simplifiée du logiciel : l'installation du logiciel sur la machine doit pouvoir se faire simplement, il ne doit surtout pas nécessiter l'installation de serveurs lourds. Le logiciel fonctionne en stand-alone.
2. Gestion simplifiée des données : qu'importe si les logiciels adoptent leur propre format, l'essentiel est que l'on puisse importer aisément des données au format texte issu de tableurs.
3. Définition des opérations par diagramme de traitements : dans le standard actuel, on parle de « *filière* » ou de « *chaîne de traitements* », dans notre logiciel l'idée est la même, définir l'enchaînement des opérations sous forme d'un graphe sans avoir à écrire une seule ligne de code.
4. Évaluation des méthodes d'apprentissage : concernant surtout l'apprentissage supervisé, le logiciel doit proposer les outils standards d'évaluation des performances (estimation de l'erreur en resubstitution, en test ou par ré-échantillonnage).
5. Affichage des résultats, aide à l'interprétation et exportation des résultats : les résultats doivent être lisibles simplement avec une mise en forme correcte. Il doit être possible de les exporter et de les reprendre dans les outils usuels d'édition (traitement de texte, diaporama, etc.)

2.1.3 Buts généraux.

Nous adoptons les quatre buts généraux suivants répérés en Génie logiciel :

Modifiabilité : Il existe deux raisons de vouloir modifier un logiciel :
- changement des spécifications (par exemple d'après une suggestion d'utilisateur)
- correction des erreurs.

ASI – MGK devrait être modifiable pour qu'on puisse introduire des changements sans complexification (ni perte de lisibilité).

Efficacité : Utilisation optimale des ressources en temps et en espace pour des systèmes en temps réel : Dans ce cas la ressource en temps est prédominante. Il faut se préoccuper d'avantage de l'efficacité, se polarisant de ce fait sur la micro-efficacité au détriment de la macro-efficacité ; dans Ross et al. (1980) : « *Une bonne perspicacité reflétant une compréhension plus unifiée d'un problème a beaucoup plus d'impact sur l'efficacité que n'importe quel tripotage de bits dans une structure déficiente* ».

Fiabilité : Peut-être critique : « *La fiabilité doit à la fois éviter les défauts de conception, d'étude et de construction, et permettre de récupérer les pannes et les défauts de performances* » (cf. Ross et al. (1980)). Pour tout bug ou dysfonctionnement, prévisible ou non, ASI-M_{GK} devrait entrer en mode dégradé en douceur, sans effet de bord dangereux.

Intelligibilité : Sans doute le but le plus crucial pour bien gérer la complexité d'un système logiciel pour qu'un système soit compréhensible, il doit être un modèle exact de notre vue du monde réel pour améliorer la compréhensibilité :

- Au bas niveau, lisibilité par un bon style de codage,
- Au plus haut niveau, il est facile d'isoler les structures de données (objets) et les actions (opérations) correspondantes à celles du monde réel.

2.1.4 Stratégie de développement.

Lorsque l'on conçoit un logiciel complexe, il est impératif de le décomposer en parties de plus en plus petites, chacune d'elles pouvant être ensuite affinée indépendamment. Dans la décomposition algorithmique, on réalise une analyse structurée descendante où chaque module du système est une étape majeure de quelque processus général. C'est pour cette raison que nous avons choisi la stratégie de développement orientée objet. La programmation orientée objets est une méthode mise en œuvre dans laquelle les programmes sont organisés comme des ensembles d'objets coopérants. Chacun représente une instance d'une certaine classe, toutes les classes étant des membres d'une hiérarchie de classes unifiées par des relations d'héritage. On désigne donc par langage orienté objets un langage répondant aux conditions suivantes. Un langage orienté objets est tel que :

- Il supporte des objets qui sont des abstractions de données avec une interface d'opérations nommées et un état interne caché.
- Les objets ont un type associé (la classe).
- Les types peuvent hériter d'attributs venant de super-types (les super-classes).

2.1.5 Modularité et localisation.

D'après le psychologue, Georges Miller (1954), l'être humain ne peut gérer plus de 7 ± 2 entités à la fois au sein d'un même niveau.
Un objet du monde réel devient un objet informatique ; abstraction et dissimulation d'informations sont la base de tout développement orienté objet. Les spécifications et leur passage en objets est un cycle en quasi-perpétuelle évolution. Il faut donc un schéma cyclique où l'on peut modifier seulement sur une partie sans toucher au reste. On opérera donc comme suit :

1. Identifier les objets et leurs attributs. Les objets découlent des groupes nominaux utilisés pour les décrire (exemple : une pile d'assiette).
2. Identifier les opérations. Ce sont les verbes que l'on utilise pour décrire les actions possibles sur l'objet.
3. Établir la visibilité. Par exemple un objet pourra avoir accès à toutes les actions d'une pile, mais ne pourra pas voir les fonctions d'allocation dont la pile se sert
4. Établir l'interface. Description de chaque opération avec ses arguments
5. Implanter chaque objet.

2.2 Présentation de l'interface d'essai ASI – MGK.

Comme son nom l'indique, il s'agit d'un logiciel d'analyse statistique (fig 2.1), basé sur la mesure de la qualité des règles d'associations M_{GK} développées par le groupe de recherche ASI-ist du centre de recherche *Education et Didactique des Mathématiques et de l'Informatique* (EDMI), CIRD-ENEST Antsiranana. Elle est écrite en langage de programmation Java, et peut fonctionner indépendamment, sans installation. On peut facilement importer des données au format texte ou à partir de fichier MS Excel au format CSV. Elle prend en charge plusieurs fenêtres pour afficher les résultats.

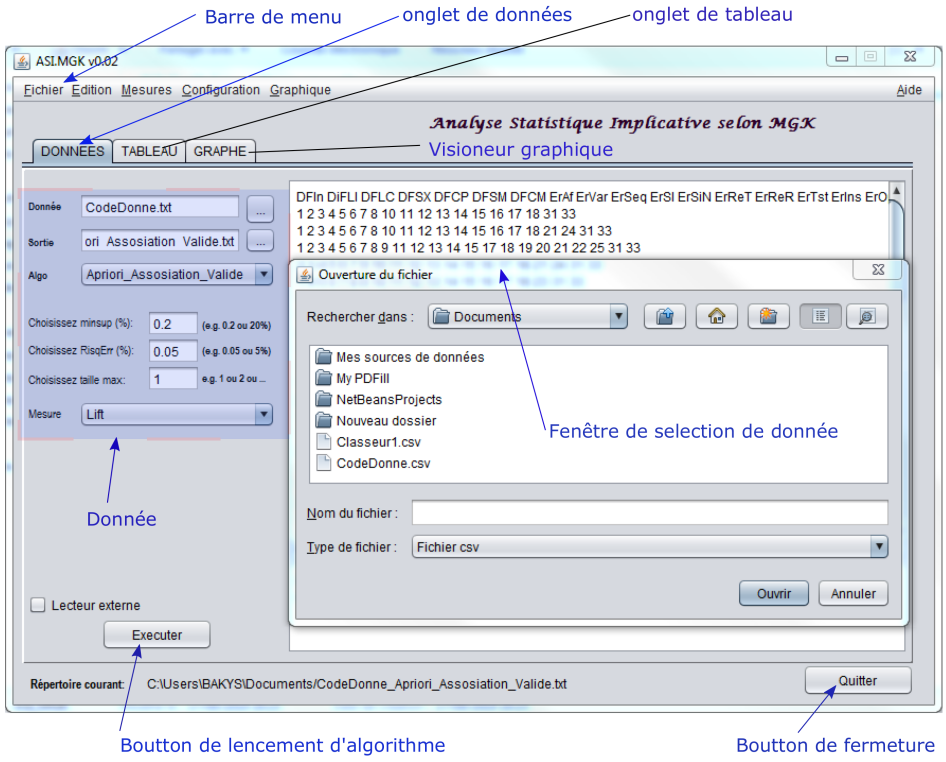


FIGURE 2.1 – Présentation de la fenêtre ASI-MGK.

La fenêtre de dialogue de message affiche toutes les informations sur les données et le traitement effectué, où nous trouvons la taille des données, le nombre de transactions et le nombre

d'individus, les détails du traitement (temps, capacité de stockage allouée, utilisation des paramètres, statistiques des résultats). La fenêtre d'affichage des données en mode texte affiche les données en mode séquence de données. La fenêtre de visualisation de données en mode tableau ; ce mode d'affiche de données permet des interactions dynamiques, triage ordonné selon les motifs, et prend également en charge une fonction d'exportation vers un tableau au format LaTeX.

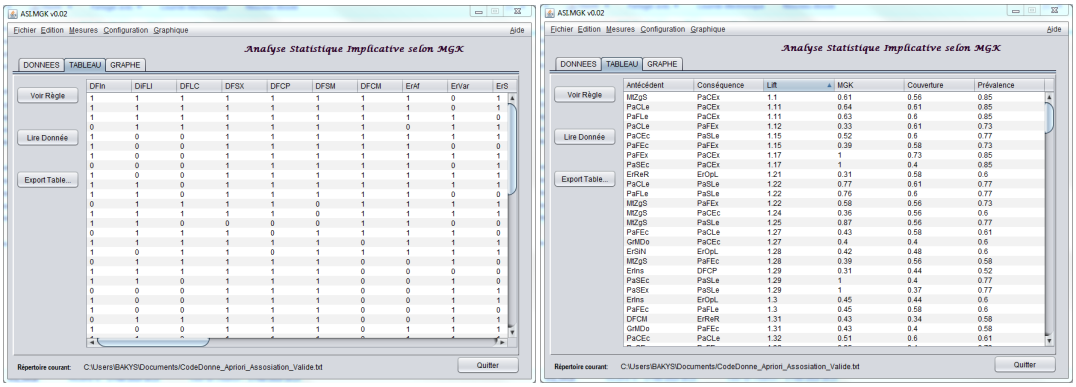


FIGURE 2.2 – Fenêtres de visualisation de ASI-MGK en Tableau : à gauche données brutes et à droite liste des régles valides.

La fenêtre d'affichage des résultats comprend trois modes de représentation :

- Représentation en mode texte ; où les résultats sont transcrits au format texte en utilisant le mode de représentation des données csv, dont les colonnes sont séparées par une tabulation ;
- Représentation en mode tableau dynamique, prenant en charge la fonction d'exportation de tableau au format LaTeX ;
- Représentation en mode graphique dans une fenêtre interne ou externe ; dans lequel les résultats sont illustrés sous forme d'un graphe implicatif, elle prend en charge tous les outils d'améliorations de la vue. Nous allons présenter ces modes d'affichage dans la section 2.8.

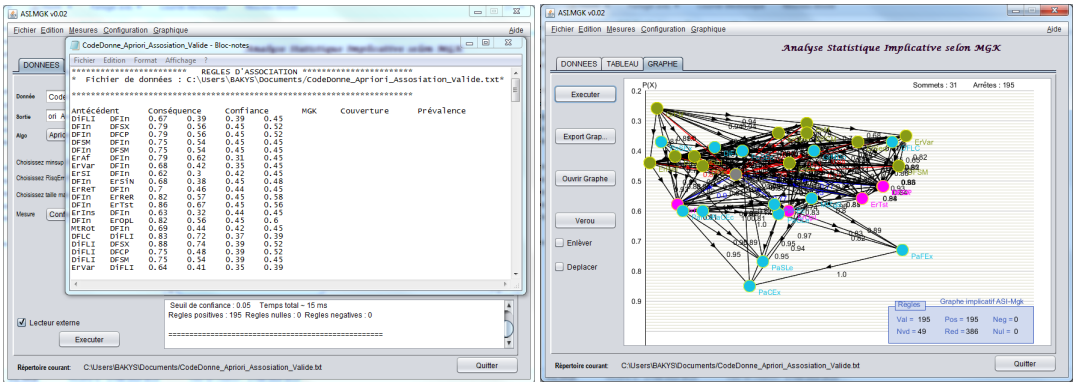


FIGURE 2.3 – Fenêtres de visualisation de ASI-MGK : A gauche en fichier text et à droite en graphe implicatif.

ASI – MGK implémente les démarches standard de traitement des données dans ECD, notamment : le prétraitement (cf. Section 2.3), la découverte fréquente d'éléments (cf. Section 2.4), l'extraction efficace des règles d'association (cf. Section 2.7) et la présentation et l'analyse des résultats (cf. Section 2.8).

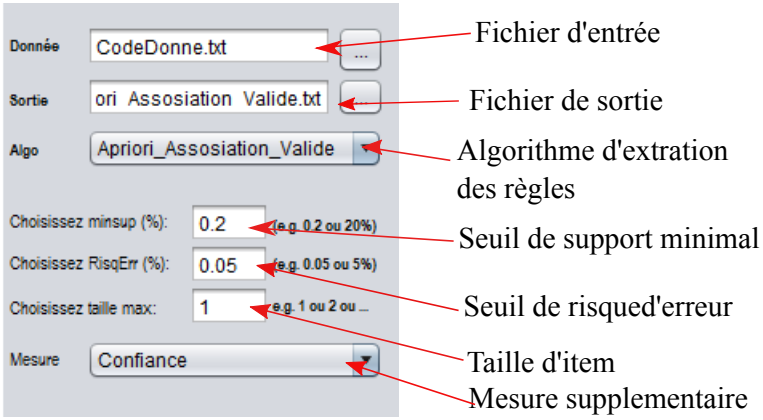


FIGURE 2.4 – Fenêtre de l'entrée des données de l'ASI-MGK.

L'outil "CSV to TRANS" parcouru depuis le menu Fichier de la barre de menu sera utilisé pour demander la matrice booléenne traitée. L'algorithme utilisé sera fourni dans la liste d'option. Des paramètres tels que le seuil support minimal (*minsup*), seuil de risque d'erreur (α) et taille maximal de l'itemset doivent être entrés via les zones de saisies comme le montre la figure 2.4.

2.3 Pré-traitement.

Un prétraitement est nécessaire de manière à mettre en forme la structure des données brutes, les codages, le formatage de données et plus généralement pour représenter les données en une autre forme. Cette nouvelle forme facilitera tous les traitements ultérieurs.

2.3.1 Attribute-Relation File Format (ARFF)

ARFF signifie Attribut Relation File Format. C'est un fichier texte ASCII qui décrit une liste d'instances partageant un ensemble d'attributs. Les fichiers ARFF ont été développés par le *Machine Learning Project* du Département d'informatique de l'Université de Waikato (Nouvelle Zelande) pour être utilisés avec le logiciel d'apprentissage automatique de Weka. Il s'agit d'une extension du format ARFF décrit dans le manuel d'exploration de données écrit par Ian H. Witten et Eibe Frank Witten & Frank (2002)(*les nouveaux ajouts sont des attributs de chaîne, des attributs de date et des instances éparses*). Les fichiers ARFF comportent deux sections distinctes. La première section contient les informations d'en-tête, suivies des informations de données. L'en-tête du fichier ARFF contient le nom de la relation, une liste des attributs (*les colonnes dans les données*) et leurs types. Un exemple d'en-tête sur notre jeu de données ressemble à ceci.

```
\% 1. Title: Donnée ditactique \%
\% 2. Sources:
\% (a) Creator: R. B. Bakys

\@ ATTRIBUTE NOM NUMERIC
\@ ATTRIBUTE CHN NUMERIC
\@ ATTRIBUTE LIST NUMERIC
\@ ATTRIBUTE SIN class {0, 1}
\@ ATTRIBUTE POU class {0, 1}
```

```
\@DATA
1, 0, 0, 1, 1, 1, 1, 0, 0, 0
1, 0, 0, 1, 0, 0, 1, 0, 0, 0
1, 1, 1, 0, 1, 0, 1, 0, 0, 0
1, 0, 1, 0, 1, 0, 1, 0, 0, 0
```

Les lignes commençant par un \% sont des commentaires. Les déclarations \@ RELATION, \@ ATTRIBUTE et \@ DATA sont sensibles à la casse.

2.3.2 Fichier Comma-separated values (CSV)

Comma-separated values, connu sous le sigle CSV, est un format texte ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules. Un fichier CSV est un fichier texte, par opposition aux formats dits « *binaires* ». Chaque ligne du texte correspond à une ligne du tableau et les virgules correspondent aux séparations entre les colonnes. Les portions de texte séparées par une virgule correspondent ainsi aux contenus des cellules du tableau. Une ligne est une suite ordonnée de caractères terminée par un caractère de fin de ligne (*line break LF ou CRLF*), la dernière ligne pouvant en être exemptée.

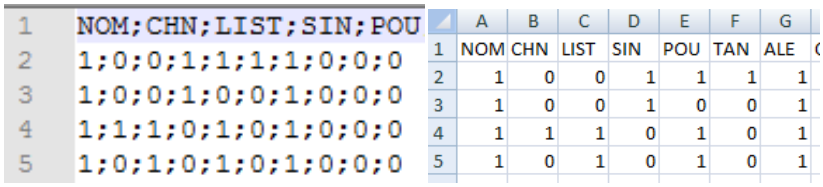


FIGURE 2.5 – Présentation des données au format CSV.

2.4 Algorithme Apriori.

L'algorithme Apriori procède de manière itérative pour identifier les $k-$itemsets fréquents, c'est-à-dire que l'ensemble des itemsets fréquents est parcouru par niveau (parcours en largeur du haut vers le bas du treillis d'itemsets). Lors de la $k$eme itération, on recherche les $k-$itemsets fréquents, c'est-àă dire les itemsets de longueur k qui sont fréquents. (Agrawal & Srikant, 1994a)

Remarque 5 *L'opération consistant à ne pas introduire les $k-$itemsets infréquents dans F_k, et donc à ne pas les prendre en compte pour la suite du traitement, est appelée pruning ou élagage.*

Propriété 3 *Tous les sous-ensembles d'un itemset fréquent sont fréquents.*

La propriété 3 permet une limitation du nombre de candidats générés lors de la keme itération par une jointure conditionnelle des itemsets fréquents, de taille $k-1$, découverts précédemment.

Propriété 4 *Tous les sur-ensembles d'un itemset infréquent sont infréquents.*

La propriété 4 permet la suppression d'un candidat de taille k lorsqu'au moins un de ses sous-ensembles de taille $k-1$ ne fait pas partie des itemsets fréquents découverts précédemment.

2.5 Notre apport au choix de régle d'association.

2.5.1 Etude de la validité des règles.

Notons $r : X \rightarrow Y$ une règle d'association positive dans une base de données $\mathcal{B}$. Une mesure d'intérêt est une fonction qui associe à une règle d'association un nombre réel caractérisant l'intérêt que l'on peut porter à cette règle. La figure 2.6 présente une telle table de contingence, dans laquelle nous notons p_x la fréquence du motif X.

	X	$\overline{X}$	
Y	n_{xy}	$n_{\overline{x}y}$	n_y
$\overline{Y}$	$n_{X\overline{y}}$	$n_{\overline{xy}}$	$n_{\overline{y}}$
	n_x	$n_{\overline{x}}$	n

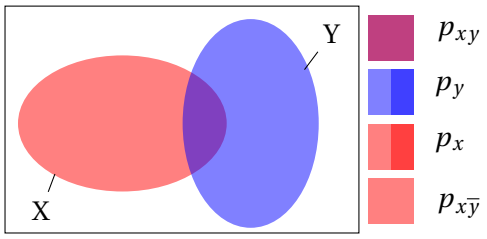


FIGURE 2.6 – Tableau de contingence de $r : X \rightarrow Y$.

Le tableau de contingence (fig 2.6) possédant trois degrés de liberté, une fois ces trois degrés choisis, il est possible de considérer les mesures comme des fonctions de $\mathbb{R}^3$ dans $\mathbb{R}$ et d'y appliquer tous les outils de l'analyse. Pour pouvoir étudier les mesures comme de simples fonctions de trois variables, il est nécessaire de bien définir le domaine de définition. Ce domaine dépend de la paramétrisation choisie : à l'aide des exemples, des contre-exemples, ou encore de la confiance. Guillaume (2000); Lenca et al. (2008) ont souligné l'importance du nombre de contre-exemples dans l'évaluation de l'intérêt des règles. Par conséquent, nous nous intéresserons ici au comportement des mesures vis-à-vis de la variation des contre-exemples des règles, c'est-à-dire qu'une règle d'association $r : X \rightarrow Y$ peut être caractérisée par les trois quantités $(n_{X\overline{Y}}, n_X, n_Y)$. Les mesures d'intérêt sont alors des fonctions d'un sous-domaine D du cube unité de $\mathbb{R}^3$:

$$D = \left\{ (a,b,c) \left| \begin{array}{rcccr} 0 & < & b & < & 1 \\ 0 & < & c & < & 1 \\ max(0,b-c) & < & a & < & min(b,1-c) \end{array} \right. \right\}$$

où a (resp. b, c) représente $p_{x\overline{y}}$ (resp. p_x, p_y). Lorsque l'on projette une règle dans $\mathbb{R}^3$, elle peut être étudiée comme un vecteur et on a alors la possibilité d'étudier un voisinage de la règle et d'observer le comportement d'une mesure sur ce voisinage.

2.5.2 Réalisation et simulation sous Scilab.

La commande `mgkrand(n, alpha)` simule `Mgk(X=>Y)` des vecteurs aléatoires binaires X et Y de taille n au seuil de confiance α. On a programmé sous Scilab [1]. Sorties : M_{GK}, M_{GKcr}, règle, supports, confiance, vecteur binaire aléatoire X et Y. On part d'une taille $n = 10$, et on étudie le

1. Scilab est un logiciel libre de calcul numérique multi-plateforme fournissant un environnement de calcul pour des applications scientifiques. Il possède un langage de programmation orienté calcul numérique de haut niveau. Il peut être utilisé pour le traitement du signal, l'analyse statistique, le traitement d'images, les simulations de dynamique des fluides, l'optimisation numérique, et la modélisation et simulation de systèmes dynamiques explicites et implicite

comportement de la confiance et M_{GK} face à la variation de nombre d'exemples et de contre-exemples.

```
function m = mgkcr(n,nX,nY,s,p)
  // mgkcr = valeur critique de mgk
  //n = effectif total nX = card(X)
  //nY = card(Y)
  //s = seuil de confience
  //p = 1 valeur critique favorable 0 defavorable
  r =(1-s);
  if p==1 then
    m = sqrt(1/n*((n-nX)/nX)*(nY/(n-nY))*cdfchi("X",1,r,s));
  else
    m = -sqrt(1/n*((n-nX)/nX)*((n-nY)/nY)*cdfchi("X",1,r,s));
  end
endfunction
```

```
function [ConfXY,SupXY,SupY,SupX,Regle,ValCr,MGkXY]= mgkrand(n,s)
  //sumulation de mgk de
  X =[]; Y =[];
  for i=1:n
          x = rand(1);
          y = rand(1);
          if x<0.5 then
              X(i)=0;
          else X(i)=1;
          end
          if y<0.5 then
              Y(i)= 0;
          else Y(i)= 1;
          end
  end
  nx = sum(X);   ny = sum(Y);   nxy = sum(X.*Y);
  SupX = nx/n;   SupY = ny/n;   SupXY = nxy/n;
  if SupX >0 & SupY >0 then
          ConfXY = SupXY/SupX;
          if ConfXY > SupY  then //m ="Positive";
             MGkXY =(ConfXY-SupY)/(1-SupY);
                ValCr = mgkcr(n,nx,ny,s,1);
             if MGkXY > ValCr tèen
                Regle ='X renforce Y : Règle valide ';
             else Regle ='Règle non valide';
             end
          elseif ConfXY==SupY
             MGkXY = 0;
             Regle = "X indépendant de Y"
             ValCr ¡[];
          else
```

```
            Regle ="Negative";
            MGkXY =(ConfXY-SupY)/SupY;
            ValCr = mgkcr(n,nx,ny,s,2);
            if MGkXY<ValCr then
               Regle ='X Repousse Y : Règle valide ';
            else Regle ='Règle non valide';
            end
         end
  else
       ConfXY =[];Regle = "Pas de regle";ValCr =[];MGkXY =[];
  end
  X=X'; Y=Y';
endfunction
```

Après avoir exécuté le programme d'ajout dans la console du logiciel scilab, comme indiqué dans la figure 2.7, nous obtenons les résultats ci-après ; Le premier tableau montre les modèles générés, le second tableau montre le tableau de contingente correspondant aux modèles et le dernier tableau montre les valeurs M_{GK}, M_{GKcr}, $Conf$ et leurs supports.

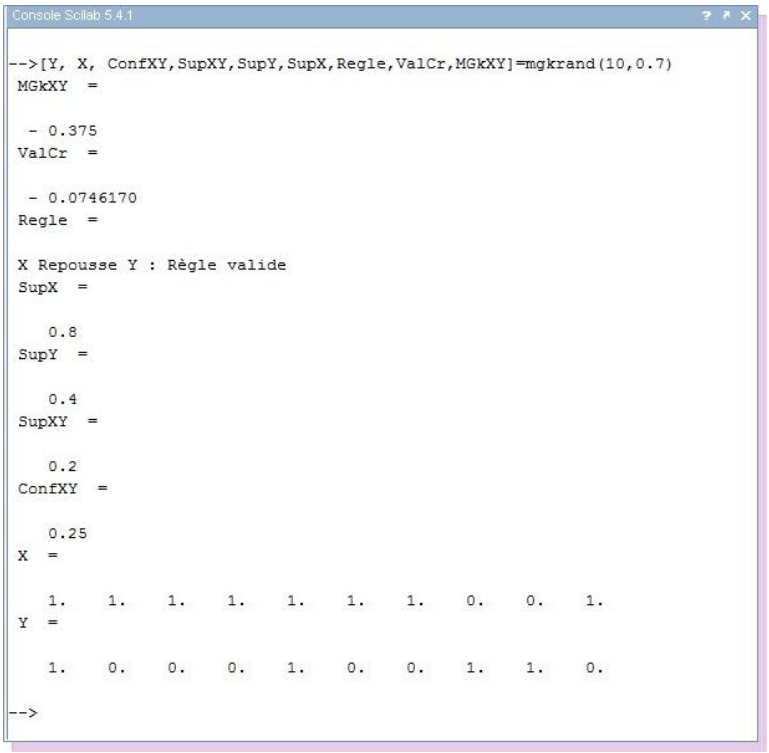


FIGURE 2.7 – Console Scilab.

Confiance élevée M_{GK} non valide (X Renforce Y)

Motifs générés

X	0.	1.	ă1.	0.	0.	1.	1.	0.	1.	1.
Y	1.	0.	1.	0.	1.	1.	1.	1.	1.	1.

	X	$\bar{X}$	
Y	0.5	0.3	0.8
$\bar{Y}$	0.1	0.1	0.2
	0.6	0.4	1

Résultats

M_{GK}	M_{GKcr}	règle	SupX	SupY	SupXY	ConfXY
0.1666667	0.1989786	non valide	0.6	0.8	0.5	0.8333333

Nous avons $M_{GK}(X \to Y) = 0,16$ inférieur à la valeur critique de M_{GK}, qui vaut 0,19, donc la règle $X \to Y$ est une règle positive et n'est pas valide, en revanche avec un seuil de confiance minimum de 0,7 car $conf(X \to Y) = 0,83$ dépassant $minconf$, la règle $X \to Y$ est valide selon la mesure de confiance.

Confiance élevée M_{GK} négatif non valide (X ne repousse pas Y)

X	1.	0.	0.	0.	0.	0.	1.	1.	1.	0.
Y	1.	1.	0.	1.	1.	1.	1.	0.	1.	1.

	X	$\bar{X}$	
Y	0.3	0.5	0.8
$\bar{Y}$	0.1	0.1	0.2
	0.4	0.6	1

Résultats

M_{GK}	M_{GKcr}	règle	SupX	SupY	SupXY	ConfXY
- 0.0625	- 0.0746170	non valide	0.4	0.8	0.3	0.75

Nous avons $M_{GK}(X \to Y) = -0,06$ où son module est inférieur au module de valeur critique de M_{GK}, qui vaut 0,074, donc la règle $X \to Y$ est négative et également non valide, d'autre part, a un seuil de confiance $minconf$ de 0,7, $Conf(X \to Y) = 0,75$ supérieur à $minconf$, la règle $X \to Y$ est également valide selon la mesure de confiance.

Confiance élevée M_{GK} positif non valide (X ne renforce pas Y)

X	1.	1.	1.	1.	1.	1.	1.	1.	0.	0.
Y	1.	1.	0.	1.	1.	0.	0.	1.	1.	0.

	X	$\bar{X}$	
Y	0.5	0.1	0.6
$\bar{Y}$	0.3	0.1	0.4
	0.8	0.2	1

Résultats

M_{GK}	M_{GKcr}	règle	SupX	SupY	SupXY	ConfXY
0.0625	0.074617	non valide	0.8	0.6	0.5	0.625

Nous avons $M_{GK}(X \to Y) = 0,062$ légèrement en dessous de la valeur critique de M_{GK}, qui est 0,07, donc la règle $X \to Y$ est d'abord positive et aussi non valide, elle peut être correcte si nous augmentons le risque avec $Conf(XY) = 0,625$, la règle $X \to Y$ sera probablement valide selon la mesure de confiance, en prenant $minconf$ inférieur à 0,6.

Indépendance de X et Y Confiance = 0.5

X	1.	1.	1.	1.	1.	1.	1.	1.	0.	0.
Y	0.	1.	0.	0.	1.	1.	1.	0.	0.	1

	X	$\bar{X}$	
Y	0.4	0.1	0.5
$\bar{Y}$	0.4	0.1	0.5
	0.8	0.2	1

Résultats

M_{GK}	M_{GKcr}	règle	SupX	SupY	SupXY	ConfXY
0		non valide	0.8	0.5	0.4	0.5

Nous avons une valeur de M_{GK}nulle, selon cette mesure, il est évident que la règle $X \to Y$ est nulle, et aussi invalide, par contre la valeur de sa confiance vaut 0,5, elle peut être valide solon confiance si le choix de $minconf$ baisse jusqu'à 0,5.

Confiance faible, M_{GK}négatif valide (X repousse Y)

X	1.	1.	1.	1.	1.	1.	1.	0.	0.	1.
Y	1.	0.	0.	0.	1.	0.	0.	1.	1.	0.

	X	$\bar{X}$	
Y	0.2	0.2	0.4
$\bar{Y}$	6	0	0.6
	0.8	0.2	1

Résultats

M_{GK}	M_{GKcr}	règle	SupX	SupY	SupXY	ConfXY
- 0.375	- 0.074617	valide	0.8	0.4	0.2	0.25

Nous avons $M_{GK}(X \to Y) = -0,37$, où son module est plus grand que le module de valeur critique de M_{GK}, qui est 0,074, donc la règle $X \to Y$ est négative et également valide, d'autre part, a un seuil de confiance minconf accepté , $Conf(X \to Y) = 0,25$ sera toujours inférieur à mincof, la règle $X \to Y$ n'est pas valide selon la mesure de confiance.

2.5.3 Contexte binaire et notions de règles d'association.

Un contexte d'extraction de règles d'association est un triplet $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ dans lequel $\mathcal{O}$ appelé ensemble d'objets (ou transactions), $\mathcal{I}$ ensemble d'attributs (ou items), et $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$ est une relation binaire. Une règle d'association est un couple $(X, Y) \in 2^{\mathcal{I}} \times 2^{\mathcal{I}}$ de motifs, noté : $X \to Y$, où X et Y sont de motifs disjoints ($X, Y \subseteq \mathcal{I}$ et $X \cap Y = \varnothing$), appelés respectivement prémisse et conséquent de la règle.

Règle d'association.

Dans ce qui suit, désignons $n = |\mathcal{O}|$ et P une probabilité sur $(\mathcal{O}, \mathcal{P}(\mathcal{O}))$, définie pour tout $X \subseteq \mathcal{I}$, par : $P(X) = \frac{n_X}{n}$.

Définition 54 (Support) *Le support d'une règle d'association* $X \to Y$ *est la proportion de transactions dans la base qui contiennent* $X \wedge Y$. $supp(X \to Y) = P(X \wedge Y)$

Une regle $r : X \to Y$ *est valide selon support si* $supp(X \to Y) \geq minsup$

Définition 55 (Confiance) *La confiance d'une règle d'association* $X \to Y$ *est le rapport entre le nombre de transactions qui contiennent* $X \wedge Y$ *et nombre de transactions qui contiennent* X. $conf(X \to Y) = P_{X'}(Y') = \frac{P(X' \cap Y')}{P(X')}$

Une regle $r : X \to Y$ *est valide selon confiance si* $conf(X \to Y) \geq minconf$

Définition 56 (Lift) *La valeur de lift d'une règle d'association* $X \to Y$ *est* $lift(X \to Y) = \frac{P(X' \cap Y')}{P(X')P(Y')}$

Une regle $r : X \to Y$ *est valide selon lift si* $lift(X \to Y) \geq minlift$

Remarque 6 *La validité d'une règle selon les mesures support, confiance et lift dépend d'une valeur constante préalablement choisi ; le seuil est déterminé a priori par l'utilisateur, ce seuil ne prend pas en considération la nature des données.*

Proposition 17 *Une règle entre deux motifs est valide seulement en fonction de leur contribution.* $r : X \to Y$ *est valide si* $X \to Y > f(X, Y, \alpha)$ *ou* α [2] *est un seuil d'erreur et* f *est une fonction critique de cette mesure.*

Définition 57 (Coefficient de corrélation) *Mesure de corrélation deux motifs (Brin et al., 1997a).* $\phi(X \to Y) = \frac{P(X' \cap Y') - P(X')P(Y')}{\sqrt{P(X')P(X')P(\overline{X}')P(\overline{Y}')}}$

Sa validité dépend de la statistique χ_2 et elle détermine une dépendance entre deux motifs.

Définition 58 (M_{GK}) *Soit* X *et* Y *deux motifs d'un contexte de fouille de données. On définit la mesure* M_{GK} *(Totohasina et al., 2004) par : etoary*

$$M_{GK}(X \to Y) = \begin{cases} \frac{P_{X'}(Y') - P(Y')}{1 - P(Y')}, & si\ P_{X'}(Y') \geq P(Y') \\ \frac{P_{X'}(Y') - P(Y')}{P(Y')}, & si\ P_{X'}(Y') < P(Y') \end{cases}$$

Proposition 18 *Une règle* $r : X \to Y$ *est valide selon* M_{GK} *si* $M_{GK}(X \to Y) \geq M_{GKcr}(X \to Y, \alpha)$

Définition 59 (Règles valides) *Étant donnés un seuil minimal* $minsupp$ *pour la mesure de support et seuil* α *pour* M_{GKcr}*, l'ensemble* $\mathcal{R}$ *de règles valides est :*

$$\mathcal{R} = \{(X, Y) \subseteq \mathcal{I} \times \mathcal{I} | supp(X \to Y) \geq minsupp\ et\ M_{GK}(X \to Y) \geq M_{GKcr}(X \to Y, \alpha)\}.$$

La valeur critique M_{GKcr} pour la mesure M_{GK}, proposée dans Feno et al. (2006), est obtenue de la manière suivante. Considérons un tableau de contingence par le croisement de ces deux motifs X et Y, s'appuyant sur la statistique de Khi-carré (χ^2) de Pearson à un degré de liberté, tel que : Rakotomalala et al. (2019)

$$M_{GKcr}(X \to Y, \alpha) = \sqrt{\frac{1}{n} \frac{n - n_X}{n_X} \frac{n_Y}{n - n_Y} \chi^2_{cr(\alpha)}}, \text{avec X favorise Y} \tag{2.1}$$

A partir de la table de χ^2, cette relation permet d'élaborer facilement les abaques des valeurs critiques de M_{GK} pour décider sa signification envers une règle d'association.

2. Un seuil statistique d'élagage calculés directement à partir des données.

Proposition 19 *Soit* $\mathcal{D} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ *un contexte d'extraction. Pour tous motifs* X ; Y *de* $\mathrm{P}(\mathcal{I})$ *et pour tout* $\alpha \in]0;1[$ *; on a :(Ramanantsoa, 2016)*

1. *Si* X *favorise* Y *, alors* $\mathrm{M}_{\mathrm{GK}cr}(\mathrm{X} \to \mathrm{Y}, \alpha) = \mathrm{M}_{\mathrm{GK}cr}(\overline{\mathrm{Y}} \to \overline{\mathrm{X}}, \alpha)$ *;*
2. *Si* X *defavorise* Y *, alors* $\mathrm{M}_{\mathrm{GK}cr}(\mathrm{X} \to \mathrm{Y}, \alpha) = \mathrm{M}_{\mathrm{GK}cr}(\mathrm{Y} \to \overline{\mathrm{X}}, \alpha)$ *;*

Preuve. : un seuil d'erreur α, si X favorise Y , alors

$$\begin{aligned}\mathrm{M}_{\mathrm{GK}cr}(\mathrm{X} \to \mathrm{Y}, \alpha) &= \sqrt{\frac{1}{n}\frac{n-n_{\mathrm{X}}}{n_{\mathrm{X}}}\frac{n_{\mathrm{Y}}}{n-n_{\mathrm{Y}}}\chi^2_{cr(\alpha)}} \\ &= \sqrt{\frac{1}{n}\frac{n_{\overline{\mathrm{X}}}}{n-n_{\overline{\mathrm{X}}}}\frac{n-n_{\overline{\mathrm{Y}}}}{n_{\overline{\mathrm{Y}}}}\chi^2_{cr(\alpha)}} \\ &= \mathrm{M}_{\mathrm{GK}cr}(\overline{\mathrm{Y}} \to \overline{\mathrm{X}}, \alpha)\end{aligned}$$

Par contre, si X défavorise Y , alors X favorise Y et on peut appliquer la première propriété :

$$\begin{aligned}\mathrm{M}_{\mathrm{GK}cr}(\mathrm{X} \to \overline{\mathrm{Y}}, \alpha) &= \mathrm{M}_{\mathrm{GK}cr}(\overline{\overline{\mathrm{Y}}} \to \overline{\mathrm{X}}, \alpha) \\ &= \mathrm{M}_{\mathrm{GK}cr}(\mathrm{Y} \to \overline{\mathrm{X}}, \alpha)\end{aligned}$$

Corollaire 6 *Pour tout couple de motifs* X ; Y *, si* X *favorise* Y *, alors les deux régles* $\mathrm{X} \to \overline{\mathrm{Y}}$ *et* $\overline{\mathrm{Y}} \to \overline{\mathrm{X}}$ *ont la même mesure selon* M_{GK} *et même valeur critique. Par conséquent, si les motifs* $\overline{\mathrm{X}}$ *et* $\overline{\mathrm{Y}}$ *sont fréquents, nous avons les équivalences, au sens de* M_{GK} *:*
$\mathrm{X} \to \mathrm{Y}$ *valide* $\Longleftrightarrow \overline{\mathrm{Y}} \to \overline{\mathrm{X}}$ *valide ;*
$\mathrm{X} \to \overline{\mathrm{Y}}$ *valide* $\Longleftrightarrow \mathrm{Y} \to \overline{\mathrm{X}}$ *valide*

Selon le corollaire 6 (Ramanantsoa, 2016), on peut déduire la validité et la mesure de $\mathrm{Y} \to \mathrm{X}$ (respectivement de $\mathrm{Y} \to \mathrm{X}$) à partir de celles de $\mathrm{X} \to \mathrm{Y}$ (respectivement de $\mathrm{X} \to \mathrm{Y}$). Il est donc inutile de mettre $\mathrm{Y} \to \mathrm{X}$ ainsi que $\mathrm{Y} \to \mathrm{X}$ lorsqu'on a déjà $\mathrm{X} \to \mathrm{Y}$ et $\mathrm{X} \to \mathrm{Y}$. Contrairement aux anciennes bases des règles MGK-valides, ces deux types de règles ($\mathrm{Y} \to \mathrm{X}$ et $\mathrm{Y} \to \mathrm{X}$) ne feront pas partie des nouvelles bases des règles MGK-valides. Cette restriction nous permet de réduire considérablement la taille des bases des règles tout en conservant les traces de toutes les informations valides.

2.6 Evaluations Expérimentales.

2.6.1 Nombres des règles extraits.

Dans ces deux graphes de la figure 2.8 nous avons montré le résultat de comparaison de nombre des règles d'association valides selon la mesure M_{GK} en fonction de seuil d'erreur. A gauche, le nombre des règles valides selon M_{GK} est comparé avec celui de confiance pour $minconf$ = 0.6, 0.7 et 0.8 à seuil d'erreur α = 0.05 M_{GK}-valide est comparable confiance (0.6), et 0.01 0.7 et 0.00 0.8. A droite, il est comparé avec les règles valides selon Lift pour $minlift$ = 1.0, 1.05, 1.15, 1.25. Ralahady & Totohasina (2019b)

2.6.2 Seuils de validités.

Les deux volets de la figure 2.9 montrent les 129 règles extraites à partir de tous les motifs fréquents supérieur à $minsup$ = 0.3. A droite les mesures d'interêts lift et confiance sont comparées

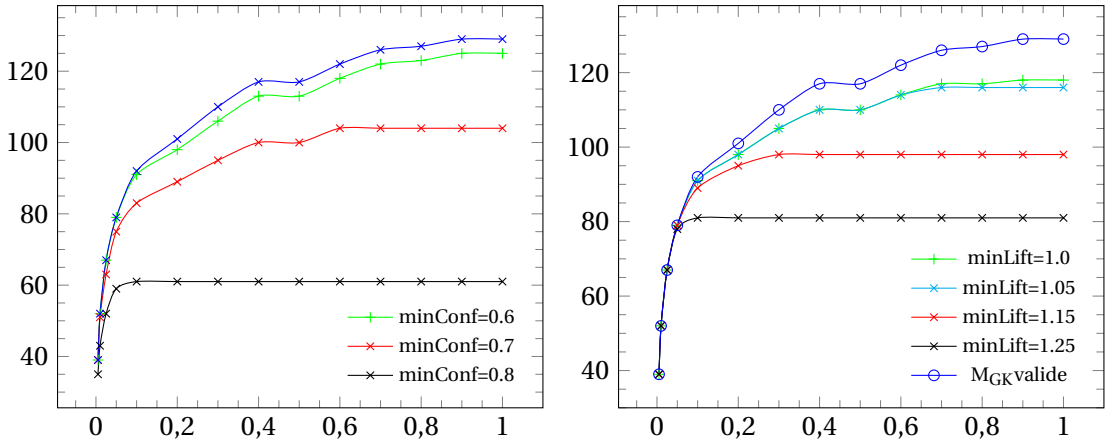


FIGURE 2.8 – Variation des règles valides selon seuils d'erreurs.

à la fonction mgkcr(α) pour seuil d'erreur α = 0.025. La figure nous montre que les règles extraites à ce seuil sont plus confiantes que les règles extraites utilisant $minlift = 1.32$ et $minconf = 0.77$.

Lift ne permet pas de faire le choix entre X → Y et Y → X ; au sens de Lift les règles X → Y et Y → X sont équivalentes. La forte variation à la fin de la courbe indique sa sensibilité à la taille de données.

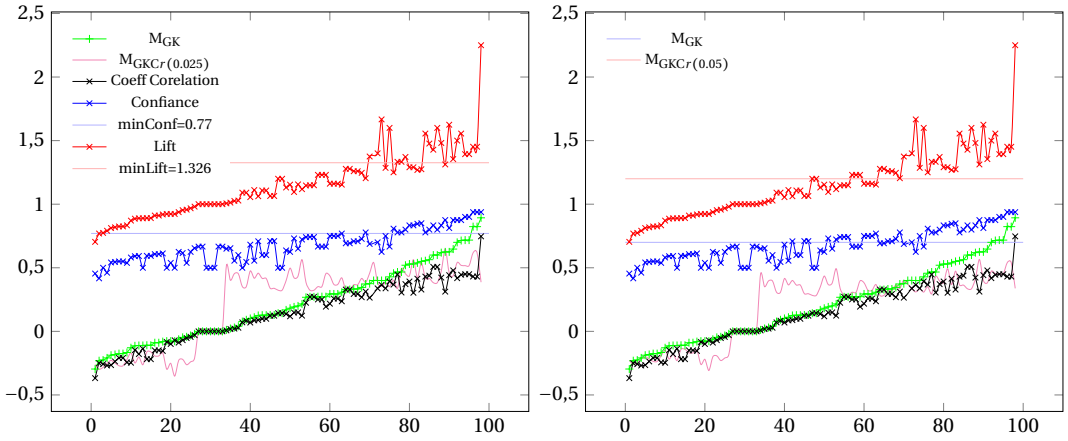


FIGURE 2.9 – Variation de nombre des règles extraites selon les seuils utilisés.

La courbe représentative de M_{GK} montre sa robustesse, sa faible variation et sa normalité (ses valeurs qui sont comprises dans l'intervalle $[-11]$). Comparaison avec coefficient de corrélation linéaire, ces deux mesures peuvent extraire deux types de règles ; règles négatives et règle positives. Le coefficient de corrélation linéaire ne nous permet pas de distinguer les règles X → Y et Y → X. Ce qui n'est pas le cas pour la mesure M_{GK}. Comme lift, elle est sensible à la taille de données. M_{GK} est plus discriminative

Il semble que M_{GK} présente beaucoup plus d'interêt par rapport à ces mesures.

2.6.3 Statut des règles extraites.

Selon le graphique (fig 2.10), en utilisant $minconf = 0.7$ et $minlif = 1.26$ et le risque d'erreur α = 0.05%, ces quatre mesures génèrent toutes 29 règles valides. En utilisant la fenêtre de visuali-

sation dynamique du graphe implicatif de l'outil ASI-MGK, après quelques réorganisations, nous analyserons le graphe obtenu.

Le graphique résultant est un graphe orienté du même ordre (29 étapes) de différentes composantes connexes, c'est-à-dire que les règles générées sont différentes. Comme le montre la figure, nous voyons que M_{GK} et le coefficient de corrélation produisent le même graphique.

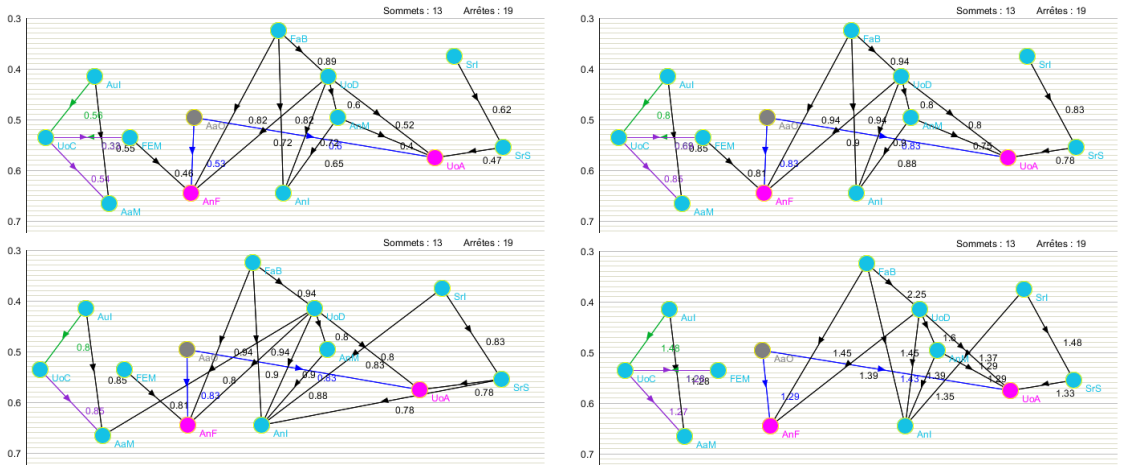


FIGURE 2.10 – Graphe implicatif des 29 règles extraites.

Par contre, Lift diffère de M_{GK} en quelque règles ; l'absence de la $\{FEM\} \Rightarrow \{AnF\}$ et l'apparition de la règle $\{SrS\} \Rightarrow \{AnI\}$. Pour celle de la confiance la différence est très remarquable ; les disparitions des règles équivalents :$\{FEM\} \Rightarrow \{AnF\}$ et $\{AnF\} \Rightarrow \{FEM\}$ et les apparitions des régles $\{FEM\} \Rightarrow \{AnF\}$ et $\{FEM\} \Rightarrow \{AnF\}$.

Selon la figure (fig. 2.11), en utilisant $minconf$ = 0.7 et $minlif$ = 1.26 et risque d'erreur α = 0.05%, ces quatre mesures génèrent chacun 29 règles valides. En profitant la fenêtre de visualisation dynamique du graphe implicatif de l'outil ASI – MGK, après quelques réorganisations, nous allons analyser les graphes obtenus.

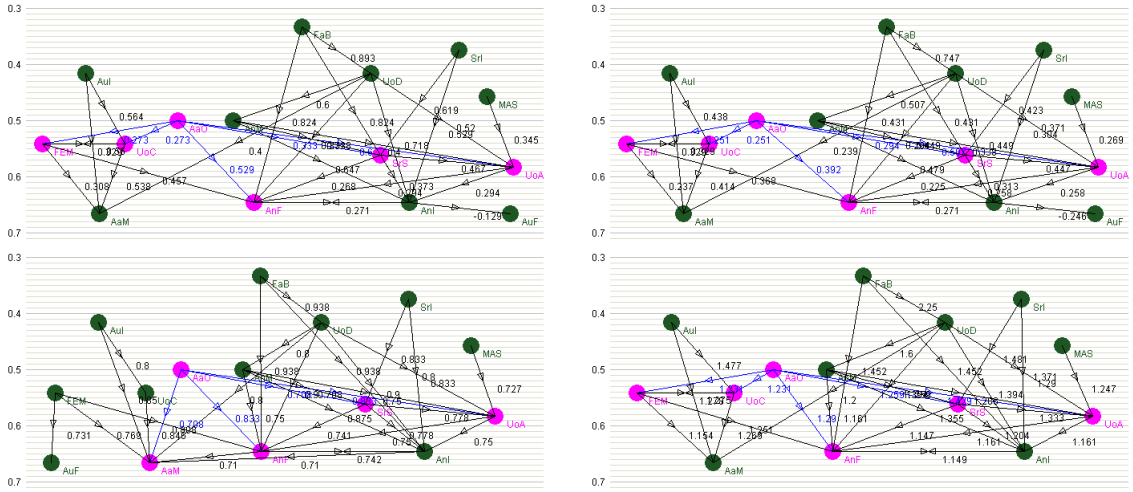


FIGURE 2.11 – Graphe implicatif des 34 règles extraites.

2.7 Logiciels explorateurs de règles.

Différents "explorateurs de règles" ont été développés pour aider l'utilisateur dans le post-traitement des règles. A l'instar des explorateurs de fichiers, ce sont des interfaces interactives qui présentent l'information sous forme textuelle.
L'idée a d'abord été proposée par Klemettinen et al. (1994), puis implémentée dans le logiciel TASA pour l'analyse des pannes d'un réseau de télécommunication. A l'aide de l'explorateur de règles intégré à TASA, l'utilisateur peut isoler les règles qui l'intéressent en ajustant des seuils sur des indices de règle et en spécifiant certaines contraintes syntaxiques (contraintes indiquant les items qui doivent apparaître ou ne pas apparaître dans les règles). Dans ce travail, l'explorateur exploite un résumé de l'ensemble des règles obtenu par la méthode de Liu et al. (1999). En sélectionnant une règle dans le résumé, l'utilisateur peut accéder aux règles plus spécifiques correspondantes.

Plus récemment, les explorateurs de règles dotés de nombreuses fonctionnalités permettent de filtrer les règles par des contraintes syntaxiques plus ou moins générales, puisqu'elles peuvent prendre en compte une taxonomie des items. L'outil propose également à l'utilisateur de programmer les indices de règle de son choix pour trier et filtrer les règles. De plus, l'utilisateur a la possibilité de sauvegarder les règles qu'il juge intéressantes au fur et à mesure de son exploration.

Certains explorateurs de règles exploitent des indices de règle subjectifs, et peuvent ainsi tirer profit des connaissances de l'utilisateur sur les données. Par exemple, dans Liu & Hsu (1996) et Liu et al. (1999), l'explorateur ordonne les règles ou classe les règles en catégories selon qu'elles contredisent plus ou moins les croyances de l'utilisateur. Ces croyances sont exprimées différemment dans les deux approches.

- Pour Liu & Hsu (1996), les croyances sont constituées des règles que l'utilisateur a jugées valides lors des explorations précédentes. Ces règles sont converties dans une représentation floue (fuzzification), sous le contrôle de l'utilisateur, pour plus de souplesse lors de la comparaison aux autres règles.
- Plus récemment Liu et al. (1999), proposent une syntaxe pour que l'utilisateur puisse exprimer ses croyances avec différents degrés de précision. Une taxonomie des items peut être exploitée.

La principale limite des explorateurs de règles existants réside dans le mode de présentation textuel, qui ne convient pas à l'étude de grandes quantités de règles. Ces outils ont également le défaut de n'implémenter qu'un faible nombre d'indices de règle (trois maximum, souvent deux).

2.7.1 Liste des mesures d'intérêt intégrées dans ASI – MGK.

De nombreux outils d'analyse déjà existants utilisent généralement le couple Support Confiance comme la mesure d'intérêt de règle d'association, certains outils implémentent aussi la mesure d'intérêt de règle d'Association Lift. L'outil expérimental, développé par?? en 2004 implémente 22 mesures. D'ailleurs, les règles valides sont sélectionnés par un seuil minimum (comme par exemples *minconf* ou *minlift*) qui est fixé préalablement par l'utilisateur.

Dans notre outil ASI–MGK, nous avons intégré 44 mesures d'intérêts, dont la liste des mesures supportées est présentée dans le tableau 2.1. Le seuil de validité de règle valide sera d'obtenir à

partir de la valeur critique la mesure M_{GK} calculé par raport à la taille de données et du valeur du Test de khi-deux à 1 degré de liberté avec le risque d'erreur α.

Enfin, dans la fenêtre de visualisation des résultats de l'ASI – MGK, les intensités des règles d'associations valides pourront ainsi être représentées selon M_{GK} ou une autre mesure (M*s*) choisi dans la liste, ou bien, être représentées en même temps par le couple des mesures (M_{GK}, M*s*).

N°	Mesures	expressions
1	Dépendance causale estimée	$\frac{3}{2}+2P(X')-\frac{3}{2}P(Y')-\frac{3}{2}P(\overline{Y}'/X')-2P(X'/\overline{Y}')$
2	Loevinger ou Satisfaction	$\frac{P(Y'/X')-P(Y')}{1-P(Y')}$
3	Fiabilité négative	$P(\overline{X}/\overline{Y})$
4	Force collective	$\frac{P(X'\cap Y')+P(\overline{X}'\cap\overline{Y}')}{P(X')P(Y')+P(\overline{X}')P(\overline{Y}')}\frac{1-P(X')P(Y')+P(\overline{X}')P(\overline{Y}')}{1-P(X'\cap Y')-P(\overline{X}'\cap\overline{Y}')}$
5	Intérêt ou Lift	$\frac{P(X'\cap Y')}{P(X')P(Y')}$
6	Laplace	$\frac{n_{XY}+1}{n_X+2}$
7	Laverage	$P(Y'/X')-P(X')P(Y')$
8	M_{GK}	$\begin{cases}\frac{P(Y'/X')-P(Y')}{1-P(Y')}, \textit{ si } X \text{ favorise } Y\\ \frac{P(Y'/X')-P(Y')}{P(Y')}, \textit{ si } X \text{ défavorise } Y\end{cases}$
9	Moindre contraduction ou Surprise	$\frac{P(X'\cap Y')-P(X'\cap\overline{Y}')}{P(Y')}$
10	Nouveauté	$P(X'\cap Y')-P(X')P(Y')$
11	Pearl	$P(X')\lvert P(Y'/X'-P(Y')\rvert$
12	Piatetsky-Shapiro	$n(P(X'\cap Y')-P(X')P(Y'))$
13	Precision ou S upport Causal	$P(X')+P(\overline{Y}')-2P(X'\cap\overline{Y}')$
14	Rappel	$P(X'/Y')$
15	Risque relatif	$\frac{P(X'\cap Y')}{P(Y'/\overline{X}')}$
16	Spécificité	$P(\overline{Y}'/\overline{X}')$
17	Support	$P(X'\cap Y')$
18	Coefficient de corrélation	$\frac{P(X'\cap Y')-P(X')P(Y')}{\sqrt{P(X')P(X')P(\overline{X}')P(\overline{Y}')}}$
19	Cohen ou Kappa	$2\frac{P(X'\cap Y')-P(X')P(Y')}{P(X')P(\overline{Y}')+P(\overline{X}')P(Y')}$
20	Confiance ou précision	$P(X'/Y')$
21	Confiance causale	$1-\frac{1}{2}P(\overline{Y}'/X')-\frac{1}{2}P(X'/\overline{Y}')$
22	Confiance centrée descriptive	$P(Y'/X')-P(Y')$
23	Confiance confirmée descriptive	$1-2P(\overline{Y}'/X')$

24	Confiance confirmée causale	$1-\frac{3}{2}P(\overline{Y}'/X')-\frac{1}{2}P(X'/\overline{Y}')$
25	Confirmation causale	$P(X')+P(\overline{Y}')-4P(X'\cap\overline{Y}')$
26	Confirmation descriptive	$P(X')-2P(X'\cap\overline{Y}')$
27	Cosinus ou Ochiai	$\frac{P(X'\cap Y')}{\sqrt{P(X')P(Y')}}$
28	Czekanowski-Dice ou F-mesure	$\frac{2P(X'\cap Y')}{P(X')+P(Y')}$
29	Dépendance	$\left\|P(\overline{Y}')-P(\overline{Y}'/X')\right\|$
30	Multiplicateur de côte	$\frac{P(X'\cap Y')P(\overline{Y}')}{P(X'\cap\overline{Y}')P(Y')}$
31	Sebag	$\frac{P(X'\cap Y')}{P(X'P(\overline{Y}')}$
32	Conviction	$\frac{P(X')P(\overline{Y}')}{P(X'\cap\overline{Y}')}$
33	Odd-Ratio	$\frac{P(X'\cap Y')P(\overline{X}'\cap\overline{Y}')}{P(\overline{X}'\cap Y')P(X'\cap\overline{Y}')}$
34	Gain-Informelle	$\log\frac{P(X'\cap Y')}{P(X'P(Y')}$
35	Exemple contre exemple	$\frac{P(X'\cap Y')-P(X'\cap\overline{Y}')}{P(X'\cap Y')}$
36	Klosgen	$\sqrt{P(X'\cap Y')}(P(X'/Y')-P(Y'))$
37	Support à sens unique	$P(Y'/X')\log\frac{P(X'\cap Y')}{P(X')P(Y')}$
38	Support à double sens	$P(X'\cap Y')\log\frac{P(X'\cap Y')}{P(X')P(Y')}$
39	Couverture	$P(X')$
40	Prévalence	$P(Y')$
41	Jaccard	$\frac{P(X'\cap Y')}{P(X')+P(Y')-P(X'\cap Y')}$
42	Q de Yule	$\frac{P(X'\cap Y')P(\overline{X}'\cap\overline{Y}')-P(X'\cap\overline{Y}')P(\overline{X}'\cap Y')}{P(X'\cap Y')P(\overline{X}'\cap\overline{Y}')+P(X'\cap\overline{Y}')P(\overline{X}'\cap Y')}$
43	Y de Yule	$\frac{\sqrt{P(X'\cap Y')P(\overline{X}'\cap\overline{Y}')}-\sqrt{P(X'\cap\overline{Y}')P(\overline{X}'\cap Y')}}{\sqrt{P(X'\cap Y')P(\overline{X}'\cap\overline{Y}')}+\sqrt{P(X'\cap\overline{Y}')P(\overline{X}'\cap Y')}}$
44	J-Mesure	$P(X'\cap Y')\log\frac{P(X'\cap Y')}{P(X')P(Y')}+P(X'\cap\overline{Y}')\log\frac{P(X'\cap\overline{Y}')}{P(X')P(\overline{Y}')}$

TABLEAU 2.1 – Liste des mesures d'intérêt integré dans ASI – MGK

2.7.2 Vue rapide des logiciels disponibles.

Voici une revue rapide des progiciels gratuits accessibles dans la section SUITES du site KDNUGGETS[3].

Adam[4] est une collection d'exécutables qui implémentent des méthodes de traitement de données. Le format de fichier de données standard est le format WEKA (ARFF, voir section 2.3.1 page 42) . Si on veut enchaîner des traitements, il faudrait donc mettre à la suite la succession des traitements dans un fichier script BAT. Particularité intéressante, certains modules peuvent traiter directement des images.

3. http://www.kdnuggets.com/software/suites.html
4. http://datamining.itsc.uah.edu/adam/

Alphaminer [5] est un logiciel stand-alone qui implémente la chaîne de traitements sous forme de graphes. Il peut prendre en entrée des fichiers textes ou EXCEL. Sa principale particularité est qu'il est capable de s'interfacer avec des plug-ins écrits dans d'autres contextes, il peut notamment récupérer des classes compilées dans WEKA par exemple.

Databionic esom tools [6], l'idée est de mettre les SOM à toutes les sauces : visualisation, classification et classement. Le logiciel est totalement dédié à ce paradigme.

Gnome data mining tools [7] est un logiciel fonctionnant sous LINUX et implémentant quelques méthodes de traitement de données (arbres de décision, règles d'association, graphiques). Les modules sont indépendants, ils possèdent leur propre interface, il ne semble pas possible de les enchaîner sans passer par la programmation.

IBM intelligent miner [8] Apparemment, ce logiciel ne peut pas fonctionner dans le serveur de base de données DB2. Il s'appuie dessus pour la gestion des données. La panoplie des méthodes usuelles (classement avec les arbres, classification, etc.) semble présente. En revanche, les possibilités d'enchaînement ne sont pas très claires. Le logiciel est gratuit pour l'enseignement.

Mining mart [9], ce logiciel met l'accent sur la préparation des données, préalable primordial avant la mise en œuvre des techniques exploratoires. Sa particularité est la nécessité de s'interfacer avec un SGBD relationnel. Il peut traiter une série de tables reliées entre elles ; il ne peut pas en revanche appréhender un fichier texte externe. Dans la version 0.22, il faut apparemment une version d'ORACLE pour le package fonctionne. L'interface respecte le standard des chaînes de traitements, plusieurs composants sont dédiés au pré traitement des données : jointures de tables, requêtes, etc. Le rapprochement avec les outils d'ETL (Extraction Transfer Loading) est tout à fait approprié.

Mlc++ [10] est une librairie de classes C++ destiné à l'apprentissage supervisé. Le code source est disponible, il est impossible de le mettre en œuvre sans le compiler soi-même, de même si on veut enchaîner une série de manipulations, il faut passer par la programmation. Le principal intérêt de cette bibliothèque est qu'elle a été écrite par Kohavi et al. (1994), un acteur important de la communauté MACHINE LEARNING, et qu'elle constitue le cœur du logiciel MINESET de SGI.

MLJ (Machine Learning in JAVA) [11] est un portage de MLC++. Il ne semble pas présenter d'intérêt supplémentaire, mis à part que l'on dispose également du code source.

5. http://www.eti.hku.hk/alphaminer/
6. http://databionic-esom.sourceforge.net/
7. http://www.togaware.com/datamining/gdatamine/
8. http://www.developer.ibm.com/university/scholars/
9. http://mmart.cs.uni-dortmund.de/
10. http://www.sgi.com/tech/mlc/
11. http://www.kddresearch.org/Groups/Machine-Learning/MLJ/

Star probe [12] est une application implémentant plusieurs techniques (arbres de décision, visualisation, etc.) qui peut fonctionner soit en stand-alone, soit comme une application client-serveur. Une version limitée est accessible directement sur le web, il s'agit d'une APPLET JAVA. Il existe une interface graphique, en revanche, il ne semble pas possible de définir une chaîne de traitements.

Orange [13] Développé par l'Université de Slovénie, le logiciel correspond exactement à notre cahier de charges. Il est possible de programmer les traitements à l'aide de scripts en PYTHON, il est également possible de définir les traitements à l'aide d'un graphe représentant les « filières ». Le cœur des algorithmes de calcul sont compilés dans des DLL écrits en C++.

Yale [14] ce logiciel s'appuie sur le cœur de WEKA et propose une interface graphique permettant de représenter l'enchaînement des traitements sous forme d'arbre : arborescence avec, à la racine, la source de données. Ce logiciel intègre une particularité intéressante : des traitements types ont été définis et un wizard permet de définir assez aisément la suite de traitements que l'on veut mettre en place. Il n'est donc pas nécessaire de composer soi même les icônes représentant les traitements.
Point très important, tant que WEKA n'intégrait pas d'interface graphique pour définir les chaînes de traitements, YALE pouvait représenter une alternative intéressante pour les allergiques de la programmation. Il semble quand même que YALE propose une alternative intéressante, bien que compliquée, pour définir des opérations complexes.

WEKA [15] Une bibliothèque de méthodes faramineuse, un code assez bien écrit la plupart du temps, avec des architectures très accessibles : la reprise du code ne pose aucun problème. Un bond en avant a été réalisé avec l'introduction d'une interface graphique permettant de définir en enchaînement de traitements sans avoir recours à la programmation.

2.8 Méthodes de visualisation des règles.

Les méthodes et outils de visualisation de règles existantes incluent généralement les fonctionnalités de base des programmes pour le tri et le filtrage des règles, en fonction des items qui les constituent ou bien selon quelques indices de règle (peu d'indices en fait : le support, la confiance, et parfois une troisième mesure comme le lift).

Une première méthode de visualisation des règles d'association est la représentation par matrice (Wong et al., 1999) et le groupe de recherche Quest (Agrawal et al., 1996), ainsi que les logiciels DBMiner (Han, 1998), MineSet (Brunk et al., 1997), Enterprise Miner, et DB2 Intelligent Miner Visualization, en donnent différentes implémentations.

Dans une matrice itemset-à-itemset, chaque colonne correspond à un itemset en prémisse et chaque ligne à un itemset en conclusion. Cette technique de visualisation a été améliorée en matrices item-à-règle (Wong et al., 1999), où chaque ligne correspond à un item et chaque colonne à

12. http://www.roselladb.com/starprobe.htm
13. http://magix.fri.uni-lj.si/orange/
14. http://www-ai.cs.uni-dortmund.de/SOFTWARE/YALE/index.html
15. http://www.cs.waikato.ac.nz/ml/weka/

une règle. La cellule à l'intersection d'un item et d'une règle est pleine ou vide suivant que l'item appartient ou non à la règle, la couleur de remplissage indiquant si l'item participe à la prémisse ou à la conclusion. Par rapport aux matrices itemset-àă-itemset, les matrices item-à-règle sont moins encombrées et permettent une meilleure représentation des règles de plus de deux items. La principale limite de ces représentations matricielles est qu'elles atteignent des tailles considérables dans le cas de grands ensembles de règles portant sur de nombreux items.

Les ensembles de règles d'association peuvent également être visualisés à l'aide d'un graphe orienté (voir (Klemettinen et al., 1994; Rainsford & Roddick, 2000), et les logiciels DBMiner (Han, 1998), CHIC (Couturier & Gras, 2005), et DB2 Intelligent Miner Visualization). Dans ce type de représentations, les noeuds et les arcs symbolisent respectivement les items et les règles. Dans Hao et al. (2001), la méthode est implémentée en 3D avec un algorithme de type masses ressorts qui optimise le placement des noeuds dans l'espace. D'abord, elle fait implicitement apparaître les règles comme des relations transitives, alors que dans le cas général, les règles ne sont pas transitives (avec la plupart des indices de règle, la qualité des règles ne se propage pas par transitivité).Ensuite, elle ne convient pas non plus à la visualisation de grands ensembles de règles portant sur de nombreux items : le graphe est surchargé de nœuds et d'arcs qui se croisent, d'autant plus si des règles de plus de deux items sont considérées

2.8.1 Visualisation d'information dans l'ASI-MGK.

Card et al. (1999) proposent un modèle générique des outils informatiques de visualisation d'information. Il consiste en une suite de traitements interactifs permettant de passer des données en entrée à une visualisation en sortie : les transformations sur les entrées, puis l'encodage graphique, puis les transformations sur la vue (figure 2.12). Les données en entrée sont un ensemble d'individus décrits par des variables qui peuvent être qualitatives nominales, qualitatives ordinales, ou quantitatives. L'utilisateur contrôle chacune des transformations en interagissant avec l'outil de visualisation (directement dans l'interface où s'affiche la vue, ou bien dans une interface séparée).

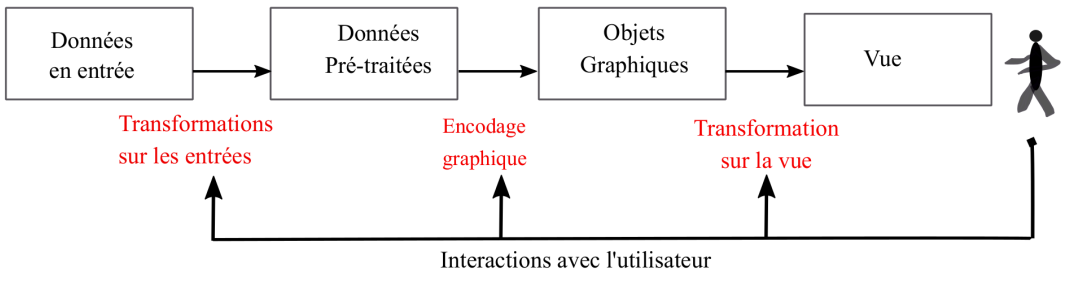


FIGURE 2.12 – Modèle générique pour la visualisation d'information.

1. Les transformations sur les entrées sont les traitements à effectuer sur les données avant qu'elles ne soient visualisées (sélection d'individus ou de variables, regroupement d'individus, formatage de variables, ordonnancement des modalités d'une variable, etc.). Pour ce qui concerne ces transformations, les techniques d'interaction les plus courantes sont les suivantes :

- le filtrage dynamique (introduit dans Williamson & Shneiderman (1992)), consiste à sélectionner les individus à visualiser par des requêtes dynamiques (l'affichage est mis à jour instantanément) portant sur les variables et soumises généralement par l'intermédiaire de cases à cocher (pour des variables qualitatives) ou de sliders (pour des variables quantitatives) (Ahlberg & Shneiderman, 1994) ;
- le détail à la demande, qui consiste à choisir un élément dans la représentation et à faire apparaître des informations supplémentaires le concernant (souvent dans une fenêtre de type pop-up ou infobulle) ;
- le brushing, qui consiste à inclure dans ou exclure de la visualisation tout un sous-ensemble d'individus sélectionnés par l'utilisateur à l'aide du pointeur (Wills, 1996).

2. L'encodage graphique est au cœur de la visualisation des informations : il s'agit du problème de la réecriture des données sous forme d'objets graphiques en associant chaque variable des données à des variables graphiques (position, longueur, surface, couleur, luminosité, saturation, forme, texture, angle, courbure ...). Les objets graphiques peuvent être de zéro à trois dimensions, c'est-àă-dire des points, des lignes, des surfaces ou des volumes. L'évolution des objets dans le temps (modification de variables graphiques) peut constituer des dimensions supplémentaires. Les interactions consistent à modifier les associations entre données et graphisme (comme par exemple changer de variable sur un axe). Une interface classique pour cela consiste à présenter à l'utilisateur un graphe dont les noeuds symbolisent les variables des données et les variables graphiques, et dont les arcs sont fixés par l'utilisateur pour indiquer les associations choisies.

3. La transformation sur la vue implique la présentation d'un objet graphique à l'utilisateur. La vue affichée à l'écran peut être en 3D (même si l'objet graphique n'est pas en 3D) ou en 2D. Pour la transformation visuelle, les techniques d'interaction les plus courantes sont les suivantes :

- Le contrôle de l'angle de vue est effectué par panoramique, rotation ou zoom et peut être extraverti (l'utilisateur déplace la représentation lorsque le point de vue est fixe) ou autocentré (l'utilisateur déplace la vue du point de vue pour indiquer qu'il est fixe) ;
- Vues multiples (overview+details) , permettant à l'utilisateur d'obtenir une vue globale et une vue détaillée de la même représentation à travers deux fenêtres (le coup effectué dans une vue est également visible dans une autre vue, on parle de linking+brushing) ; (Shneiderman, 1996)
- les techniques dites focus+context qui intègrent les détails dans la vue globale en les révélant autour du focus (point d'intérêt) de l'utilisateur, soit par affichage direct (fish eye filtrant, introduit dans Furnas (1986)), soit par distorsion de la vue (fish eye déformant (Sarkar & Brown, 1992) et plans hyperboliques (Lamping et al., 1995).

Certaines techniques combinent plusieurs types de transformations. Par exemple, le zoom sémantique est un zoom (transformation sur la vue) qui change les données visualisées (transformation sur les entrées) : plus l'utilisateur zoome et plus le niveau de détail des données s'élève (Hascot & Beaudouin-Lafon, 2001). Il est donc possible de zoomer continuellement tant que le niveau de détail le plus faible n'est pas atteint. Ce type d'outils de visualisation est appelé interfaces zoomables.

2.8.2 Tâche de l'utilisateur.

Lors du post-traitement des règles d'association, l'utilisateur est confronté à un grand nombre de règles décrites par des indices. La tâche de l'utilisateur est alors de rechercher les règles pour trouver des connaissances intéressantes pour la prise de décision. Pour ce faire, il doit interpréter les règles et évaluer la qualité de la sémantique. Par conséquent, les deux indicateurs de la prise de décision sont la syntaxe des règle (items participant à chaque règle) et les indices.

La tâche de l'utilisateur est difficile pour deux raisons. Premièrement, un grand nombre de règles à la sortie de l'algorithme d'exploration de données interdit l'exploration détaillée. Ensuite, de par leur nature non supervisée, les règles d'association sont souvent utilisées lorsque l'utilisateur ne sait pas exactement ce qu'il recherche dans la terminologie de données. Il ne peut pas formuler spontanément des contraintes qui isolent directement les règles qui l'intéressent.

2.8.3 Transformations sur les entrées.

Nous profitons du fait que l'utilisateur concentre son attention sur des sous-ensembles pour ne visualiser que le sous-ensemble en cours d'exploration.

Ainsi, ce sont les relations de voisinage qui réalisent les transformations sur les entrées en ciblant les sous-ensembles. Des opérateurs interactifs doivent être intégrés à la visualisation pour que l'utilisateur puisse "activer" les relations.

2.8.4 Encodage graphique.

Certaines méthodes de visualisation des règles ont l'inconvénient de privilégier rarement les indices de règles. Par exemple, les méthodes de visualisation par coordonnées matricielles, graphiques et parallèles utilisent la couleur pour représenter certains indinces de règles, tandis que les choix de codage de modèle pour de telles variables quantitatives sont connus pour leur médiocre visualisation des informations (Bertin, 1967).

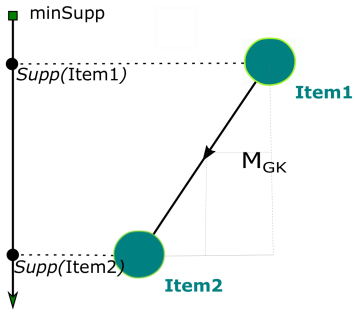


FIGURE 2.13 – Métaphore d'une règle d'association entre deux items.

Ainsi, nous avons décidé d'utiliser un graphe orienté pour représenter les règles d'association (fig.2.13). Chaque règle est symbolisée par un graphe orienté composé des nœuds qui matérialisent les items.

La représentation en graphe implicatif que nous proposons dans la figure 2.13 implémente les concepts suivants :

1. **seuil minimal de support** : la borne inférieure sur l'axe correspond au seuil minimal de support choisi arbitrairement par l'utilisateur pour filtrer les motifs ou ensemble des motifs candidats à étudier ;
2. **représentation d'item** : le nœud en formant de cercle généralement coloré en bleu symbolise la présence d'un item ou d'une modalité correspondant. Chaque item est directement identifiable par son code inscrit-en-dessous du nœud légèrement placé à droite. Sa position sur l'axe horizontale est non significative ainsi que celle sur l'axe verticale indique sa proportionnalité ou son support ;
3. **relations inter-items** : la relation implicative indique si la présence d'un item favorise la présence d'un autre item au sein de son ensemble (prémisse). La relation existante entre deux items sera représentée par un arc entre deux nœuds. La force de cette relation correspond au poids de l'arc qui seront évaluées en se basant sur la mesure du M_{GK} et/ou une autre mesure choisie par l'utilisateur ;
4. **représentation des mesures** d'intérêt support et M_{GK} :
   - l'indice de support pour positionner les règles dans le plan.
   - l'indice de M_{GK} pour traduire le poids de l'arc qui relie la prémisse de la conclusion.

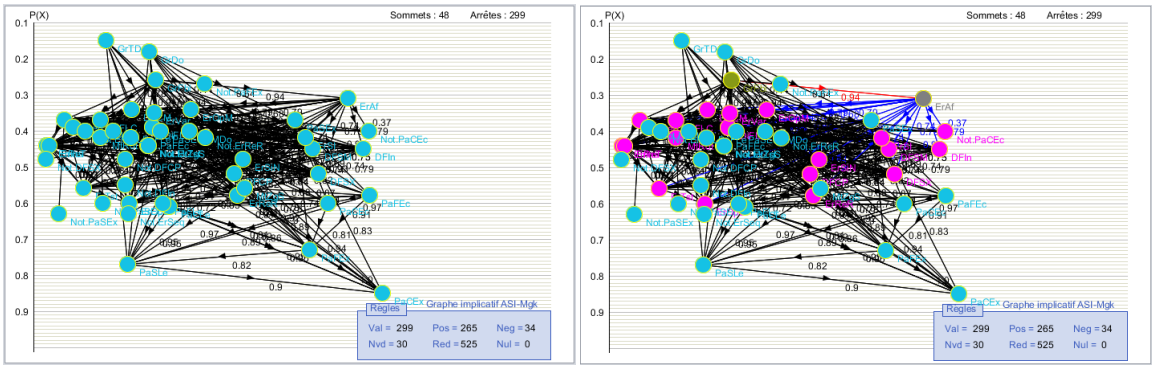


FIGURE 2.14 – Représentation graphique des règles et la transformation des vues via ASI-MGK.

Dans la méthode RF, afin de coder l'indice de règle de manière liée, il est possible de se référer au principe de visualisation de Bertin (Bertin, 1967). En ce qui concerne les variables quantitatives, elles indiquent que la position ou la taille doit être codée efficacement. En particulier, la position est l'information visuelle perceptivement dominante dans une représentation (Bertin, 1967; Wills, 1996; Card et al., 1999) ; elle devrait être utilisée pour les indices les plus importants.

Une autre limitation des méthodes de visualisation abordées dans (Bertin, 1967) est qu'elles encodent un petit nombre d'index de règles (jusqu'àătrois, généralement deux). Pour la méthode RF, le codage a été effectué via au moins quatre indices : un indice descriptif de l'écart d'équilibre, un indice descriptif du biais d'indépendance, un indice statistique des différences d'équilibre et un indice statistique de l'écart d'indépendance.

2.8.5 Amélioration de la visualisation.

Selon la perception visuelle humaine, la scène dans son ensemble attire plus d'attention que sur les détails (Hascot & Beaudouin-Lafon, 2001) il s'agit de la vue d'ensemble + les détails et la

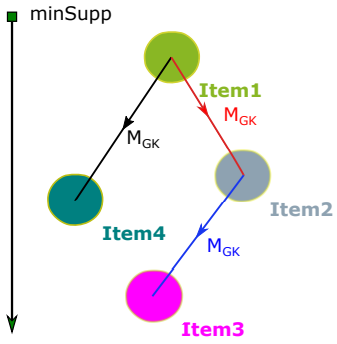


FIGURE 2.15 – Transformation sur la vue par focus.

motivation + le développement de l'arrière-plan. Surtout dans une formule bien connue décrite par Shneiderman, il est considéré comme le mantra de la visualisation d'informations : "Vue d'ensemble d'abord, redimensionnement et filtrage, puis détails à la demande" (Shneiderman, 1996).

Dans l'ASI – MGK, nous avons privilégié à l'utilisateur de pouvoir passer facilement de la vue globale à la vue détaillée en interagissant avec les fonctionnalités existantes pour l'amélioration de la visualisation.

Transformation de vue par le focus de couleur Il s'agit d'affecter des couleurs différents aux nœuds et aux arcs pour améliorer la visualisation.

- La couleur Gris pour le nœud sélectionné.
- La couleur vert à tous les nœuds prédécesseurs ou prémisse vis-à-vis au nœud sélectionné, correspondant a tous les items favorisants l'item en question. (Item1 favorise l'Item2)
- La couleur Rose pour tous les nœuds successeurs ou conséquent par rapport au nœud sélectionné, correspondant a tous les items qui sont favorisé par l'item marqué, Item3 est favorisé par l'Item2.
- La couleur Rouge pour tous les arcs entrant au sommet sélectionné, correspondant à toutes les règles d'association qui favorise l'item marqué.
- La couleur Bleu pour tous les arcs sortant au sommet sélectionné, correspondant à toutes les règles d'association favorisé par l'item marqué.

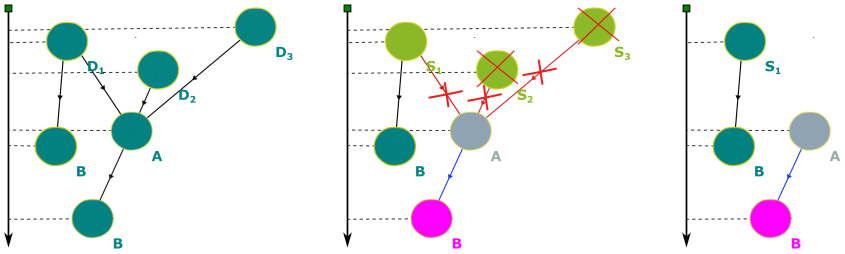


FIGURE 2.16 – Transformation sur la vue par suppression.

Transformation de vue par suppression (Figure 2.16) Cette amélioration consiste à présenter un sous-graphe partiel obtenu après la suppression de tous les arcs entrant à un sommet sélectionné ce qui va entraîner directement des disparition de tous les sommets isolés. Dans, l'illustration présentée dans la figure 2.16, après avoir sélectionné le sommet A, les sommets S_1 S_2 et S_2 dûs

au focus (cf sec. 2.8.5) changent de couleur vert et celui de B en rose, les arcs qui représentent les Règles d'Associations $r_1 : S_1 \longrightarrow A$, $r_2 : S_2 \longrightarrow A$ $r_3 : S_3 \longrightarrow A$, sont entrant en sommet A. Cette fonctionnalité va provoquer la suppression directe de ces trois arcs et va entrainer par la suite la disparition des sommets S_2 et S_3. Comme, le sous-graphe partiel obtenu après la suppression n'est plus connexe, le sommet S_1 regagne sa couleur par défaut.

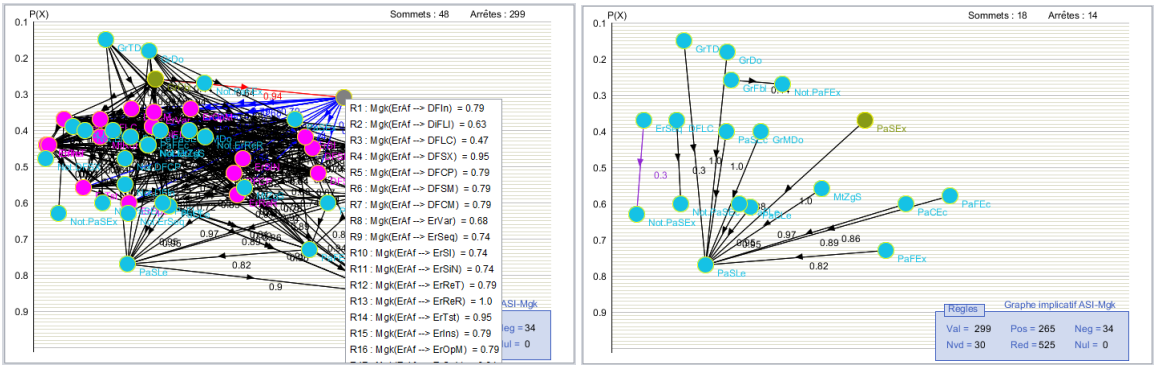


FIGURE 2.17 – Réprésentation de vue d'ensemble et détails à la demande et l'amélioration de vue par suppression avec ASI-MGK.

Transformation de vue par deplacement A l'issue de cette fonctionnalité, l'utilisateur interagit dynamiquement aux positions des sommets sur le graphe, en vue d'obtenir une visibilité plus claire. L'outil ASI – MGK implémente deux modes de deplacement :

1. Deplacement manuel ; c'est-à-dire, l'utilisateur pourrait arranger la disposition des sommets voulus en les déplaçant un à un, mais seulement suivant l'axe horizontal.
2. Deplacement automatique ; en sélectionnant sur un sommet, le programme optimise les positions de tous les sommets prédécesseurs du sommet sélectionné sur l'axe horizontal.

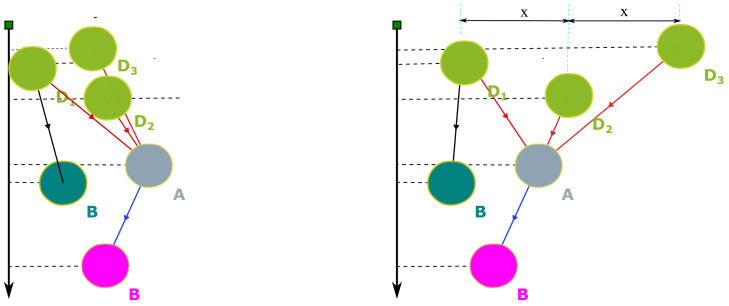


FIGURE 2.18 – Transformation sur la vue par deplacement.

La figure 2.18 montre le mécanisme de deplacement automatique implémenté dans l'ASI-MGK ; après avoir cliqué sur le sommet A, le programme va déplacer les sommets D_1, D_2 et D_3 linéairement equiréparti sur l'axe horizontal avec une distance x.

La figure 2.19 montre les deux graphes implicatifs comprenant chacun 11 sommets et 12 arêtes, c'est-à-dire le même nombre de motifs et le même nombre de règles générées, mais avec des niveaux de complexité différents. A gauche se trouve le graphique directement sorti par l'outil ASI – MGK et à droite, celui obtenu par l'application de l'amélioration de la vue par deplacement.

Enfin, la figure 2.20 illustre les deux graphiques implicatifs extraits de l'application ASI – MGK du même ensemble de données et avec le même seuil de support minimum, mais avec des seuils

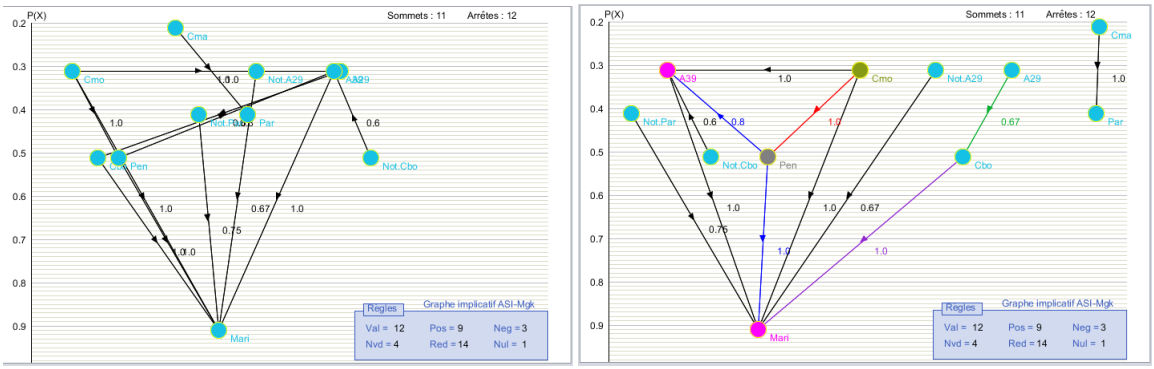


FIGURE 2.19 – Méthode d'amélioration de vue dans ASI-MGK par deplacement.

de confiance différents. A gauche le graphique implicatif de 123 règles formées par 29 motifs est obtenu au seuil de confiance égal à 90%, et à droite nous avons appliqué la méthode d'amélioration de la vue par seuil afin de réduire le nombre de règles générées ; nous avons obtenu un graphique contenant 54 règles générées par 27 éléments et obtenu au niveau de confiance de 95%.

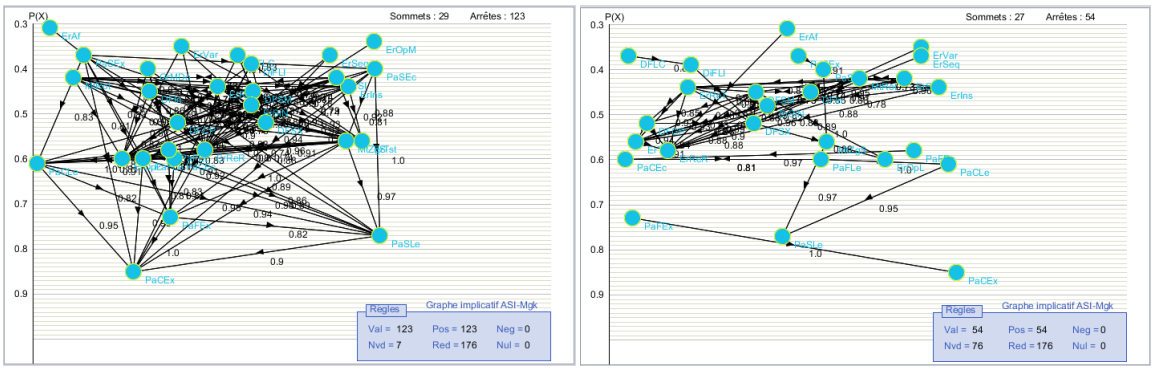


FIGURE 2.20 – Méthode d'amélioration de vue dans ASI-MGK par seuil.

2.9 Conclusion partielle.

Nous avons développé l'outil ASI – MGK, un outil d'analyse statistique implicative basé sur la mesure de la qualité des règles d'association M_{GK}, inspiré du CHIC qui est basé sur l'indice d'implication statistique de Gras.

Cette nouvelle approche d'extraction de règles valides introduites dans cette section est déjà intégrée dans l'outil ASI – MGK [16], a repoussé la limite de sorte que la plupart les outils n'extraient que des règles positives. Nous avons montré sa capacité à éliminer les règles non informatives et son intérêt pour l'analyse graphique à l'aide d'un graphe implicatif des règles d'association valides. Les résultats obtenus sont préliminaires mais confirment que l'approche proposée est prometteuse et peut être utilisée dans diverses recherches dans le domaine de l'analyse et de l'exploration de données.

Dans le chapitre chap.4.9, nous utiliserons cette application pour les données réelles collectées durant la partie de nos recherches sur la didactique de l'informatique dont il est question à partir du chapitre qui suit.

16. Analyse statistique implicative basée sur la mesure de la règle d'association M_{GK}

Deuxième partie

DIDACTIQUE L'INFORMATIQUE

Chapitre 3

Etat des connaissances dans le système informatique des collèges malgaches

Sommaire

Dans ce chapitre, après avoir présenté rapidement ce que recouvre le terme informatique nous détaillerons les différents facteurs qui, d'après nous, peuvent agir et interagir sur l'enseignement de l'informatique au collège.

3.1 Informatique

3.1.1 Étymologie

En 1957, le terme « **Informatik** » est créé par l'ingénieur allemand Karl Steinbuch dans son essai intitulé « *Informatik : Automatische Informationsverarbeitung* », pouvant être rendu en français par « *Informatique : traitement automatique de l'information* ».
Alors qu'aux USA, on véhicule l'appellation de « computer science » jusqu'au début des années 2000, où les Américains intègrent le mot « informatics » plus adapté.

En mars 1962, le terme « **Informatique** » est utilisé pour la première fois, en France, par Philippe Dreyfus, ancien directeur du Centre national de calcul électronique de Bull, pour son entreprise Société d'Informatique Appliquée (SIA). Ce néologisme est formé par la combinaison du terme « **informatio**n », réduit à « **infor** », et du terme « **automatique** », réduit à « **matique** »

Le même mois, Walter Bauer inaugure la société américaine Informatics Inc., qui dépose son nom et poursuit toutes les universités qui utilisent ce mot pour décrire la nouvelle discipline, les forçant à se rabattre sur computer science, bien que les diplômés qu'elles forment soient pour la plupart des praticiens de l'informatique.

Pour le Centre National de ressources textuelles et lexicales français lié au CNRS, « L'informatique est la Science du traitement rationnel, notamment par machines automatiques, de l'information considérée comme le support des connaissances humaines et des communications dans les domaines technique, économique et social » (B.O.E.N., 26 févr. 1981, no8).

L'informatique est un domaine d'activité scientifique, technique et industrielle qui se préoccupe du traitement automatique de l'information via l'exécution de programmes informatiques par des machines que les ordinateurs, les systèmes embarqués, les robots, les automates, etc.

Cette définition officielle de l'informatique montre que c'est la science de tous les traitements effectifs applicables à des données discrètes et le Petit Larousse Illustré l'explique comme « la science du traitement automatique et rationnel de l'information en tant que support des connaissances et des communications » ainsi que « l'ensemble des applications de cette science, mettant en œuvre des matériels (ordinateurs) et des logiciels ».

Dans l'absolu, si on se réfère à la définition officielle qui se rapporte au traitement de l'information par un système, on pourrait faire de l'informatique avec n'importe quel système se comportant comme un circuit logique : machines mécaniques (comme par exemple la Pascaline de Blaise Pascal en 1642 ou les automates), machines pneumatiques, ou hydrauliques ⋯ les premiers programmes permettant de traiter de l'information datent même d'avant l'invention de l'ordinateur comme celui de Ada Lovelace en 1843 qui est à l'origine de la première boucle conditionnelle, véritable concept informatique. Mais dans les faits, seule l'électronique numérique permet d'avoir une puissance de calcul accessible quant au cout financier, à la place occupée par la machine, aux ressources nécessaires à la fabrication et au fonctionnement, à la durée de vie de la machine, à la

rapidité du traitement et sa souplesse d'utilisation···

La séparation du concept informatique par rapport au support informatique reste donc très abstraite. De ce fait l'informatique recouvre trois domaines :

1. De manière officielle, l'informatique désigne l'automatisation du traitement de l'information par un système, concret (machine) ou abstrait
2. L'acception courante, l'informatique désigne l'ensemble des sciences et techniques en rapport avec le traitement de l'information.
3. Et dans le parler populaire, l'informatique peut aussi désigner ce qui se rapporte au matériel informatique (l'électronique), et la bureautique.

Cette dyarchie voire cette triarchie qui existe dans l'acception du mot informatique nous posera plusieurs problèmes au niveau de l'enseignement. Il est évident que pour les jeunes l'informatique est le moyen de communiquer avec les copains/copines sur les réseaux sociaux, tandis que pour le professeur, souvent de mathématiques, l'informatique est d'abord la Science du traitement rationnel de l'information.

Il est prudent d'abord de donner quelques définitions, très utilisées en Informatique.

3.1.2 Définitions

Un ordinateur est une machine électronique qui fonctionne par la lecture séquentielle d'un ensemble d'instructions, organisées en programmes, qui lui font exécuter des opérations logiques et arithmétiques sur des chiffres binaires. Dès sa mise sous tension, un ordinateur exécute, l'une après l'autre, des instructions qui lui font lire, manipuler, puis réécrire un ensemble de données. Des tests et des sauts conditionnels permettent de changer d'instruction suivante, et donc d'agir différemment en fonction des données ou des nécessités du moment.
Un système embarqué est défini comme un système électronique et informatique autonome, souvent à temps réel, spécialisé dans une tâche bien précise. Le terme désigne aussi bien le matériel informatique que le logiciel utilisé. Ses ressources sont généralement limitées. Cette limitation est généralement d'ordre spatial (encombrement réduit) et énergétique (consommation restreinte).
Un robot est un dispositif mécatronique (alliant mécanique, électronique et informatique) accomplissant automatiquement, soit des tâches qui sont généralement dangereuses, pénibles, répétitives ou impossibles pour les humains, soit des tâches plus simples, mais en les réalisant mieux que ce que ferait un être humain.
Un automate est un dispositif se comportant de manière automatique, c'est-à-dire sans l'intervention d'un humain. Ce comportement peut être figé, le système fera toujours la même chose, ou bien peut s'adapter à son environnement.

Remarque 7 (informatique) *Dans le vocabulaire universitaire américain, il désigne surtout l'**informatique théorique** : un ensemble de sciences formelles qui ont pour objet d'étude la notion d'information et des procédés de traitement automatique de celle-ci, l'algorithmique. Par extension, la mise en application de méthodes informatiques peut concerner des problématiques annexes telles que le traitement du signal, la calculabilité ou la théorie de l'information.*

« La science informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes. » Fellows & Parberry (1993)

La science informatique est une science formelle dont l'objet d'étude est le calcul au sens large, c'est-à-dire non pas exclusivement arithmétique, mais en rapport avec tous types d'information que l'on peut représenter de manière symbolique par une suite de nombres.

3.1.3 L'informatique dans les programmes scolaires malgaches.

L'enseignement du premier degré : défini comme la préparation à la vie scolaire. Premièrement, le préscolaire qui est un enseignement facultatif est dispensé dans des écoles de la maternelle ; il est attribué aux enfants de 2 à 6 ans pour leur enseigner ce qu'est la vie en collectivité (socialisation de l'enfant). Deuxièmement, l'enseignement primaire élémentaire est dispensé aux enfants de 6 à 11 ans afin de les induire vraiment dans l'apprentissage des logiques de base de la vie courante. Il est reparti en cours préparatoire, cours élémentaire et cours moyen. Il est caractérisé par l'obtention du diplôme de CEPE : le certificat de fin d'études dans l'enseignement du premier degré et une transition vers l'enseignement du second degré. Dans cette section aucun enseignement formel de l'informatique nest présent.

L'enseignement du second degré comprend deux cycles. Le premier cycle, subdivisé en deux cycles : cycle d'observation (6e et 5e) et cycle d'orientation (4e et 3e) intègrent les élèves de 11 à 15 ans. À la fin de la classe de 3e se passe un examen (le BEPC) permettant d'entrer dans la classe de seconde.

Le Ministère de l'Éducation nationale malgache n'a pas encore élaboré un programme ni même un projet d'introduction de la science informatique et/ou la notion d'algorithmique dans l'enseignement. La numérisation Informatisation par des formations des responsables administratifs et par des dotations en équipements informatiques par le projet de ministère de l'Éducation nationale avec comme partenaire le ministère de la Télécommunication.

Certains établissements publics et/ou privés ont introduit dans leurs programmes une discipline à caractère informatique qui s'est basée encore essentiellement sur l'apprentissage des logiciels bureautiques présents dans la suite Microsoft Office.

Le second cycle intègre les élèves de 15 à 18 ans. Il est plus une orientation vers l'étude supérieure. À la fin du second cycle, l'élève passe à l'examen du baccalauréat dont le diplôme est nécessaire pour accéder à l'enseignement supérieur.

Depuis l'année 2018, dans le nouveau programme du Ministère tutelle, l'enseignement des Technologies de l'information et de la communication pour l'enseignement (TICE) et de l'algorithme s'est apparu en classe de seconde et poursuivi en classe de première pour quelques lycées publics pilotes.

3.1.4 L'enseignement de l'informatique.

L'informatique : discipline scolaire.

L'informatique en tant que discipline scolaire est une matière relativement jeune. En effet, même si ses origines remontent au temps antique avec l'invention des calculs, de la logique, de l'écriture, du codage et que son histoire s'est poursuivie à travers les siècles avec le développement des mathématiques, la mécanisation du calcul, l'émergence de l'automatisme et de l'électronique, elle n'a commencé à prendre forme en tant que discipline indépendante que dans les années soixante du siècle dernier. Elle a été reconnue en tant que champ disciplinaire par l'Académie française en 1966 et définie comme « science du traitement rationnel, notamment par des machines automatiques, de l'information considérée comme le support des connaissances humaines et des communications dans les domaines techniques, économiques et sociaux »(Baron, 2001).

Actuellement, ce statut de l'informatique « discipline scolaire » est admis par la quasi-totalité de tous les systèmes éducatifs dans le monde entier. Cependant, le débat sur les savoirs et les compétences informatiques que les jeunes d'aujourd'hui devront apprendre à l'école est toujours d'actualité (Diethelm et al., 2013; Seehorn et al., 2011a). Ainsi, même si la nécessité d'introduire la discipline informatique dans le cursus scolaire fait l'unanimité, les approches adoptées et les expériences menées dépendantes du contexte social et/ou culturel de chaque pays diffèrent d'un pays à l'autre (Hubwieser et al., 2011; Sturman & Sizmur, 2012; Diethelm et al., 2013). Ces différences se traduisent par une multiplicité de conceptions sur les domaines de l'informatique que l'école doit dispenser et par une grande variété de démarches d'implémentation. Ainsi, la divergence apparait déjà au niveau de la nomenclature et plus précisément au niveau de l'arsenal vocabulaire désignant la discipline scolaire informatique. En fait, le survol bibliographique effectué a permis de relever plusieurs intitulés de cette discipline (tableau 3.1).

La lecture des attributs relatifs à ces différents intitulés reflète le fait que la discipline oscille entre : le technique et le théorique. Outre l'intitulé, les contenus de la discipline informatique et les démarches de leur intégration dans le cursus éducatif suscitent encore le débat et bien qu'ils soient communs, ils manifestent quelques divergences.

Intitulé	Définition
Les Technologies d'Information et de Communication (TIC).	Terme global désignant une gamme complète d'outils électroniques utilisés pour transmettre, traiter, stocker, créer, afficher, partager et échanger les informations par voie électronique. Il s'agit d'une large définition englobant des technologies variées (Anderson, 2010)
Informatique, science informatique « Informatics, Computer Science »	Discipline qui englobe l'étude des ordinateurs et des processus algorithmiques, leurs principes, leurs conceptions matérielles et logicielles, leurs applications et leur impact sur la société (Seehorn et al., 2011a; Wilson & ACMCS, 2010)
Technologie éducative « Educational Technology »	Derrière ce vocable, on met l'utilisation des outils informatiques (matériels et logiciels) pour faire progresser l'apprentissage des élèves dans d'autres disciplines (Seehorn et al., 2011a; Cheung & Slavin, 2013). Par exemple, le professeur de sciences physiques peut recourir à des simulations informatiques préexistantes pour que les apprenants comprennent mieux des principes spécifiques des sciences physiques.
L'alphabétisation et la maîtrise numérique « Digital literacy and fluency »	représentent un ensemble de curriculums allant de l'alphabétisation et de la simple utilisation des outils technologiques à la maîtrise et la capacité d'exprimer des idées de façon créative via ces outils (Pernia, 2008; Seehorn et al., 2011a).

TABLEAU 3.1 – Différents intitulés de la discipline scolaire informatique

3.2 Didactique de l'informatique (D.I.) : discipline autonome.

À propos de quels types de situations d'enseignement de l'informatique peut-on se poser des problèmes de didactique ? Où cela prend-il du sens de travailler la D.I. ?
Ce sont des questions qui se posent à tout enseignant devant enseigner l'informatique.

Le survol des multiples perspectives d'enseignement de l'informatique oriente la réponse à ces questions, sans trancher. L'informatique éducative, *sensu stricto*, est bien représentée par le « Logo project » : l'outil informatique (dont la programmation) apparait comme un moyen de développer d'autres acquisitions, de découvrir des compétences diverses, générales des élèves. L'enseignement n'est pas tourné vers l'appropriation de contenus informatiques précis.

L'information « *outil* » d'acquis disciplinaires spécifiques est utilisée en mathématiques pour opérationnaliser des concepts de façon différente. La programmation en Prolog opérationnalise des concepts de logique, celle en « Logo » des concepts de géométrie « intrinsèque » etc. ⋯ La proximité conceptuelle des objets en jeu ouvre la question des acquisitions « parallèles » en informatique.

L'informatique comme technologie (objet) est le parent pauvre des études en France : orientation didactique et formations peu structurées au collège, peu d'échos didactiques sur l'enseignement dans les lycées techniques (programmation d'automates, informatique en section F). Quelques travaux considèrent cependant le long terme de cet enseignement : à Gênes en Italie, au Mans en France avec (Bruillard & Vivet, 1994) sur la programmation (LOGO) de robots.

L'alphabétisation informatique (« objet » et « *outil* ») est aujourd'hui structurée dans les lycées en France en tant que discipline scolaire (mais elle est optionnelle) : il existe des programmes nationaux, des enseignants identifiés, des manuels, une épreuve au bac. L'enjeu est de faire acquérir aux élèves des concepts informatiques (en particulier, mais pas seulement, en programmation) et de leur permettre de se construire des représentations de l'informatique compatibles avec l'activité professionnelle informatique et l'utilisation de l'informatique comme citoyen.

L'initiation pré professionnelle informatique concerne les formations techniques au lycée, et surtout celles du premier cycle universitaire scientifique. Elle vise à la fois des acquis en informatique et la maîtrise d'un outil pour travailler dans les domaines d'engagement professionnel ultérieur (mathématiques, sciences physiques, gestion ...).

La formation professionnelle a deux visées : produire des concepts, des méthodes, des outils informatiques nouveaux, former à l'utilisation de l'informatique pour résoudre des problèmes professionnels particuliers (production, gestion, information ...). Les attentes sont différentes en ce qui concerne les acquis conceptuels.

3.3 Émergence de l'enseignement de l'informatique.

Dans leur souci de promouvoir la discipline scolaire « informatique », la renforcer dans le cursus scolaire et lui conférer le statut d'une science au même niveau que les mathématiques, les sciences physiques et la chimie (Seehorn et al., 2011a), plusieurs pays ont procédé ou ont commencé à la rénovation de leurs curriculums. Nous relatons ci-dessous des exemples d'expériences décrivant l'enseignement de l'informatique dans un certain nombre de pays. Le choix a porté sur des pays industrialisés et réputés pour leur développement économique et technologique tels que : l'Angleterre, les États-Unis, la France et le Japon. On présente également le cas de la Pologne, qui suite à des réformes menées à partir de 2005, a été constaté une progression nette chez les jeunes Polonais. En effet, ces derniers obtiennent régulièrement de bons résultats lors des compétitions IOI [1] (International Olympiad in Informatics) depuis la réforme. Nous donnons aussi un bref aperçu sur l'enseignement de l'informatique en Tunisie vue la convergence avec le Maroc au niveau économique et éducatif.

a) En Angleterre.

Les demandes de renouvèlement de programmes informatiques se sont multipliées pour faire de l'informatique une discipline. Ces appels réclamaient une discipline stricte et contraignante dans le programme d'études et ont donc le même statut que les autres disciplines, à savoir : les mathématiques, la physique [2] (Livingstone & Hope, 2011). Ainsi en septembre 2013, de nouveaux programmes d'enseignement ont été publiés [3]. Ces programmes présentaient la tendance actuelle pour l'enseignement de l'informatique au primaire et au collège : en plus de l'informatique outil, enseigner ; la pensée informatique, donner plus d'importance à l'enseignement basé sur les jeux

1. http://www.ioinformatics.org/www.frantice.netfrantice.net,numéro11,décembre2015
2. http://www.computingatschool.org.uk/index.php?id=documents
3. https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study

vidéo [4], [5] et d'introduire des environnements visuels comme les environnements Scratch (Wilson et al., 2013) et Kodu (Stolee & Fristoe, 2011) qui rendent la programmation amusante et accessible aux jeunes enfants (Dagiene et al., 2013).

b) Aux États-Unis.

Malgré le fait que la« *the Computer Science Teachers Association* » (CSTA) et « *the Association of Computing Machinery* » (L'ACM) aient développé « l'ACM Model Curriculum for K12 Computer Science Education » publié en 2003 et révisé en 2011, plusieurs rapports et recherches (Seehorn et al., 2011a,b; Wilson & ACMCS, 2010; Lang et al., 2013) ont précisé que l'enseignement de la science informatique est en crise. Parmi les problèmes évoqués dans ces rapports, on note le manque dans les certifications des professeurs dans la science informatique (Lang et al., 2013). En effet, plusieurs États ne disposent pas de normes standards pour l'enseignement de la science informatique au secondaire, et même quand elles existent, ces normes confondent souvent l'informatique en tant que science et l'informatique en tant qu'outil (Seehorn et al., 2011a).

Il faut noter que ces dernières années, pour encourager les élèves à apprendre la science informatique plusieurs efforts ont été menés. On peut citer le projet « *code.org* » qui invite des millions d'élèves à suivre une heure de formation d'initiation à l'algorithmique et à la programmation. On note également l'élaboration de projets axés sur la création des jeux, animations et histoires avec les environnements de programmations comme Scratch [6] et Alice [7]. Enfin, les recommandations, les appels pour réformer et d'innover dans l'enseignement de la science informatique du primaire au secondaire aux États-Unis afin de s'adapter aux besoins technologiques et économiques du pays, ont été récemment soutenus par le gouvernement américain [8], [9].

c) En France.

Les tentatives d'introduction de l'informatique dans l'enseignement français remontent aux années 1960. Plusieurs expériences d'intégration ont été menées principalement dans les domaines de l'enseignement supérieur et de l'ingénierie, alors que les expériences, en général, étaient isolées et moins répandues (Pélissier, 2002). Ce n'est qu'au début de 1970, que les ordinateurs ont été réellement introduits dans l'enseignement général français (Baudé, 2014).

Il convient de noter que les expériences et les initiatives menées concernent principalement l'utilisation d'outils informatiques dans d'autres disciplines (Baudé, 2010), par exemple l'expérience de « 58 lycées » entre 1972 et 1976 et le projet consistant à équiper les collèges de 10 000 ordinateurs par an (Pélissier, 2002). En 1981, la deuxième option informatique expérimentale a été mise en place dans douze lycées. Cette option est généralisée depuis 1985, mais supprimée au cours de la réforme de 1989 (Baudé, 2010).

4. http://www.creativefront.org/news/education-secretary-unveils-new-plan-to-use-computer-games-in-schools
5. http://www.bbc.com/news/technology-29550486
6. https://scratch.mit.edu/
7. http://www.alice.org/index.php
8. https://www.youtube.com/watch?feature=player_embedded&v=kp_zigxMS-Y
9. http://www.gamesindustry.biz/articles/2013-02-19-obama-games-can-make-education-relevant-for-young-people

En effet, le système éducatif français s'est engagé à mettre en place des attestations et ou certificats de compétences obligatoires à acquérir après certaines périodes du cours. Pour cette raison, des certificats ont été délivrés indiquant le contrôle progressif des Technologies de l'Information et de la Communication (TIC), par exemple, ce sont le brevet informatique et Internet (B2i) et le Certificat Informatique et Internet (C2i).

- Le B2i atteste que le titulaire est en mesure de s'approprier un environnement informatique de travail, adopter une attitude responsable, créer, produire, traiter et exploiter des données, s'informer et se documenter, communiquer et échanger.
- Le C2i créé en 2002 pour renforcer et valider les acquis des étudiants en TIC est une certification française délivrée par les établissements supérieurs français. Un étudiant ayant à son actif le C2i doit en principe être capable de mener correctement les activités exigées aujourd'hui par un cursus de l'enseignement supérieur et notamment : recherche, création, manipulation, gestion de l'information, récupération et traitement des données, gestion de données

Depuis le début de l'année scolaire 2012, l'informatique est proposée en enseignement dans la spécialité « Informatique et sciences du numérique » (ISN) pour les élèves de terminale série S, à partir de la rentrée 2013 son enseignement est en option pour les autres filières de terminales (Drot-Delange, 2013).

L'étude des documents officiels, des travaux de recherches et des publications en particulier ceux de la revue EPI [10] montrent que l'informatique en France apparait dans un double perspective : une perspective pédagogique qui renvoie à l'utilisation de l'informatique pour des objectifs d'apprentissage et une autre curriculaire qui vise à placer l'informatique en tant que discipline scolaire (Texier, 2002; Baudé, 2013b). Récemment le débat sur l'enseignement de l'informatique, discipline au même titre que la physique, la chimie, les mathématiques et les sciences de la vie et de la terre, devient très vif (Baudé, 2013b,a). L'Académie des sciences, dans son rapport [11] de mai 2013 mentionnait que « *l'enseignement général de l'informatique devra d'abord donner à tous les citoyens les clés du monde du futur, qui sera encore bien plus numérique et donc informatisé que ne l'est le monde actuel, afin qu'ils le comprennent et puissent participer en conscience à ses choix et à son évolution plutôt que de le subir en se contentant de consommer ce qui est fait et décidé ailleurs* ». Dans ce contexte la tendance actuelle du gouvernement vise à rendre obligatoire l'enseignement du codage informatique dès le primaire [12].

En dehors du système scolaire, plusieurs initiatives ayant pour objectifs de sensibiliser et former les jeunes à l'informatique (Drot-Delange, 2013) ont vu le jour. On peut citer par exemple le concours Castor [13] qui permet d'aborder l'informatique de façon ludique. Ce concours, actuellement organisé dans 40 pays, est considéré par les chercheurs comme un excellent moyen pour promouvoir l'enseignement de l'informatique « science » (Tort, 2011; Tort & Dagiene, 2012). L'édition française de l'année 2015 a connu la plus grande participation [14] : plus de 344.976 élèves (dont

10. http://www.epi.asso.fr/
11. http://www.academie-sciences.fr/activite/rapport/rads_0513.pdf
12. http://www.assemblee-nationale.fr/14/propositions/pion2022.asp
13. http://castor-informatique.fr/
14. http://www.bebras.org/?q=countries

49% étaient des filles) issus de 2286 établissements à travers le pays.

d) Au Japon.

En avril 2012, le Japon a lancé un nouveau programme d'études pour l'enseignement de l'informatique « information studies education » dans les écoles primaires et au collège. Il a été généralisé aux écoles secondaires en avril 2013 [15]. Selon ce programme, au primaire, l'accent est mis sur les TIC et l'usage des dispositifs informatiques à travers des activités intégrées pendant la période d'étude intégrée « the Period for Integrated Studies ». Au collège, on note également l'absence de matière spécifique d'informatique. En revanche, l'enseignement des technologies d'information fait partie de la matière « *Technology and Home Economics generally* » qui couvre l'internet, la conception et la création de produits numériques, ainsi que la mesure et le contrôle par des programmes informatiques. Au secondaire l'informatique est enseignée soit comme :

- Sujet général (*General information*) : l'informatique est perçue comme outil dans la matière « *Society and Information* ». Les bases de la programmation sont timidement abordées dans la discipline « *Science of Information* » ;
- Sujet principal (*Major information*) : en plus des notions de base, on donne une grande importance à la résolution des problèmes et à la science informatique. Il faut noter, cependant, que ce type d'enseignement s'effectue dans un nombre limité d'établissements.

Pour ce, (Nakano & Izutsu, 2013) proposent de mettre plus l'accent sur la compréhension scientifique de l'information et ainsi généraliser l'enseignement de l'informatique en tant que sujet principal au secondaire dans les prochaines réformes.

e) En Pologne.

L'informatique « *informatyka* » [16] et les TIC existent depuis 1985 comme une matière de formation en Pologne (Gańko-Karwowska, 2015; Syslo & Kwiatkowska, 2008). À la fin de 2008, la réforme du système éducatif polonais était en faveur de l'enseignement de l'informatique, elle a apporté des améliorations importantes dans les programmes pour le primaire, collège et lycée [17]. Le système éducatif polonais donne une grande importance à l'informatique comme science (Sysło & Kwiatkowska, 2013). Ainsi, depuis plusieurs années des activités de sensibilisation à l'intérêt de l'algorithmique et la programmation ont été réalisées (Sysło, 2011). La Pologne organise le concours castor informatique « *bebras* » [18]. Comme résultats de ces efforts les jeunes Polonais sont bien classés dans les olympiades d'informatique. En effet, la Pologne est classée en 2éme position au niveau du nombre de médaillés avec un total de 97 honorés après la Chine (107 honorés) [19].

f) En Tunisie.

Le système éducatif tunisien a introduit la discipline informatique comme matière optionnelle dans les deux dernières années du secondaire scientifique et économique (3éme année et 4éme

15. http://www.mext.go.jp/a_menu/shotou/new-cs/youryou/index.htm
16. Le terme (pl. informatyka) désigne la science informatique
17. http://www.men.gov.pl/index.php/2013-08-03-12-10-01/podstawa-programowa/
18. http://www.bobr.edu.pl/
19. http://stats.ioinformatics.org/countries/,consultéle20décembre2015

année) en 1992 et au collège en 2003. Après la réforme de 2005, elle est devenue obligatoire dans toutes les sections du lycée et du collège. Il a même été instauré la spécialité « Informatique » pour laquelle l'élève peut opter à partir de la 2éme année secondaire (Trabelsi, 2010). Selon le curriculum, au secondaire [20] le contenu dispensé, tout en étant très varié, dépend de la nature de la filière. En revanche, le programme de la matière informatique au collège [21] consiste en l'enseignement de l'informatique outil. Au primaire, bien que l'informatique ne soit pas programmée, selon les moyens des établissements, un module TIC est intégré dans « l'éducation technologique » depuis le niveau de la 3éme année. Enfin, parmi les recommandations actuelles, on note l'appel à la généralisation de l'enseignement des TIC aux six années du primaire et à la distinction entre l'informatique outil et l'informatique technique. À l'issue de l'étude des différents curriculums de l'enseignement de l'informatique ci-dessus, on peut distinguer l'informatique outil et l'informatique science :

- L'informatique outil : consiste au simple usage des technologies numériques qui va de l'alphabétisation « **Digital literacy** » à la maîtrise numérique « Digital fluency ».
- L'informatique science : Il s'agit de l'identification des principes de fonctionnement et de conception d'un système informatique, de la résolution des problèmes et de la pensée.

3.4 Enseignants en informatique à Madagascar.

À Madagascar, un aspect crucial de l'introduction de l'informatique à l'école est la nécessité de former le corps enseignant : on parle à la fois d'évolution du programme de formation initiale, de recrutement de formateurs *ad hoc*, de formation continue proposée aux enseignants sur un critère de « motivation », etc.

3.4.1 Les enseignants en mathématiques sont réticents

Indépendamment de leur nationalité, les enseignants mentionnent des contraintes et difficultés dans leur introduction de l'algorithmique qui sont habituellement celles soulevées lors des discours autour de l'intégration des TIC : augmentation du temps de préparation, manque de formation, perte de temps dans la mise en place du matériel informatique, notions trop consommatrices en temps au détriment des autres notions à enseigner, manque de matériel. À cela, s'ajoutent aussi des difficultés nouvelles liées aux conceptualisations des élèves en algorithmique face auxquelles les enseignants apportent des réponses communes.

Enfin, certains enseignants affirment travailler dans des conditions matérielles difficiles. Le manque d'ordinateurs en nombre suffisant dans les salles informatiques, la non-disponibilité de la salle et les pannes techniques non réparées, est les principaux problèmes mis en évidence.

À côté de ces difficultés usuellement mentionnées dans les discours sur l'intégration des technologies, des difficultés propres à l'algorithmique sont mentionnées : celles conceptuelles des élèves et plus particulièrement les variables et leur affectation. Reliant l'algorithmique à l'informa-

20. http://www.edunet.tn/ressources/pedagogie/programmes/nouveaux_programme2011/secondaire/info.pdf

21. www.edunet.tn/ressources/pedagogie/programmes/nouveaux_programme2011/preparatoire/technologie/info_college.pdf

tique, les enseignants ne disposent pas de formation requise pour l'enseigner « l'algorithmique » et ils sont totalement décalés par rapport à leur formation initiale. Ainsi ils manquent de compétente en algorithmique puisque la formation qu'ils ont est une formation d'autodidacte et pendant leur parcours de formation il n'y avait pas de cours d'informatique.

Les enseignants ressentent l'introduction de l'algorithmique comme un domaine en tension entre deux pôles distants non connectés, les mathématiques et l'informatique. Cette distance génère des difficultés d'enseignement face auxquelles ils opèrent des choix dans leurs pratiques.

3.4.2 Les enseignants en technologie

Comme cette discipline est nouvelle, il existe encore très peu d'établissements à vocation pédagogique de formation initiale des enseignants qui disposent des parcours d'informatique. En particulier, pour le cas de l'ENSET, c'est encore quasiment sa première promotion qui est sortie l'année 2018.

Ainsi en résumé, l'immense majorité des professeurs des écoles ont une vision très confuse de l'informatique. La raison en est simple : c'est une discipline qu'ils n'ont pour la plupart jamais abordée en tant que telle. Par suite, ils en ont une perception par défaut qui est celle de l'utilisateur (d'un ordinateur, disposent de logiciels de bureautique, etc.). Ceci ne permet pas de comprendre ce qu'est la discipline informatique ou la « pensée informatique » et, pire, crée des confusions. L'informatique, c'est souvent un domaine mal défini mêlant « savoir utiliser un ordinateur » et « savoir comment marche un ordinateur », et dont on se méfie, car on n'y connait/comprend pas grand-chose, d'autant que « souvent, ça ne marche pas ». Quant à « l'informatique à l'école », c'est faire utiliser aux élèves un programme existant (p. ex., un jeu a vocation éducative) ou un outil de bureautique. La notion de « pensée informatique » et même souvent d'algorithme est inconnue. Ce qui tombe bien, car si, de façon générale, « faire de l'informatique » génère une certaine appréhension, alors s'il s'agit de construire des programmes et pas simplement de les utiliser.

Pour enseigner l'informatique à l'école, il faut, comme pour tout enseignement :

1. Disposer ou construire une certaine conceptualisation du domaine considéré et des enjeux d'apprentissage (identifier, définir, dissocier, les notions/compétences en jeu).
2. Identifier les objectifs pédagogiques que l'on se propose de considérer.
3. Maîtriser les notions/compétences en jeu et les moyens de les faire travailler par les élèves (types d'exercices, etc.). Ce qui inclut, si l'on se propose de faire travailler les élèves sur la construction de programmes, de savoir utiliser le moyen « langage de programmation » (comme on utilise le moyen « calculatrice » dans certains exercices de maths, ou le moyen « dictionnaire » dans certains exercices de français.

Alors, sans attendre les résultats d'un plan pluriannuel de formation initiale et continue, un recrutement des enseignants volontaires et compétents (titulaires d'un diplôme universitaire en informatique ou d'une compétence validée institutionnellement) pourrait permettre le démarrage rapide de l'enseignement de l'informatique au collège.

3.4.3 La formation des enseignants

À Madagascar, la situation de l'éducation n'a cessé d'empirer de jour en jour. Nombreux sont les paramètres ayant entrainé cette dégradation. Cela a commencé par le Service national (S.N) que l'État a décrété pour envoyer les nouveaux bacheliers, dépourvus de formation initiale ou professionnelle dans le domaine de l'enseignement, pour venir en aide aux établissements nécessiteux, souvent en brousse.

Est venu ensuite l'avènement du FRAM (Fikambanan'ny Ray Aman-drenin'ny Mpianatra), au début, censé être une solution d'appoint pour les nouvelles écoles créées par des parents qui soucieux de l'éducation de leurs enfants, et se sont donnés la main pour construire ces écoles et en même temps subvenir aux besoins des maîtres (on parle de Maitre FRAM) recrutés en catimini pour faire fonctionner ces établissements scolaires.

Par la suite, la création d'écoles est devenue une sorte de business, alors qu'il manquait cruellement de nouveaux enseignants formés. Le phénomène est devenu national et généralisé, en ce moment, il y a plus d'enseignants FRAM, n'ayant reçu aucune formation initiale, que des normaliens, qui devraient être formés pour la profession et devraient être recrutés dans nos écoles publiques.

Les écoles normales réparties dans tout le pays n'arriveront pas encore combler ce manque, tant la politique sur l'éducation, ou les stratégies d'intervention resteront les mêmes. On loue toutefois les efforts entrepris, en régionalisant l'INFP (Institut National de Formation Pédagogique) pour avoir mis en place des CRINFP un peu partout (actuellement plus d'une vingtaine). La formation des enseignants selon le besoin de l'Éducation nationale demeure la solution pour arriver à bout de leurs peines. À part la formation de ceux qui sont recrutés et devenus actuellement des enseignants-fonctionnaires, qui doivent être une priorité absolue, l'extension ou la ramification des CRINFP est vivement encouragée ainsi que le projet de création des IREMI (Instituts de Recherche sur l'Enseignement des Mathématiques et de l'Informatique) initié par le Directeur de l'équipe d'accueil « Éducation et didactique des mathématiques et de l'informatique ». Tous ces projets ont vocation à remédier à la :

- la désaffection des élèves pour les études de mathématiques et de sciences physiques ;
- l'insuffisance de la formation initiale des enseignants de mathématiques ;
- la formation continue quasiment inexistante ;
- la nécessité de former des doctorants pour renouveler les enseignants-chercheurs de mathématiques.

3.5 Algorithme la pierre angulaire de l'informatique.

La notion d'algorithme remonte à l'antiquité. Cela s'est précisé dans le domaine des mathématiques par l'emploi de variables. L'algorithme au sens informatique apparait avec l'invention de premières machines dotées d'automatismes.

3.5.1 L'algorithme.

Origine du mot. Le mot algorithme vient du nom du mathématicien perse du neuvième siècle (apr. J.-C.) Abu Abdullah Muhammad ibn Musa al-Khwarizmi. Le mot algorithme se référait à l'origine uniquement aux règles d'arithmétique utilisant les chiffres indo-arabes numéraux, mais cela a évolué par la traduction en latin européen du nom Al-Khwarizmi en algorithme au 18ième siècle. L'utilisation du mot a évolué pour inclure toutes les procédures définies pour résoudre un problème ou accomplir une tâche.

L'algèbre à l'origine des variables. Le travail des anciens géomètres grecs, du mathématicien perse Al-Khwarizmi souvent considéré comme le « père de l' algèbre », des mathématiciens chinois et européens de l'est a atteint un point culminant avec Leibniz (1646-1716) dans la notion de « calcul ratiocinateur », une algèbre logique.

Les anciens Euclide (vers 300 av. J.-C). à formaliser un algorithme auquel on a donné son nom. Cet algorithme qui sert à calculer le plus grand diviseur commun (PGCD) est le suivant :

- diviser a par b, on obtient le reste r
- remplacer a par b
- remplacer b par a
- continuer tant que c'est possible, sinon on obtient le PGCD.

L'algorithme d'Archimède (287-212 av. J.-C.) donne une approximation du nombre π. Eratosthènes (3eme sicle av J.-C.) a défini un algorithme pour retrouver les nombres premiers. Averroès (1126-1198) utilise un procédé algorithmique pour ses calculs. Au 12ième siècle Adelard de Bath introduit le mot *algorismus* dérivé de Al Kwarizmi. Symboles, règles, et paradoxes entre les années 1800 et jusqu'au milieu des années 1900 ce furent :

- Boole (1847) invente l'algèbre binaire, la base des ordinateurs. En fait, il a unifié la logique et les calculs dans un symbolisme commun ;
- Frege (1879) introduit le langage des formules, qui est une *lingua characterica*, un langage écrit dans des symboles spéciaux, « pour la pensée pure », sans aucun embellissement rhétorique... construit avec des symboles spécifiques manipulés selon des règles bien définies ;
- Peano (1888) écrit les principes de l'arithmétique, présentés par une nouvelle méthode, qui est la première tentative d'axiomatisation des mathématiques dans un langage symbolique ;
- Alfred North Whitehead et Bertrand Russell dans leur Principia Mathematica (1910-1913) ont à leur tour simplifié et amélioré le travail de Frege ;
- Gödel (1931) cite le paradoxe du menteur qui réduit les règles de récursion entièrement en nombres.

Un algorithme est alors une méthode générale pour résoudre un ensemble de problèmes. Il est correct si la sortie correcte est créée pour chaque instance du problème, c'est-à-dire que le problème est résolu. L'efficacité d'un algorithme est notamment due à son temps de calcul, à sa consommation de mémoire RAM (en supposant que chaque instruction a un temps d'exécution constant), à la précision des résultats obtenus (par exemple, en utilisant des méthodes probabilistes telles que la méthode de Monte-Carlo). Leur évolutivité (leur capacité de parallélisation efficace), etc.

Les ordinateurs qui exécutent ces algorithmes ne sont pas infiniment rapides : le temps machine reste une ressource limitée malgré une augmentation constante des performances de l'ordinateur. On dit donc qu'un algorithme fonctionne bien en conservant les ressources dont il dispose, à savoir le temps processeur et la mémoire RAM, ou, plus récemment, la consommation d'énergie. L'analyse de complexité algorithmique permet de prévoir l'évolution du temps de calcul nécessaire pour compléter un algorithme en fonction de la quantité de données à traiter.

3.5.2 Quelques définitions.

Donald Knuth (1938) éminent logicien, également professeur à l'université Stanford, lista les cinq propriétés suivantes comme étant les prérequis d'un algorithme :

- **La finitude :** « Un algorithme doit toujours se terminer après un nombre fini d'étapes. »
- **Une définition précise :** « Chaque étape d'un algorithme doit être définie précisément, les actions à transposer doivent être spécifiées rigoureusement et sans ambigüité pour chaque cas. »
- **Des entrées :** « ⋯ des quantités qui lui sont données avant qu'un algorithme ne commence. Ces entrées sont prises dans un ensemble d'objets spécifiés. »
- **Des sorties :** « ⋯ des quantités ayant une relation spécifiée avec les entrées. »
- **Un rendement :** « ⋯ toutes les opérations que l'algorithme doit accomplir doivent être suffisamment basiques pour pouvoir être en principe réalisées dans une durée finie par un homme utilisant un papier et un crayon. »

George Boolos (1940-1996), philosophe et mathématicien, proposa la définition suivante :

Définition 60 *Des instructions explicites pour déterminer le énième membre d'un ensemble, pour n un entier arbitrairement grand. De telles instructions sont données de façon bien explicite, sous une forme qui puisse être utilisée par une machine à calculer ou par un humain qui est capable de transposer des opérations très élémentaires en symboles.*

Gérard Berry (1948-), chercheur en science informatique, en donne la définition grand public suivante citée par : Bloch (2016)

Définition 61 *Un algorithme, c'est tout simplement une façon de décrire dans ses moindres détails comment procéder pour faire quelque chose. Il se trouve que beaucoup d'actions mécaniques, toutes probablement, se prêtent bien à une telle décortication. Le but est d'évacuer la pensée du calcul, afin de le rendre exécutable par une machine numérique (ordinateur, ⋯). On ne travaille donc qu'avec un reflet numérique du système réel avec qui l'algorithme interagit.*

Simon Modeste, actuellement enseignant-chercheur à Montpellier et didacticien des mathématiques français, propose, en collaboration avec Ouvrier-Buffet et Gravier (2010), puis seul dans sa thèse (Modeste et al., 2010; Modeste, 2012), d'étudier cette introduction de l'algorithmique dans les cours de mathématiques au lycée. Les travaux sur l'algorithmique du point de vue de la didactique des mathématiques sont relativement nouveaux, car les chercheurs précédents s'intéressaient davantage aux algorithmes en tant que tels. L'intérêt pour l'étude des algorithmes d'un point de vue mathématique réside dans le fait qu'il n'est pas seulement un outil de résolution de

problèmes, il fait également l'objet des mathématiques.

Avant d'aller plus loin, il convient de définir formellement la notion d'algorithme. Après avoir comparé diverses définitions de ce terme, Modeste en 2012 donne la définition suivante :

Définition 62 *Un algorithme est une procédure de résolution de problème, s'appliquant à une famille d'instances du problème et produisant, en un nombre fini d'étapes constructives, effectives, non ambigües et organisées, la réponse au problème pour toute instance de cette famille.*

Et son corolaire.

Corollaire 7 *L'algorithmique est la science qui étudie l'application des algorithmes à l'informatique.*

3.5.3 Différents aspects de l'algorithme.

En comparant les différentes définitions du concept d'algorithme et en plus de divers exemples, elles apportent des considérations importantes concernant les algorithmes au langage. Le plus important dans ces définitions est que l'on retrouve les différents aspects des algorithmes. Ces cinq aspects sont le problème, l'efficacité, la preuve, la complexité et les modèles théoriques. (Modeste, 2012)

L'aspect problème. Un algorithme apporte la réponse à une question posée par un problème. C'est courant parce que cela doit fonctionner pour toutes les instances d'une famille d'instances. Dans cet aspect, nous trouvons également le fait qu'un algorithme doit avoir des entrées et des sorties.

L'aspect effectivité. L'algorithme fonctionne avec un nombre limité d'étapes sur des données limitées. En outre, il doit pouvoir être placé sous la forme d'un programme à implémenter par un ordinateur ou une machine.

L'aspect preuve. Quelle que soit l'instance saisie, l'algorithme doit fournir les résultats attendus pour le problème pour lequel il est conçu. Pour prouver que l'algorithme fournit le résultat correct, on parle de correction. Nous parlons d'un algorithme correct. En outre, les résultats doivent être obtenus en un nombre limité d'étapes. L'algorithme de preuve se termine par un nombre limité d'étapes afin de prouver la terminaison. Bien entendu, il ne suffit pas de tester l'algorithme dans quelques exemples pour voir s'il fournit les résultats attendus en un nombre limité d'étapes.

L'aspect complexité. Le concept de complexité est un concept mathématique unique à un algorithme. Il est calculé en fonction de la taille des données entrées dans l'algorithme. De plus, il peut être mesuré en temps ou en taille de mémoire. L'étude de la complexité algorithmique a pour but de pouvoir classer les différents algorithmes répondant à la même question afin de trouver l'algorithme le plus efficace.

L'aspect modèle théorique. Cet aspect est basé sur un modèle théorique, tel que la machine de Turing ou la fonction récursive de Kleene. Ces différents modèles peuvent être considérés comme des définitions plus théoriques de l'algorithme. Grâce à eux, on peut remarquer que tous les problèmes ne peuvent pas être résolus par un algorithme. Pour ceux-ci, ces modèles peuvent les classer en fonction de leur complexité respective.

3.5.4 L'émergence de l'enseignement de l'algorithmique.

Dans les premiers enseignements de programmation, le mot « *algorithme* » n'est pas encore présent. L'examen des divers rapports du CNRS dans les années 60 permet à Baron de le constater :

Ainsi le rapport national de conjoncture scientifique du CRNS de 1964 ne mentionne le mot « algorithme » qu'à deux reprises, les deux fois en faisant référence au langage Algol qui est présenté comme étant particulièrement adapté : une fois dans le cadre de l'analyse numérique, et une fois sous le titre « langage de programmation ». (Baron, 1987)

Expliquant la naissance du langage avancé d'algo dans les années 1960, Moreau en 1987 montrait que la solution aux problèmes technologiques, en particulier la compilation de langages, a permis de réaliser que les algorithmes sont le sujet de la communication personne-machine. C'est donc l'une des conditions principales pour l'émergence de l'algorithme pour lui :

Jusqu'en 1960, l'effort avait principalement porté sur la mise au point d'une traduction efficace des langages de programmation. [...] Mais, une fois les problèmes de compilation bien maîtrisés, il était naturel que les théoriciens étudient cette fois, comme ils avaient commencé à le faire avec Algol, la forme que devait avoir la définition même d'un langage. C'est pourquoi, à la fin de la période historique, les jalons ont été posés d'une réflexion sur la programmation. Cette réflexion est liée, non plus à la communication entre l'homme et la machine, mais à l'objet même de cette communication, les algorithmes (voir Moreau, 1987, p.188).

Le développement des langages de programmation et la réflexion sur la conception des programmes (preuves, comparaisons, etc.) à la fin des années 1960 ont fait apparaitre une nouvelle branche de l'informatique, à savoir l'algorithmique. La recherche algorithmique conduit à la croissance des applications informatiques dans la société. Moreau décrit cette étape comme cruciale pour l'émergence de l'informatique (par opposition aux mathématiques) :

Or cette génération des applications a fait que, en 1963, l'informatique n'était plus l'apanage des mathématiciens. Une nouvelle science venait de naître avec ses domaines de recherche propres, ses applications propres, et une industrie qui l'entraînait ou la suivait (Moreau, 1987, p. 200)

L'enseignement de l'algorithmique va alors se séparer à l'université de celui des langages de programmation. Ainsi Duchâteau (2002) écrivait :

Elle [La méthode LCP - Logique de Conception des Programmes] a permis de passer d'un enseignement visant essentiellement l'acquisition de connaissances techniques à un enseignement orienté vers l'apprentissage d'une méthode, en mettant en évidence les étapes fondamentales du processus d'élaboration des programmes ; plus généralement la pratique de la méthode LCP a conduit à dissocier au niveau de la formation l'enseignement de l'algorithmique de celui des langages de programmation.

Les jalons historiques que nous venons d'évoquer très rapidement, nous permettent de faire un choix de moments clés pour mieux repérer l'arrivée dans l'enseignement universitaire des objets informatiques fondamentaux comme boucle et variable, les conditions de leur mise en place et les relations qu'ils peuvent entretenir avec les machines et les langages de programmation (par

exemple) :

- **1957** : Premier enseignement de programmation mis en place à l'université Joseph Fourier à Lyon par J. Kuntzmann.
- **1968** : Première édition du volume « Fundamental algorithms » dans la série « The art of computer programming » de Knuth (1968).
- **1978** : Parution d'un ouvrage de référence sur l'algorithmique « Fundamentals of Computer Algorithms » de Horowitz (1978).

3.5.5 Présence universelle des algorithmes.

La Commission de réflexion pour l'enseignement des mathématiques à souligné que« *les mathématiques sont partout présentes dans la vie courante : traitement des données, statistique, codage, compression de données, simulation numérique, ⋯ Mais cette présence qui se renforce est souvent occultée aux yeux du public qui ne voit que le produit fini.* Cette observation est entièrement applicable aux algorithmes et les résultats sont plus communs que les principes de base. Les algorithmes qui existent dans le monde technologique qui nous entoure sont très matures. Des automates les plus simples aux systèmes les plus complexes, les algorithmes contrôlent bon nombre de nos comportements quotidiens. Cependant, leur existence ne se traduit pas par un contact direct avec l'utilisateur, et l'utilisateur peut facilement assimiler la « machine » à son mode de fonctionnement.

Pour cette raison, il est apparu nécessaire d'indiquer dans le projet de programme pour la classe de Seconde :
La démarche algorithmique est, depuis ses origines, une composante essentielle de l'activité mathématique. Au collège, les élèves ont rencontré des algorithmes (algorithmes opératoires, algorithme des différences, algorithme d'Euclide, algorithmes de construction en géométrie). Ce qui est proposé dans le programme est une formalisation en langage naturel.

En mathématiques, les algorithmes sont utilisés très tôt à l'école. Opérations, calcul mental, réduction au même dénominateur, résolution d'une équation du second degré, factorisation d'une expression polynomiale, identification d'une situation de proportionnalité, tous ces algorithmes déplacent parfois chez les élèves les objets qui les manipulent. En travaillant dans un univers plus petit dans lequel le nombre de règles et de symboles utilisés est limité, nous tenterons de clarifier et de formaliser partiellement la notion d'algorithme.

3.5.6 Le langage informatique.

La programmation s'exprime par le code, qui est un ensemble d'instructions écrites en langage informatique. Il existe différents types de langages informatiques. Malgré leurs différences, leurs structures logiques sont assez similaires. La programmation nous permet de donner des instructions à des appareils numériques programmables comme les ordinateurs ou les robots.

Un langage de programmation graphique ou visuel est un langage de programmation dans lequel les programmes sont écrits par assemblage d'éléments graphiques. Sa syntaxe concrète est

composée de symboles graphiques et de textes, qui sont disposés spatialement pour former des programmes.

Les outils de programmation visuelle (p. ex. Scratch, Blockly, Kodu) permettent d'assembler des blocs de code en faisant un « glisser-déposer » sur l'espace de programmation de l'interface.

Il existe de très nombreux outils de programmation visuelle. Nous avons fait le choix d'utiliser le logiciel Scratch, en raison de sa qualité remarquable, sa simplicité d'utilisation, sa gratuité, et sa communauté très active, y compris dans le champ de l'éducation (primaire et collège). Ce choix est cependant arbitraire et, bien entendu, l'enseignant peut suivre la même progression en utilisant d'autres environnements (comme snap, mblock ou blockly, très similaire à Scratch, et eux aussi gratuits, ou kodu et tangara, qui sont payants)

L'apprentissage de la programmation développe les stratégies cognitives et métacognitives liées à la pensée informatique dont : l'abstraction, l'algorithmique, l'identification, la décomposition et l'organisation de structures complexes et de suites logiques.

La pensée informatique est en lien avec tous les systèmes symboliques permettant la modélisation de connaissances comme les mathématiques, les langues, les sciences et les technologies.

3.6 Pensée informatique et d'algorithme.

3.6.1 Définitions des notions de pensée informatique.

Le terme « pensé informatique » a fait florès depuis la parution d'un article « point de vue » d'une chercheuse en informatique aux États-Unis où elle argumente que la pensée informatique est : « un ensemble d'attitudes et de connaissances universellement applicables, que nous gagnerions toutes et tous à apprendre et à maîtriser, et que nous devrions transmettre à nos enfants ». Il y a toute une littérature proposant des définitions plus précises, dont celle proposée par cette même chercheuse en 2011 : *Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent.* En France un article souvent citéprésente les choses ainsi : *Le souci du caractère algorithmique de la description des objets, du langage dans lequel ces descriptions sont exprimées, des flux d'information et des instruments sont plus généralement caractéristiques d'une « pensée informatique ».* On retrouve les mêmes idées dans les textes (cosignés par différents acteurs plus ou moins institutionnels) demandant la prise en compte de l'enseignement de l'informatique dans les programmes du primaire et du secondaire. Michel et al. (2016)

La définition à laquelle font référence les promoteurs de l'enseignement de la pensée informatique à l'aide du langage Scratch est intéressante à double titre. D'une part, elle détaille les choses. D'autre part, elle permet de comprendre la vision sous-jacente aux ressources pédagogiques proposées autour de Scratch :

1. La pensée informatique implique d'appréhender le monde selon l'approche employée en programmation par les développeurs de logiciels.
2. Cette approche peut être scindée en cinq grandes catégories :
   – appréhender un problème et sa solution à différents niveaux (abstraction) ;

- réfléchir aux tâches à accomplir sous forme d'une série d'étapes (algorithmes) ;
- comprendre que pour résoudre un problème complexe il faut le décomposer en plusieurs problèmes simples (décomposition) ;
- comprendre qu'il est probable qu'un nouveau problème soit lié à d'autres problèmes déjà résolus par l'élève (reconnaissance de formes), et
- réaliser que la solution à un problème peut servir à résoudre tout un éventail de problèmes semblables (généralisation).

Alors, la pensée informatique est :

- savoir décomposer un problème en sous-problèmes plus simples ;
- savoir réfléchir aux tâches à accomplir pour résoudre un problème en termes d'étapes et d'actions (algorithme) ;
- savoir décrire les problèmes et les solutions à différents niveaux d'abstraction.

Ce qui permet d'identifier des similitudes entre problèmes et, par suite, de pouvoir réutiliser des éléments de solutions.

La pensée informatique ne se réduit donc pas à l'algorithmique, mais le concept d'algorithme est au cœur de la pensée informatique.

3.6.2 Pourquoi enseigner la pensée informatique ?

L'enseignement de la pensée informatique relève fondamentalement des objectifs pédagogiques autour de l'algorithmique et de la résolution de problèmes. L'avis de Baron & Bruillard (2011) est que :

1. Des compétences sous-jacentes à la pensée informatique (l'algorithmique, la logique, la décomposition de problèmes en sous-problèmes, les processus d'abstraction, de généralisation et d'instanciation sont des compétences utiles et importantes à tous et pas simplement aux informaticiens, surtout qu'on retrouve dans la vie de tous les jours dans l'organisation de l'action.
2. D'enseigner ces compétences de différentes façons, y compris sans informatique. L'approche algorithmique/programmation me semble cependant présenter des propriétés intéressantes. Elle amène notamment à travailler sur l'Action de transformer en chose concrète une idée ou un concept abstrait (réifications). Par exemple, quand on travaille sur un algorithme et qu'on veut le rendre « plus abstrait », on travaille sur un objet concret, qu'on voit à l'écran et qu'on modifie. Il y a un support concret à la réflexion et à l'action. Il est donc possible que l'enseignement de l'informatique favorise ces apprentissages plus que d'autres approches et/ou en permette un apprentissage alternatif pertinent pour certains élèves.
3. Pratique de l'algorithmique. Ceci passe, classiquement, par des exercices de type « écrire un programme qui fait xxx », le fait de « faire xxx » nécessitant que les élèves utilisent les notions algorithmiques que l'on veut leur faire travailler. Ce xxx peut être sans intérêt autre que de créer un cadre à l'exercice ou relever d'autres compétences du programme (en maths, en français, etc.). Il est donc possible de coupler enseignement de la pensée informatique et d'autres domaines. Il est également possible de coupler enseignement de la pensée infor-

matique et créativité. Cela ouvre un large champ pour l'exploration d'innovations pédagogiques tissant des liens entre différents apprentissages.

4. Prévoir des points d'entrées « construction d'algorithmes » et « programmation » qui sont des moyens que certains élèves peuvent percevoir (ou percevoir à certains moments) comme ludiques et motivants (notamment par la dimension de défi de ce type d'activité).

Les points d'entrées « construction d'algorithmes » et « programmation » permettent, en passant, d'aborder des objectifs de plus hauts niveaux comme :

– démystifier. Un ordinateur, ce n'est pas magique, ce qu'il fait est régi par des programmes faits par des humains, qui appliquent bêtement -ou plutôt, mécaniquement- ce qu'on leur a dit de faire ;
– inciter à la réflexion. Par exemple, une fois compris ce qu'est un algorithme, il doit être convaincu qu'un moteur de recherche ne renvoie pas « l'information » sur un sujet, mais les données qu'un algorithme particulier a identifiée comme pertinente et donc, en fait, l'information que quelqu'un a décidée, pour certaines raisons, légitimes ou pas, de renvoyer).

3.7 Logiciel Scratch comme outil d'enseignement.

3.7.1 Présentation.

Scratch est une application en ligne (*ou Offline*) conçues pour s'initier à des concepts fondamentaux en mathématiques et en informatique. Il repose sur une approche ludique de l'algorithmique, pour créer de petits programmes. Scratch est développé par le groupe de recherche *Lifelong Kindergarten* auprès du laboratoire Média du MIT (USA).

L'intérêt de Scratch est son approche basée sur l'utilisation de blocs de programmation, ce qui permet d'éliminer la difficulté majeure de devoir mémoriser et taper des instructions selon une syntaxe rigoureuse. Les avantages de ce logiciel sont nombreux :

Scratch est dynamique : il permet de modifier le code du programme en cours d'exécution. Il est orienté multimédia pour l'enseignement à l'univers informatique des enfants, il traite avec une grande facilité les concepts de base de la programmation comme les boucles, les tests, les affectations de variables, et surtout de la manipulation des objets, tout comme les sons et les vidéos.

Scratch est visuel : tout le code est directement inscrit dans la langue maternelle de l'enfant (une vingtaine de langues européennes est disponible) sous forme de briques de couleurs (par exemple les contrôles en jaune, les variables en rouge, les mouvements en bleu, etc. ⋯).

3.7.2 Scratch est un outil parfaitement adapté pour la programmation au collège.

Sur l'environnement de programmation visuel du Scratch, en glissant-déposant des blocs colorés, il est possible de créer des histoires interactives, des jeux, des animations, de la musique, ou des présentations. Il est ensuite possible de les télécharger sur Internet pour les partager avec les autres utilisateurs du monde entier. Scratch est conçu pour jouer, apprendre par soi-même et créer.

Scratch dispose surtout de compléments très intéressants :

- sa prise en main par les élèves est quasi-immédiate ;
- l'environnement est simple et efficace : constitué de trois parties, on a les instructions, la fenêtre du programme et la fenêtre d'exécution du programme sur le même écran ;
- il n'y a pas de syntaxe à connaître, ni à écrire : on déplace simplement des blocs d'instructions qui s'imbriquent par aimantation ;
- il est adapté à la programmation évènementielle : les scripts démarrent à partir d'un évènement et les objets peuvent communiquer entre eux par des messages ;
- un simple double-clic sur une instruction permet de l'exécuter, ce qui permet de vérifier facilement la bonne programmation d'un objet ;
- il apporte des rendus visuels grâce à des scènes et des costumes et constitue une interface attractive pour le jeune public ;
- à partir de l'interface de développement, on peut envoyer sur un espace de partage en ligne son programme ce qui facilite les échanges et favorise les interactions entre élèves.

3.7.3 L'interface de Scratch.

La figure 3.1 est la copie d'écran de l'interface :

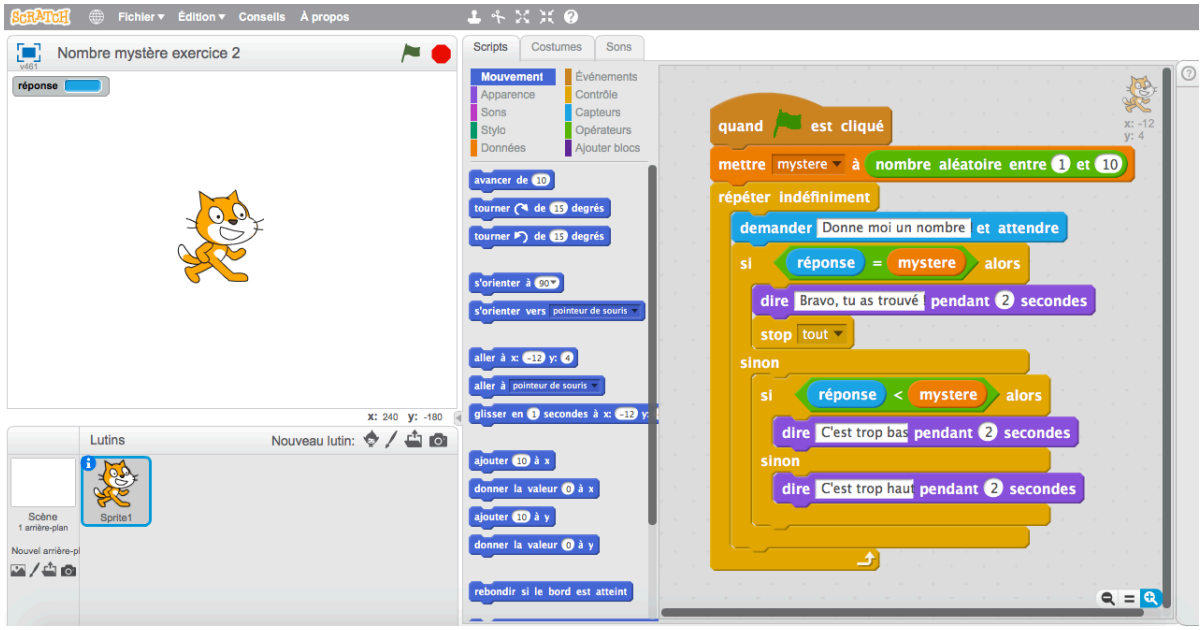


FIGURE 3.1 – Interface de Scratch : à gauche la scène, palette des blocs et à droite aire des scripts.

3.7.4 L'écriture d'un programme avec Scratch.

Le logiciel Scratch offre un environnement d'édition et une interface d'exécution des programmes. C'est un logiciel gratuit et disponible sur différents environnements usuels (*Windows, Mac et Linux*).

Ses principales qualités sont sa simplicité, sa fiabilité et sa robustesse. Il permet par ailleurs de travailler tous les concepts algorithmiques : la programmation évènementielle et la gestion de scripts

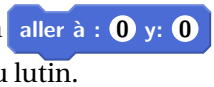
s'exécutant en parallèle. Scratch est donc dynamique : il permet de modifier le code directement en cours d'exécution. Son utilisation est ludique, il suffit de manipuler les briques et d'observer immédiatement le résultat.
Il est utilisable en ligne et offre la possibilité de partager ses projets au sein de la communauté internationale. De nombreuses ressources sont disponibles sur internet concernant Scratch : tutoriels, livres ou encore exemples de programmes à télécharger par la communauté sur le portail Scratch.

Les objets Scratch Lutin (Lutin ou arrière-plan).

Les lutins sont des objets manipulés par Scratch. Par défaut, il s'agit du chat, mais on peut utiliser plusieurs lutins, qui peuvent interagir en échangeant des messages, en détectant des collisions, etc. Chaque lutin a ses propres séquences d'instruction (*scripts*).
Tout lutin est muni d'un stylo, inactif (*car relevé en position haute*) par défaut, qui se déplace avec le lutin et qui, s'il est en position d'écriture, laisse une trace lors de ses déplacements.
Le stylo se déplace avec le lutin, mais il faut le mettre en position d'écriture pour que le tracé s'effectue lors des déplacements du lutin, ou le relever en position haute pour déplacer le lutin sans rien dessiner.

Les déplacements.

Il y a deux façons de se déplacer :
- en déplacement relatif. Les blocs d'instruction 

, déplacent le lutin dans la direction qu'il regarde ou le font tourner, par rapport à la direction courante ;
- en déplacement absolu. Les blocs d'instruction 

, indiquent la nouvelle position ou la nouvelle direction du lutin.

L'origine est au centre de l'écran de travail, ses bords gauche et droit sont d'abscisses -240 et +240, les bords bas et hauts d'ordonnées -180 et +180.
Les angles sont mesurés dans le sens des aiguilles d'une montre, la direction 0° visant le haut de l'écran, la direction 90° la droite, etc.
Les costumes : Un lutin (objet) peut porter différents costumes, ce qui permet par exemple d'animer les déplacements d'un lutin ou encore d'en avoir plusieurs représentations dans un même programme.
Les variables : Une variable permet de stocker une information. Les différentes variables possibles avec le logiciel Scratch sont présentées dans la figure 3.2.
Déclarer une variable, c'est indiquer le nom et le type (nombre, texte, ...) d'une variable que l'on utilisera dans l'algorithme. Déclarer une variable revient à « créer la boîte ».
La déclaration des variables se fait au début de l'algorithme avant la première instruction.
Affecter une variable, c'est attribuer une valeur à cette variable. Affecter une variable revient à « remplir la boîte ».

Toute affectation d'une valeur à une variable détruit sa valeur précédente. Dans `scratch`, les commandes permettant de créer et de gérer les variables sont dans la catégorie donné

Type de variable	Booléen	Nombre	Texte
Valeurs possibles	vraies / Faux (2 états uniquement)	Nombre réel (*Plusieurs valeurs)*	Chaîne de caractères (*Plusieurs caractères*)

TABLEAU 3.2 – Variables possibles avec le logiciel Scratch

Les listes : Une liste ou tableau permet de stocker plusieurs éléments les uns à la suite des autres. Cette liste peut contenir des variables d'un type donné : booléen / nombre / texte.
Les évènements : Un évènement permet de déclencher un ensemble d'opérations. Un évènement est une modification de l'état du système.

Exemple d'évènement : 

Dès que la touche espace sera enfoncée au clavier, l'instruction suivante sera exécutée. Cet évènement est indépendant des autres objets.
Les scripts : Un script (ou un programme informatique) est la traduction de l'algorithme dans un langage donné (pour nous, `scratch`), qui respecte scrupuleusement les codes de ce langage.
Un script est une séquence d'instructions associée à un objet (*lutin, arrière-plan*) et démarrant par un évènement. Dans un algorithme ou un script, une variable est une boîte qui possède :

- un nom (une lettre ou un mot),
- une valeur (un nombre, par exemple) qui peut changer au cours de l'exécution de l'algorithme ou du script.

Un programme Scratch comprend une ou plusieurs séquences, scripts (*actions en parallèle*).

3.7.5 Les structures algorithmiques.

Les tests ou structures alternatives.
Le test « **si alors** » peut être complété par une seconde instruction : « **sinon** ». Il se transforme ainsi en « **Si alors sinon** ». Le bloc d'instruction « **SI (condition) alors... sinon** » permet d'exécuter deux séquences d'instructions : une séquence si la condition est vérifiée et une autre séquence si la condition n'est pas vérifiée. Il est possible d'imbriquer le bloc d'instruction « **Si (condition) alors sinon** » dans un autre bloc d'instruction « **Si (condition) alors sinon** ».

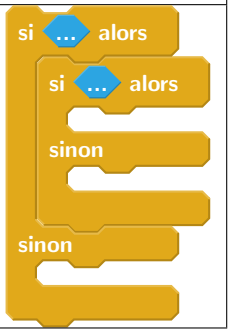
Si alors	Si alors sinon	Si alors sinon imbriquée
		

TABLEAU 3.3 – Les trois types de structure alternative (test).

Boucles ou structures répétitives.
Une boucle (bornée) permet de répéter un certain nombre de fois les mêmes opérations. On distingue trois types boucles dans Scratch : 1) Répéter *n* fois, 2) Répéter indéfiniment et 3) Répéter jusqu'à (tabaleau 3.4).

Répéter n fois	Répéter indéfiniment	Répéter jusqu'à
		

TABLEAU 3.4 – Structures répétitives dans Scratch.

Scratch permet donc d'aller assez loin dans la programmation. Il est possible d'imaginer un grand nombre d'activités autour du logiciel : jeux, énigmes, problèmes mathématiques, commandes d'un robot, dessin, etc.
À l'école primaire et au collège, on peut former les élèves à l'algorithmique et au codage en utilisant le cahier d'algorithmique et programmation Cycle 3, ou les 3 cahiers d'algorithmique et programmation 5ième 4ième et 3ième. Ces cahiers se trouvent directement sur le Web selon les modalités suivantes :

- Testez-vous sur le site web du MIT : https://scratch.mit.edu/
- Pour installer la version locale sur votre PC, c'est par ici : https://scratch.mit.edu/scratch2download/

Maintenant le logiciel Scratch est bien connu, nous présentons, dans la suite, les choix fondamentaux de construction de la séquence, qui reposent sur des choix didactiques et/ou des propriétés de l'environnement informatique. Pour cela, nous nous référons à la Théorie Anthropologique du Didactique (Chevallard, 1999), et plus précisément au cadre T4TEL (Chaachoua, 2018).

3.8 Les élèves

Durant cette recherche nous avons ciblé l'enseignement du second degré. Les établissements où nous avons travaillé se situent administrativement dans la DREN[22] Diana et Circonscription Scolaire Antsiranana I. Comme nous avons travaillé avec les établissements catholiques, ces deux collèges se situent aussi dans la DIDEC[23] Antsiranana.

Dans ces deux établissements nous occupons toutes les classes de 6ième, 5ième et 4ième pour l'expérimentation, mais d'ailleurs nous avons décidé uniquement réalisé nos études uniquement pour la classe de 4ième et principalement sur le chapitre d'enseignement et apprentissage d'algorithme. Selon le tableau, notre population cible compte 135 élèves pour le collège sainte Thérèse et 243 élèves pour le collège Notre-Dame de lourds, et 1885 élèves au total de cette classe d'éducation dans la région d'Antsiranana. Ceci elles représentent un pourcentage de 15% des élèves en classe de 4ième dans la circonscription d'Antsiranana.

22. Direction Régionale de l'Éducation Nationale
23. Direction Diocesien de l'Éducation Catholique

D'abord les classes sont dominées par les filles. En effet, les élèves de sexe féminin occupent le 63% contre 37% pour le sexe masculin. Et puis sur le point de vue psychologique, dans ces deux établissements, les élèves de la classe de quatrième appartiennent encore plus à la classe des pré-adolescents, les plus petits sont âgés de 10ans (0.3%) et le plus âgé à 18ans (0.36%). L'âge moyen est de 14ans (soit 26%). Enfin, les classes sont un peu surchargées, l'effectif par classe est en moyenne 64 élèves par classe. En classe, ces élèves s'assoient généralement en binôme sur des bancs de deux places. Selon les données statistiques que nous avons eues aux établissements privés d'Antsiranana les 51% de chaises existant sont en deux places.

Ces deux établissements sont différents sur l'aspect socioéconomique. Pour l'établissement Sainte-Thérèse on perçoit qu'un pourcentage élevé des élèves est issu de la bonne classe à l'inverse de l'établissement Notre-Dame-de-Lourdes, certains élèves sont issus de classes élevées, d'autres de classe plus populaire.

Ces deux établissements disposent tous les deux d'une bibliothèque comme salle de lecture et de documentation, du centre Informatique pour les TICE et du terrain mixte pour l'Éducation physique et sportive. Dans la matinée, ces deux établissements débutent leurs activités pédagogiques à 7h15 mais quelque fois, ils commencent à 6h pour le cours d'informatique et d'Éducation physique et sportive. Le soir, ils finissent généralement à 17h, même si parfois certains cours se prolongent à 18h, ce qui est souvent le cas pour cours d'informatique.

FIGURE 3.2 – Les élèves en cours de Scratch Collège Notre Dame de Lourdes.

3.8.1 Antécédents en mathématiques.

Mathématiques et informatique partagent des champs communs en plusieurs concepts. L'algorithme est un concept qui est à la fois présent dans les mathématiques et aussi dans l'informatique ceci montre la difficulté de travailler en informatique sans mathématiques.

Depuis la première année du collège, l'enseignement des mathématiques a rencontré déjà plusieurs algorithmes de résolution connus ; algorithme d'Euclide pour la recherche de PGDC d'un nombre et algorithme Eratosthène pour les nombres premiers. Depuis, l'enseignement des mathématiques utilise déjà plusieurs méthodes algorithmiques pour assimiler de nouvelles connaissances. Citons par exemple :
En primaire
En classe maternelle les nombres s'écrivent par une suite d'instructions.

– Pour écrire 1 : je fais un trait incliné vers la droite et je descends à la verticale.

– Pour écrire 2 : je fais un demi-cercle qui tourne à droite et un trait qui s'incline à droite et un trait horizontal.

Les opérations sont effectuées en suivant une suite d'instructions avec des conditions que ce soit pour faire les additions, les fractions ... De même en géométrie, plusieurs algorithmes sont énoncés de façon implicite dans les propriétés.

L'algorithme et les mathématiques partagent en commun, les variables, les constantes, les fonctions et les structures de contrôles. En première année du cycle secondaire, les élèves ont commencé à rencontrer les notions d'ensembles de nombres ; l'ensemble $\mathbb{N}$, $\mathbb{D}$ et $\mathbb{Z}$ aussi l'abstraction des calculs par l'utilisation des alphabets pour symboliser les variables.

En classe de sixième du collège de Madagascar, les élèves ont fait cinq heures de cours de Mathématiques par semaine pour un programme de six chapitres d'activités arithmétiques et sept chapitres d'activités géométriques. En arithmétique, on introduit généralement la notion d'ensembles ainsi que les proportionnalités et on accentue les calculs sur les nombres décimaux. Les figures géométriques planes et les figures symétriques sont introduites en géométrie.

En classe de cinquième, les élèves font les opérations sur les termes, les facteurs, la notion de puissance entière et les ordres de priorité dans les opérations. En classe de sixième, les élèves font cinq heures hebdomadaires de Mathématique. Le programme est constitué de deux parties : cinq chapitres d'activités arithmétiques et sept chapitres d'activités géométriques.
Les activités en arithmétiques priorisent les notions de la puissance entière et la décomposition en produit de facteurs premiers ainsi que les calculs sur PGCD et PPCM et le renforcement de connaissance sur les calculs sur les ensembles $\mathbb{N}$, $\mathbb{D}$ et $\mathbb{Z}$.

Par ailleurs, les activités géométriques se focalisent les notions d'angle et les triangles ainsi que sur les propriétés (Théorèmes de Thalès et de Pythagore) enfin sur les figures symétriques planes.

Ainsi, ces différents antécédents nous permettront facilement d'introduire en troisième année du cycle secondaire la notion d'algorithme selon plusieurs formes :

– Algorithme en tant qu'outils mathématiques. Ceci nécessite d'introduire des méthodes ou des démarches bien structurées lors de la résolution des problèmes mathématiques. Il faut savoir décrire le problème à résoudre et savoir présenter les données nécessaires ainsi qu'illustrer les suites finies d'opérations. Enfin il faut savoir présenter les solutions du problème.
– Algorithme en tant qu'objet, il faudra introduire la notion d'algorithmique ; sa définition, ses différents concepts, ses modes de présentation, sa validité, sa finitude et aussi sa complexité.
– Ainsi l'algorithme informatique, permet d'introduire un algorithme connu par des machines informatiques en particulier ceux ayant trait à la mémoire de l'ordinateur.

3.8.2 Antécédents en manipulation des ordinateurs

Pour mieux commencer l'apprentissage de l'algorithme et de la programmation en langage visuel, selon (Baron & Bruillard, 2011) dans sont intitulé, les nouveaux apprentis ont besoin des divers prérequis en manipulation des ordinateurs et ses périphériques. Les antécédents nécessaires peuvent se présenter comme suit :

– Connaissance sur la mise en tension et hors tension des équipements informatiques tels que, vérification ou branchement des cordons d'alimentation d'un ordinateur, l'ordre de démarrage.

– Connaissance sur l'interaction sur une fenêtre Windows, les apprenants devront savoir ouvrir, réduire et fermer une fenêtre Windows, effectuer des recherches d'un mot-clé simple, modifier, créer et renommer un dossier ou un fichier.
– Connaissance sur les différentes manipulations d'une souris à savoir les différents clics (clic, doubleclic, clic droit et clic glissé-déposé) et le déplacement de la flèche sur l'écran avec précision.
– Connaissance sur le saisi d'un mot simple et une phrase courte ou bien une page de document à structure simple à l'aide d'un clavier.

L'ensemble de ces connaissances est le résultat des formations qu'on leur a données en classes antérieures. Ces apprenants ont déjà bénéficié de deux années d'initiations en informatiques depuis la classe de sixième.
La présence des élèves nouvellement inscrits en classe de quatrième dans ces établissements, qui n'ont pas suivi ces formations, nous permettrons d'étudier davantage le besoin de ces prérequis.

Ainsi, en classe de quatrième, les chapitres qui précèdent l'initiation en algorithmique et programmation comme PAO, PréAO et Tableur leur permettent non seulement le renforcement des acquis, mais aussi le rappel des connaissances acquit antérieurement afin d'y rattacher de nouvelles connaissances.

3.9 Environnement informatique des élèves participant à l'expérimentation

Un parc informatique est un ensemble des matériels et logiciels informatiques (Ordinateur, les applications...). Lorsque plusieurs dispositifs informatiques sont mis à la disposition de groupe de personnes d'un établissement, leur ensemble est qualifié de parc informatique.

Par définition, un parc informatique est, comme il a été souligné précédemment, un ensemble de solutions informatiques dont dispose un établissement et comprend tous les équipements intégrant ce domaine. Il est important de souligner que la notion de parc informatique et aujourd'hui indissociable au domaine d'enseignement de l'informatique, autrement dit, les solutions informatiques sont interdépendantes et interconnectées. Il est évident que le système informatique dont on fait allusion ici fait référence à ces deux composantes, à savoir le hardware qui est la partie matérielle et le software qui comprend les logiciels. Comme il a été précisé plus haut, un parc informatique est composé de tous les éléments de nature informatique et non exclusivement, les ordinateurs en tant que machine.

Les matériels dont il est question ici sont bien évidemment les solutions qui peuvent être considérées comme étant des machines, dénommées hardware dans le jargon informatique, à savoir les ordinateurs et tous ses composants internes, mais également les matériels accessoires.

Pour ce qui est des programmes ou softwares, ce sont toutes les applications informatiques qui peuvent être considérées comme étant des logiciels et qui sont nécessaires au fonctionnement des machines (ordinateurs) ainsi que dans parc informatique d'établissement scolaire, ils peuvent être subdivisés en trois catégories, à savoir :

– Les programmes dédiés à l'opérationnalité des machines tels que les systèmes d'exploitation ou les antivirus.

– Les softs destinés aux activités d'apprentissage.

3.9.1 Disposition en « rangées » ou en « autobus ».

C'est un mode de disposition dans laquelle les élèves sont assis dans une même orientation en deux (ou rarement trois) colonnes. De cette configuration, pour assurer une bonne visibilité, chaque groupe dispose un ordinateur pour trois élèves.

Avantages : Cette configuration présente quelques avantages au niveau de la visibilité de la classe dans sa globalité. Il est favorable pour l'usage d'un vidéo projecteur ou d'un tableau. En effet, tous les apprentis ont le même aperçu. Aussi, en se déplaçant au fond de la classe, le formateur peut contrôler l'ensemble de la classe.

Inconvénients : Cette configuration présente des inconvénients : En effet, les groupes assis par derrière seront influencés par les travaux des autres groupes devant. Cette disposition favorise la copiée-collée et diminue la créativité et l'individualisation. Enfin, si on souhaite amener une situation de débat ou même de dialogue et d'entraide entre élèves durant une séance, ce n'est pas la configuration idéale puisqu'ils ne sont pas orientés les uns vers les autres.

3.9.2 Disposition en ilots.

La disposition en ilots permet de rassembler 4 a 6 élèves autour d'une même table (fig 3.3). C'est un mode disposition dans laquelle les groupes d'élèves assises face à face devant leurs ordinateurs, les élèves ont des orientations différentes.

Avantages : Aspect didactique, individualisation et autonomie de travail. Cette disposition met davantage dans une situation d'accompagnement et rend plus accessible les dialogues. Comme les élèves se trouvent en face a face dans un groupe restreint, l'entraide est favorisee. Les ilots sont plutôt pratiques pour mettre en œuvre la différenciation en regroupant des élèves de niveau homogène, ou au contraire, en les mélangeant pour que les plus faibles soient aidés par les élèves les plus avances.

Inconvénients : La gestion de la classe est difficile, car il y a un besoin de beaucoup plus de mobilité et de déplacement. Cette disposition est tres sensible au bruit. On met en garde : le volume sonore est plus élevé que dans une disposition plus classique. Il sera peut-être nécessaire de trouver une alternative a la prise de notes au tableau puisque certains élèves peuvent etre de dos ou de biais, surtout dans les petites salles.

3.9.3 Outils logiciels mis à la disposition des élèves.

Chaque machine dispose d'une même version de système d'exploitation (Microsoft Windows XP service pack 2), suite bureautique Microsoft Windows 2007 (Word, Excel, Publisher et PowerPoint), logiciel de dessin Microsoft Paint, logiciel Dactyl, logiciels de navigation internet (Firefox et Google Chrome) et logiciel de programmation Algobox et Scratch 2.0.

Nous avons mis les icônes de chaque application sur le bureau, dans tous les programmes et dans la barre de lancement rapide de la barre de tâches afin de permettre aux apprenants toute sorte de lancements de logiciels connus.
Remarque : Seul un établissement dispose un parc informatique connecté à l'internet. Alors toutes séances de cours lié à l'internet ont enseigné à l'aide un vidéoprojecteur, et les activités sont à faire chez soi.

3.9.4 Mode de travail.

Pendant chaque séance, nous avons divisé le temps de travail par rapport aux effectifs des membres de chaque groupe selon la disposition présentée dans la figure 3.3. Alors pour une durée de 60 minutes chaque membre de groupe sera 10 à 15 minutes, à la tête du groupe. Ainsi, en une heure chaque membre de groupe pour être à la tête du groupe. Cette règlementation permet pour chaque apprenant membre du groupe de bénéficier des mêmes compétences.

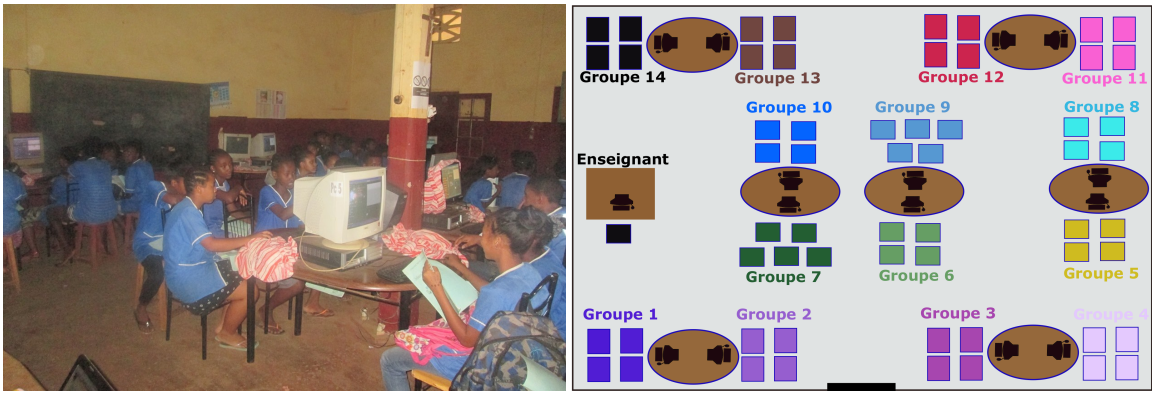


FIGURE 3.3 – Disposition des élèves dans la salle de classe.

3.10 Méthode de conduite de groupe « Jigsaw classroom ».

« Jigsaw classroom », ou « classes-en puzzle », ou « apprentissage coopératif avec décloisonnement en équipes d'experts » est une technique d'enseignement inventée en 1971 par le psychologue social américain Elliot Aronson (né le 9 janvier 1932). Il utilise une stratégie d'apprentissage coopératif destinée à l'enseignement primaire et secondaire. Celle-ci encourage fortement les élèves à l'écoute, à l'engagement, à l'interaction, au partage et donc, confère à chacun un rôle essentiel à jouer dans l'activité académique. Les « Jigsaw classroom » ont également pour avantage considérable la réduction de l'hostilité et des préjugés interethniques. C'est ce modèle que nous avons choisi pour notre initiation à l'informatique.

3.10.1 Fonctionnement.

Les élèves sont placés dans des groupes d'apprentissage de six personnes. Chaque jour, la leçon est divisée en six paragraphes de sorte que chaque élève reçoit une seule partie de l'information. Toutes les parties, comme les pièces d'un puzzle, doivent alors être mises ensemble pour que chacun puisse étudier le sujet en entier.

Le groupe expert : Chaque élève se sépare de son groupe d'apprentissage et doit lire son paragraphe, le comprendre et tenter de le mémoriser. Ensuite, les groupes experts sont constitués. Il s'agit de réunir tous les élèves devant mémoriser la même partie de la leçon. Ils se consultent alors les uns les autres de manière à répéter et à clarifier quels sont les principaux aspects importants du paragraphe.

L'organisation de la classe prévoit trois étapes d'élèves qui se succèdent dans le temps voire dans l'espace comme le montre la figure 3.4

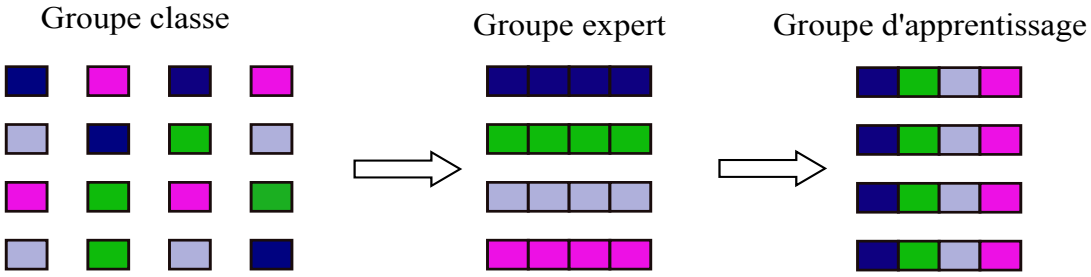


FIGURE 3.4 – Les trois étapes de la technique d'enseignement des Jigsaw classroom.

Le groupe d'apprentissage : Après avoir passé entre 10 et 15 minutes dans son groupe expert, chaque élève revient dans son groupe d'apprentissage originel de six personnes. Le professeur leur signale alors qu'ils disposent d'un temps limité (20 à 30 minutes) pour communiquer leur savoir aux autres membres du groupe. Ensuite, ils seront interrogés sur la leçon complète. Livrés à eux-mêmes, les élèves apprennent rapidement à enseigner et à écouter les autres. Ils se rendent compte qu'aucun d'eux ne peut réussir le test sans l'aide de tous les autres. Chaque élève fait l'expérience de la compétence, d'un don essentiel à apporter au groupe qui ne peut être apporté que par lui.

3.10.2 Bénéfices et Lacunes de la Jigsaw classroom.

Bénéfices : Les élèves des classes en puzzle montrent une évolution positive dans leur attrait pour leurs camarades de classe, peu importe qu'ils fassent partie de leur groupe ethnique ou non. Leur estime de soi à également progresser significativement plus que dans les classes contrôles, aussi bien pour le groupe majoritaire que pour les groupes minoritaires. D'après LUCKER (1977), il apparait qu'une augmentation importante des scores des élèves minoritaires des classes puzzle, comparativement aux élèves majoritaires qui ont obtenu des résultats semblables dans les deux conditions expérimentales.

Lacunes : Dans certains cas, un élève dominant peut prendre trop de place, ou tenter de diriger le groupe. Certains élèves sont naturellement plus faibles que la majorité et éprouvent des difficultés à résumer ce qu'ils ont lu. Il peut également arriver qu'un élève plus brillant que les autres trouve peu d'intérêt à la tâche proposée, s'ennuie et ne participe pas.

Chapitre 4

Méthodes pédagogiques mises en œuvre.

Sommaire

Pour répondre aux objectifs préétablis de notre recherche, nous allons emprunter la méthodologie « fouille de données », appelée aussi par bon nombre des chercheurs « Extraction des connaissances à partir des données (ECD) » (cf. :1.1).

D'abord, nous avons considéré la connaissance, selon la définition fournie par Platon ; *la connaissance est une croyance vraie et justifiée*. Sur laquelle, elle implique deux conditions :

- pour accéder au statut de connaissance, une croyance doit non seulement vraie (c'est-à-dire qu'elle correspond à une réalité),
- le sujet doit également être fondé à croire (c'est-à-dire que l'on ne peut pas savoir par hasard ou par erreur).

Ainsi, nous résolvons notre problème en application la théorie platonicienne de la connaissance à la justification. En effet, selon cette théorie, nous devons connaitre si notre croyance est justifiée. Dans quels cas peut-on considérer que notre croyance est justifiée, sur ce problème, deux familles de théories existent :

Internalisme La première famille est celle de l'internalisme, pour laquelle nous avons, au moins en théorie, accès aux justifications de nos croyances. Mais là aussi, on distingue deux types d'énoncés à savoir :

- les énoncés d'observations qui décrivent des objets et des faits ou les données immédiatement accessibles par l'observation, directement observables
- et les énoncés théoriques qui se réfèrent à des choses non directement perçues. Ici, nous avons réalisé des analyses des faits exécutés par les élèves qui seront donc directement observables en classe par leur copie soit lors d'expérimentations ou par le remplissage des questionnaires. Nous sommes dans le premier cas.

Il faut cependant noter qu'en temps que chercheur, nous allons chercher les justifications de nos croyances observées par les travaux des chercheurs antérieurs.

Externalisme La seconde famille est celle de l'externalisme. Selon cette théorie, il n'est pas nécessaire que nous ayons-nous même accès à la justification de nos croyances pour ce qu'il y est des connaissances : il suffit qu'un autre travail de recherche extérieur possède cette justification. Les théories externalistes sont souvent formulées comme des théories causales de la connaissance.
Afin d'explorer toutes les relations causales des connaissances cachées, dans la première analyse, nous avons regroupé toutes les données et nous allons utiliser et avons profité l'analyse statistique implicative en se servant de l'outil, ASI – MGK, que nous avons présenté dans le chapitre (cf. :2)

4.1 Les écoles qui vont servir à l'expérimentation

Avant de commencer la recherche sur le terrain, nous avons mené des visites de courtoisie suivie d'entretiens avec les institutions.

Le premier entretien a été établi avec la Sœur-Directrice du collège Notre-Dame-de-Lourdes Tananmbao - Antsiranana. Lors de cet entretien, nous lui avons demandé son consentement et son autorisation pour nos recherches. La direction a favorablement accepté sans hésitation ; dans son établissement, nous avons eu à notre disposition les élèves de la classe de sixième jusqu'à la classe de quatrième. Nous avons obtenu des cours hebdomadaires d'une heure pour toutes les classes et l'accès à la salle informatique tous les jours ouvrables.

Le deuxième entretien a eu lieu avec la Sœur Directrice du collège Sainte-Thérèse Place Kabary - Antsiranana, du fait que l'école a déjà des cours d'informatique, la direction a accepté sans aucun souci. L'école dispose d'une salle d'informatique avec 11 postes en marche. Nous avons a notre disposition trois classes de sixième, trois classes de cinquième et deux classes de quatrième, nous avons obtenu une séance de deux heures par semaine par classe. En raison d'encombrement et du manque d'ordinateur disponibles, l'accès à la salle pour chaque classe se faisait en deux groupes. Une formation pédagogique des enseignants responsables et une révision du contenu des cours ont aussi été sollicitées durant l'entrevue.

Le troisième entretien a été avec le Père responsable du lycée Saint-Jean Place Kabary - Antsiranana. Nous avons également obtenu l'approbation pour toutes les classes de seconde et de première L, les autres classes comme la première S et les terminales ne sont pas disponibles. Nous avons effectué une séance de 2 heures de cours par semaine et ainsi que l'accès à une salle de médiathèque informatique comptant 16 ordinateurs disponibles.

Avec d'autres établissements comme l'Avotra College et l'ABC qui font tous deux de l'informatique, nous n'avons pu parler qu'aux professeurs responsables du cours. Les deux enseignants sont tous convaincus du bien fondé des réformes que nous avions prévu d'introduire.

Pour tous les établissements, nous avons vocation à introduire une réforme sur le contenu du cours à enseigner. Les établissements suivront le même programme et auront le même matériel de cours, les professeurs étant prêts à suivre une formation de recyclage de deux semaines.

4.2 Apprentissage des élèves

L'informatique, en tant que science comme en tant qu'outil, a envahi notre quotidien nous apportant maints services. Elle a contribué à une transformation rapide de la société partout dans le monde. Le progrès qu'elle a enregistré au niveau des logiciels comme au niveau matériel a provoqué le développement rapide de la plupart des secteurs vitaux. L'accès libre et immédiat à une source de données prodigieuse a changé radicalement les mentalités et a permis l'émergence d'un autre mode de pensée. Ce progrès continu et rapide nécessite une mise à jour perpétuelle des connaissances. Le marché de l'emploi a lui aussi vu l'émergence d'un large éventail de nouveaux métiers. L'éducation, secteur pilier dans la formation de l'individu, s'est trouvée impliquée dans cette transformation. En effet, l'enseignement de l'informatique, qui était il y a peu de temps optionnel et réservé aux dernières classes du secondaire, devrait être obligatoire dans toutes les classes depuis le collège. Ce choix vise à développer, de plus en plus tôt, des compétences spécifiques en matière d'informatique chez les élèves.

L'enseignement de l'informatique dans ces deux établissements existe depuis 2010 et l'apprentissage est basé sur les suites bureautiques (MS Word en sixième, MS Excel en cinquième et MS Po-

wer Point en quatrième). Lors de notre étude, notre première contribution a porté principalement sur une amélioration du contenu du programme enseigné et une orientation vers un aspect le plus en plus académique. Il s'agit d'un enseignement de culture générale qui apporte connaissances, savoirs et savoir-faire, qui développera l'esprit d'initiative, de création et le travail en équipe, ce qui établira des liens étroits avec l'ensemble des disciplines (notamment pour le choix et la réalisation des projets).

L'enseignement de l'informatique que nous avons introduit au collège vise les objectifs généraux suivants :

- Exploitation des fonctions élémentaires d'un ordinateur,
- production d'un document numérique ,
- savoir documenter et communiquer avec les TICs ,
- développer son esprit critique face à l'information et à son traitement.
- acquérir les notions de base en programmation et en algorithmique ,
- et acquérir les savoir-faire relatifs à l'écriture et à l'exécution d'un programme simple.

Alors, depuis l'année scolaire 2015/2016, nous travaillé avec douze classes au collège Notre-Dame-de-Lourdes ; cinq classes de sixième, quatre classes de cinquième et trois classes de quatrième. Au collège Sainte Thérèse, ce fut seulement avec huit classes dont trois classes de sixième, trois classes de cinquième et deux classes de quatrième. L'enseignement était assuré par un enseignant qui ayant bénéficié de la formation.

Durant l'année scolaire 2017/2018, les classes avec lesquelles nous avons travaillé, avaient déjà reçu deux années de formation en informatique, avec l'initiation en informatique en classe de sixième et le traitement de texte et le tableur en classe de cinquième. Selon notre programme en informatique que nous avons décrit pour la classe de quatrième, l'enseignement de l'algorithme et la programmation sur Scratch se situent au quatrième chapitre. Cet enseignement a débuté qui débute vers le deuxième mois du deuxième trimestre et a duré douze semaines. Les apprentis en algorithme et programmation sur Scratch ont déjà les prérequis nécessaires pour introduire ce nouveau concept.

Selon le règlement de ces deux établissements, tous les supports de cours que nous avons utilisé sont donnés en version papier sous forme d'un livret de 22 pages au début de l'année pendant l'inscription ou la réinscription. Les activités d'évaluation sont partagées par la suite en polycopiés au fur et à mesure de l'avancement du programme.

Les cours théoriques sont faits en mode débranché dans la salle de classe tandis que les séances pratiques sont faites en mode connecté dans la salle informatique. Chaque classe a une séance de une heure par semaine repartie selon la plage d'horaire différent de 6h 30 à 11h 30 la matinée et de 14h 30 à 17h l'après-midi.

Plusieurs modèles et méthodes pédagogiques furent utilisés durant cette expérimentation, à savoir :

- L'introduction de la notion algorithmique en activités débranchées, quand elles ne demandent aucune utilisation d'objets numériques. Ces dernières abordent essentiellement des concepts de base de la science informatique : algorithme, langage, représentation de l'information, etc.
- L'introduction de la notion algorithmique en activités branchées utilise des outils électro-

niques tels que les ordinateurs, et permet surtout l'apprentissage de la programmation. Pour celle-ci nous avons introduit deux types de langages : programmation en pseudocode utilisant algobox et programmation en langage visuel utilisant logiciel Scratch.

4.3 Utilisation du Logiciel Scratch

Les objectifs sont clairs :

– acquérir les notions de base en programmation et en algorithmique ;
– acquérir les savoir-faire relatifs à l'écriture et à l'exécution d un programme simple avec l'environnement Scratch.

Le cours d'initiation en programmation est important, car, rappelons-le, l'informatique n'est pas la science des ordinateurs, mais celle du calcul considère comme l'art de réaliser des opérations complexes à partir d'un jeu d'opérations élémentaires, au demeurant arbitraires, mais données et invariantes. Pour ce faire, nous avons privilégié, au moins au début, les problèmes simples ne nécessitant aucune connaissance et très peu d'efforts de compréhension, et ne soulevant, dans un premier temps, aucune difficulté mathématique : problème de remise en ordre des éléments d'une suite, énumération et listage d'éléments pris dans un ensemble défini par des conditions simples (par exemple : mots de n lettres écrits sur l'alphabet (a,b) qui ne contiennent pas deux occurrences consécutives des b, permutations, triangulations d'un polygone, etc.), recherche d'un élément dans un ensemble, vérification de propriété ou théorème, procédure de construction d'une figure plane.

À l'école élémentaire, l'exemple d'une utilisation de Scratch en géométrie permettra plus facilement aux élèves de s'imprégner des propriétés des figures qu'ils doivent représenter, en les combinant avec les contraintes inhérentes au langage de programmation.

Les algorithmes décrits en premier seront des algorithmes parfaitement naïfs qui seront explicités en langage ordinaire avant d'être traduits en programmes (par exemple, on explicitera complètement l'algorithme de construction d'une figure géométrique plane, l'algorithme d'une division euclidienne). Alors dans cette section nous avons présenté d'une part les séquences d'enseignement effectuées en géométrie et d'autre part celles qui sont effectuées en arithmétique.

4.4 Séquence d'enseignement reposant sur la géométrie

4.4.1 Tache 1 : construction d'un carré avec le logiciel Scratch

Propriété :
Un carré est une figure géométrique qui a quatre côtés égaux dont les deux côtés adjacents font un angle de 90 degrés.

Principe de mise en œuvre :
On procède quatre fois l'opération tracé un côté de longueur l et tourné un angle de 90°.
- En séquence d'instruction on pourra le traduire comme suit :
`AVANCER 1 PAS` et `TOURNER DE 90°`, `AVANCER 1 PAS` et `TOURNER DE 90°`,`AVANCER 1 PAS`

et `TOURNER DE 90°` et puis `AVANCER 1 PAS` et `TOURNER DE 90°`.

En Scracth, elle se code comme le montre l'algorithme (a)

- Mais en structure répétitive, il se traduit de cette manière (Figure 4.1) :
`REPETER 4 FOIS (AVANCER 1 PAS` et `TOURNER DE 90° )`

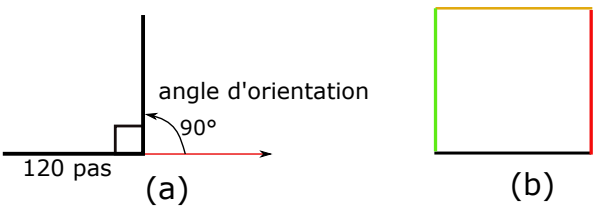


FIGURE 4.1 – Modélisation de la construction d'un carré en Scratch.

Mise en œuvre :
Avec un support écrit, sans support visuel (donc oralement), nous avons demandé à nos élèves de la classe quatrième de reproduire le programme de construction d'un carré de côté 120 pixels algorithme (fig-(a)).

Après quelques minutes de travail, d'après sa représentation un peu ludique et sa forme concrète colorée, les élèves se lancent rapidement dans la construction et parviennent à la construction sans difficulté.

Puis on leur demande par la suite d'utiliser une structure répétitive, en regroupant les instructions qui sont répétées dans la boucle `REPETER n FOIS`, et les élèves avec quelques interactions sont parvenue à la construction.

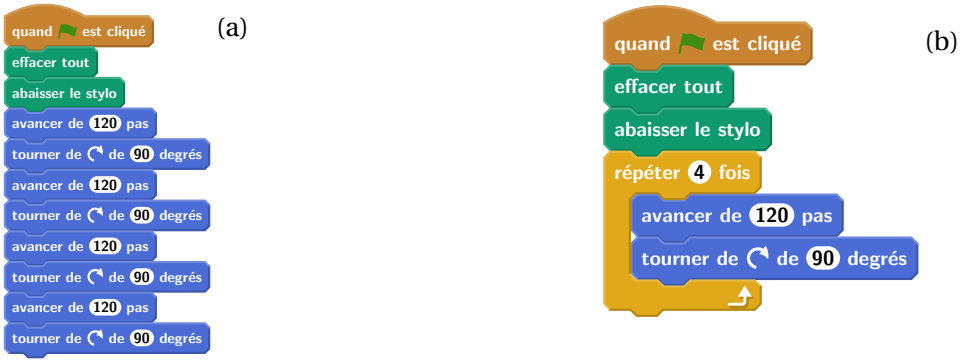


FIGURE 4.2 – Codes de construction d'un carré en Scratch : (a) en séquence, (b) en boucle.

Pour mieux comprendre les rôles des chaque bloc d'instructions nous avons décidé de rajouter oralement une autre instruction au programme. Cette instruction a pour but de comprendre tout d'abord le rôle du bloc de répétition et aussi l'ordre des instructions. Pour les faire, il faut que le lutin fasse une pause de une seconde à la fin de chaque côté tracé.

Ainsi, on fait comprendre aux élèves que pendant la construction le lutin s'est avancé de 120 pas et puis il a tourné d'un angle de 90 degrés et a répété quatre fois ses opérations.

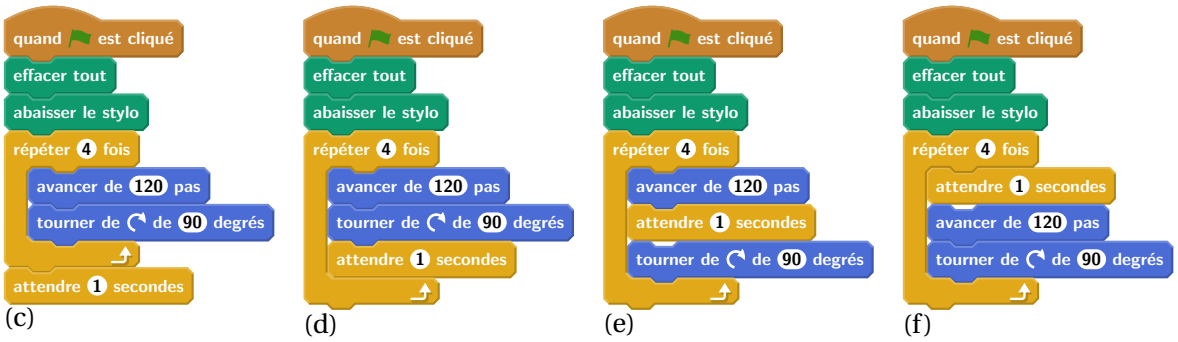


FIGURE 4.3 – Programmes en Scratch réalisés par les élèves : (c) incorrect, (d,e) acceptables et (f) correct.

Les élèves se sont mal adroitement précipité à faire la figure 4.3 (c) ce qui visiblement ne répond pas au programme voulu et après quelque exécutions manquées, ils sont tombés presque au hasard sans grande réflexion sur l'une de figure 4.3 (d), (e) et (f) et sans comprendre les rôles de l'instruction `AVANCER` et un peu sur l'instruction `TOURNER`, mais en majorité, ils voient que le lutin s'est arrêté quatre fois.

La compréhension du bloc répétition va avoir lieu ici ; ils ont bien constaté que, pendant la construction, le lutin a avancé de 120 pas et puis tourné de 90 degrés quatre fois ces opérations. Reste à savoir maintenant l'importance de l'ordre des instructions ; qu'est ce qui va se passer en premier entre les deux `AVANCER` ou `TOURNER` ?

Pour les faire, il faut que les élèves puissent répondre à cette question, il a fallu qu'on leur demande de modifier la couleur du stylo à la fin de chaque côté tracé avant de `TOURNER`. Cette instruction leur a permis de bien identifier chaque côté tracé avec leur couleur et constater la rotation après le traçage.

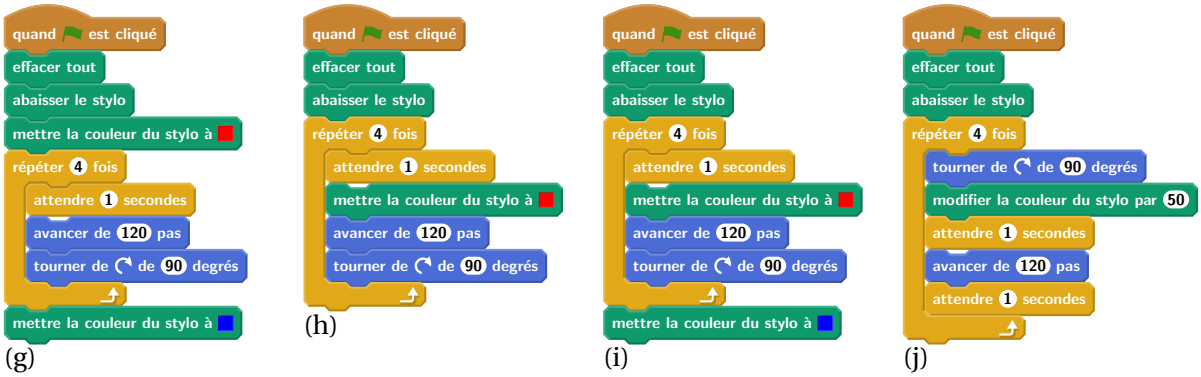


FIGURE 4.4 – Programmes en Scratch réalisés par les élèves : (g,h,i) incorrects et (j) correct.

Les résultats des élèves vont vers un échec. Une autre instruction additionnelle sera nécessaire pour débloquer les élèves et pour leur faire découvrir le sens exact des instructions `AVANCER` et `TOURNER`. On a leur demande encore d'attendre une seconde après `TRACER` et une autre seconde après une rotation. Et il faut bien faire attention aux noms de chaque instruction c'est-à-dire `MODIFIER LA COULEUR` et ne pas confondre avec `METTRE LA COULEUR`.

À la fin les élèves vont presque à tous coups avoir la figure 4.4 (j). Mais pour améliorer la visibilité on peut leur suggérer de modifier la longueur de pas de 10 à 20cm ou plus.

La plupart des élèves ne comprennent pas réellement le rôle du drapeau vert. Le lancement du programme se fait majoritairement par le doubleclic au script. Le lutin quitte le tracé et revient au point (0,0) ce qui permet de dire aux élèves « le carré est tracé ».

4.4.2 Tache 2 : Construction d'un triangle équilatéral

Propriétés :

- Un triangle équilatéral est une figure géométrique qui a trois côtés de même longueur dont les deux côtés adjacents font un angle de 60 degrés. Il est possiblede dire aussi un triangle équilatéral est une figure géométrique qui a trois côtés de même longueur et trois angles de même mesure.
- Deux angles sont dits angles supplémentaires lorsque leur somme fait 90 degrés

À partir de la tâche précédente, on a leur demande d'utiliser directement la structure répétitive.

Après quelques tentatives de construction par la méthode essai-erreur en utilisant à la fois un papier et un crayon. La majorité des élèves tombent sur l'une des algorithmes présentés dans la figure 4.5 (k), (l) et (m) qui rebouclent quatre fois un avancement de longueur donnée suivi d'une orientation de 60 degrés par rapport à la direction en cours. Dans les quatre premières tentatives de construction avec le logiciel Scratch, rares sont les groupes qui trouvent le bon nombre de boucles appropriées et aucun na trouvé le bon angle d'orientation.

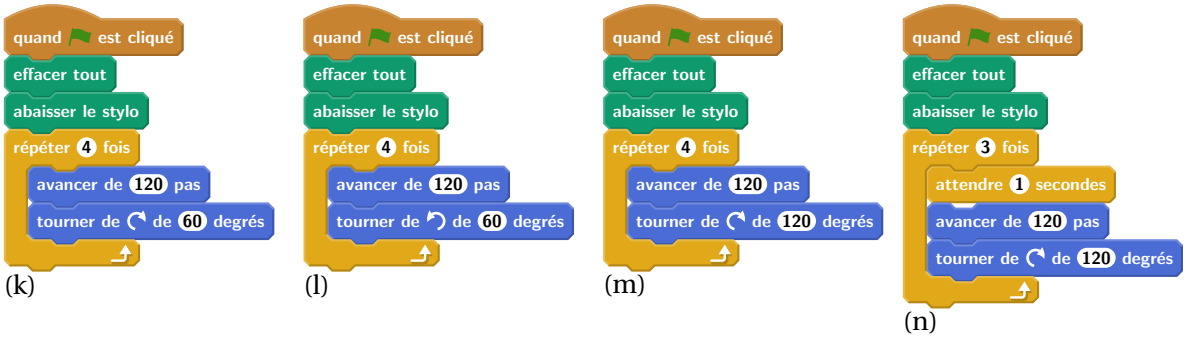


FIGURE 4.5 – Programmes de constructions d'un triangle équilatéral en Scratch réalisé par les élèves : (k,l,m) incorrects et (n) correct.

La majorité des groupes parviennent en tâtonnement et après avoir effectué plusieurs essais, seuls les groupes doués parviennent, en se faisant cependant aider par la modélisation schématique du problème, ils ont trouvé que l'angle de rotation est celui de l'angle supplémentaire de l'angle formé par les deux côtés adjacents. C'est-à-dire $180° - 60° = 120°$.

Première phase

L'ajout de l'instruction de pause de lutin après chaque tracé permet aux groupes en difficulté de remédier à l'erreur sur le nombre de répétitions.

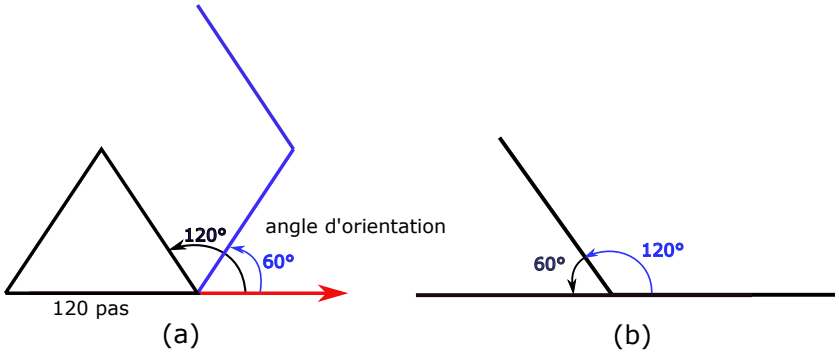


FIGURE 4.6 – Difficulté observé sur la modélisation de construction d'un triangle en Scratch.

4.5 Séquence d'enseignement reposant sur l'arithmétique

4.5.1 Activités basées sur la division euclidienne

Comment exploiter les caractéristiques de l'algorithmique et de la programmation pour créer des situations didactiques faisant travailler le sens de la division euclidienne dans le secondaire?

Pour répondre à cette question, nous avons considéré l'enjeu d'apprentissage « donné du sens aux notions de quotient et de reste » et construit un ensemble de tâches présentant les caractéristiques suivantes :

1. L'algorithme/programme que l'élève doit décrire. Il définit le comportement d'un objet ou d'un personnage (par exemple, son déplacement) à l'écran.
2. La tâche de l'élève consiste à construire un algorithme qui permet d'obtenir un comportement cible, qui est stipulé par l'énoncé de l'exercice.
3. L'écriture de l'algorithme nécessite la mobilisation des notions enjeux d'apprentissage.
4. L'algorithme attendu est une explicitation de la procédure de calcul attendue.
5. L'exécution de l'algorithme permet de visualiser la procédure proposée par l'élève.

Propriétés

Présenter une multiplication en une somme

$8 \times 7 = 8+8+8+8+8+8+8$

Multiplier un entier b par un entier a signifie ajouter a fois la même quantité b

Présenter une division en soustraction successive

$\mathrm{D} d - d - \ldots - d = r$ ou $q = (\mathrm{D} - r)/d$ le nombre de soustractions effectuées

L'activité à réaliser correspond au découpage d'un ruban de longueur D à l'aide d'un gabarit de longueur d. Trouver le nombre de rubans découpé équivaut à trouver le quotient de la division euclidien de D/d.

Étant donnée un ruban de longueur D et un gabarit de longueur d inférieur ou égal à D, il existe un couple d'entiers uniques (q, r) tels que $\mathrm{D} = d \times q + r$

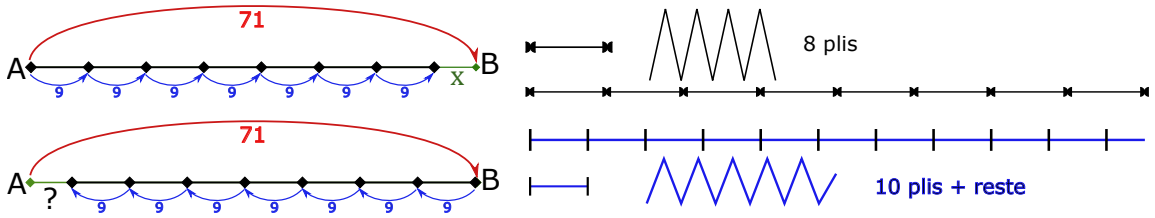


FIGURE 4.7 – Modélisation de la division euclidienne en image.

On retrouve ici les mêmes ingrédients que dans la recherche d'un algorithme : on a une situation initiale qui doit être précisée (distance AB et longueur de l'étape), un résultat souhaité (nombre d'étapes et distance restantes de la dernière étape et la destination), et un petit nombre d'opérations possibles (avancer retourner, récupérer la distance de la position) ; il est conseillé de construire une liste ordonnée de ces opérations qui conduit certainement au résultat.

Les élèves ont agi comme la plupart des chercheurs à qui ce problème est posé. Ils ont fait un essai, puis un autre, et après chaque échec, ils ont réessayé et ont cherché autre chose. Bref, ils ont progressé, partant de la situation donnée au départ, vers la situation finale souhaitée, c'est-à-dire passant du connu à l'inconnu. Il se peut qu'après de nombreuses tentatives, nous finissions par réussir, mais ce sera par hasard.

Les données de la tâche, les contraintes de l'interface à respecter et les propriétés de la mise en œuvre avec Scratch sont liées à différentes variables didactiques qui permettent de générer des types de tâches et des sous-types de tâches à partir de problèmes.

Les différentes tâches selon les variables didactiques

Tache T1 : soustraction successive présentée en séquence d'instruction

- q nombre de séquences soustractives effectuées ;
- r la position finale du lutin.

Pour le cas des additions successives, le travail déconnecté sur papier crayon est nécessaire pour retrouver le reste de la division de 71 par 9. Pour passer de A à B, on ordonne au lutin d'avancer sept fois successives de neuf étapes et une fois avancer de 8 pas. Cette instruction peut être traduite en une expression algébrique de la forme : $71 = 9 * 7 + 8$.

Afin de respecter les critères (3) et (4) (sec 4.5.1), nous avons défini des blocs Scratch spécifique correspondant a une série soustraction : d'abord « Avancer de D pas » et puis « Avancer de -d pas », $\cdots$, « Avancer de -d pas ». Et le programme scratch est le suivant (fig 4.8) :

FIGURE 4.8 – Code des séquences soustractives en Scratch.

Certains groupes d'élèves résolvent le problème par essais-erreurs, le nombre de répétitions d'instructions obtenues par essais-erreurs et la distance restante obtenue par la différence entre la longueur totale et celle de sa position, donc pour passer de A à B, nous commandons au lutin d'avancer sept fois successives de neuf pas et un avancer de distance restante. Cette instruction peut être traduite en une expression algébrique de la forme : $9*7<71<(9*8)$.

Si le lutin arrive à la bonne destination B, alors la distance AB est un multiple de l'étape ou bien la division de la distance AB avec l'étape est exacte. Dans le cas contraire AB n'est pas multiple de l'étape choisie ou c'est une division euclidienne.

La détermination de l'instruction correcte entraîne une expression booléenne *position* == *distance* AB, ce qui est une bonne occasion d'introduire l'expression conditionnelle et la structure conditionnelle `SI..ALORS` ou `SI ..ALORS..SINON`.

La distance $71-9*7$ correspond à la position du lutin. On peut présenter cet algorithme de cette façon (fig 4.9).

FIGURE 4.9 – Code d'une division euclidienne en Scratch.

Ces modes de résolution sont tous bénéfiques, d'une part dans l'utilisation de la méthode du crayon papier, les connaissances mathématiques sur le concept de division ou multiple s'éveillent. En revanche, la méthode directe ou par essais-erreurs conduit les groupes à trouver la solution en ne se servant que de l'environnement de programmation qui est en dehors du concept de mathématiques enseigné en classe. Citons, par exemple, dans l'environnement Scratch, le capteur de position, les coordonnées de la souris, le chronomètre, la distance, ⋯

Tache T2 : additions successives présentées en séquence d'instructions

- q nombre de séquences d'instructions additives effectuées ;
- r la distance restante pour arriver à la destination.

Comme nous avons D = $d \times q + r$ donc r = D – $(d \times q)$

Dans le cas d'une soustraction successive, nous présentons la situation finale comme dans la figure 2 (partons de ce qui est demandé et admettons que ce que l'on cherche est déjà trouvé). À partir de quelle situation précédente, nous avons pu obtenir la situation souhaitée (cherchons à partir de quel antécédent le résultat final pourrait être obtenu). Nous arrivons à destination, il faudrait revenir en arrière successivement de même pas. Pour cela ... il faudrait être à l'arrivée B. C'est l'idée ! Mettons à nouveau à la position qui pourrait être la position prédécesseur de la position d'arrivée B. Cette méthode provoque un retour progressif du même pas. 71 – 9 – 9 enfin le reste a été trouvé et le quotient ce n'est que le nombre de soustractions effectuées.

Afin de respecter les critères (3) et (4) (sec.4.5.1), nous avons défini des blocs Scratch spécifiques correspondant à une série d'additions : « Avancer de d pas », ⋯ , « Avancer de d pas ».
Avec Scratch, cette séquence d'instructions est traduite comme suit (fig 4.10).

FIGURE 4.10 – Code des séquences additives en Scratch.

L'expression du pas négatif ou l'affectation de variable par une valeur négative sont les principales difficultés que nous avons notées chez les élèves. Écrire une instruction avancée (-1) * variable n'a pas été facile pour les étudiants. Nous avons profité de cette occasion pour introduire la notion de variable et d'affectation. Un programme décrit la série de transformations qui passent de la situation initiale à la situation finale, en passant par toute une série de situations intermédiaires. L'algorithme implique la notion de temps. « *Avant* » l'instruction est la pré affirmation. « *Après* » est post-affirmation. Ces assertions décrivent des états (relation entre variables), les instructions passent d'un état à un autre. La complexité inhérente du concept de variable dans le contexte informatique et la rupture qui opère ainsi avec le concept de variable mathématique.

Le tâtonnement est la forme la plus primitive de recherche d'une solution, Jacques Arsac (Arsac, 1980) en a fait une illustration remarquable dont nous nous sommes inspirés pour montrer aux élèves qu'un programme qui fonctionne n'est pas nécessairement un algorithme correct. La principale difficulté est d'éviter de travailler à tâtons, « *On error try again* » ; il convient de donner, dès que possible, du sens à ce que l'on écrit.

Notre démarche est semblable aux actions présentées par Papert (1981) comme favorisant l'accès à la « *pensée procédurale* », c'est-à-dire la capacité de penser à une répétition d'actions telles qu'une procédure configurable et réutilisable dans un traitement plus complexe. Les élèves devraient pouvoir voir la valeur du bloc d'instructions qui peut être répété, qui fait l'objet des tâches T3 et T4.

À partir de la tâche T1, l'opération avancée est répétée 7 fois, afin que nous puissions améliorer les séquences répétées dans les structures répétitives : `REPETER n FOIS`, `POUR` ou `Répéter jusqu'à TANT QUE`

Tâche T3 : additions successives présentées en structure répétitive

- q nombre d'instruction additive répétée ;
- r la distance restante pour arriver à la destination.

Comme nous avons D = $d \times q + r$ donc r = D – $(d \times q)$

Les valeurs de D et d connues, permettent de favoriser ou non la mobilisation du répertoire de la multiplication. Ainsi, on s'intéresse à la combinaison de ces deux variables comme suit : D > $d \times q$

Au cours de cette troisième tâche, nous pouvons introduire deux formes de structure répétitive selon le mode de présentation de l'algorithme ; avec Scratch, où le mode de présentation est visuel, il est plus pratique et intuitif d'utiliser l'instruction de contrôle pour `REPETER n FOIS`, qui résulte directement de la répétition d'actions avancées.

Parce que le nombre de répétitions d'instructions est obtenu par la même opération décrite dans la tâche 1 ou le travail sur papier et crayon, donc avec Scratch (fig 4.11), l'algorithme est écrit comme suit :

$71 = 9 * 7 + 8$ équivaut à `REPETER 7 FOIS  AVANCE 9 PAS et AVANCE 8 PAS`

Afin de respecter les critères (3) et (4), nous avons défini des blocs Scratch spécifique correspondant a une boucle d'addition : `REPETER q FOIS( AVANCER d PAS)`

FIGURE 4.11 – Code d'une boucle d'addition en Scratch.

Pour l'écriture habituelle d'algorithmes en utilisant le langage Algol ou le pseudocode, ou en utilisant le logiciel Algobox, la boucle `POUR` semble un peu plus simple que l'instruction `REPETER JUSQU'A` sans l'instruction `REPETER n FOIS`. Nous utilisons la boucle `POUR` pour réécrire le même algorithme que ci-dessus

`POUR compteur =1 à 7 FAIRE  AVANCER 9 PAS et AVANCER 8 PAS`

Le concept de compteur et le pas d'incrément des itérations intégrées dans la boucle ne sont pas intuitifs chez les enfants, une connaissance préalable des concepts et des affectations variables est généralement nécessaire.

Tâche T4 : soustractions successives présentées en structure répétitive

- q nombre d'instructions soustractives répétées ;
- r la position final du latin .

Comme nous avons D = $d \times q + r$ donc r = D – $(d \times q)$

Les valeurs de D et d connues, permettent de favoriser ou non la mobilisation du répertoire de la multiplication. Ainsi, on s'intéresse à la combinaison de ces deux variables comme suit : D > $d * q$

De cette dernière tâche, nous présentons les deux autres formes de structures répétitives. Si nous ne demandons pas de travaux préparatoires antérieurs et si nous voulons tout program-

mer pour le processus et son interruption, nous avons besoin de nouvelles instructions sanctionnées par les conditions d'arrêt. La suite de l'opération de soustraction se termine lorsque la distance restante est inférieure à la marche. Alors d'une part, lorsquon utilise la structure répétitive `REPETER JUSQU'A`, nous ayons : reculer toujours (Répété reculé) de 9 pas jusqu'à ce que la position du lutin soit plus petite que la marche.

`AVANCER 71 TANT QUE POSITION >9 RECULER 9 PAS (AVANCER -9 PAS)`

Afin de respecter les critères (3) et (4), nous avons défini des blocs Scratch spécifique correspondant a une boucle de soustraction : « répéter q fois (retourner de d pas) »

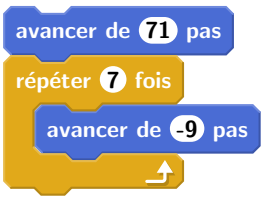


FIGURE 4.12 – Code d'une boucle de soustraction en Scratch.

D'autre part, avec la structure répétitive `TANT QUE`, nous avons ; `TANT QUE` la distance est plus grande que le pas `RECULER 9 PAS (AVANCER -9 PAS) AVANCER 71 REPETER AVANCER -9 PAS JUSQU'A POSITION <9`

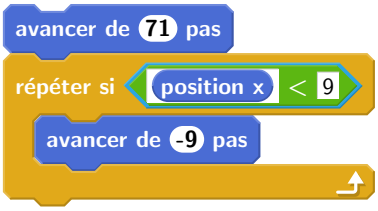


FIGURE 4.13 – Code d'une division avec la structure répétitive en Scratch.

Ces deux structures répétitives, avec condition d'arrêt, dépendent fortement des expressions booléennes.

Ainsi, nous pouvons introduire les activités en vue de s'approprier la connaissance algorithmique et mettre en pratique les connaissances mathématiques comme :

- les critères de divisibilité, donc tester la position du lutin ;
- la fonction modulo, on retourne à la position ;
- la division euclidienne, on retourne à la position et au compteur ;
- le test de primalité (*extension de modulo et divisibilité*) ;
- le PGCD (*extension de modulo et divisibilité*) ;
- les multiples. Toutes les positions retournées du lutin lors de chaque répétition.

Bref, les séquences d'activités en géométrie illustrent de nouvelles techniques ou pratiques pédagogiques permettant l'introduction ou l'enseignement des algorithmes au collège. Les étapes de constrictions des côtés d'une figure géométrique offrent des méthodes très tangibles pour l'appropriation de séquences d'instruction sur l'enseignement des concepts algorithmique. Les constructions géométriques comme les polygones réguliers sont aussi les meilleurs exemples pour familia-

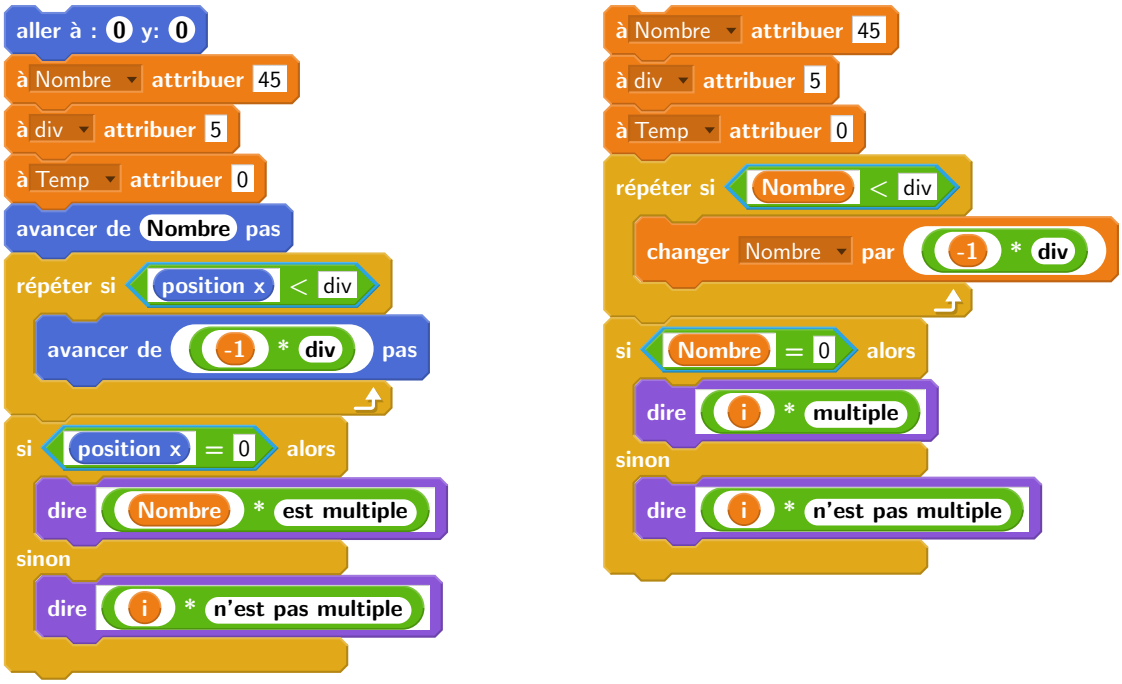


FIGURE 4.14 – Programmes de test de divisibilité en Scratch.

riser les apprenants avec la recherche du nombre total d'itérations et l'initialisation de l'invariant d'un bloc d'instructions répétitif. Les aspects preuve et finitude sont observables, les apprentis en algorithmique peuvent facilement vérifier la justesse de leurs constructions ou peuvent identifier de manière très intuitive (déboguer) leurs erreurs (bugs). L'ajout de pauses entre les instructions est fortement recommandé, cette dernière montre de manière plus visuelle, étape par étape, les séquences d'instructions.

L'introduction des structures conditionnelles ne semble pas facile en géométrie, donc une certaine activité arithmétique est nécessaire pour combler cette lacune. Le programme d'arithmétique fait en sixième est une mine d'or et permet de tirer des exemples ou des activités favorables aux structures conditionnelles et aux structures répétitives avec conditions d'arrêt. En se servant des critères de divisibilités, test de primalité, division exacte, etc. suffisent à introduire l'algorithmique.

Dans toutes ces activités, il ne s'agit pas de familiariser les apprenants avec les résolutions de problèmes mathématiques, mais de leur donner les connaissances de base des algorithmes, à savoir : comment ordonner les instructions ? Comment simplifier une séquence d'instructions répétitives ? Comment choisir la bonne structure algorithmique adéquate ?

4.5.2 Avantages et inconvénients de Scratch

Avantages de Scratch

Si on dresse un tableau comparatif des divers langages énumérés dans cette étude, avec les critères cités et avec quelques tests faits en classe, Scratch est idéal pour mettre en œuvre de manière ludique et rapide des algorithmes au collège. En effet :

– sa prise en main par les élèves est quasi-immédiate après une très courte démonstration au

vidéoprojecteur ;
- l'environnement est simple et efficace. Il est constitué de trois parties, on a les instructions à disposition, la fenêtre du programme et la fenêtre d'exécution du programme dans la même fenêtre ;
- il n'y a pas de syntaxe à connaître ni à écrire, on déplace simplement des lignes d'instructions qui s'imbriquent par aimantation ;
- un simple doubleclic sur une instruction permet de l'exécuter ;
- les élèves sont séduits par l'interface ;
- le débogage c'est très facile pour deux raisons. En premier, exécution en mode pas-à-pas dont on peut régler la vitesse puis en second les blocs de programme peuvent être détachés les uns des autres et testés par simple doubleclic dessus. On pourra donc effectuer un débogage par bloc ;
- c'est un excellent logiciel pour s'initier à la programmation (on peut faire travailler les élèves par groupes pour effectuer de petits projets de programmation comme des simulations de promenades aléatoires) ;
- il permet de faire des rendus visuels intéressants grâce à des scènes et des costumes. On pourra alors proposer aux élèves de programmer des jeux par groupe.

Plusieurs outils d'enseignement des algorithmes sont disponibles sur Internet. Il convient de noter les compilateurs de pseudocodes Visualg (De Souza, 2005) et IDE portugais (MANSO, 2005). Ces outils permettent à l'élève de modifier l'algorithme, d'effectuer la correction syntaxique et sémantique (compilation) puis l'exécuter.

Inconvénients de Scratch

Après quelques analyses, Scratch présente un grand nombre d'inconvénients. En voici une petite liste :
- il est lent ;
- l'éditeur est pénible (codes couleurs pas très pratiques, pas de touche « annuler » ni « copier-coller », la méthode du « glisser-coller » est vite fastidieuse si on veut faire des choses un peu élaborées, les sauvegardes automatiques permanentes entérinent toutes les erreurs...) ;
- il fait par défaut des choses qu'on ne demande pas (par exemple : le simple fait de créer un « lutin » (ou sprite) fait que celui-ci sera automatiquement affiché à l'écran : il faut préciser « cacher » si on ne le veut pas) ;
- certaines actions (comme « si lutin touché ? alors ») fonctionnent mal ou de façon très imprécise.

Mais surtout :
- Scratch exécute toutes les boucles et toutes les instructions en même temps : on trouve que cela n'est pas propice à un apprentissage structuré de la logique de déroulement d'un programme ;
- les instructions sont éclatées en plusieurs fenêtres : une par lutin. Il est très compliqué de lire un tel programme, d'en avoir une vision d'ensemble ;
- pour la même raison, mais aussi à cause du système des « briques d'instruction » un pro-

gramme ne peut pas imprimer.

Enfin, Scratch réalise beaucoup de choses en très peu d'instructions (c'est un langage de très haut niveau). Les élèves auront l'impression de pouvoir programmer « Call of Duty » en quelques clics seulement ! Ça peut être attrayant dans un premier temps, mais c'est vite agaçant, car quand on veut faire autre chose que ce que la logique Scratch suggère, on est amené à devoir bidouiller, contourner, tricher et, finalement, très souvent renoncer.

4.6 Appréciation de l'acquisition par les élèves

L'évaluation des acquis scolaires des élèves joue un rôle important dans le processus des apprentissages. Elle permet de voir si les objectifs sont convenablement atteints et aide à faire les divers ajustements nécessaires.

4.6.1 Critères d'évaluations

Dans notre étude, nous avons utilisé les modèles d'évaluations définis dans PISA 2015, P21 et le Programme de formation de l'école québécoise (PFÉQ) qui évalue la compétence d'usage des technologies de l'information (niveaux 1 et 2 du référentiel UNESCO) et de la communication en trois compétences : compétence à la résolution de problèmes, compétence en créativité et la compétence collaboration. Et l'évaluation de compétence en pensée informatique décrit par les équipes de Scratch du MIT et par l'initiative *Computating At School*.

Évaluation de la compétence pensée informatique

La pensée informatique est un ensemble de stratégies cognitives et métacognitives liées à la modélisation de connaissances et de processus, à l'abstraction, à l'algorithmique, à l'identification, à la décomposition ainsi qu'à l'organisation de structures complexes et de suites logiques.

C 1 : Comprendre la logique d'un algorithme.

C 2 : Concevoir et développer un programme informatique.

C 3 : Organiser des données de manière efficiente.

C 4 : Comprendre le fonctionnement d'un appareil numérique.

C 5 : Concevoir et développer des projets créatifs par le biais de la programmation.

Autres composantes et critères pour évaluer cette compétence : Au Royaume-Uni, l'initiative Computating At School(http ://barefootcas.org.uk/) identifie six concepts et cinq processus pour le développement et l'évaluation de la pensée informatique. Au niveau des concepts Barefoot identifie la logique, les algorithmes la structure des instructions et de l'exécution du code (par exemple les structures « si alors » ; « répéter X fois », « tant que »), la décomposition, les patterns ou patrons, l'abstraction et l'évaluation. Au niveau des processus, ils identifient le « bidouillage » ou « bricolage » informatique (tinkering), la création, le débogage ou debugging(la résolution de bogues ou dysfonctionnements informatiques), la persévérance et la collaboration.

Pour l'équipe Scratch du MIT (Brennan, Chung et Hawson, 2011 ; Brennanet Resnick, 2012), la pensée informatique est

– la capacité à comprendre et faire usage des différents concepts en lien avec la programmation : séquences, boucles, processus en parallèle, évènements, conditions (si...alors), opérateurs, variables et listes ;
– la capacité à comprendre et faire usage des différentes pratiques en lien avec la programmation : l'approche itérative et incrémentale, les tests et corrections d'erreurs, la réutilisation du code, la modularisation et l'abstraction.

Évaluation de la compétence résolution de problèmes

La résolution de problèmes est la capacité d'identifier une situation-problème pour laquelle le processus et la solution ne sont pas connus d'avance. C'est également la capacité de déterminer une solution, de la construire et de la mettre en œuvre de manière efficace.

C 1 : Analyser les éléments de la situation.

C 2 : Explorer une variété de solutions.

C 3 : Mettre à l'essai des pistes de solutions ; évaluer sa démarche.

Modèles	Critères
PISA 2015	Exploration et compréhension Représentation et formulation Planification et exécution Suivi et réfléxion
PFÉQ	Pertinence des éléments identifiés Formulation de solutions plausibles et imaginatives Utilisation de stratégies efficaces et variées Dynamisme de la démarche Reconnaissance des éléments de réussite et de difficulté Transposition des stratégies à d'autres situations
P21	Résolution de différents types de problèmes non conventionnels de manière innovante Questionnements qui permettent d'explorer la situation-problème et d'avancer vers de meilleures solutions Argumentation dans le but de comprendre Prise de décisions complexes Compréhension des interconnexions entre des systèmes Cadrage, analyse et synthèse de l'information pour résoudre des problèmes

TABLEAU 4.1 – Autres composantes et critères d'évaluation pour la compétence de résolution de problèmes.

Évaluation de la compétence créativité

La créativité est un processus de conception d'une solution jugée nouvelle, innovante et pertinente pour répondre à une situation-problème.

C 1 : Explorer une variété de solutions nouvelles.

C 2 : Utiliser des sources d'inspiration pour orienter la recherche créative.

C 3 : Sélectionner une solution en tenant compte du contexte de la situation-problème.

Autres composantes et critères pour évaluer cette compétence :

Modèles	Critères
PFÉQ	Appropriation des éléments de la situation Diversité des possibilités de réalisation inventoriées Originalité des liens entre les éléments et dynamisme du processus Détermination d'améliorations possibles dans le processus d'innovation
P21	Développement d'idées diverses qui tiennent compte des besoins et des contraintes de la réalité Création d'idées nouvelles et pertinentes Élaboration, raffinement, analyse et évaluation des idées dans le but de les améliorer Capacité à communiquer des idées de manière efficace Ouverture à différentes perspectives et capacité à intégrer des rétroactions dans le travail commun Conception de la créativité comme un processus d'amélioration progressive et prise en compte des échecs comme une opportunité d'apprentissage

TABLEAU 4.2 – Évaluation de la compétence créativité.

Évaluation de la compétence collaboration

La collaboration est la capacité de développer une compréhension partagée et de travailler de manière coordonnée avec plusieurs personnes dans un objectif commun.

C 1 : Capacité à identifier la situation-problème et définir en équipe un objectif commun.

C 2 : Établir et maintenir une compréhension et une organisation partagée.

C 3 : Développer une compréhension des savoirs, compétences, forces et limitations des autres membres de l'équipe pour organiser les tâches vers l'objectif commun.

C 4 : Savoir gérer les difficultés du travail en équipe dans le respect et la recherche de solutions.

Autres composantes et pour évaluer cette compétence :

Modèles	Critères
PISA 2015	Établir et maintenir une compréhension partagée Entreprendre des actions appropriées pour résoudre le problème Établir et maintenir l'organisation de l'équipe
PFÉQ	Reconnaissance des besoins des autres Attitudes et comportements adaptés Engagement dans la réalisation d'un travail de groupe Contribution à l'amélioration des modalités d'un travail de groupe
P21	Prise de responsabilité individuelle sur le processus d'apprentissage Optimisation de la performance de l'équipe au cours de la collaboration Gestion des relations interpersonnelles

TABLEAU 4.3 – Évaluation de la compétence collaboration.

4.6.2 Indicateurs d'évaluation

Tout d'abord, il faut remarquer que l'informatique, en tant que domaine scientifique et technique, est un domaine complexe, dont on peut avoir différentes perceptions (liens avec les maths, liens avec les machines, liens avec les sciences humaines, etc.). C'est également un domaine qui

en recoupe beaucoup d'autres, et c'est parfois difficile de faire la part des choses (par exemple, la notion d'algorithme, ou la logique relèvent de l'informatique et d'autres domaines.). C'est enfin un domaine dont les dimensions scientifiques (notions abstraites, principes, méthodes) et techniques (ordinateur, langage de programmation) se mêlent et, parfois, se confondent. Par ailleurs, il est difficile d'utiliser les indicateurs existants pour évaluer les connaissances informatiques.

En premier lieu, leur compétence à la résolution de problèmes, c'est-à-dire, les connaissances de base que les étudiants doivent posséder pour pouvoir passer a une activité ; analyse des éléments de la situation, exploration d'une variété de solutions et la mise à l'essai des pistes de solutions ; évaluer sa démarche. Il est donc possible de mesurer ces connaissances et attribuer une note afin que, comme dans toute autre matière, trouver la bonne réponse induisent une note correspondante.

Deuxièmement, leur capacité computationnelle, c'est-à-dire de traduire une démarche ou une résolution trouvée en un algorithme. Difficile d'établir une échelle de connaissances devant la multitude de formes d'algorithmes possibles, en plus aucun algorithme n'est considéré efficace pour résoudre un problème donné et puis, dans les sciences des algorithmes, un algorithme est jugé pertinent par sa faible complexité, c'est-à-dire sa capacité à trouver des solutions dans un minimum de temps.

Troisièmement, leur capacité de codage et de débogage pour utiliser et gérer les objets qui les entourent afin qu'ils puissent contrôler l'ordinateur pour exécuter le programme. Les critères nécessaires pour noter une activité deviennent de plus en plus difficiles, car le meilleur programme est celui qui résout le problème avec un minimum de ressources (mémoire ou variable utilisée) et avec une interaction simple avec les entrées-sorties.

Et enfin, leur compétence dans le travail collaboratif, dans la communication et l'entente sur le dialogue. Les attitudes et les caractéristiques que doit avoir un opérateur informatique : client/-serveur, commande/réponse, capacité de communication entrée/sortie. C'est aussi une bonne occasion de présenter ou de parler de la notion de programmation parallèle ainsi que de la programmation évènementielle.

Alors, nous avons emprunté et avons utilisé la courbe d'évaluation ou la courbe d'apprentissage. Il s'agit de la technique d'évaluation utilisée par les psychologues pour mesurer le degré d'apprentissage et la vitesse de compréhension d'un sujet. Alors, nous avons utilisé une mesure bidimensionnelle stockant le score obtenu et la durée et l'avons représenté sur un repère orthonormé où l'axe des x représentent la durée de l'activité et l'axe des y représente le cumul des scores obtenus par l'apprenant.

La courbe ainsi obtenue décrit notamment une relation entre le score cumulé d'une activité et le temps de réalisation nécessaire pour accomplir chaque unité de l'évaluation.

4.7 Méthode d'évaluation utilisant une courbe d'apprentissage

Nous pensions que le processus d'évaluation était défectueux et inapproprié, et nous recherchions des moyens d'améliorer et de développer de nouvelles méthodes pour évaluer les nouvelles connaissances en fonction de la « *courbe d'apprentissage* ».

Par exemple, lors de l'évaluation des connaissances de deux élèves qui ont été soumis à la

résolution d'un problème d'algorithmique, le premier élève a terminé en seulement 35 minutes et le second a terminé en 50 minutes. Ils ont tous deux résolu le problème avec succès. L'évaluation des connaissances que nous utilisons classe ces élèves d'après leurs succès et leurs notes, avec les mêmes qualifications, sont toutes les deux dans la catégorie des bons élèves.

Cependant, la recherche montre que l'évaluation des connaissances a de nombreux effets, sans parler de « *l'effet Halo* » ; effet qui favorise les élèves qui ont plus de relations avec les enseignants.

Et surtout dans le concept d'algorithmique, on parle de la complexité algorithmique, ce qui signifie qu'en algorithmique, il ne suffit pas d'obtenir la réponse exacte, mais il doit être capable de donner cette réponse dans un délai très raisonnable et avec un très faible coût de mémoire.

Ainsi, l'élève qui obtient rapidement la bonne réponse ne devra pas tomber dans la même qualification que le dernier. C'est pourquoi nous avons proposé cette méthode d'évaluation basée sur la courbe d'évaluation afin de mesurer la connaissance et à la fois la performance des apprenants.

4.7.1 Essai de définition

Le terme courbe d'apprentissage est utilisé de deux manières principales : lorsque la même tâche est répétée dans une série d'essais ou lorsqu'un ensemble de connaissances est acquis au fil du temps. Hermann Ebbinghaus a été le premier à décrire la courbe d'apprentissage en 1885 (Shakow, 1930), dans le domaine de la psychologie de l'apprentissage, bien que le nom n'ait été utilisé qu'en 1903. En 1936, Theodore Paul Wright (Wright, 1939) a décrit l'effet de l'apprentissage sur les coûts de production dans l'industrie aéronautique. Cette forme, dans laquelle le coût unitaire est représenté par rapport à la production totale, est parfois appelée courbe d'expérience.

Une courbe d'apprentissage est une représentation graphique de la manière dont une augmentation de l'apprentissage (mesurée sur l'axe vertical) provient d'une plus grande expérience (axe horizontal) .

Ainsi, dans le domaine de l'éducation, lors de l'apprentissage la courbe sert à déterminer le rythme qui convient le mieux à un sujet donné et lors de l'évaluation la courbe sert à déterminer les succès des apprenants, mesurer à la fois les savoirs et qualifier leur compétence. Elle sert aussi à analyser et à repérer les difficultés des élèves, durant ces activités.

4.7.2 Observations empiriques

Ci-dessous, nous montrons l'expérimentation et les illustrations utilisées pour prouver que la courbe d'évaluation est robuste et utilisable.

L'expérience a été réalisée dans la classe de quatrième d'un établissement privé avec 45 élèves. La classe a été choisie selon la composition suivante : 24 élèves ont commencé un cours d'informatique à partir de la classe de sixième, 21 élèves ont commencé leurs cours d'informatique à partir de la classe de cinquième et 20 élèves sont de nouveaux apprentis. Conformément au concept constructiviste de l'apprentissage, nous avons privilégié la confrontation des élèves aux situations

problématiques, en créant une classe en 14 groupes : 12 groupes de quatre élèves et un groupe de cinq élèves.

	Groupes	Classe	Notions enseignés
Initiation en informatique	5, 6, 8, 9	6e	Historique, Logicièl, Ordinateur, Souris, Clavier, Environement Windows et MS Paint
Traiment de text et tableur	4 à 11	5e	Stucture du Text, Mise en forme, MS Word et MS Excel
PAO, PréAO et Algo	1 à 11	4e	MS Publisher, MS Power Point programmation avec Scratch

TABLEAU 4.4 – Tableau de répartition des groupes par rapport à leur antécédent en informatique.

Nous avons demandé aux étudiants de faire une heure de travaux pratiques, composée de cinq séries d'exercices, dont l'échelle est de 4 points chacun. Après une heure de travail, dans une situation didactique et sous la supervision d'un enseignant, nous avons enregistré l'heure à laquelle les groupes ont terminé leurs tâches de base. Le tableau 4.5 ci-dessous répertorie le temps d'exécution et les résultats obtenus

	Gr1	Gr2	Gr3	Gr4	Gr5	Gr6	Gr7	Gr8	Gr9	Gr10	Gr11	Gr12	Gr13	Gr14
Q1	40	38,5	36,5	31,5	25	21,5	21,5	20	18,5	10	10	10	15	13
Q2	x	x	51,5	45	35	28,5	35	27	26	20	20	20	22	17,5
Q3	x	x	x	x	42,5	32	50	33	30	37,5	39	27	25	22
Q4	x	x	x	x	60	36,5	x	39	34,5	42,5	50	32,5	29	24
Q5	x	x	x	x	x	50	x	60	45	x	x	40	40	35

TABLEAU 4.5 – Notes obtenues par chaque groupe lors de l'expérimentation.

Interpretation

Lorsque l'on essaie de représenter la variation de score (note) obtenu en fonction de temps dans un repère orthogonal, les courbes ainsi obtenues sont des familles de courbes en "**S**", appelées courbes d'évaluation ou courbes d'apprentissage.

Selon leur forme, elles sont également appelées courbes en S, mais selon leur expression fonctionnelle, elles peuvent être appelées courbes sigmoïdes. Le terme courbes en S représente la tendance de la courbe cumulative à former un S peu profond ; plus plat au début, plus raide au milieu et s'aplatissant de nouveau vers la fin.

Le psychologue Arthur Bills a donné une description plus détaillée des courbes d'apprentissage en 1934. Il a également évoqué les propriétés des différents types de courbes d'apprentissage, telles que l'accélération négative, une accélération positive, plateaux et ogivales courbes.

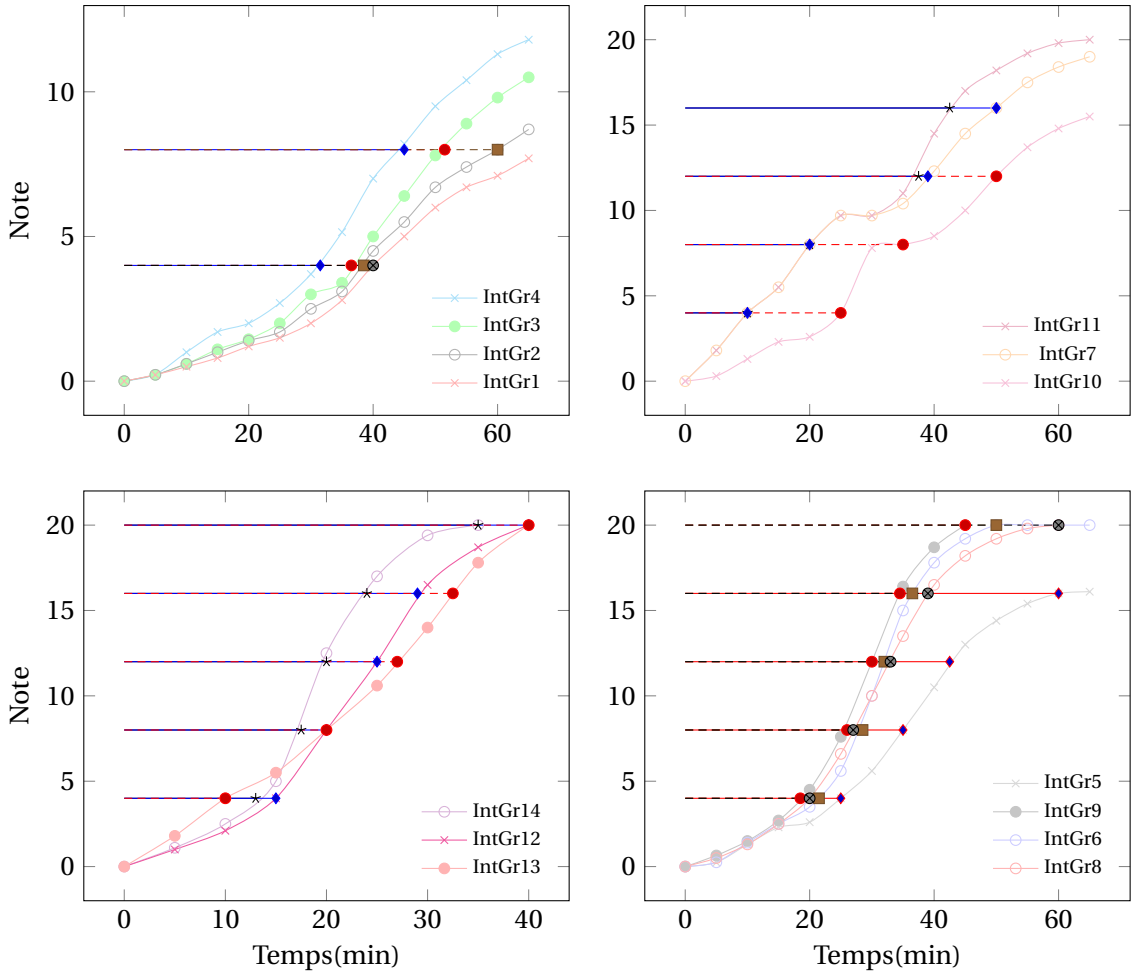


FIGURE 4.15 – Courbes d'évaluations des groupes obtenues après l'expérimentation.

Il y a une confusion, un glissement sémantique, plus précisément un basculement sémantique concernant la signification des termes qui qualifient la courbe d'apprentissage (... douce, rapide, abrupte, raide ...), selon que l'auteur les attribue à la pente de la courbe ou à la difficulté d'apprentissage.

Dans l'article « *Paroscopic Colon Resection Early in the Learning Curve - What Is the Appropriate Setting ?* » à la section « Discussions » le Dr C. Daniel Smith (Atlanta, Géorgie) remarque à propos de l'utilisation du terme « courbe d'apprentissage abrupte » : « *Tout d'abord, la sémantique. Une courbe d'apprentissage abrupte est celle où vous gagnez la compétence avec un petit nombre d'essais. Cela signifie que la courbe est raide. Je pense que sémantiquement nous parlons vraiment d'une courbe d'apprentissage prolongée ou longue. Je sais que c'est une distinction subtile, mais je ne peux pas rater l'occasion de faire ce point.* »

Dans l'article de Ben Zimmer « *A "Steep Learning Curve" for "Downton Abbey"* » dans lequel il reprend la signification de la phrase "une courbe d'apprentissage abrupte" de 1885 (courbe introduite par le psychologue allemand Hermann Ebbinghaus). Relevons le passage : « *Si l'on considère une courbe d'apprentissage dans lequel une certaine mesure de compétence est sur l'axe des y et le nombre de tentatives d'apprentissage au fil du temps est sur l'axe des x, alors que représente une*

pente raide ? Une ascension rapide à un niveau plus élevé de l'apprentissage. Et pourtant, la façon dont la courbe d'apprentissage abrupte est utilisée dans les années 70 suggère le contraire : une ascension ardue plutôt qu'une ascension rapide et facile. Le sens est devenu ainsi inversé par rapport à celui de la technique antérieure ».

Expression analytique

La méthodologie que nous avons utilisée c'est l'interpolation non linéaire, utilisant les modèles de fonctions logistiques. Les paramètres du modèle, les erreurs et les équations du modèle trouvé sont donnés dans les tableaux (4.6 et 4.7)

$$\frac{pr1}{1+\mathrm{E}xp(-pr2\times \mathrm{T}-pr3)}$$

où $pr1$ performances initiale, $pr2$ performances asymptotiques et $pr3$ est un coefficient de taux constant.

Coef d'ajustement :

	Gr1	Gr2	Gr3	Gr4	Gr5	Gr6	Gr7	Gr8	Gr9	Gr10	Gr11	Gr12	Gr13	Gr14
Observations	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00	13,00
DDL	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00
R^2	0,99	0,99	0,99	0,99	0,99	0,99	0,97	1,00	0,99	0,97	0,96	0,99	0,98	0,98
SCE	0,20	0,28	0,39	0,57	1,18	2,46	7,21	0,28	0,72	11,99	13,56	2,94	10,92	13,92
MCE	0,02	0,03	0,04	0,06	0,12	0,25	0,72	0,03	0,07	1,20	1,35	0,29	1,09	1,39
RMCE	0,14	0,17	0,19	0,24	0,34	0,49	0,85	0,17	0,27	1,09	1,16	0,54	1,04	1,18
Itérations	12,00	10,00	11,00	9,00	12,00	11,00	9,00	9,00	9,00	9,00	8,00	12,00	7,00	9,00

Statdescriptives :

Minimum	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Maximum	7,10	8,00	9,80	11,30	16,00	21,36	14,80	20,00	21,61	19,80	18,40	23,67	21,33	20,90
Moyenne	2,91	3,28	3,82	4,84	7,18	10,59	6,56	10,16	11,39	10,64	9,83	13,27	12,89	13,46
Ecart-type	2,56	2,84	3,42	4,01	6,02	8,68	5,14	8,02	8,81	6,71	5,88	8,73	8,47	8,79

TABLEAU 4.6 – Les parametres statiqtiques obtenus apres la regression

4.7.3 Observations analytique

L'équation d'apprentissage exponentielle a été dérivée analytiquement par plusieurs chercheurs (Estes, 1950; Thurstone, 1919) et est l'une des équations standards pour décrire l'amélioration dans l'exécution de tâches avec la pratique (Heathcote et al., 2000)

$$\mathrm{P}_n = \mathrm{P}_\infty - (\mathrm{P}_\infty - \mathrm{P}_0).e^{-\alpha.\mathrm{S}_n}$$

où n représente le numéro de l'essai, P_n la mesure de performance à n essai et p_0, p_∞ sont respectivement les performances initiales et asymptotiques, et α est un coefficient de taux constant. La concavité de P_n implique une amélioration décroissante de façon monotone $(\Delta_n = \mathrm{P}_n - \mathrm{P}_{n-1} < \Delta_{n-1})$.

Cependant, un comportement sigmoïde dans lequel l'amélioration initialement les augmentations diminuent ensuite a été suggéré de façon persistante sur des observations empiriques

(Culler, 1928; Culler & Girden, 1951; Gallistel et al., 2004; Leibowitz et al., 2010) ou sur une dérivation analytique à partir d'hypothèses sur le processus sous-jacent de l'apprentissage (Gulliksen, 1934; Mazur & Hastie, 1978; Newell et al., 2001).

$$\begin{aligned} P_{n+1} &= P_\infty - (P_\infty - P_0).e^{-\alpha.S_{n+1}} \\ &= P_\infty - (P_\infty - P_0).e^{-\alpha.S_n}.e^{-\alpha.P_n} \end{aligned}$$

$$P_{n+1} = P_\infty - (P_\infty - P_0).e^{-\alpha.P_n}$$

$$p(t) = P_\infty - (P_\infty - P_0).e^{-\alpha.s(t)}$$

p (t) est continu, il est aussi différentiable

$$S(t) = \int_0^t p(x)dx$$

$$p' = \alpha p(p_\infty - p_0) \quad p(0) = p_0$$

$$\begin{aligned} p(t) &= \frac{p_0 \times p_\infty}{p_0(-P_0 - P_\infty).e^{-\alpha.p_\infty.t}} \\ &= \begin{cases} 0 & p_0 = 0 \\ \dfrac{p_\infty}{1 + e^{-\alpha.p_\infty.t} + C} & p_0 \neq 0 \end{cases} \end{aligned}$$

$C = ln|(p_\infty - p_0)/p_0|$

4.7.4 Le facteur de pente ou la pente de la colline

L'inclinaison est quantifiée par la pente de la colline, également appelée facteur de pente.

1. une courbe dose-réponse avec une pente standard a une pente de Hill de 1,0 ;
2. une courbe plus raide a un facteur de pente plus élevé et ;
3. une courbe moins profonde a un facteur de pente plus faible.

4.7.5 Propriétés

Selon l'échelle de compétence, en pratique, on constate effectivement l'existence de quatre grandes catégories d'utilisateurs :

- les apprenants qui maîtrisent bien le concept et ont un niveau de compétence très élevée,
- les apprenants qui maîtrisent bien le concept et ont un niveau de compétence moyenne,
- les apprenants qui maîtrisent un peu le concept avec un niveau de compétences pouvant être améliorées,
- les apprenants qui maîtrisent mal le concept ou ont un faible niveau de compétence.

Différentes phases.

La phase de latence : La courbe classique indique que l'apprentissage, d'abord lent, se fait rapidement, puis se stabilise à un certain niveau. Les termes : « *continue, douce, facile, progressive, rapide*

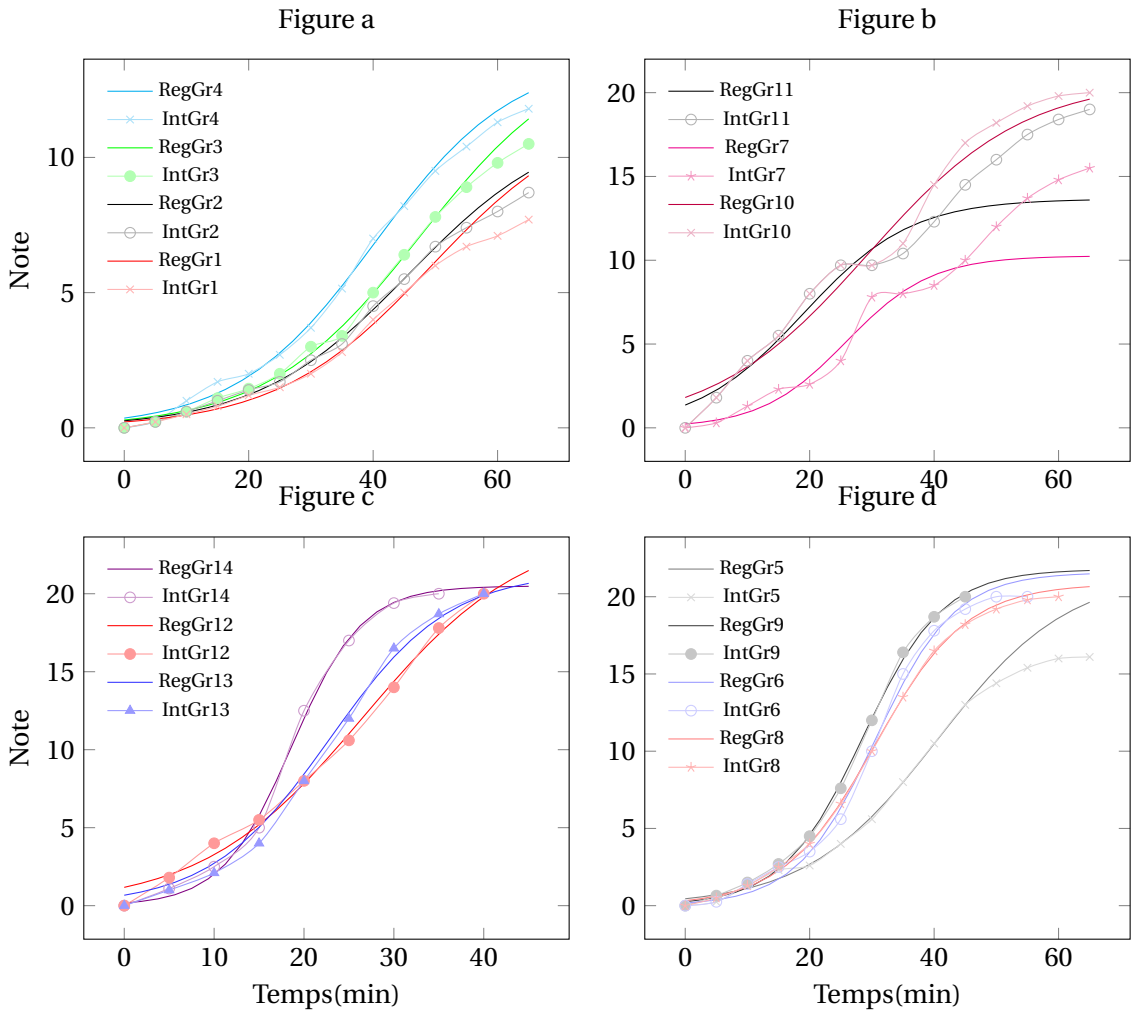


FIGURE 4.16 – Courbes d'évaluation théoriques et observées de 13 groupes.

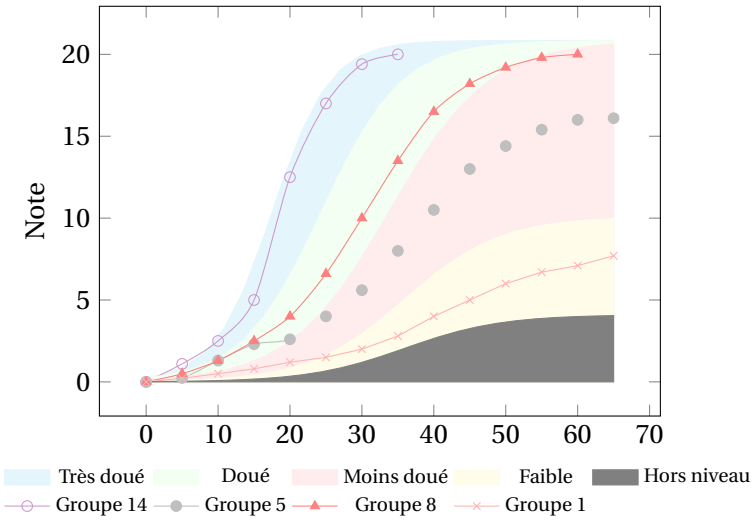


FIGURE 4.17 – Tendance de la courbe cumulative.

... » qui sont attribués à la courbe d'apprentissage qualifie la nature de l'apprentissage, ils correspondent à une courbe d'apprentissage dont la pente est forte, sa croissance progresse rapidement.

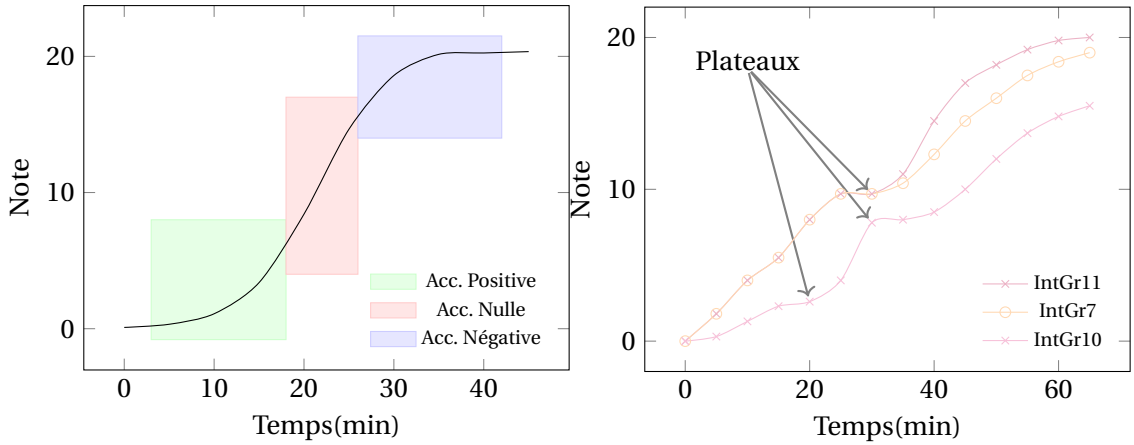


FIGURE 4.18 – Différents types de courbes (à gauche) et courbes avec plateaux (à droite).

La phase linéaire : Il se produit lorsque le corps et l'esprit ont intégré avec succès les éléments travaillés et l'apprentissage donne ses fruits

La phase de ralentissement : L'expression familière « *une courbe d'apprentissage abrupte* » signifie que l'activité est difficile à apprendre, même si une courbe d'apprentissage avec un départ rapide représente en réalité un progrès rapide.

Différents types de courbes

Une courbe d'apprentissage linéaire : sous forme de droite ascendante, l'apprentissage ou l'évaluation présentent convenablement au niveau des apprenants.

Une courbe d'apprentissage en escalier : On franchit rapidement un palier énorme, mais ensuite on stagne à un niveau, l'apprentissage ou l'évaluation présente des obstacles chez les apprenants.

Une courbe d'apprentissage progressive : Elle est cher aux apprenants et demande, certes, un peu d'effort dès le départ, mais en contrepartie, elle permet de maîtriser le concept, et évite ensuite les pics brutaux de compétences nécessaires qui peuvent bloquer toute évolution.

Une courbe d'apprentissage avec plateaux : On observe assez souvent des paliers dans la courbe d'apprentissage (fig. 4.18). Ces « *plateaux* » signifient presque toujours que les apprenants passent brusquement à une nouvelle connaissance et/ou à une meilleure technique après un coup de pouce. La courbe de la figure 4.18(à droite) indique clairement qu'il faut toujours aider ou débloquer les apprenants.

height	Paramètre	Valeur	Erreur standard	Equation du modèle
Gr1	pr1	-3,915	0,140	$\frac{8,4513}{1+Exp(-0,09478\times T+3,9153)}$
	pr2	0,095	0,005	
	pr3	8,451	0,346	
Gr2	pr1	-3,746	0,132	$\frac{9,6873}{1+Exp(-0,08941\times T+3,7460)}$
	pr2	0,089	0,005	
	pr3	9,687	0,456	
Gr3	pr1	-3,842	0,125	$\frac{12,4694}{1+Exp(-0,08627\times T+3,8424)}$
	pr2	0,086	0,005	
	pr3	12,469	0,695	
Gr4	pr1	-3,604	0,135	$\frac{12,8548}{1+Exp(-0,09268\times T+3,6037)}$
	pr2	0,093	0,005	
	pr3	12,855	0,497	
Gr5	pr1	-3,914	0,162	$\frac{17,4099}{1+Exp(-0,1085\times T+3,9136)}$
	pr2	0,109	0,006	
	pr3	17,410	0,480	
Gr6	pr1	-4,815	0,252	$\frac{21,4081}{1+Exp(-0,1579\times T+4,8153)}$
	pr2	0,158	0,009	
	pr3	21,408	0,348	
Gr7	pr1	-3,151	0,307	$\frac{16,3485}{1+Exp(-0,08535\times T+3,1508)}$
	pr2	0,085	0,013	
	pr3	16,349	1,712	
Gr8	pr1	-4,207	0,075	$\frac{20,4115}{1+Exp(-0,1396\times T+4,2065)}$
	pr2	0,140	0,003	
	pr3	20,412	0,130	
Gr9	pr1	-4,443	0,121	$\frac{21,7696}{1+Exp(-0,1557\times T+4,4428)}$
	pr2	0,156	0,005	
	pr3	21,770	0,176	
Gr10	pr1	-2,387	0,208	$\frac{22,1978}{1+Exp(-0,07625\times T+2,3870)}$
	pr2	0,076	0,011	
	pr3	22,198	1,872	
Gr11	pr1	-2,156	0,212	$\frac{20,3771}{1+Exp(-0,06943\times T+2,1557)}$
	pr2	0,069	0,012	
	pr3	20,377	2,268	
Gr12	pr1	-2,987	0,133	$\frac{24,2025}{1+Exp(-0,1123\times T+2,9869)}$
	pr2	0,112	0,006	
	pr3	24,203	0,446	
Gr13	pr1	-3,433	0,360	$\frac{21,3447}{1+Exp(-0,1503\times T+3,4330)}$
	pr2	0,150	0,017	
	pr3	21,345	0,586	
Gr14	pr1	-5,390	0,796	$\frac{20,8566}{1+Exp(-0,2563\times T+5,3903)}$
	pr2	0,256	0,038	
	pr3	20,857	0,506	

TABLEAU 4.7 – Paramètres et équations de modèles des courbes d'apprentissage des 13 groupes

4.8 Méthode d'Analyse des résultats de manière empirique

4.8.1 Attentes des élèves sur l'utilité de l'informatique.

En début d'année scolaire, pour les deux établissements, nous avons pris deux classes de quatrième dans chacune. Et lors de la prise de contact avec les élèves, on leur a demandé leurs attentes quant à l'utilité de l'ordinateur, où chacun d'eux leur a permis de répondre d un à trois rôles.

Après l'analyse de la feuille de réponses, nous avons regroupé leurs réponses en 6 catégories ;
Usage multimédia : Clip vidéo ou lecteur de film, montage vidéo ou photo, gravure de CD / DVD.
Usage des télécommunications ou d'Internet : pour toute réponse de nature relationnelle à distance sur les réseaux sociaux, ou messagerie.
Usage bureautique : l'utilisation habituelle de l'ordinateur au bureau ; saisie, mise en forme et impression.
Usage ludique ou jeux : toutes sortes de divertissements sur ordinateur,
Usage pédagogique : Corresponds aux réponses centrées sur l'aspect pédagogique et didactique.
Autre réponse : toute autre catégorie des réponses.

4.8.2 Prise en compte des facteurs environnementaux pouvant influencer les étudiants dans leur comportement.

Trois mois après l'introduction au cours d'informatique, toujours sur un enseignement déconnecté, des plaintes et des avis ont été reçus par le personnel enseignant et administratif, concernant l'augmentation du nombre d'élèves restés en dehors de la classe, en retard et punis.

L'analyse du cahier de registre de présence nous a permis de constater que ce défaut provenait principalement des cours d'informatique et d'EPS. Le surveillant général a déclaré que la plupart des billets d'entrée qu'il avait signés étaient, presque toujours pour les absences ou les retards pendant ces deux cours.
Nous avions engagé une analyse des attitudes et de leurs comportements en classe afin de pouvoir vérifier ces informations.

Nous les avons classées par les qualificatifs suivants :
Retard : les élèves qui entrent plus de cinq minutes après le début du cours
Pas de livre : les élèves qui n'ont pas encore de livre d'informatique, après trois mois de la rentrée
Paresse : les élèves qui laissent ou oublient intentionnellement leur livre d'informatique ou tenue de sport pour ne pas participer aux cours.
Pas intéressé : les élèves qui oublient leur livre d'informatique parce qu'ils détestent le cours d'informatique. Ou les étudiants osent dire qu'ils ne sont pas intéressés par le cours d'informatique.
Pas de tenue : les élèves qui n'ont pas apporté leur tenue de sport.
Dispensé : les élèves actuellement dispensés de sport pour des raisons de santé.

4.8.3 Autres facteurs enregistrés

Au début du deuxième trimestre, nous étions au courant de toutes les plaintes reçues concernant l'augmentation des retards, perturbations, bruits, abcès, lors de l'enseignement des cours

d'informatique et d'EPS et ce par des reçus des enseignants et des administrateurs scolaires. Il faut reconnaître que les étudiants étaient déçus de la façon dont les cours d'informatique étaient dispensés

Nous avons donc décidé de faire des tests et des analyses sur le comportement des élèves. L'étude a été immédiatement menée dans une École qui avait travaillé avec nous dans cette recherche, dans une école ayant deux catégories d'apprenants ; la première catégorie est une classe d'élèves qui ont suivi des cours d'informatique débranchés (*sans ordinateur*) et la deuxième catégorie est une classe d'élèves qui ont suivi des cours d'informatique branchés (*avec ordinateur dans la salle informatique*).

L'étude faite dans ces deux classes était similaire, les élèves étaient autorisées à travailler librement par groupe et les enseignants tentaient de ne fournir aucune instruction ou rétroaction pendant la supervision. Pendant ces cours, nous avons relevé le nombre des élèves toutes les 15 minutes, suivant leurs comportements : bavards, déconcentrés, perturbateurs, actifs, participants et hyperactifs.

Une fois les tests terminés, les données compilées et les deux catégories sont présentées ensemble sur un même graphique selon leurs types pour faciliter l'analyse. Les séries d'images dans la figure 5.6 représentent les résultats de cette recherche, le trait en bleu est la première catégorie et le trait en rouge est la deuxième catégorie.

4.8.4 Étude du mode de présentation des algorithmes

« *C'est au pied du mur qu'on voit le maçon !* » Ce proverbe bien connu reflète une idée prévalant dans la formation, soutenue récemment par le prix Nobel de physique Georges Charpak, dans le programme « *La main à la pâte* » mais déjà en œuvre dans les pédagogies actives, comme la pédagogie Freinet : apprendre par l'action est plus efficace qu'en parole. Dans le sillage de la théorie du double codage, le chercheur allemand Engelkampk et al. (2016) avait montré un meilleur rappel d'actions (j'ouvre la porte, je tourne les pages d'un livre), par rapport à leur présentation imagée, elle-même supérieure à la présentation de phrases. Comme la supériorité des images est expliquée par un double codage (imagé + verbal), il a proposé la théorie du triple codage, l'action permettant un codage verbal, imagé plus un codage moteur.

Une deuxième double expérimentation s'est déroulée toujours en classe de quatrième A du Collège Notre-Dame et une autre classe de quatrième du collège Sainte-Thérèse pendant l'année scolaire 2018-2019, puisque les élèves ont déjà pratiqué l'informatique en classe antérieure (donc ils ont des prérequis en informatique), l'enseignement basé sur la méthode active et utilisant « *essai-erreur* ». Les élèves travaillent sur les mêmes concepts de base en utilisant la méthode courante sur le langage de programmation visuel Scratch.

4.8.5 Objectifs, facteurs étudiés

Objectifs :

- Trouver les modèles de représentions approprié à l'enseignement/apprentissage de l'algorithme .
- Appliquer les enseignements en trois niveaux d'abstraction proposés par la méthode de Singapour en Informatique.

Principes : Nous allons évaluer la vitesse de compréhension et d'acquisition de l'enseignement d'algorithmique selon trois niveaux d'abstraction, portant nous avons choisi cette adéquation :

- algorithme exprimé en langage naturel correspondant à la phase rhétorique (abstrait) ;
- algorithme construit en pseudocode phase syncopée (abstrait) ;
- algorithme traduit en langage de programmation phase symbolique (abstrait) ou graphique (imagé).

Nous avons essayé d'analyser les facultés de compréhension des élèves ainsi que de mise œuvre et d'exécution d'un algorithme en plusieurs formats en leur demandant d'écrire, de lire ou d'exécuter un algorithme avec le logiciel Scratch et quelquefois avec pseudocode.

4.8.6 Savoir lire un algorithme/programme

Cette compétence suppose de savoir reconnaître un algorithme. Par exemple, une recette de cuisine n'en est pas un car elle contient des implicites, des instructions ambigües (pincée de sel, four doux)...Et le résultat n'est pas garanti !

L'algorithme doit être reconnu, quelle que soit sa forme :

En français	En pseudo-code	En langage scratch
Je pars de 0. J'ajoute 1 au résultat précédent. Puis j'ajoute 2. Je continue ainsi jusqu'à ce que j'ajoute 20. J'annonce le dernier résultat obtenu.	SM := 0; POUR k ALLANT de 1 à 20 FAIRE SM := SM + k FINPOUR RETOURNER SM	

Les élèves parviendront-ils à lire un algorithme ? Comment ? Et par quel moyen ?

Consignes

Nous avons demandé à chaque groupe de lire trois algorithmes, qui sont représentés dans trois formats différents. Afin d'éviter la possibilité de rappel lors du processus de lecture, nous leur avons donné un nouvel algorithme peu connu. Chaque groupe dispose de 15 minutes de préparation avant la lecture.

Critères d'évaluations

La vitesse de lecture (aisance) : La vitesse de lecture acceptée de la part d'un bon lecteur est de 1 minute 20 secondes. Les élèves ont été évalués par rapport à cette performance.
Modalités d'évaluation : très bon 1 minute à 1 minute 30 secondes, satisfaisant : de 1 minute 30 secondes à 2 minutes 10 secondes, faible de 2 minutes 10 secondes à 3 minutes, insatisfaisant : plus de 3 minutes.
Verbalisation : Elle a été évaluée par rapport à la compréhension des syntaxes et de la reconnaissance de différentes expressions.
Erreurs (exactitude) : L'exactitude est définie par l'absence d'erreurs. Il s'agit, dans cette épreuve, de prendre en compte le nombre de toutes les erreurs commises.
Modalités d'évaluation : Très bon : moins de 5 erreurs, satisfaisant : de 6 à 10 erreurs. Faible : de 11 à 15 erreurs, insuffisant : plus de 15 erreurs.

4.8.7 Savoir exécuter un algorithme/programme

La finalité d'un algorithme est d'être traduite sous la forme d'un programme exécutable sur un ordinateur. Il est donc indispensable d'avoir une idée précise (bien plus qu'une idée en réalité !) de la façon dont va « fonctionner » le programme en question.
Pour cela, il est très formateur « d'exécuter soi-même » (on dit faire tourner) l'algorithme que l'on conçoit. Cela permet de mieux comprendre le fonctionnement des différentes opérations et structures et, bien souvent, de découvrir des anomalies dans l'algorithme que l'on a conçu.

Pour exécuter un algorithme, il suffit de conserver une trace des valeurs en cours des différentes variables et d'exécuter une à une les opérations qui composent l'algorithme (en respectant la sémantique des structures de contrôle) en reportant leur éventuel impact sur les valeurs des différentes variables.

Afin de déterminer si les élèves peuvent exécuter l'algorithme / programme, nous pouvons poser quelques questions :

1. *Les élèves arrivent-ils à exécuter un algorithme présenté en Scratch coloré ?*
2. *Les élèves arrivent-ils à exécuter un algorithme présenté en Scratch non coloré ?*
3. *Les élèves arrivent-ils à exécuter un algorithme présenté en pseudocode en algo box ?*

Consignes

Nous avons demandé à chaque groupe d'exécuter des algorithmes écrits dans différents formats. Afin de réduire le biais des travaux de recherche, nous avons choisi un algorithme simple pour éviter les éventuels obstacles de la transposition mathématique à l'informatique. Par exemple, la division euclidienne est un concept mathématique qui a été découvert et étudié depuis la classe antérieure.

Critères d'évaluations

La vitesse de codage (aisance) : Le codage normal de la part d'un bon étudiant est de 10 minutes. Les élèves ont été évalués par rapport à cette performance.
Modalités d'évaluation : très bon de 8 à 11 minutes, satisfaisant de 12 à 15 minutes, faible de 16 à 20 minutes, insatisfaisant plus de 21 minutes.
Erreurs (bugs) : Le succès si le groupe réussit l'essai sans erreurs. Il s'agit, dans cette épreuve, de prendre en compte le nombre d'essais effectués.
Modalités d'évaluation : Très bon moins de 5 essais, satisfaisant de 6 à 10 erreurs. Faible de 11 à 15 erreurs, insuffisant plus de 15 erreurs.

4.8.8 Savoir écrire/compléter/corriger un algorithme/programme

On peut également vouloir déterminer si les élèves ont une bonne compréhension des boucles et de l'initialisation des variables. Les élèves doivent aussi être capables de détecter et de corriger des erreurs dans un algorithme/programme. Pour cela, il peut être intéressant de leur faire compléter un algorithme/programme.
Les élèves parviendront-ils à écrire un algorithme ? Comment ? Et par quel moyen ?

Consignes

Nous avons également demandé à chaque groupe de coder l'algorithme dont la représentation restait à choisir par eux. Pour ne pas biaiser nos travaux de recherche, nous avons choisi un problème simple pour éviter d'éventuels obstacles lors de la transposition mathématique à l'informatique.

Critères d'évaluation

La démarche qu'ils ont choisie (méthode). La méthodologie donnée s'est déroulée en trois étapes : 1. résolution du problème, 2. réalisation, 3. vérification.
Modalités d'évaluation très bonne en trois étapes, satisfaisant en deux étapes. Faible en une étape, insuffisant si pas de réalisation.
Durée de réalisation Les élèves ont été évalués par rapport au temps écoulé après la dernière vérification réussite.
Modalités d'évaluation : très bon mois de 20 minutes, satisfaisant de 20 à 30 minutes, faible de 30 à 40 minutes, insatisfaisant plus de 40 minutes.

4.9 Méthodes d'analyse statistique des résultats par L'ASI-MGK

Dans cette section, nous visons à trouver des réponses à toutes les questions posées antérieurement lors des analyses observatoires. L'ensemble des énoncés d'observation qui peuvent être attachés à un énoncé théorique forme son contenu empirique. Parmi les énoncés d'observation, il y a une différence entre les énoncés singuliers qui se réfèrent à un unique fait observable et à une classe donnée et les énoncés universels, qui portent sur la totalité des classes.

Dans la définition du chapitre et dans la section du chapitre, nous avons décrit les démarches communes ou le processus itératif et interactif de l'extraction des connaissances causales est présenté sous forme des règles d'associations.

4.9.1 Données théoriques de L'ASI-MGK

Collecte de données

Au cours de cette étude, nos données ont d'abord été collectées à partir d'une enquête statistique, où les modalités ou items étudiés ont été définis comme des informations observées durant leur apprentissage et des questions dans le questionnaire et le nombre d'individus est le nombre total de questionnaires retournés et remplis, pris dans une classe de quatrième du collège Notre-Dame Lourdes à Antsiranana. Les données recueillies sont les reponses au questionnaire suivant :

1. Sexe
   - □ Masculin
   - □ Féminin
2. Parcours d'élève
   - □ Nouvel élève
   - □ Ancien élève
   - □ Fait l'informatique en 5ieme
3. Attente en informatique
   - □ Internet
   - □ Jeux vidéo
   - □ Educatif
   - □ Multimédia
   - □ Bureautique
   - □ Autres
4. Comportement
   - □ Bavard
   - □ Non interessé
   - □ Déteste Informatique débranché
   - □ Perturbateur
   - □ Actif
   - □ Rétardataire
   - □ Participatif
   - □ Déconcentrer
   - □ Hyper Actif
5. Compréhension des notions algorithmique enseignées
   - □ Notion de variable
   - □ Branchement conditionnel avec alternative
   - □ Affectation
   - □ Test et logique
   - □ Structure répétitive avec tant que
   - □ Branchement conditionnel simple
   - □ Structure répétitive avec boucle Pour
6. Compétence acquis
   - □ Bonne interaction avec les fenêtre
   - □ Ordre et habillage d'objet inséré
   - □ Mise en forme d'objet
   - □ Objet word'art
   - □ Publication sur Publisher
   - □ Histoire sur scratch
   - □ Présentation avec PowerPoint
   - □ Reseau informatique
7. Attente
   - □ Lecture
   - □ Exécution
   - □ Scratch coloré
   - □ Écriture
   - □ Algobox
   - □ Scratch non coloré
8. Niveau de groupe d'appartenance

□ Faible □ Moyen □ Doué

Ensuite, à partir des feuilles de copie des examens individuels des élèves, où les modalités ou les éléments sont des variables didactiques qui ont été examinées dans le questionnaire des sujets d'examen, et le nombre de personnes est la taille de la classe pendant les cours et l'apprentissage d'informatique. Les informations à collecter sont la présence / absence des variables ou la maîtrise ou non-maîtrise d'un concept qui a été examiné dans la copie de chaque élève.

4.9.2 Prétraitement

Cela comprend la transcription et le codage des informations directement observées et contenues dans chaque questionnaire, la conversion des valeurs qualitatives et/ou quantitatives en valeurs binaires ainsi que la cohérence ou le contrôle de la qualité des données.

	Q1	Q2	Q3	Q4	⋯	Q16
Ele1	F	Ancien	Multimédia	17	⋯	Doué
Ele2	M	Ancien	Jeux	16	⋯	Doué
Ele3	M	Nouvel	Jeux	7	⋯	Faible
Ele4	F	Ancien	Bureautique	10	⋯	Moyenne
⋮	⋮	⋮	⋮	⋮	⋮	⋮
Ele75	M	Ancien	Internet	9	⋯	Moyenne

TABLEAU 4.8 – Donnée transcrite. Tableau de données obtenues après la transcription de données-réponses à des questions posées aux élèves.

Dans le tableau 4.8, Q1 c'est le sexe, Q2 c'est le parcours d'élève, Q3 c'est l'attente en informatique, ⋯, et Q16 c'est le niveau de groupe d'appartenance.

Dans cette étape, nous utilisons généralement un logiciel tableur (Microsoft Excel). Les données ont la forme indiquée sur la figure (fig 2.5) et être enregistrées au format CSV (*comma separated variable*).

Regles de codages :

- Les données manquantes sont converties en 0 ;
- Les modalités avec un faible support sont rejetées ;
- Les modalités pouvant être obtenues par complémentarité sont directement éliminées de la base de données ;
- Les modalités avec des valeurs numériques ou non binaires sont transformées.

Matrices booléennes

Codes	Signification	Support
F	Sexe féminin	0.57
M	Sexe masculin	0.43
PTB	Perturbateur	0.25
ACT	Actif	0.52
PRT	Participatif	0.52
NVE	Nouvel élève	0.15
OBJ	Mise en forme objet	0.68
RES	Réasau informatique	0.77
PRS	Paressé	0.47
RTR	Retardataire	0.45
AFC	Notion d'affectation	0.77
TST	Test conditionnel	0.65
REP	Structure répétitive Pour	0.56
ITR	Bonne interaction	0.80
DOC	Avoir amené autre document	0.51
PAO	Publication avec Publisher	0.85
INT	Interaction	0.52
BUR	Bureautique	0.85
EDU	Éducatif	0.25
DCON	Déconcentré	0.67
JEUX	Jeux	0.51
ANCE	Ancien élève	0.88
GrDE	Groupe doué	0.39
GrMN	Groupe moyen	0.45
GrFB	Groupe faible	0.65
SCon	Concentré en Scratch	0.83
NINT	Non intéressé	0.56
HACT	Hyper actif	0.49
SINN	Si sinon	0.55
SIAL	Si alors	0.65
BMLT	Bon en multimédia	0.61
PrAO	Présentation avec Powerpoint	0.75
HABL	Ordre et habillage des objets	0.80
WARD	Objet wordart	0.57
MULT	Multimédia	0.44
AUTR	Autres attentes	0.23
LECT	Lecture	0.35
ECRT	Ecriture	0.25
EXEC	Exécution	0.37
ALGB	Algobox	0.22
SCOL	Scratch coloré	0.38
SNCOL	Scratch non coloré	0.31
CAlG	Concntré en Algobox	0.21

TABLEAU 4.9 – Tableau de codage obtenu à partir du tableau de donnée.

	F	M	BVR	PTB	ACT	PRT	NVE	OBJ	RES	PRS	RTR	PDL	TNQ	AFC	TST	REP	TTR	DOC	PAO	INT	BUR	EDU	DCONJEUX	INF5	ANCE	GrDE	GrMN	GrPB	AINF	SCon	NINT	DINF	HACT	SINN	SIAL	BMLT	PAO	HABL	WAR	HIST	MULT	AUTR	LECT	EGRT	EXEC	ALGB	SCOL	SNC	OCAL	G
A01	0	1	1	0	0	0	0	0	0	1	1	1	0	0	1	0	0	1	1	0	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0
A02	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1	0	0	1	1	0	0	0	1	1	1	0	1	0	1	0	1	1	1	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
A03	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A04	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A05	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A06	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A07	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A08	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A09	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A10	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A12	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A22	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A23	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A24	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A31	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A32	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A34	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A35	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A36	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A38	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A39	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A40	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A41	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A42	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A43	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A45	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A46	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A47	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A48	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A49	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A50	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A51	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A52	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A53	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A54	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A55	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A56	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A57	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A58	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A65	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A66	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A68	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A69	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A70	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A71	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A72	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A73	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A74	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A75	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLEAU 4.10 – Tableau boléen de données à 75 attributs et 50 motifs

4.9.3 Extraction des connaissances.

L'ASI – MGK implémente les résultats des recherches scientifiques effectués et publiés par notre groupe de rechercheur travaillant conjointement sur les propriétés mathématiques et statistiques de M_{GK} par Totohasina (2003, 2008), base de règles valides selon M_{GK} par Feno (2007), sur les graphes implicatifs selon M_{GK} par Bemarisika (2016), Optimisation d'algorithme extraction de RA valide par Ramanantsoa (2016), et Ralahady & Totohasina (2019a,b), sur l'implémentation, la conception et l'amélioration de vue (Ralahady et al., 2019),(Razanatsoavina et al., 2019). Dans cette section, pour l'extraction des connaissances cachées ont nos données, nous allons utiliser cet outil.

Extraction des motifs fréquents.

En choisissant l'algorithme apriori implémenté dans ASI – MGK avec support minimal égal à 40%, dans sa fenêtre de dialogue, nous avons la statistique suivante.

```
=============  STATISTIQUES APRIORI  =============
 Nombre des candidats : 8530
 L'algorithme est stoppée à la taille 9, car il n'y plus de candidat
 Nombre des motifs Frequents: 5496
 Memoire maximal utilisé : 20.976287841796875 mb
 Temps total ~ 369 ms
=================================================
```

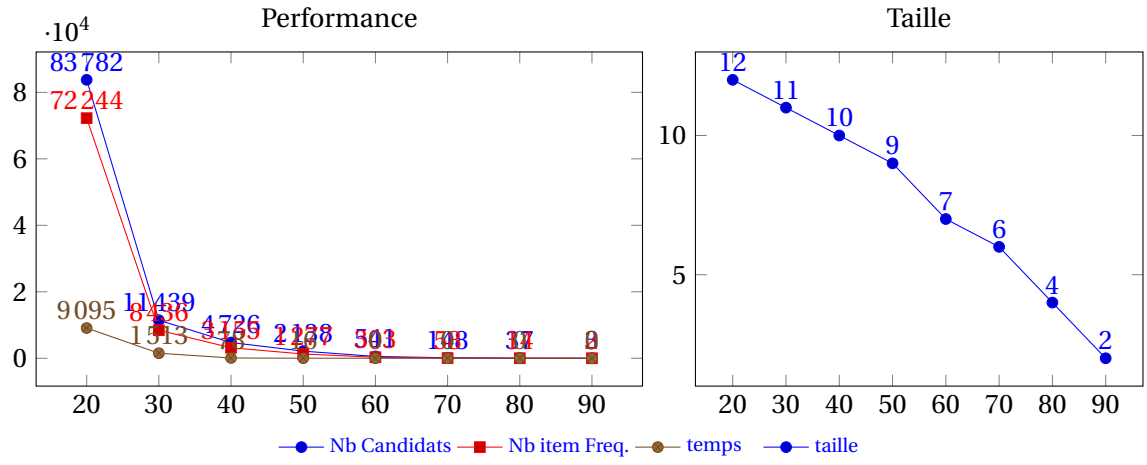


FIGURE 4.19 – Resultats des simulations de l'algorithme Apriori.

À gauche, c'est la performance de l'algorithme selon : en bleu le nombre de candidats, en rouge le nombre des items fréquents et en marron le temps en fonction de support minimal choisi.
À droite, c'est la taille de l'itemset maximale en fonction de support minimal choisi.

Extraction des règles d'associations valides.

Avant tout, nous avons les données d'entrée comme suit :
Nom du fichier d'entrée `DonneDOctorat.txt`,
nom du fichier de sortie `DonneDOctorat\_Apriori\_Assosiation\_Valide.txt`,
algorithme sélectionné `Apriori\_Assosiation\_Valide`.

Première extraction Lors de cette première extraction, nous avons utilisé la configuration par défaut de l'ASI-MGK, telle que : minsup =0.2 et minEur =0.05. Et comme l'algorithme d'extraction des règles d'associations intègre lui-même aussi l'algorithme apriori, alors le rapport de traitement dans la fenêtre de dialogue a donné cette double statistique.

```
=============  STATISTIQUES APRIORI  =============
 Nombre des candidats: 1225
 L'algorithme est  stoppée à la taille 2, car il n'y plus de candidat
 Nombre des motifs Frequents: 937
 Memoire maximal utilisé: 13.315078735351562 mb
 Temps total ~ 47 ms
==================================================
========  STATS DE GENERATION DES REGLES  =========
 Nombre de règles d'association valides générés: 280  Ecartés: 1496
 Nombre de règles redondantes:1151 règles non valides: 345 Taux: 0.16%
 Seuil de confiance: 0.05     Temps total ~ 593 ms
Règles positives: 280  Règles nulles: 1  Règles negatives : 0
==================================================
```

Au seuil de support minimal égal à 20%, en 47 millisecondes l'algorithme apriori sélectionne 937 items fréquents sur les 1225 items candidats.

Durant cette procédure, l'algorithme de génération de règles nous donne 280 règles valides. Pourtant on a écrêté 1496 règles dont 1151 règles non valides et 345 règles redondantes avec un taux de 0.16%.

Deuxième extraction : choix de support minimal. Lors de l'exécution de l'algorithme d'extraction des règles d'associations valides, afin de réduire le nombre de candidats, nous avons autorisé seulement les caractères observés dont la fréquence dépasse le 40%.

```
========  STATS DE GENERATION DES REGLES  =========
 Nombre de règles d'association valides générés: 107  Ecartés : 625
 Nombre de règles redondantes: 477 règles non valides: 148 Taux: 0.15%
 Seuil de confiance: 0.05     Temps total ~ 259 ms
Règles positives: 107  Règles nulles: 1  Règles negatives : 0
==================================================
```

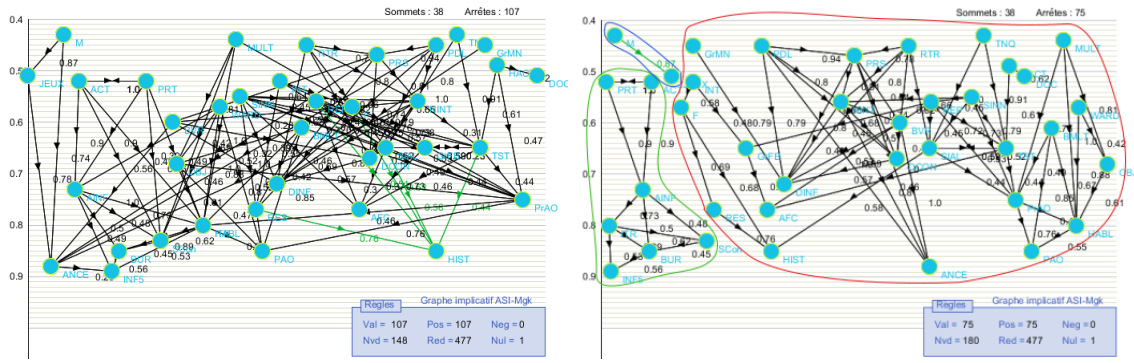


FIGURE 4.20 – Règles générés aux seuils 5.0E-2 et 5.0E-3.

En augmentant le support minimal à 40%, avec le même seuil d'erreur, nous avons obtenu un bloc de 107 règles avec un taux de réduction de règle égal à 15% (figure4.20 à gauche).

Troisième, quatrième et cinquième extractions : choix de seuil de confiance. Depuis la troisième génération, nous avons fixé le minsup à 40%, et nous avons procédé à trois suites de générations successives afin de faire apparaître les règles les plus pertinentes. Durant la première tentative, on a choisi le risque d'erreur 5.0E-3, nous avons obtenu 79 règles valides. Son graphe implicatif présente trois composantes connexes (figure 4.20 droite).

Durant la deuxième tentative, on a choisi le risque d'erreur 5.0E-4, nous avons eu cette fois-ci 51 règles valides et le graphe implicatif présente cinq composantes connexes (figure 4.21 gauche).

Lors de la troisième tentative, le risque d'erreur choisi était 5.0E-5, nous a amenés à avoir 33 règles valides. Sur la fenêtre de représentation d'ASI-MGK, figure 4.21 droite), le graphe a plusieurs composantes connexes. Trois blocs et 7 composantes formés par deux paires d'items.

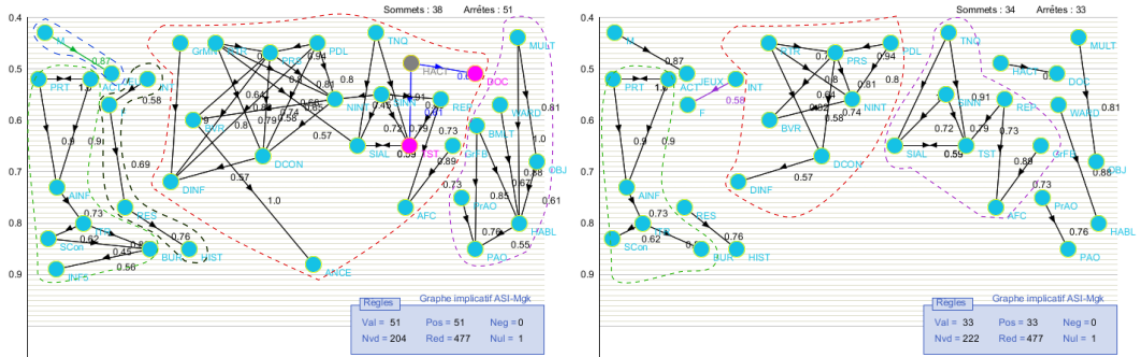


FIGURE 4.21 – Règles générés aux seuils 5.0E-4 et 5.0E-5.

Durant ces processus, nous avons observé une réduction des nombres de règles valides. Les règles les moins pertinentes disparaissent au fur et à mesure que l'on diminue le risque d'erreur. Nous pouvons désormais avancer en fixant notre résultat sur ces 33 premiers règles le plus pertinentes (tab 4.11), générées à la configuration suivante : support minimal égale à 40% et au risque d'erreur égale à 5.0E-5 et la taille maximal quant à elle étant au maximum égal à 1.

```
========  STATISTIQUES  DE GENERATION DES REGLES  =========
 Nombre de règles d'association valides générées: 33  Ecartés: 699
 Nombre de règles redondantes: 477 règles non valides: 222 Taux: 0.04%
 Seuil de confiance: 5.0E-5     Temps total ~ 16 ms
Règles positives: 33  Règles nulles: 1  Règles negatives : 0
=================================================
```

La statistique de la génération des règles après le traitement nous montre qu'au support minimum égal à 40% et avec seuil d'erreur égal à 5.0E-5, pour 741 combinaisons d'items ou d'ensembles d'items candidats, parmi lesquels 404 sont fréquents, l'outil d'analyse statistique ASI – MGK nous a généré 33 règles d'associations valides au sens de la mesure M_{GK} et toutes des règles sont positives. Elle a écarté 699 règles dont 222 règles non valides, 477 règles redondantes et 1 règle nulle. Le taux extraction est de 0.04% et la durée totale de traitement était de 16 millisecondes. Nous avons 33 relations causales significatives entre nos données, ainsi nous allons passer à la fenêtre graphique pour représenter notre résultat en graphe implicatif.

L'examen du résultat graphique nous a permis d'observer un graphe orienté à 34 nœuds et 33 arcs, c'est-à-dire 33 relations binaires entre les 34 modalités étudiées. Dans sa représentation spa-

	Antécédent	Conséquence	Confiance	MGK	Couverture	Prévalence
R_1	INT	SEXF	0.82	0.58	0.52	0.57
R_2	SEXM	JEUX	0.94	0.87	0.43	0.51
R_3	MULT	OBJ	0.94	0.81	0.44	0.68
R_4	RESX	HIST	0.97	0.76	0.77	0.85
R_5	GrFB	AFC	0.94	0.73	0.65	0.77
R_6	ACT	AINF	0.97	0.9	0.52	0.73
R_7	PRT	AINF	0.97	0.9	0.52	0.73
R_8	AINF	ITR	0.95	0.73	0.73	0.8
R_9	ITR	SCon	0.93	0.62	0.8	0.83
R_{10}	NINT	BVR	0.83	0.58	0.56	0.6
R_{11}	PRS	NINT	0.91	0.81	0.47	0.56
R_{12}	RTR	NINT	0.91	0.8	0.45	0.56
R_{13}	PDL	NINT	0.91	0.8	0.45	0.56
R_{14}	DCON	DINF	0.88	0.57	0.67	0.72
R_{15}	PRS	BVR	0.86	0.64	0.47	0.6
R_{16}	PRS	DCON	0.91	0.74	0.47	0.67
R_{17}	RTR	DCON	0.94	0.82	0.45	0.67
R_{18}	PRT	ACT	1	1	0.52	0.52
R_{19}	ACT	PRT	1	1	0.52	0.52
R_{20}	HACT	DOC	0.81	0.62	0.49	0.51
R_{21}	RTR	PRS	0.88	0.78	0.45	0.47
R_{22}	PDL	PRS	0.97	0.94	0.45	0.47
R_{23}	TNQ	TST	0.97	0.91	0.43	0.65
R_{24}	TNQ	SIAL	1	1	0.43	0.65
R_{25}	SINN	TST	0.93	0.79	0.55	0.65
R_{26}	SINN	SIAL	0.9	0.72	0.55	0.65
R_{27}	REP	AFC	0.98	0.89	0.56	0.77
R_{28}	REP	TST	0.9	0.73	0.56	0.65
R_{29}	SIAL	TST	0.86	0.59	0.65	0.65
R_{30}	TST	SIAL	0.86	0.59	0.65	0.65
R_{31}	ITR	BUR	0.98	0.89	0.8	0.85
R_{32}	PrAO	PAO	0.96	0.76	0.75	0.85
R_{33}	WARD	HABL	0.98	0.88	0.57	0.8

TABLEAU 4.11 – Les 33 règles générées à analyser

tiale, l'interprétation s'avère être impossible. Dans le chapitre 2, section 2.8.5 nous avons introduit les méthodes d'amélioration de vue.

Chapitre 5

Résultats et interprétation des études.

Sommaire

Dans le précédent chapitre, nous avons énuméré les différentes analyses que nous avons effectuées. Dans ce chapitre nous les reprenons, une par une, et nous présentons avant tous les résultats et les interprétons pour en donner des recommandations quant à l'enseignement de l'informatique au collège. Comme pour enseigner, il faut des enseignants, c'est par eux que nous allons commencer.

5.1 Formation des enseignants à l'informatique.

5.1.1 Formation au niveau du concept de base de l'informatique.

Durant cette formation, nous avons ciblé tous les personnels, ensignants et administratifs. L'objectif de la formation est de donner une bonne connaissance de base en informatique. L'informatique c'est n'est pas une séance d'ordinateur.

Concept informatique, une machine différente d'un ordinateur. L'ordinateur, l'outil le plus courant et le plus connu, ce n'est la seule machine pouvant être utilisée, dans le domaine informatique, même si elle a vocation aux traitements de l'information.

En effet, d'abord l'ordinateur représente la machine modulable et universelle qui peut s'adapter à de multiples tâches en ajoutant des logiciels et/ou des matériels. En plus, l'architecture matérielle d'un ordinateur recouvre plusieurs types : les gros ordinateurs ou superordinateurs, les miniordinateurs, les microordinateurs ou les PC et les nano-ordinateurs ou les ordinateurs de poche.

Ensuite, les systèmes embarqués représentent les machines les plus particulières qui sont destinées à une ou des tâches bien précises. Et en fin les automates et les robots représentent les machines les plus complexes dues à leur fonctionnalité semi-autonome (architecture matérielle, logiciel et réseau, protocole de communication).

Concept informatique en tant que programme. Nous avons encore expliqué le fonctionnement logique de base d'une machine informatique, elle fonctionne grâce à des programmes préalablement écrits. La machine lit et exécute les programmes écrits en langage machine (ou assembleur) mais depuis l'invasion du copulateur FORTRAN et ses successeurs, elle peut lire le programme écrit en langage évolué (langage plus proche de l'humain). On définit, alors, la programmation qui est la traduction des instructions algorithmiques (les différents langages de programmation, paradigme de programmation...).

Concept informatique en tant qu'un algorithme. Là nous avons abordé l'aspect automatique et logique de l'informatique. l'ensemble des ordres et instructions algorithmiques.

Concept informatique en tant que des informations. Nous, les humains, disposons de cinq types d'informations, par contre en informatique, on dispose que de trois formes d'informations ; information visuelle, toute sorte d'information visible (texte, image fixe et image dynamique), information auditive (son et audio) et information audiovisuelle (cinéma, vidéo). Dans celle-ci, nous

avons appréhendé son mode de présentation en fichier avec ses différents formats, sa capacité mesurée en octets et ses multiples : Ko, Mo, Go et To.

Par conséquent, les périphériques d'entrée et de sortie d'une machine varient en fonction de leurs outils de capture ou de diffusion de l'information (par exemple, la présentation visuelle de type texte est saisie par un clavier et sort à l'aide d'une imprimante).

5.1.2 Formation au niveau du concept à enseigner

Durant cette formation, nous avons ciblé les enseignants responsables du cours d'informatique au collège. L'objectif est de les amener vers un aspect plus académique. Actuellement, l'enseignement de l'informatiqure est plus pratique, puisque les jeunes sont initiés aux logiciels de bureautique. Word en 6ième, Excel en 5ième, Power Point en 4ième. Dans cette formation , nous leur avons expliqué les théories de base qui sont implémentés dans ces logiciels qu'il faut enseigner aux élèves avant la pratique.

D'abord, nous avons abordé le traitement de texte, le tableur, la publication et la présentation assistée par ordinateur. Et puis l'histoire et l'architecture des ordinateurs et enfin, nous avons terminé par la pensé informatique (le cœur de l'enseignement de l'informatique).

5.2 Formation des enseignants au logiciel Scratch

Pour enseigner l'informatique à l'école, il n'est pas nécessaire de tout savoir et tout comprendre de l'informatique. Les professeurs des écoles n'ont pas à devenir plus informaticiens qu'ils ne sont matheux, littéraires, ou spécialistes des différentes matières enseignées à l'école. Comme pour les autres matières, le fait de ne pas avoir suivi de cours d'informatique dans son cursus universitaire ne pose pas de problème insurmontable. Ce peut même être un avantage, car, dans certains cursus, les étudiants apprennent à programmer, mais pas à comprendre les concepts abstraits sous-jacents, et développent des conceptualisations de l'informatique qui n'aident pas à son enseignement.

Cependant, et c'est le point clé, pour identifier et travailler ce que l'on a besoin de savoir, il faut avoir des idées claires sur ce que l'on veut enseigner (et pourquoi). Ceci permet ensuite de faire la part des choses comme ce qu'il faut comprendre, ce qu'il faut savoir utiliser, ce qui est hors sujet.

En général, la formation des enseignants prend son sens au regard du métier tel qu'ils l'exercent au quotidien. Comment les y préparer ? Comment les "équiper" ? De quoi ont-ils besoin ?

Durant notre formation nous nous sommes basés sur la méthodologie décrite par Robitaille (2007) qui recense cinq grandes orientations : l'apport théorique, la démonstration d'habiletés, la mise en pratique, la rétroaction, la discussion.

5.2.1 Apport théorique (formation sur l'algorithme)

Cette première partie de notre formation concerne l'étude des concepts et les apports de la recherche sur un sujet donné, démarche qui peut amener les enseignants à découvrir la théorie à travers des situations problèmes.

Cette première partie de la formation a pour objectif d'initier les gens à la programmation informatique d'une manière visuelle en utilisant Scratch, à travers des exemples ludiques et concrets.

Les notions de programmation informatique telles que les instructions, les variables, les procédures, les listes, les boucles, les conditions y sont présentées avec une approche de situations-problèmes et expliquer le récit sur la raison de l'utilisation du logiciel dans les salles de classe. Nous avons donné des exemples d'algorithmes mettant en contexte l'utilisation de Scratch dans le domaine des mathématiques.

5.2.2 La démonstration de mise en pratique sur l'environnement Scratch

Une partie de la formation consiste d'abord à une observation de la mise en pratique des connaissances ou habiletés visées par la formation, puis à une simulation où les participants mettent en pratique les connaissances ou habiletés abordées précédemment. Les contenus partagés durant cette partie sont : une rapide intervention sur la version du logiciel à installer sur l'ordinateur ou sur la tablette, une découverte de l'interface de Scratch 2.0 à travers une courte démonstration comme le tracé de différentes formes géométriques à l'aide du petit lutin. Lorsque l'utilisateur demande à ce dernier de se déplacer sur le plan, il laissera une trace de son trajet. Ainsi, le site propose de réaliser un carré, un triangle équilatéral, en y découvrant les concepts de boucles et de variables.

5.2.3 Rétroaction et discussion

Cette partie consiste à récolter des informations en retour sur les activités effectuées par les enseignants formés. Elle permet aussi de vérifier leur compréhension, de partager leurs doutes, leurs insatisfactions ou leurs découvertes.

Temps de formations

Des ressources informatiques On croule sous le poids des documents informatiques, notamment sur les sites, nous ne ferons que deux remarques :

- que les documents mis à disposition des enseignants sont passables, le plus souvent le point faible reste la pédagogie.
- et que les interrogations portent surtout sur la nature même de la formation. Doit-on demander aux enseignants d'être des autodidactes qui se forment eux-mêmes à partir de leurs lectures et à partir d'internet ? Toute formation ne suppose-t-elle pas des enseignements et des échanges ?

Outils de préparation. Il faut toujours qu'ils travaillent d'abord sur l'environnement Scratch et puis effectuent des captures d'écran, et les collent dans un document.

Méthode pédagogique. Il s'agit de la difficulté majeure et concerne le choix de l'approche utilisée et la méthode pédagogique à mettre en œuvre pour conduire les apprenants vers une situation d'apprentissage. De même la méthode d'évaluation des acquis des élèves reste difficile à appréhender.

En vue d'apporter une bonne utilisation, nous avons créé un correctif de l'extension LaTeXécrit par Christian Tellechea unbonpetit@netc.fr version v 0.4 du 8 avril 2018. Le correctif comprend :

- 16 commandes dans le bloc de mouvement,
- 14 commandes dans le bloc d'apparence,
- 12 commandes dans le bloc de sons,
- 10 commandes dans le bloc de stylo,
- 15 commandes dans le bloc de contrôle,
- 21 commandes dans le bloc d'opération,
- 5 commandes dans le bloc de capteur.

x	Nouveau	Ancien
	tournerGauche[1]	`\blockmove{tourner de \turnleft{} de \ovalnum{#1} degrés}`
	tournerDroite[1]	`\blockmove{tourner de \turnright{} de \ovalnum{#1} degrés}`
	pointerVers[1]	`\blockmove{Pointer vers \ovalnum{#1\selectarrownum}}`
	orienter[1]	`\blockmove{s'orienter à \ovalnum{#1\selectarrownum}}`
	allerA[1]	`\blockmove{aller à : \ovalnum{#1}}`
	allerAXY[2]	`\blockmove{aller à : \ovalnum{#1} y: \ovalnum{#1}}`
	mettreX[1]	`\blockmove{mettre x à : \ovalnum{#1}}`
	mettreY[1]	`\blockmove{mettre y à : \ovalnum{#1}}`
	remplacerX[1]	`\blockmove{remplacer x par \ovalnum{#1}}`
	remplacerY[1]	`\blockmove{remplacer y par \ovalnum{#1}}`
	Si[2]	`\blockif{si \boolsensing{#1} alors}{#2}}`
	SiSiNon[3]	`\blockifelse{si \boolsensing{#1} alors}{#2}{#3}}`
	repeter[1]	`\blockinfloop{répéter indéfiniment}{#1}}`
	repeterNfois[2]	`\blockrepeat{répéter \ovalnum{#1} fois}{#2}}`
	repeterJusque[2]	`\blockrepeat{répéter jusaqu'à \boolsensing{#1}}{#2}}`

TABLEAU 5.1 – Extension LaTeXmise en œuvre.

5.3 Formation des élèves par les enseignants.

Pour mettre en activité les élèves, l'idéal est de disposer d'un ordinateur pour trois à cinq élèves comme recommandé Cockburn & Williams (2000). Non seulement cela facilite la gestion de classe (moins de programmes à débugger, des mises en commun plus rapides, et plus riches), mais surtout, les élèves progressent plus rapidement. En groupe, ils peuvent débattre, tester plusieurs idées concurrentes, s'appuyer chacun sur les forces de l'autre, surveiller ce qui se passe à l'écran et détecter des bugs de façon très précoce.

Il est illusoire (et inutile) d'espérer pouvoir passer dans chaque groupe pour débugger les différents programmes. Il est préférable d'utiliser pour cela des moments de mise en commun. Plutôt que demander aux « meilleurs » groupes de montrer aux autres groupes ce qu'ils ont réussi à faire. Il est préférable de proposer une aide collective ou travail collaboratif aux autres groupes qui sont

bloqués. Le groupe s'installe alors sur le poste de travail commun et explique ce qu'il a fait et pour quoi il est bloqué. C'est alors toute la classe, et non l'enseignant qui l'aide collectivement. Quand c'est terminé, on propose à un autre groupe une aide similaire (c'est souvent nécessaire), et on reproduit l'opération. Cette façon de gérer les mises en commun permet d'économiser beaucoup de temps (la plupart des élèves réussiront, ensuite, à débugger leur programme) tout en valorisant l'intelligence collective, et pas seulement la réussite de quelques-uns.

Si l'enseignant choisit de passer dans tous les groupes pour corriger leurs programmes, il risque d'y passer beaucoup de temps (et d'énergie), et de laisser les élèves aggraver leurs difficultés. En effet, plutôt que d'attendre l'enseignant sans rien faire, ces derniers ont tendance à changer de très nombreuses parties de leur programme pour tenter de le corriger (parfois, au hasard !), avec pour seule conséquence que leur programme ne fait plus rien correctement et qu'il faut repartir de zéro

programmer est une tâche complexe, qui peut dérouter les élèves, ne sachant pas « par quel bout commencer ». Dans les différentes séquences, il faut proposer un découpage qui tient compte à la fois de la difficulté des différentes tâches (autant que possible, aller du plus simple au plus complexe), mais aussi de l'importance des différentes compétences.

5.3.1 Proposition d'un curriculum informatique commun

Après une entrevue avec les enseignants actuellement responsables du cours d'informatique, et un panorama fait par les résultats de recherche sur le *curriculum* de l'informatique au niveau secondaire, nous sommes persuadés que l'introduction des notions des bases de l'informatique depuis la classe de sixième est primordiale.

L'intégration instrumentale décrit quatre étapes d'une utilisation croissante de la technologie en classe (Assude, 2007) :

1. l'initiation instrumentale (étape 1) - les étudiants s'engagent uniquement dans l'apprentissage de l'utilisation de la technologie,
2. l'exploration instrumentale (étape 2) - les problèmes mathématiques motivent les élèves à en apprendre davantage sur l'utilisation de la technologie,
3. le renforcement instrumental (étape 3) - les étudiants résolvent des problèmes mathématiques avec la technologie, mais doivent étoffer leurs compétences technologiques,
4. la symbiose instrumentale (étape 4) - la maîtrise de la technologie par les étudiants permet d'élargir la tâche mathématique si bien que ce sont à la fois les compétences technologiques et la compréhension mathématiques des élèves qui sont enrichies.

Nous associons ces étapes aux dimensions de développement de la pensée informatique d'un étudiant du cadre de Brennan et Resnick (2012) : les étapes 1 et 2 concernent les concepts informatiques, les étapes 2 à 4 initient aux pratiques informatiques et les étapes 3 et 4 révèlent aux perspectives informatiques. Et c'est à l'étape 4 que nous soutenons que l'étudiant s'est approprié la programmation comme un instrument pour les mathématiques « *comme le feraient les mathématiciens* »(à la fois en termes de pratiques et de perspectives informatiques), ce que nous nom-

mons « *la programmation en tant qu'instrument de pensée informatique pour les mathématiques* ».

Tout d'abord, les élèves dès leurs premières rencontres avec le cours d'informatique, devront savoir l'histoire de l'ordinateur et de l'informatique, quelque notion sur la théorie de l'information sur la quantité de l'information qui circule partout comme le bit, les éléments qui constituent un ordinateur, le démarrage et les interactions sur l'ordinateur et ses dispositifs de contrôles ; souris et clavier.

En deuxième année d'études, ils devront s'initier au formatage et à la production de document textuel, traitement de données et opération sur une feuille de calcul dans un tableur. Enfin, au troisième niveau, ils devront aussi savoir effectuer une recherche documentaire et partager l'information sur internet, présenter et publier leurs idées à l'aide d'un support informatique et s'initier à l'algorithme et programmation.

Ainsi nous avons décidé d'instaurer un curriculum unique pour tous les établissements qui ont déjà instauré l'enseignement de l'informatique.

En classe de sixième.

Pour la classe de sixième nous élaboré des livres d'informatique de 22 pages et de format A4 pour trois établissements. Sur la page de couverture, inscris le nom de l'établissement et leur logo. (fig 5.1)

FIGURE 5.1 – Livres informatiques classe de sixième élaborés à partir de nos recommandations.

Le livre de classe sixième comprend six chapitres :

1. Apprentissage de l'informatique, *Introduction à l'informatique*.
2. Apprentissage de l'ordinateur, *Architecture de l'ordinateur*.
3. Apprentissage de la souris, *Commandes dans l'ordinateur*.
4. Apprentissage du Clavier, *Dactylographie*.
5. Apprentissage du Paint Windows, *Logiciel du dessin*.
6. Apprentissage du Système d'exploitation, *Système d'exploitation Windows*.

En classe de cinquième

Pareillement, les livre d'informatique, élaboré pour la classe cinquième, possèdent les mêmes caractéristiques (fig 5.2).

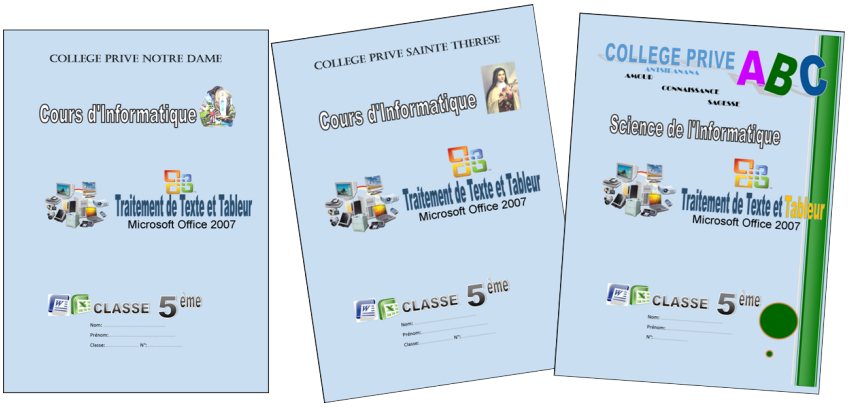


FIGURE 5.2 – Livres informatiques classe de cinquième élaborés à partir de nos recommandations.

Le livre de classe cinquième comprend cinq chapitres :

1. Traitement de texte et Tableur.
2. Présentation de MS Word.
3. La mise en forme d'un document.
4. Présentation de MS Excel.
5. Calcul élémentaire sur une feuille de calcul.

Livre de quatrième

Egalement, les livres d'informatique, élaborés pour la classe quatrième, possèdent la même caractéristique que les précédentes. Par contre, on a construis des livres informatiques de 28 pages, pour la classe de Seconde, uniquement pour l'établissement Saint Jean (fig 5.3).

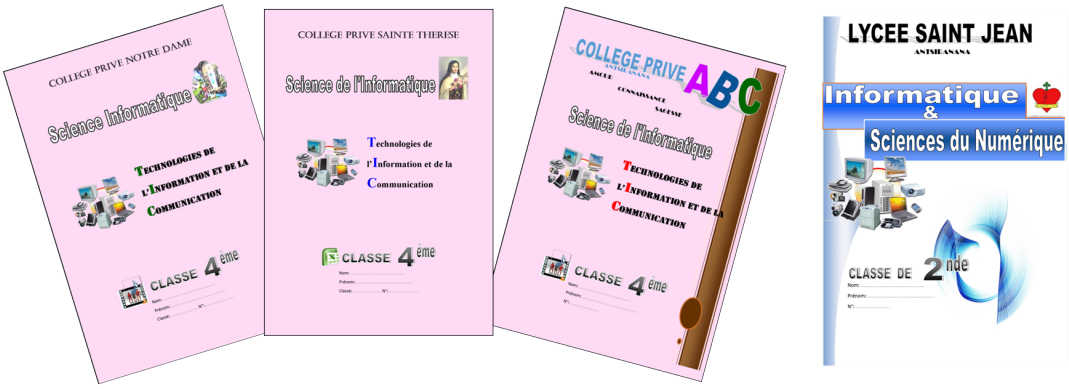


FIGURE 5.3 – Livres informatiques classe de quatrième (à gauche) et de seconde élaborés à partir de nos recommandations.

Le livre de classe quatrième comprend six chapitres :

1. Internet, *international network*
2. PAO & PréAO, *publication et présentation assisté par ordinateur*
3. Publisher, *logiciel de publication assisté par ordinateur*
4. Powerpoint, *logiciel de présentation assisté par ordinateur*
5. Tableur, *avancé traitement de tableau sous MS Excel*
6. Scratch, *Initiation à l'algorithme et la programmation*

5.3.2 Outils pédagogiques utilisés

- Un livret de cours, des exercices et des activités contenants des situations problèmes conçues pour amener les apprenants à faire l'expérience de « la prise de décision en situation d'incertitude et de la confrontation des solutions » (cf. annexe).
- Une application (*androïd*), accessible via tablette numérique ou des smart phones, intégrants tous les supports de documentation et des activités .
- Un package LaTeX [1] permettant aux enseignants de produire ou convertir des activités selon l'abstraction précise.
- Un guide d'enseignant pour favoriser le cheminement intérieur de l'élève. Ce guide propose aux enseignants un certain nombre d'activités à donner, de manière opportune, aux étudiants ainsi que des objectifs d'apprentissage précis.

5.4 Facteurs externes, attitudes, attente et taxonomie

5.4.1 Attitudes des élèves lors du cours d'informatique.

Dans la figure 5.4, ces deux graphiques (informatique à gauche et EPS à droite, montrent les effectifs records de désintéressement pour l'informatique. Les classe2 et classe1, deux classes d'établissement Notre-Dame-de-Lourdes sont toujours en tête en suivant cet ordre contre les deux classes du collège Sainte-Thérèse (classe3 et classe4).

Aucun élève n'a été relevé en retard en cours d'EPS du collège Sainte Thérèse, parce que ces cours sont tous placés aux dernières heures de l'après-midi. Ce n'est pas comme les classe1 et classe2 du collège Notre-Dame-de-Lourdes qui sont toutes aux premières heures ce qui provoque fréquemment des retards des élèves. Nous avons noté 6 élèves en retard pour la classe1 et 4 pour la classe2. Nous n'avons pas trouvé de remarques significatives pour les élèves dispensés du cours d'EPS et les élèves n'ont pas apportés leur tenue.

Nous avons sorti les élèves qui n'ont pas apporté leur livre d'informatique lors de la séance précédant de cette séance d'analyse. En effet, les effectifs avaient atteint respectivement 25, 29, 15 et 13 pour les classes 1, classe 2, classe 3 et la classe 4. Dans ces effectifs se situent des élèves ayant affirmé sincèrement qu'ils détestent et qu'ils ne s'intéressent pas au cours d'informatique. Ils se

1. LaTeXest un langage et un système de composition de documents. Il s'agit d'une collection de macrocommandes destinées à faciliter l'utilisation du « processeur de texte »

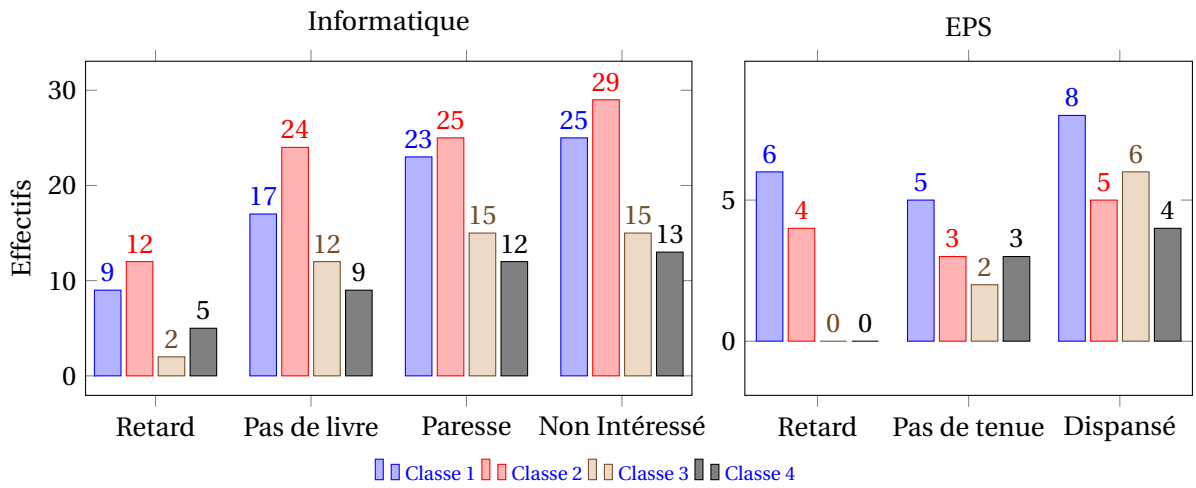


FIGURE 5.4 – Attitudes des élèves lors des cours d'informatique et d'EPS

sentaient gênés et désespérés de se présenter au cours informatique sans ordinateur, sans leurs attentes et trop centré sur le raisonnement mathématique. Et pourtant ils ont participé financièrement aux frais supplémentaires en plus d'écolage pour un cours particulier des mathématiques, tous les mercredis après-midi ou les samedis matin, pour se préparer aux examens.

Du fait que le cours informatique n'est pas noté, ils furent autorisés à s'absenter et se mettre volontairement en retard, ou de même à laisser intentionnellement leur livre pour les faire sortir du cours.

Nous avons retenu comme des élèves paresseux, tous les élèves qui n'ont pas de livre informatique avec eux ou qu'ils l'ont fréquemment retenu, et ce par analogie avec les élèves qui parfois sont sans tenue de sport en EPS. Nous avons recensé respectivement comme les élèves paresseux 23, 25, 15 et 12 pour les classe1, classe2 classe3 et clase 4. Statistiquement les différences d'effectifs ne sont pas significatives du fait des effectifs initiaux trop faibles. Malgré qu'ils n'aient pas osé le dire sincèrement leur dégout envers l'informatique, leur refus à travers leur paresse ou leur désamour le confirme toujours.

Les élèves sans livre sont ceux qui ne l'ont pas encore acheté ou l'ont déjà perdu. Le collège Sainte-Thérèse présente des effectifs un peu bas du fait que les livres font déjà partie des fournitures scolaires à exiger depuis la rentrée scolaire, ce qui n'est pas le cas pour le collège Notre-Dame-de-Lourdes.

La classe 2 du collège Notre-Dame-de-Lourdes présente plus de retard. Le cours débute la journée, mais par contre comme nous avons remarqué pour la classe 1 du même établissement, le cours est placé à la dernière heure d'enseignement. Lors du changement de professeur, certains élèves sortent et oublient de revenir à l'heure.

Les questions qui méritent d'être approfondies sont :

1. *Qu'est-ce qui a démotivé les étudiants, le cours d'informatique déconnecté ou la méthode d'enseignement utilisée ou le concept enseigné ?*
2. *Qu'est-ce qui favorise la motivation des autres élèves*?

Les discussions avec les directions ont apporté des réponses pédagogiques :

– Faire l'informatique aux premières heures augmente le nombre de retards et diminue la durée du cours, car il n'ya que une séance d'une heure par semaine.
– Il ne faut pas considérer l'informatique comme une discipline facultative, il faut l'introduire dans l'évaluation trimestrielle.
– Le livre de support de cours devrait être intégré dans la liste des matériels obligatoires lors de la rentrée scolaire.

5.4.2 Attentes des élèves sur l'utilité de l'informatique.

Nous présentons dans la figure 5.5 les attentes des élèves de la classe de 4ième, sur l'utilité de l'informatique, recueillie lors de la prise de contact. Les classe1 en bleu et classe2 en rouge sont au collège Notre-Dame tandis que les classe3 en baisent et classe4 en gris sont au collège Sainte Thérèse.

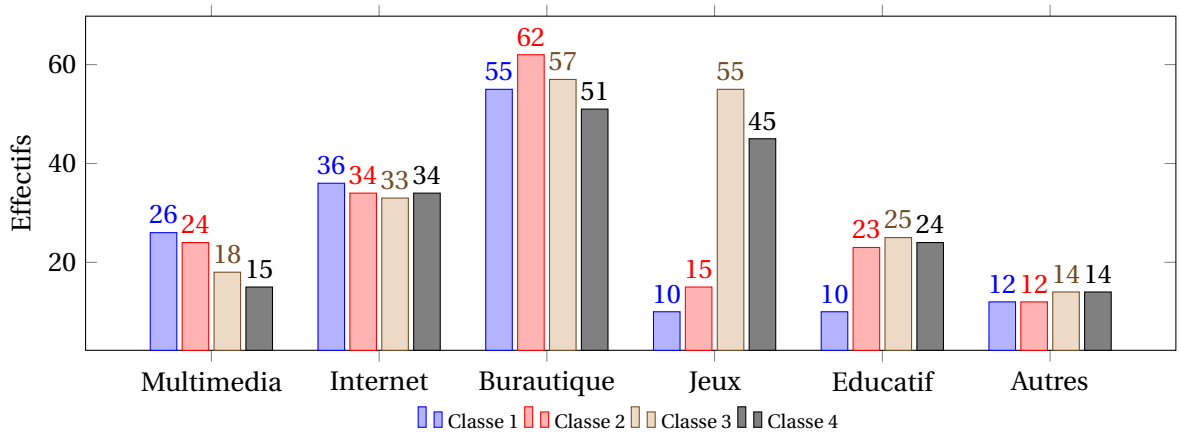


FIGURE 5.5 – Attente des élèves en informatique.

L'examen graphique reprend les intérêts des élèves pour l'informatique et qu'ils l'occasion de relever dans les pratiques à domicile journaux et les formations existences sur l'informatique, nous avons trouvé que les élèves ont des attentes variées sur le cours de l'informatique. Ceci correspond à des attentes variées qui peuvent être résumées de la manière suivante :

Bon nombre des élèves de presque toutes les classes, ont rêvé d'assister au cours d'informatique pour obtenir une formation en bureautique, dans laquelle seules les productions de document avec Microsoft Office Word, et la dactylographie et l'interaction sur ordinateur seront enseignées.

Les élèves rêvent aussi d'utiliser l'informatique pour se distraire, c'est-à-dire, l'hors de séance de travaux pratiques de l'informatique on leur a alloué un laps de temps pour qu'ils puissent jouer à des jeux vidéos. Ce désir affecte majoritairement les élèves de la classe moyenne, surtout les élèves du collège Sainte-Thérèse. Ceci pourrait laisser penser qu'ils ont déjà pratiqué une partie de leurs désirs chez eux.

Les élèves ont reconnu qu'après avoir assisté au cours d'informatique, ils seront plus habiles à choisir des sites web de rencontre ou à naviguer sur les réseaux sociaux et à établir la connexion internet sur leurs terminaux mobiles ou sur les smart phones d'OS Android. Les effectifs sont ici presque identiques entre les quatre classes de ces deux établissements.

Certaines élèves, majoritairement provenant de l'école Notre-Dame-de-Lourdes, attendaient que les montages des vidéos et des photos, les gravures des CD/DVD, la création des publications animées ou statiques soient enseignés lors du cours d'informatique. C'est le choix d'un grand nombre d'élèves des bas quartiers, où les studios des gravures et des reproductions multimédias informelles sont poussés comme de l'herbe folle.

Rares sont les élèves qui attendaient une vraie cour théorique en informatique. Et surtout que pendant le cours d'informatique, les connaissances ou des raisonnements mathématiques leur sont encore nécessaires.

De cette analyse, en nous préoccupant de notre objectif c'est-à-dire l'introduction du cours informatique, nous avons voulu connaitre les causes et/ou conséquences liées à des questions suivantes :

1. *Pourquoi les étudiants du Collège Ste Thérèse ont-ils choisi les jeux et l'éducation plus que les étudiants du Collège Notre-Dame ?*
2. *Qu'est-ce qui favorise le choix des jeux ou de l'éducation pour les collégiens de Ste Thérèse ?*

5.4.3 Influence de facteurs externes.

Après l'expérience et un traitement supplémentaire avec un logiciel de tableur, nous avons obtenu les résultats suivants. Dans tous les graphiques (fig 5.6), l'axe des abscisses représente les temps d'observation t_{15} en 15 minutes, t_{30} en 30 minutes et t_{45} en 45 minutes et l'axe des ordonnées représente les effectifs des élèves observés.

Premier graphique, bavardage des élèves

Pour les groupes des élèves travaillant en mode débranché, nous avons trouvé ; au premier quart heure 15 élèves bavards et puis 18 élèves au deuxième quart d'heure et ceci a monté jusqu'à 30 élèves vers la fin de la classe. *Nous avons remarqué la classe a été bavarde au début et devient de plus en plus bavarde.*

Par ailleurs, pour les groupes des élèves travaillant en mode branché, nous avons observé ; trois élèves bavards au début de la séance et ceci montent à six élèves à la mi-temps pour atteindre huit élèves vers la fin de la séance. *La classe est calme et quelques chuchotements sont entendus vers la fin.*

Deuxième graphique, concentration des élèves en classe

Dès le début de la séance, pour les élèves travaillant en mode débranché, 20 élèves sont déjà classés comme non concentré au cours, et ce nombre augmente à 26 élèves au moitié du temps jusqu'à s'élever élevé à 30 élèves vers la fin de la séance. *Nous avons remarqué une progression remarquable des élèves déconcentrés durant cette séance.*

Par contre pour les groupes connectés, bien que quelques élèves (sept élèves) ne soient pas concen-

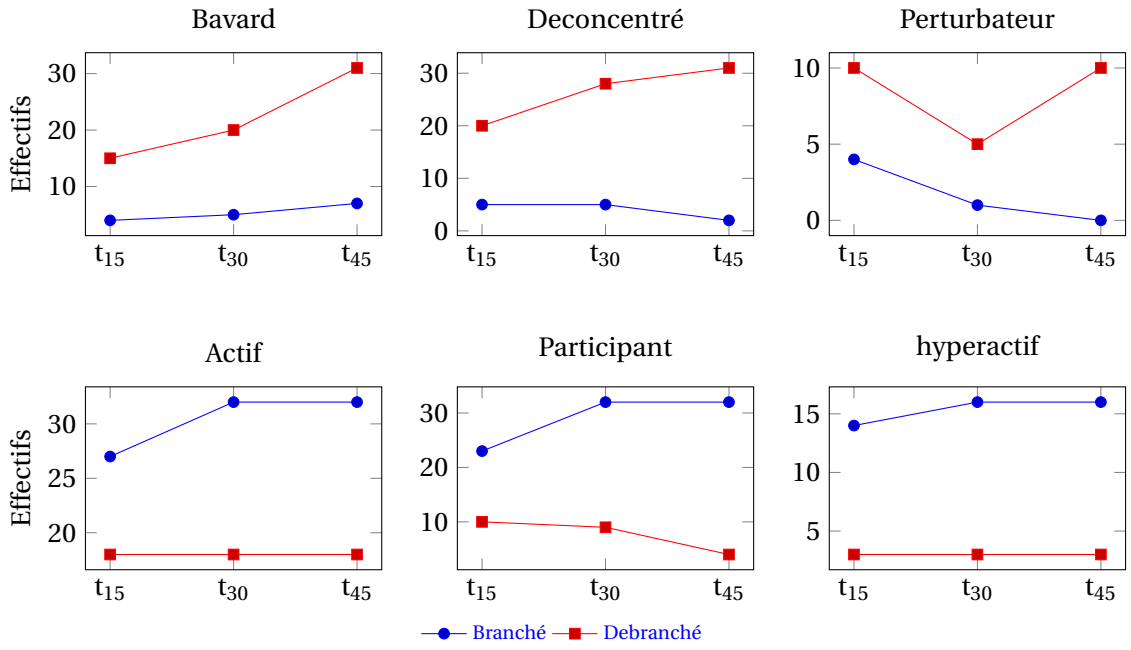


FIGURE 5.6 – Comportement des élèves.

trés au début de la séance, ce nombre reste constant jusqu'à la trentième minute de la séance et même diminue à seulement quatre élèves vers la fin. *Nous avons remarqué une classe très concentrée.*

Troisième graphique, perturbation des élèves en classe

Pour le groupe des élèves débranchés, au début de la séance dix élèves perturbent déjà la classe, cet effectif diminue de moitié après un quart d'heure, le nombre remonte de nouveau à dix élèves vers la fin. *Nous avons remarqué que la classe est un peu turbulente.*

Pour le groupe des élèves branchés, une légère perturbation est faite par quatre élèves au début puis diminue à un élève vers la moitié du temps et a disparu vers la fin. *Nous avons remarqué une classe calme.*

Quatrième graphique, activité des élèves en classe

Pour le groupe des élèves travaillant en mode branché, 25 élèves (les chefs de groupes et quelques élèves) sont déjà actifs, ce nombre augmente et dès la trentième minute on arrive à 32 élèves, cet effectif restera constant jusqu'à la fin de la séance. *Nous avons remarqué qu'au moins un membre de groupe est déjà actif au début de la séance et le nombre augmente.*

Pour le groupe des élèves débranchés, nous avons trouvé seulement que 15 élèves sont actifs et qu'ils restent actifs jusqu'à la fin de la séance. *Nous avons remarqué que seul le chef de groupe (ou le secrétaire) est actif.*

Cinquième graphique, participation des élèves en classe

Pour le groupe des élèves branchés, depuis le premier quart d'heure, l'échange et la participation de 22 élèves sont déjà remarqués, cela augmente un peu et ce sont 32 élèves au deuxième quart

d'heure. À partir de ce moment, cet effectif reste stationnaire jusqu'à la fin de la séance. *Nous avons observé une participation effective et croissante des élèves dans ce groupe classe.*
Concernant le groupe des élèves débranchés, en partant du premier temps d'observation, avec des effectifs de dix élèves, et au deuxième temps d'observation, avec des effectifs de huit élèves, le nombre de participations des élèves en classe n'a cessé de diminuer jusqu'à avoir trois élèves à la fin de la séance. *Nous avons observé une dégradation de participation des élèves dans ce groupe classe.*

Sixième graphique, hyperactivité des élèves en classe
Nous avons noté des hyperactifs, tous les étudiants utilisant ou consultant un autre outil ou autre support de documentation. Pour le groupe des élèves, le nombre des élèves hyperactifs augmente presque considérablement depuis le début jusqu'à la fin de la séance. *Nous avons remarqué une classe très active et des apprenants très motivés.*
Par ailleurs il reste constant et d'effectif très faible pour le groupe travaillant en mode branché. *Nous avons remarqué une classe passive et des apprenants découragés.*

5.4.4 Taxonomie des structures algorithmiques

Dans la figure 5.7 sur ces trois graphiques, nous présentons sur l'axe des y, les effectifs d'élèves ayant réussi les activités proposées au cours d'une séance et sur l'axe des x les séances d'enseignements réalisées (première séance S_1, deuxième séance S_2 et la troisième et dernière séance S_3).

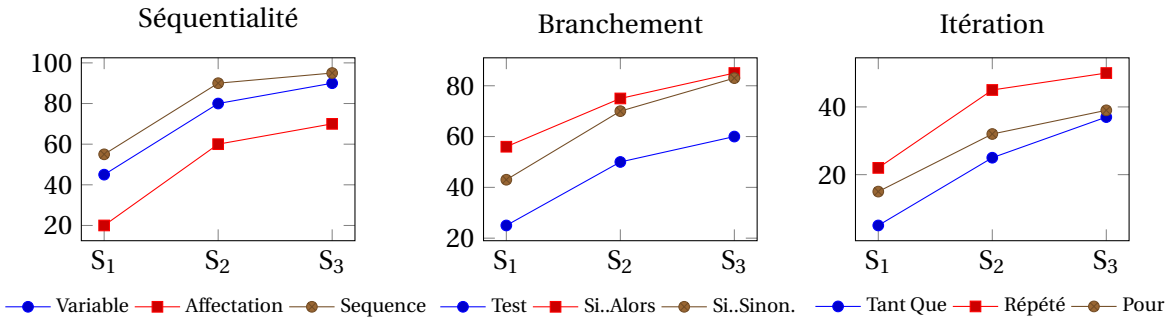


FIGURE 5.7 – Taxonomie des structures algorithmiques.

Suite aux propositions du programme que nous avons prescrit au préalable dans le tableau, nous nous sommes passés à une série de tests, qui sont basés sur l'algorithme d'enseignement en classe de quatrième. Deux classes ont été dispensées, l'une à l'école Notre-Dame et l'autre à l'école Sainte-Thérèse. Ceci à duré plus de trois mois. Voilà comment se déroule le processus. Le cours a été enseigné de manière séquentielle, suivi de près par des carnets de compétence ou port folio que les étudiants ont reçu.

Les unités d'enseignement reçues par chaque groupe portaient sur la structure. Les éléments composites ont été présentés ensemble lors des trois sessions d'étude. Ces éléments sont divisés en trois groupes ;

La première séquence d'enseignement avait porté sur la structure séquentielle où les notions de variables, l'instruction d'affectation et les séquences d'instruction d'entrée / sortie étaient enseignées. Nous sommes persuadés qu'il n'était pas difficile pour les élèves de comprendre et de s'approprier la notion de séquence, car presque plus de la moitié des élèves ont compris et assimilé cette séquence d'enseignement lors de la première séance.

En effet, les notions de variable ne sont pas difficiles pour eux, mais le langage de déclaration et de mise à jour de sa valeur présente des formes inhabituelles en Scratch. Mais après, la notion d'affectation est un peu plus difficile pour les élèves du fait de ses petites différences dans les connaissances acquises en mathématiques.
Une seule activité a suffi à introduire cette séquence d'enseignement, sur laquelle nous leur avons demandé d'afficher une suite de valeurs introduient par une tierce personne à l'aide d'un clavier, pourtant il a fallu qu'on dispose d'un espace mémoire prêt à recevoir ces valeurs (variable, lecture affectation et affichage). Cette séquence d'enseignement pourra être faite en une séance.

La seconde séquence d'enseignement avait porté sur les « branchements conditionnels », composée de expression booléenne, `SI..ALORS` et `SI..SINON`. C'est très naturel pour les élèves de comprendre `SI..ALORS` surtout si le test utilisé est simple, et ils continuent à comprendre sinon, la connaissance des expressions booléennes ou des tests conditionnels deviennent les conditions nécessaires et suffisantes pour comprendre les structures de branchements simples.

En effet, une activité pouvant afficher le signe d'un nombre introduit par une tierce personne par clavier pourra introduire brièvement cette séquence, mais ici la condition sur le signe de nombre est simple (`x : = LIRE("entrer x"), SI x < 0 ALORS DIRE(" x est négatif") SINON DIRE("x est positif")`). La parité, multiple, colinéarité perpendicularité, etc., pourront étendre ou modifie la variable didactique du test. Ainsi deux séances sont suffisantes pour introduire ce concept ; une séance d'introduction à des tests simples et une autre pour l'approfondissement avec des tests conditionnels complexes.

La troisième séquence d'enseignement porte sur les "structures répétitives" dans lesquelles trois structures ont enseignées ; `TANT QUE`, `REPETER` et `POUR`. Mais, comme qu'on n'a pas fonction `REPETER..JUSQU'A` et `REPETER n FOIS` en scratch, notre enseignement était centré seulement sur ces deux structures ;

La structure itérative prise comme équivalence à `REPETER n FOIS`, ne s'avère pas difficile pour eux, les programmes de construction géométrique en connaissant le nombre des côtés pourront introduire cette itération. (*Exemple* : Carré ; `REPETER 4 FOIS {AVANCER 10 PAS, TOURNER DE 90ř)`

`REPETER..JUSQU'A` parait être la bête noire pour les élèves. Définir la valeur initiale de la variable, trouver l'expression invariante et le pas d'incrémentation, enfin trouver ou exprimer la condition d'arrêt figure parmi les problèmes à résoudre qui demandent beaucoup plus des rai-

sonnements mathématiques. (*Exemple* : Somme des 10 premiers nombres ; `S= 0, i=1, REPETER (S=S+i, i=i+i) JUSQU'A i=10` ou `TANT QUE i<=10 FAIRE (S=S+i, i=i+i)`. C'est la structure la plus appropriée et la plus utilisée des résolutions des problèmes, une ou deux séances ne leur suffisent pas.

Genèse épistémologique des structures algorithmiques

En 820 : Le mathématicien **El Khawarizmi** publie à Bagdad un traité intitulé « *La science de l'élimination et de la réduction* » qui importé en Europe occidentale lors des invasions Arabes aura une grande influence sur le développement des mathématiques.

Séquentialité. : Collaboratrice de Babbage, Ada Lovelace mathématicienne, définit en 1840, le principe des itérations successives dans l'exécution d'une opération. En l'honneur du mathématicien Arabe **El Khawarizmi** (820), elle nomme le processus logique d'exécution d'un programme : **algorithme**.

Branchement. En 1854 G Boole démontre que tout processus logique peut être décomposé en une suite d'opérations logiques (ET, OU, NON) appliquées sur deux états (ZERO-UN, OUI-NON, VRAI-FAUX, OUVERT-FERME).(Boole, 1854) Babbage l'a clairement senti lorsqu'il écrivait que sa machine pouvait résoudre n'importe quelle équation et exécuter les opérations les plus compliquées de l'analyse. Son intuition recevra un fondement théorique en 1936, lorsque Turing montra que toute fonction calculable pouvait l'être à l'aide d'une unique machine numérique, abstraite certes, mais dont la machine de Babbage était une préfiguration. (Moreau, 1987, p20)

Variable informatique et affectation. Depuis 1945, la contribution fondamentale de Von Neumann à la conception de la machine appelée « *ordinateur* » est nommée couramment « *Principe de Von Neumann* ». Les instructions peuvent alors être modifiées par la machine durant l'exécution même d'un programme. L'idée de programme enregistré présente deux avantages principaux, celui de l'accélération des calculs et celui de la modification des instructions par la machine elle-même

Variable et boucle, instruction. Cependant, en 1957, Kuntzmann accorde une place centrale à la notion d'instruction (Kuntzmann, 1957). Ce qu'il appelle « *mémoire convenable* » est une variable informatique, en l'occurrence la variable « *compteur décrémenté* ». Cependant les premiers langages en restant proches des caractéristiques technologiques de la machine opacifient la signification du programme et des notions qui y sont fondamentalement liées comme variable et boucle. L'écriture de programmes de calculs itératifs dans des langages éloignés de la machine est une condition nécessaire à l'émergence d'une notation d'affectation dans une boucle qui permettra de rendre pleinement compte de la signification de la notion de variable en informatique.

Itération. Depuis Ada (1840), la boucle « *Goto* » est présente dans tous les langages de haut niveau. Or cette boucle, attachée au fonctionnement de la machine, peut produire des programmes peu structurés. En 1968, prenant acte de ces risques, Dijkstra rédige dans les Communications of the ACM l'article fondateur « *On the Go To Statement Considered Harmful* » (Dijkstra, 1968), dans

lequel il recommande l'abandon de l'usage de « *Goto* » dans les pratiques de programmation. Pour Dijkstra, l'usage incontrôlé de la boucle `GOTO` rend difficile la lecture du programme, et devient facultative du fait de l'existence d'autres structures d'itération comme `WHILE` et `REPEAT`.

Ainsi le paradigme d'une programmation structurée dans le style impératif repose sur trois types de boucles : « *compteur* », `TANT QUE` et `REPETERJUSQU'A`. Nous regroupons ces différentes écritures de boucles en deux catégories selon que le nombre d'itérations est connu à l'avance ou non :

- Boucle « *compteur* » : le nombre d'itérations est déterminé à l'avance `REPETER n FOIS`. La variable compteur peut être décrémentée (comme dans le premier programme informatique écrit par Ada) ou incrémentée. Ce type de boucle simplifie fortement la syntaxe du langage de programmation puisque l'instruction de boucle se ramène à `REPETER n FOIS`.
- Boucle dont le nombre d'itérations n'est pas connu à l'avance. Dans ce cas il y a nécessité de formuler une condition logique d'arrêt pour sortir de la boucle. Cette condition peut être placée juste après l'entrée de la boucle « *vérifier condition et répéter* » ou juste avant la sortie de la boucle « *répéter* ⋯ *vérifier condition* ».

La construction d'un algorithme ou d'un programme informatique nécessite la combinaison de trois types d'instruction qui interviennent Nguyen & Bessot (2003) : instructions d'affectations, instructions conditionnelles et instructions itératives. Par ordre de difficulté croissante des structures algorithmiques chez les novices en informatique, notre analyse et le panorama de la littérature de recherche semblent donner une classification suivante :

- Affectations.
  - ◇ Les structures séquentielles.
  - ◇ Variables.
- Les branchements ou les structures conditionnelles.
  - ◇ Test conditionnel.
  - ◇ `SI..ALORS`
  - ◇ `Si..Alors..Sinon`.
- Les boucles ou structures répétitives.
  - ◇ Répéter ⋯ jusqu'a ce que.
  - ◇ Pour ⋯ jusqu'a ⋯ faire.
  - ◇ Tant que ⋯ faire.

5.5 Evaluation des acquis des élèves en algorithmique

5.5.1 Compétence analysée à la lecture

Compétence des élèves sur la lecture d'un algorithme (réponse Question. 4.8.6)

Nous avons trouvé sur ce tableau que tous les groupes sont parvenus à la lecture malgré quelques erreurs pour certains groupes. L'écart où la différence se trouve sur le nombre des erreurs commises et aux vitesses de lecture. Les groupes ont commis plus de fautes de lecture en Scratch qu'en

	Scratch coloré			Scratch non coloré			Algo-box		
	Aisance	Erreur	Succ.	Aisance	Erreur		Aisance	Erreur	Succ.
Gr1	insat	Satisf	Non	insat	Satisf	Non	Satisf	Satisf	Non
Gr2	faible	Satisf	Non	faible	Satisf	Non	Satisf	Satisf	Oui
Gr3	faible	Satisf	Oui	faible	Satisf	Oui	Satisf	T. bon	Oui
Gr4	faible	Satisf	Oui	faible	Satisf	Oui	faible	T. bon	Oui
Gr5	faible	Satisf	Oui	Satisf	Satisf	Oui	Satisf	T. bon	Oui
Gr6	faible	Satisf	Oui	faible	Satisf	Oui	faible	T. bon	Oui
Gr7	Satisf	Satisf	Oui	faible	Satisf	Oui	Satisf	T. bon	Oui
Gr8	Satisf	Satisf	Oui	Satisf	Satisf	Oui	Satisf	T. bon	Oui
Gr9	Satisf	Satisf	Oui	Satisf	Satisf	Oui	T. bon	T. bon	Oui
Gr10	Satisf	Satisf	Oui	Satisf	Satisf	Oui	T. bon	T. bon	Oui
Gr11	Satisf	Satisf	Oui	Satisf	Satisf	Oui	T. bon	T. bon	Oui
Gr12	Satisf	T. bon	Oui	Satisf	T. bon	Oui	T. bon	T. bon	Oui
Gr13	Satisf	T. bon	Oui	Satisf	T. bon	Oui	T. bon	T. bon	Oui

TABLEAU 5.2 – Savoir lire un algorithme

pseudocode. Et la vitesse est beaucoup plus lente lorsqu'ils ont fait le passage aux opérations algébriques.

La lecture des opérations algébrique ou booléenne est très intuitive en algobox ou en pseudocode ne présente aucune difficulté. Par ailleurs, comprendre un algorithme présenté en langage Scratch ne parait pas très compliqué aux élèves, mais sa verbalisation, ou sa traduction en langage naturel parait très difficile pour eux.

En effet, en langage Scratch le problème de lecture ou de verbalisation des algorithmes se pose sur les structures semblant les plus élémentaires, à savoir : l'initialisation et déclaration des variables, l'affectation d'une valeur à une variable et les expressions booléennes ou algébriques, comme la manipulation des chaines des caractères sont difficile.
Par exemple :

signifie attribuer la valeur 10 au variable : A ; A ⟵ 1

signifie incrémenter la valeur de F par 1 : F ⟵ F + 1

concatener "texte1" avec " texte2" : "texte1" + " texte2"

A ⟵ Lire("entrer un nombre")

La couleur et la forme ne présentent aucune différence d'où une difficulté remarquable en lecture, l'obstacle n'est pas sur l'aspect sémiotique, mais sur l'aspect sémantique, c'est à dire lors de la définition de ces objets informatiques en Scratch. Il s'agit des erreurs lors de la traduction de ces objets algorithmiques en des objets informatiques sur l'environnement Scratch. Ces sont des

erreurs de transposition informatique que les auteurs définissent comme un obstacle didactique.

Interprétation

Les lenteurs en lecture et l'abondance des erreurs commises sur des algorithmes présentés en Scratch indiquent la difficulté de la verbalisation. Pourtant la lecture des pseudocodes semble un peu plus naturelle pour tous les groupes, la vitesse est rapide avec une moindre faute commise.
Classification à quatre degrés, selon la nature sémiotique des ILEIS [2]

1. Analogique : ressemblance, homomorphisme (*traitement figuratif sur base d'une donnée perceptif*)
2. Analogique conventionnel : homomorphisme, convention (*traitement figuratif. signification conventionnelle*)
3. Symbolique analogique : (arbitraire), mais incorporant valeur onomatopéique linguistique et/ou scripto-visuelle.
4. Symbolique : langue naturelle (signe linguistique) ou artificielle (*langage formulaire*).

5.5.2 Compétence analysée à l'écriture

	Outils de vérification	Durée	démarche	Succ.
Gr1	Scratch	insuff	insuff	Non
Gr2	Scratch	insuff	insuff	Non
Gr3	Scratch	insuff	insuff	Non
Gr4	Scratch	insuff	insuff	Non
Gr5	Scratch	insuff	insuff	Non
Gr6	Scratch	insuff	Satisf.	Non
Gr7	Scratch	insuff	Satisf.	Non
Gr8	Scratch	Faible	Satisf.	Non
Gr9	Scratch	Faible	Satisf.	Non
Gr10	Papier Crayon, Scratch	Faible	Satisf.	Non
Gr11	Papier Crayon, Scratch	Faible	Satisf.	Oui
Gr12	Papier Crayon, Scratch	Faible	Satisf.	Oui
Gr13	Papier Crayon, Scratch	Faible	Satisf.	Oui

TABLEAU 5.3 – Savoir écrire un algorithme

Compétence des élèves sur l'exécution d'un algorithme (Réponse à la question. 4.8.8)

Concernant l'écriture d'un algorithme pour résoudre un problème donné, beaucoup d'élèves oublient de stipuler les entrées et les sorties. De plus, certains ne semblent pas encore maîtriser les différentes structures comme l'affectation (notée = dans certaines copies) et les deux boucles.

2. Les ILEIS sont les représentations figurées à petite taille (Icônes).

Certains élèves de trois groupes sur 13 ont de très bons algorithmes, les groupes qui ont commencé la résolution du problème par papier crayon, et puis l'ont formulé en langage algorithmique avant de le tester sur le logiciel Scratch. Tandis que d'autres n'ont pas réellement réfléchi au problème, ils ont directement tenté leur réussite comme d'habitude par essai erreur sur Scratch et n'y apportent donc aucune solution. Enfin, chez certains élèves, les algorithmes sont presque corrects, à un ou deux détails près. Mais ils donnent des résultats faux, car ils se sont lâchés et n'ont pas résolu le problème. Nous sommes persuadés que si ces groupes avaient testé au fond leurs algorithmes étape par étape, ils auraient probablement aperçu les problèmes. Il s'agit là d'un reproche que nous avions déjà formulé à l'égard des étudiants universitaires.

Confronter les élèves à de vrais problèmes (construction des algorithmes), lorsque nous demandons leur avis, nous constatons qu'ils sont assez d'accord pour dire qu'ils ont trouvé que ce soit la partie le plus intéressant, même si elle comporte quelques difficultés.

5.5.3 Compéténce analysée à l'exécution

	Scratch coloré			Scratch non coloré			Algo-box		
	Vitesse	Erreur	Succ.	Vitesse	Erreur	Succ.	Vitesse	Erreur	Succ.
Gr1	faible	Satisf.	Non	faible	Satisf.	Non	insuff	insuff	Non
Gr2	faible	Satisf.	Oui	faible	Satisf.	Non	insuff	insuff	Non
Gr3	Satisf.	T. bon	Oui	Satisf.	T. bon	Non	insuff	insuff	Non
Gr4	Satisf.	T. bon	Oui	Satisf.	T. bon	Oui	insuff	insuff	Non
Gr5	Satisf.	T. bon	Oui	Satisf.	T. bon	Oui	insuff	insuff	Non
Gr6	Satisf.	T. bon	Oui	Satisf.	T. bon	Oui	insuff	insuff	Non
Gr7	T. bon	T. bon	Oui	Satisf	T. bon	Oui	insuff	insuff	Non
Gr8	T. bon	T. bon	Oui	Satisf	T. bon	Oui	insuff	Faible	Non
Gr9	T. bon	T. bon	Oui	Satisf	T. bon	Oui	insuff	Faible	Non
Gr10	T. bon	T. bon	Oui	Satisf	T. bon	Oui	insuff	Faible	Non
Gr11	T. bon	T. bon	Oui	T. bon	T. bon	Oui	insuff	Faible	Non
Gr12	T. bon	T. bon	Oui	T. bon	T. bon	Oui	insuff	Faible	Oui
Gr13	Satisf.	T. bon	Oui	Satisf.	T. bon	Oui	insuff	Satisf.	Oui

TABLEAU 5.4 – Savoir exécuter un algorithme

Competence des élèves sur l'exécution d'un algorithme présenté en Scratch coloré (Réponse à la question. 1)

Sur les 13 groupes, 12 parviennent à exécuter cet algorithme correctement. Nous nous attendions à ce que cette question soit bien réussie, mais nous constatons que 2 groupes (16%) ne sont pas parvenus à trouver la bonne réponse qu'après plusieurs tentatives, ce qui n'est pas négligeable. Et nous observons que cinq groupes ont réussi à faire fonctionner cet algorithme, mais ce fut un peu lent.

En plus des erreurs commises, nous trouvons, entre autres, des groupes qui ne savent pas bien interagir avec l'ordinateur. Malgré les structures éligibles déjà à partir de sa forme et de sa coloration, le parcours de choix des structures dans les menus leur présente encore des obstacles.

Compétence des élèves sur l'exécution d'un algorithme présenté en Scratch non coloré (Réponse à la question.2)

Légèrement moins bien qu'avec la forme colorée sur les 13 groupes, 11 parviennent à exécuter cet algorithme correctement, mais parmi eux, la majorité est parvenue à trouver la bonne réponse qu'après plusieurs tentatives infructueuses. De plus, une lenteur par rapport à la première forme colorée a été remarquée. Parmi les erreurs commises, nous trouvons, entre autres, des groupes qui ne savent pas bien interagir avec l'ordinateur.

Compétence des élèves sur l'exécution d'un algorithme présenté en pseudocode en Algo-box (Réponse à la question.3)

La réponse est non, seuls 2 groupes y sont parvenus.Un a réussit après beaucoup d'essais. Nous pensions que réécrire un code donné ne devait poser aucun problème. Nous relatons les difficultés des groupes n'ayant pas réussi comme les suivants :

- d'erreurs commises lors des saisies, de difficultés liées aux manques de compétence en traitement de texte, de confusion sur les ponctuations comme des virgules et point-virgule et même aussi d'espaces ;
- d'erreurs de syntaxe, de manque de respect ou de rigueur sur les syntaxes demandées, de confusions des caractères en majuscules et en minuscules.

Interprétation

Donc, exécuter des algorithmes présentés dans un langage visuel et coloré est plus favorable et apprécié par les élèves que de manipuler les pseudocodes.

Les résultats démontrent que la majorité des groupes a réussi à faire fonctionner l'algorithme présenté sous sa forme colorée en Scratch, puis avec un peu d'écart celui présenté sous sa forme non coloré. L'aspect visuel et chromatique accélère l'orientation rapide des choix de structures à utiliser chez les élèves, la manipulation simplifiée par clic-glisser réduit le temps de conception et diminue le nombre d'erreurs syntaxiques à commettre. Peu de groupes ont réussi à faire fonctionner l'algobox. D'ailleurs, les erreurs syntaxiques commises sont nombreuses en algobox et la difficulté de saisie à l'aide du clavier pour les élèves débutants en informatique provoque des obstacles majeurs.

5.5.4 Résultats

L'écriture ou le code d'un algorithme présenté sous forme de problèmes est encore très difficile pour les élèves de collège. Déjà la partie résolution réserve des difficultés, la notion d'algorithme est encore nouvelle pour eux. Les choix de la structure utilisée amplifient les échecs.

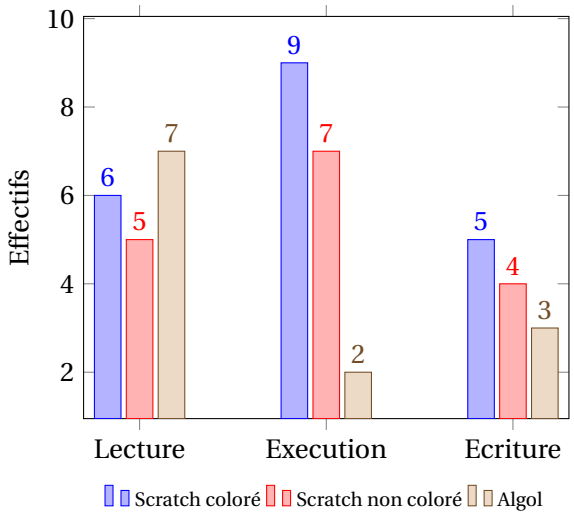


FIGURE 5.8 – Histogramme (Lecture Ecriture Exécution).

On a observé que pratiquement tous les étudiants qui ont participé ont réussi à faire fonctionner, en langage Scratch coloré, le codage et l'exécution du programme proposé à l'activité.

Dans la deuxième activité, concernant l'enchainement de la structure de sélection à l'aide langage Scratch non coloré, environ 50% les élèves ont réalisé que le chainage était la solution la plus appropriée. Cependant, 50% ont proposé la solution de manière séquentielle.

5.5.5 Interprétation empirique

Les résultats sont très intéressants, car ils montrent que l'apprentissage de l'algorithme par la représentation textuelle est lent, mais en revanche, le score est double grâce à des apprentissages par les représentations imagées, colorées et formelles. Les résultats s'expliquent bien dans la conception des multiples représentions : sémantique, syntaxique, graphique. La représentation imagée devient donc utile si la performance requiert des perceptions et des capacités codages qui dénotent une iconification de l'algorithme.

À l'inverse, une autre expérience sur la lecture révèle que la représentation imagée est inutile. Tout d'abord, la lecture est un processus complexe qui nécessite au moins quatre principaux codages, sémantique , lexical, phonologique et visuographique . Le décodage de lexique et de phonème à partir d'une représentation imagée soulève beaucoup de difficultés chez les apprenants. Lors de la lecture on insiste sur « la maîtrise des correspondances entre les lettres et les mots » (codage phonologique)

Les représentations imagées seraient donc essentiellement inutiles lorsque la performance requiert des capacités linguistiques, dont on souhaite que les élèves puissent reconnaitre des mots anticipés dans les syntaxes, qui dénotent une verbalisation de l'algorithme.

En fonction des recherches, quatre représentations sont fondamentales dans l'apprentissage des algorithmes. En s'inspirant de la signalétique des « étiquettes énergie », une méthode sera d'autant meilleure qu'elle permet un entrainement dans tous les codes et non seulement dans quelques-uns.

Conclusions de ces études partielles.
D'après cette étude nous avons pris les dispositions suivantes :

- Suggérer l'apprentissage et interaction avec l'environnement Windows depuis la classe antérieure comme : les déplacements / clics de la souris, le lancement /la fermeture de logiciels, les rôles de outils de base dans la barre d'outils standards.
- Motiver les apprentis à l'étude des algorithmiques en utilisant le langage de programmation visuelle comme Scratch en complément de sa représentation en pseudo-code.

5.6 Proposition de mettre en œuvre une méthode de Jigsaw classroom.

5.6.1 Modélisation d'études.

Nous proposons une stratégie d'animation de groupe inspirée de la méthode de Jigsaw. Cette méthode consiste à partager les membres d'un groupe très doué pour aider les trois autres groupes moins avancés. La construction de groupes homogènes se ferait en trois étapes, de sorte que dans la dernière phase, les niveaux des étudiants soient presque homogènes.

En classe de quatrième avec 42 élèves, répartis en 14 groupes, nous avons expérimenté cette méthode. Des cours ont été organisés en 12 groupes de quatre étudiants et deux groupes de cinq étudiants. Nous avons utilisé le modèle d'éducation socioconstructiviste et permettons aux étudiants des groupes de s'autoétudier et de communiquer entre eux avant de prendre des actions ou des décisions.

L'activité à réaliser est donnée avec des instructions très claires. Ils étaient autorisés à utiliser tous les documents et à utiliser tous les outils sélectionnés.

Le rôle de l'enseignant est de garantir que la situation didactique se déroule en classe, ainsi il s'assurera du silence dans la classe, répondra à toutes les questions qui pourraient survenir, surveillera les progrès de chaque groupe, identifiera ceux qui sont en retard et déterminera ceux qui sont très avancés.

5.6.2 Résultats obtenus.

Premiere phase. Après trente minutes de travail, nous avons remarqué une bonne progression du groupe 13, qui a mis fin à toutes leurs activités, alors que, le groupe 1, le groupe 2 et le groupe 3 ont rencontré quelques difficultés qui ont bloqué le système dès la première activité. Les difficultés observées qui ont retardé ces groupes sont liées à une mauvaise compréhension du sujet, l'ajout d'un nouveau membre qui maîtrise le sujet dans chacun de ces groupes peut apporter des solutions.

Cependant, nous avons dissout le groupe 14 et recréé les membres des 3 derniers groupes. Pour ce faire, nous avons donc remplacé un membre de chacun de ces trois groupes par trois membres du groupe 14, et les autres formeront un nouveau groupe A.

La figure 5.9 à droite montre la répartition des groupes au cours de cette étape et à gauche une courbe d'évaluation qui montre le niveau avancé du groupe 14 et les retards des trois derniers groupes.

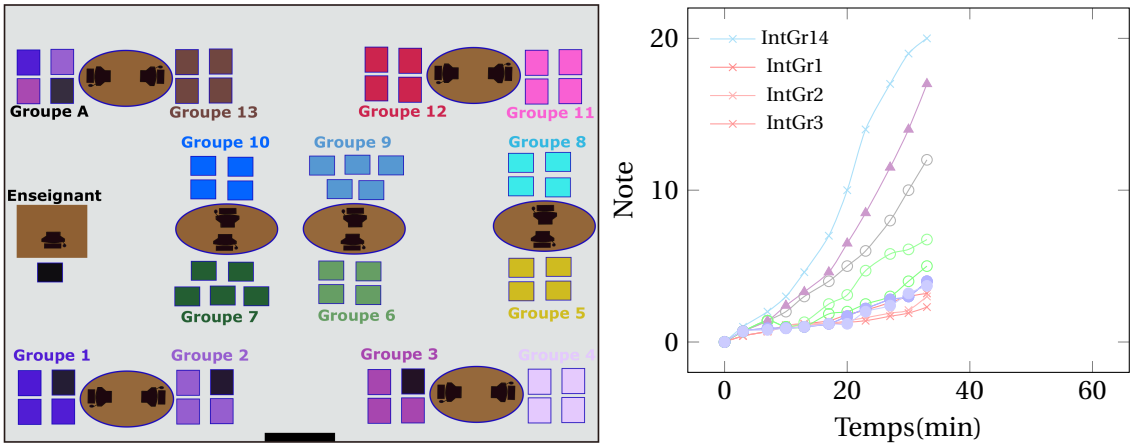


FIGURE 5.9 – Travail de groupe Jigsaw amélioré stade 1.

Deuxième phase

Après la nouvelle reconstitution effectuée dans la première phase, chaque groupe continue son activité et le groupe nouvellement créé (groupe A) reprend rapidement ses activités. Après 40 minutes de temps régulier, soit dix minutes plus tard après la première phase, le nouveau groupe 13 à son tour termine ses activités (fig 5.10), et nous avons noté les trois autres qui occupent la dernière position, dont le groupe quatre, le groupe sept et le groupe dix. Les difficultés principales observées à ce stade sont basées sur la résolution du problème, le choix de la méthode de résolution. Une petite discussion avec un tiers peut résoudre ces difficultés. Comme nous l'avons fait dans la première phase, du groupe 13, trois de ses membres rejoindront les groupes quatre, sept et dix, les autres créeront un nouveau groupe B.

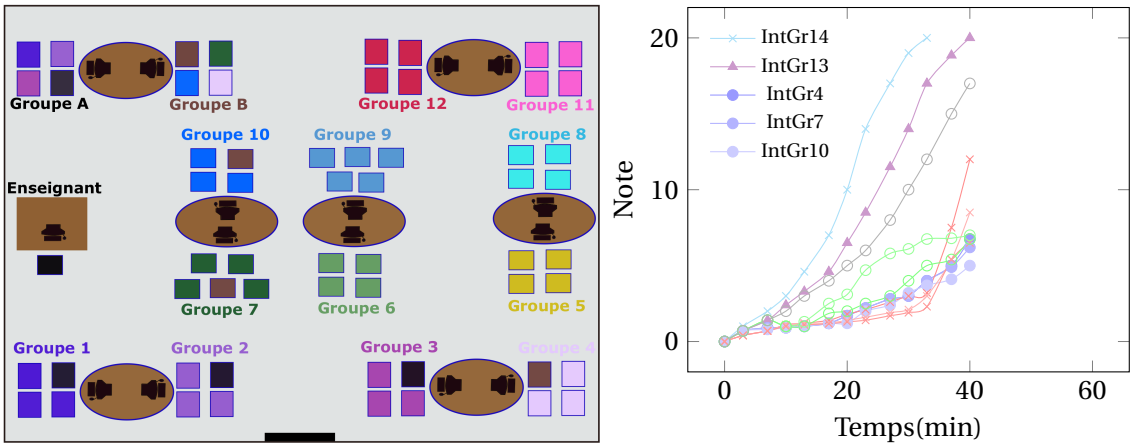


FIGURE 5.10 – Travail de groupe Jigsaw amélioré stade 2.

Nous avons présenté dans la figure 5.10 ci-dessus, à gauche, la disposition des groupes lors de cette deuxième phase et à droite, les courbes d'évaluations de chaque groupe correspondant à cette phase.

Troisième phase

A cette phase ce sont les dernières reconstitutions adoptées, à environ 50 minutes de travail. Deux groupes étaient à nouveau en tête (groupes 12 et 11), et les groupes quatre, groupes cinq

et groupes neuf étaient au dernier rang. Les principales difficultés observées ici concernent les erreurs sémantiques et syntaxiques ou les manipulations incorrectes, qui peuvent conduire à un échec total, voire une bonne compréhension du problème et un bon choix de méthode de résolution. Lorsqu'une de ces difficultés a été levée, ils ont„ sans aucun problème, réussi à y parvenir.

Comme toutes les autres phases, nous avons recréé ces trois groupes et dissout le groupe 11 pour créer le dernier nouveau groupe C.

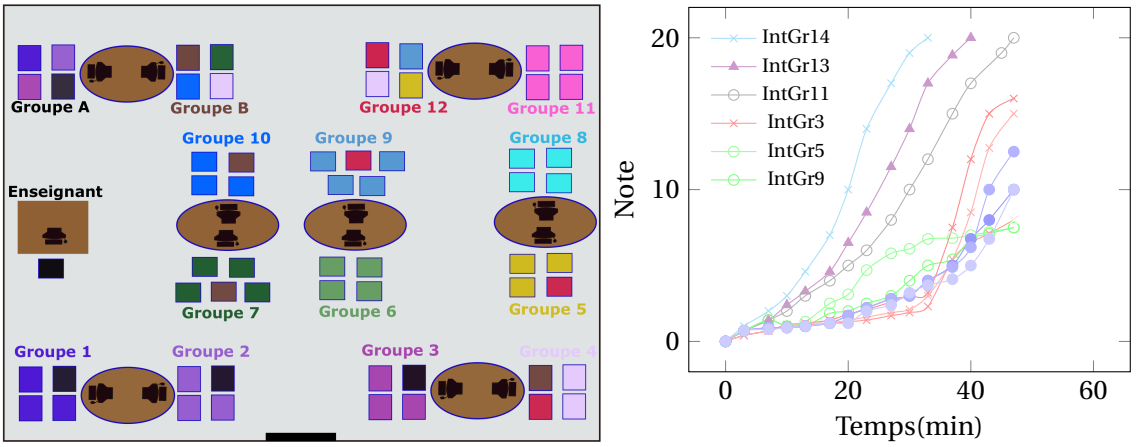


FIGURE 5.11 – Travail de groupe Jigsaw amélioré stade 3.

La représentation du système concerné ainsi que les courbes d'évaluation associées sont illustrées dans la figure 5.11.

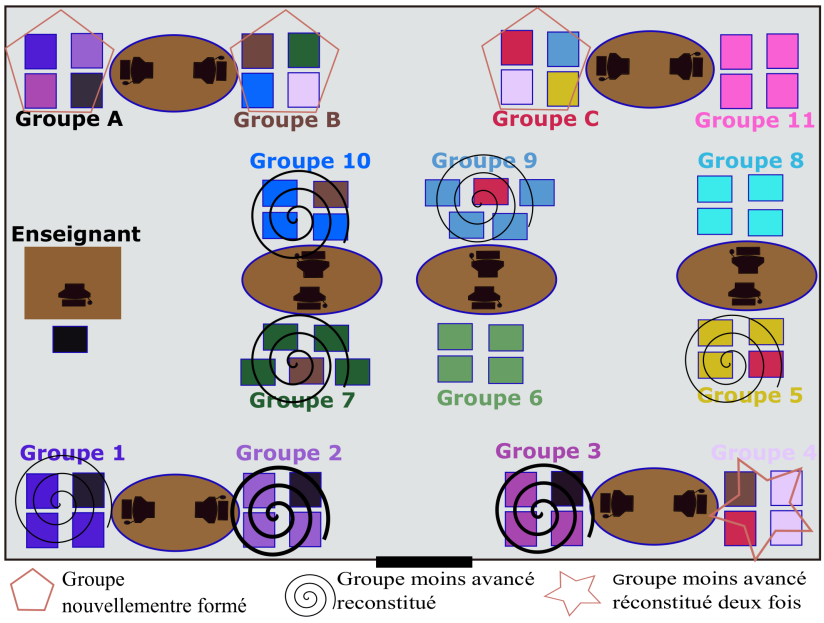


FIGURE 5.12 – Travail de groupe situation finale.

Cette nouvelle méthodologie de conduite du travail de groupe recréé, que nous avons pratiquée en classe de quatrième au collège, a également été utilisée à l'université lors des cours pratiques au laboratoire. Les résultats que nous avons observés au collège sont à nouveau apparus à

l'université.

Après avoir répété cette méthode plusieurs fois, nous remarquons que dans la phase finale (dans la seconde ou la troisième phase) nous avons observé l'homogénéité des niveaux de groupe, aucun groupe talentueux ne mettra fin à son activité avant la durée normale et aucun groupe faible n'échouera, tous les groupes ont tendance à mettre fin à leur activité (surtout, si on consacre un peu de temps aux derniers groupes nouvellement créés).

5.7 Etude globale à l'aide de l'ASI-MGK

L'examen du résultat graphique nous a permis d'observer un graphe orienté à 34 nœuds et 33 arcs, c'est-à-dire 33 relations binaires entre les 34 modalités étudiées. De sa représentation spatiale, l'interprétation s'avère être impossible. Dans le chapitre 2, section 2.8.5 nous avons introduit les méthodes d'amélioration de vue.

5.7.1 Analyse et interprétations

Nous avons donc appliqué quelques interactions à notre graphe. Il s'agit de l'amélioration de vue par changement de donnée d'entrée (réduction de nombre des règles valides en diminuant le seuil d'erreur et $minSupp$) et amélioration de vue par déplacement. Le graphe ainsi obtenu comprend, huit composantes connexes, que nous avons pu présenter en six sous graphes.

Affection.

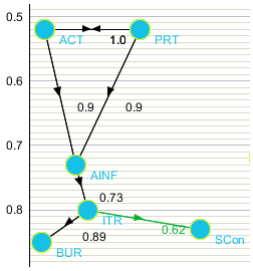


FIGURE 5.13 – Graphe implicatif des relations affectives des élèves.

Règle$\{ACT\} \Longleftrightarrow \{PRT\}$: 1.0 avec $Supp(\{ACT\}) = 0.52$ et $Supp(\{PRT\}) = 0.52$ Cette règle peut se traduire comme suit : une même proportion des élèves actifs est équivant à une même propotion des élèves participants

Règle$\{ACT\} \Rightarrow \{AINF\}$: 0.9 avec $Supp(\{ACT\}) = 0.52$ et $Supp(\{AINF\}) = 0.73$, Règle$\{PRT\} \Rightarrow \{AINF\}$: 0.9 avec $Supp(\{PRT\}) = 0.52$ et $Supp(\{AINF\}) = 0.73$ Les élèves travaillent activement et participent tout au long de l'activité parceque'ils aiment beaucoup l'informatique.

D'abord, presque tous les élèves qui sont restés actifs jusqu'à la dernière heure de l'activité ont répondu adorer l'informatique lors de l'enquête. Et puis, si les élèves ont beaucoup participé lors

de l'activité, alors ils sont presque affirmés qu'ils ont adoré le cours informatique.

Cette relation justifie la question, sur laquelle nous avons demandé ce qui motive les élèves. Ici nous avons une règle significative justifiant qu'une pédagogie active, non seulement qui augmente la concertation et l'autonomie des élèves, mais favorise aussi l'affection et l'intention des élèves à apprendre l'informatique.

Règle$\{\text{AINF}\} \Rightarrow \{\text{ITR}\}$: 0.7 avec $Supp(\{\text{AINF}\}) = 0.73$ et $Supp(\{\text{ITR}\}) = 0.80$ D'abord, à la première impression, nous interprétons cette règle comme suit : les élèves qui adorent l'informatique sont généralement en bonne interaction avec l'ordinateur, une dépendance significative du désir, une attirance avec interaction aux technologies informatiques. Puisque la mesure M_{GK} est une mesure implicative, nous avons la relation d'équivalence des deux règles contreposées, selon la propriété énoncée dans la thèse HDR de A. Totohasina (Totohasina, 2003, 2008) que nous avons évoquée dans le chapitre 1 (proposition 8). $M_{GK}(X \longrightarrow Y) = M_{GK}(\overline{Y} \longrightarrow \overline{X})$, ainsi, si la règle $X \longrightarrow Y$ est valide selon M_{GK} au risque d'erreur α alors $\overline{Y} \longrightarrow \overline{X}$ l'est aussi.

Portant, utilisons sa contreposée pour l'interpréter : $R : Non\{\text{ITR}\} \longrightarrow Non\{\text{AINF}\}$. Les élèves qui sont en mauvaise interaction avec l'ordinateur ont presque tous répondu leur désintérêt au cours d'informatique.
Règle$\{\text{ITR}\} \Rightarrow \{\text{BUR}\}$: 0.89 avec $Supp(\{\text{ITR}\}) = 0.80$ et $Supp(\{\text{BUR}\}) = 0.86$. L'interaction sur l'ordinateur favorise la maîtrise des notions de base en bureautique. Comme précédemment, utilisons aussi la forme contreposée, $R : Non\{\text{BUR}\} \longrightarrow Non\{\text{ITR}\}$. Les élèves qui n'ont jamais fait ou assisté au cours de bureautique en classe antérieure ont souvent des difficultés sur l'interaction avec l'ordinateur.

Genre et interêt.

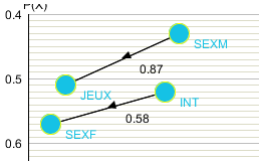


FIGURE 5.14 – Graphe implicatif de genre et intérêt des élèves.

Règle$\{\text{SEXM}\} \Rightarrow \{\text{JEUX}\}$: 0.87. avec $Supp(\{\text{SEXM}\}) = 0.43$ et $Supp(\{\text{JEUX}\}) = 0.51$ Règle$\{\text{INT}\} \Rightarrow \{\text{SEXF}\}$: 0.58 avec $Supp(\{\text{INT}\}) = 0.52$ et $Supp(\{\text{SEXF}\}) = 0.57$ Ici, nous constatons que la plupart des garçons sont généralement intéressés par les jeux vidéos. En revanche, presque toutes les filles de la classe ont tendance à s'habituer à Internet. Nous avons obtenue une forte dépendance de choix de technologie ou service en informatique avec le genre.

De la même manière, au sein de l'entreprise, le cliché est encore coriace : les filles sont au marketing, les garçons au développement. Il n'y a pas beaucoup de femmes dans les studios de jeux vidéos parce qu'il n'y a pas beaucoup de femmes ingénieures, développeuses ou spécialisées dans le game design. Ce dernier point est la clé du problème : si les entreprises veulent la parité au

sein de leurs équipes, elles doivent soutenir les formations où les jeunes femmes sont encore trop minoritaires. À commencer par les écoles d'ingénieurs.

Interêt.

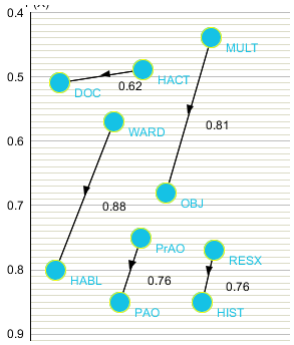


FIGURE 5.15 – Graphe implicatif des intérêts des élèves.

Règle$\{\text{MULT}\} \Rightarrow \{\text{OBJ}\}$: 0.81 avec $Supp(\{\text{MULT}\})$ = 0.44 et $Supp(\{\text{OBJ}\})$ = 0.68 Règle$\{\text{HACT}\} \Rightarrow \{\text{DOC}\}$: 0.62 avec $Supp(\{\text{HACT}\})$ = 0.49 et $Supp(\{\text{DOC}\})$ = 0.51 . Les élèves qui sont trop actifs amènent d'autres documents de support durant l'activité.
Règle$\{\text{WARD}\} \Rightarrow \{\text{HABL}\}$: 0.88 avec $Supp(\{\text{WARD}\})$ = 0.58 et $Supp(\{\text{HABL}\})$ = 0.80 . Les élèves qui sont capable aux objets **Word'art** sont également réceptifs aux habillages des objets insérés.
Règle$\{\text{P}rAO\} \Rightarrow \{\text{PAO}\}$: 0.76 avec $Supp(\{\text{P}rAO\})$ = 0.75 et $Supp(\{\text{PAO}\})$ = 0.85 . Les élèves maîtrisent le moins l'outil de présentation que l'outil de publication assistée par ordinateur.

Règle$\{\text{RESX}\} \Rightarrow \{\text{HIST}\}$: 0.76 avec $Supp(\{\text{RESX}\})$ = 0.77 et $Supp(\{\text{HIST}\})$ = 0.85 . Cette règle révèle une dépendance significative au niveau de connaissance sur les réseaux informatiques et des jeux de rôle avec Scratch.

Attente.

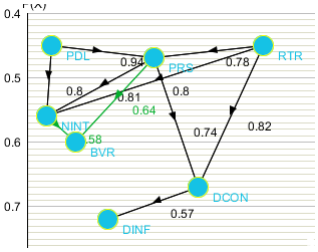


FIGURE 5.16 – Graphique implicatif des attente des élèves.

Règle$\{\text{DCON}\} \Rightarrow \{\text{DINF}\}$: 0.57 avec $Supp(\{\text{DCON}\})$ = 0.67 et $Supp(\{\text{DINF}\})$ = 0.72 *Déconcentrer implique déteste informatique.* visiblement avec un pourcentage de 70%, les élèves détestent l'enseignement de l'informatique privé d'ordinateur (débranché) parce qu'ils sont déconcentrés, ce n'est pas ce qu'ils ont voulu apprendre lors de la prise de contact. Les élèves se sentent trompés et désespérés par le changement en mode débranché.
Règle$\{\text{RTR}\} \Rightarrow \{\text{PRS}\}$: 0.78 avec $Supp(\{\text{RTR}\})$ = 0.45 et $Supp(\{\text{PRS}\})$ = 0.46 *Etre en retard implique être paresseux,*
Règle$\{\text{RTR}\} \Rightarrow \{\text{NINT}\}$: 0.80 avec $Supp(\{\text{RTR}\})$ = 0.45 et $Supp(\{\text{NINT}\})$ = 0.56 *Etre en retard implique*

ne pas intéresser au cours.

Règle{RTR} ⇒ {DCON} : 0.82 avec Supp({RTR}) = 0.45 et Supp({DCON}) = 0.67 *Etre en retard implique, déconcentré.*

Nous avons recensé 45% étudiants qui se mettent volontairement en retard pour ne pas s'impliquer activement et avec attention aux activités soi-disant informatiques, mais sans ordinateur. Pour le cas de Madagascar, par rapport aux effectifs et à la disposition en classe, faire de l'informatique débranchée demande beaucoup de préparation morale et nécessite aussi beaucoup d'efforts pour développer leur motivation.

Règle{PRS} ⇒ {NINT} : 0.81 avec Supp({PRS}) = 0.45 et Supp({NINT}) = 0.46 Paresseux implique non intéressé,

Règle{PRS} ⇒ {BVR} : 0.64 avec Supp({PRS}) = 0.56 et Supp({BVR}) = 0.60 *Paresseux implique bavard en classe,*

Règle{PRS} ⇒ {DCON} : 0.80 avec Supp({PRS}) = 0.56 et Supp({DCON}) = 0.67 *Paresseux implique déconcentré,*

Règle{PDL} ⇒ {PRS} : 0.94 avec Supp({PDL}) = 0.45 et Supp({PRS}) = 0.46 *Pas de livre implique paresseux,*

Règle{PDL} ⇒ {NINT} : 0.80 avec Supp({PDL}) = 0.45 et Supp({NINT}) = 0.56 *Pas amené leur livre implique non intéressé.*

Leur paresse et leur haine à ce mode d'enseignement caractérisé par le non-achat, ou la non-présentation du livre d'informatique. Cette attitude n'est pas remarquée pour les élèves en classe de sixième aussi bien pour ceux en classe de cinquième.

Règle{NINT} ⇒ {BVR} : 0.58 avec Supp({NINT}) = 0.46 et Supp({BVR}) = 0.60 *Ne pas être intéressé au cours implique être bavard en classe.*

Les désordres, les bavardages en classes et aussi les absentéismes signalés par les administrations pour les classes pour lesquelles nous avons pratiqué ce mode d'enseignement ne sont que des effets de la haine et du désamour des apprenants pour l'enseignement débranché.

Structure : Relation et difficulté.

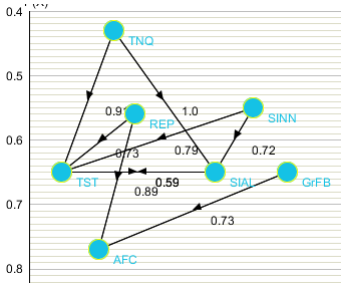


FIGURE 5.17 – Graphe implicatif des difficultés.

Règle{REP} ⇒ {AFC} : 0.89 avec Supp({REP}) = 0.56 et Supp({AFC}) = 0.77 *Si un apprenant a maitrisé la notion de structure répétitive, il maîtrisera presque surement la notion d'affectation.*

Selon les supports des motifs lus sur l'axe vertical, désignant les proportions des élèves ayant trouvé que chacune de ces structures présentées comme des sommets dans ce graphe. L'instruction d'affectation est la plus appropriée, et puis par ordre croissant de difficulté la condition boo-

léenne et la structure de branchement simple, structure répétitive (boucle traduit `REPETER n FOIS` en Scratch), la structure de branchement complet, et le dernier c'est la structure répétitive (tant que traduit `REPETER JUSQU'A` en Scratch).

Pour être effectif, un programme utilise une structure d'itération (programme à boucles), comprenant, l'initialisation des variables et la mise à jour des variables.
Une variable mathématique se distingue d'une variable informatique. Une variable doit être déclarée avant d'être utilisée. Déclarer une variable consiste à l'énoncer et donc à lui réserver une place dans l'espace mémoire. Contrairement à la variable informatique, une variable mathématique est, par définition, « *un symbole représentant un élément non spécifié ou inconnu d'un ensemble* » (Samurçay, 1985). De plus, dans la manipulation de variables, certaines instructions changent de propriétés selon qu'elles sont utilisées avec une orientation mathématique ou informatique, ajoute Samurcay. Par exemple, en mathématique, la relation d'affectation d'une valeur a une variable est une égalité régie par des propriétés d'une relation symétrique alors qu'en informatique, cette relation est asymétrique. L'affectation consiste en l'attribution de la valeur à la variable déjà déclarée. Cette opération se fait à travers une instruction élémentaire appelée instruction d'affectation Dagdilelis et al. (1990). Cette instruction comprend une variable compteur et une variable d'accumulation. Les auteurs définissent une variable compteur comme un générateur d'entiers, obtenus par valeurs successives par rapport à la valeur initiale et peuvent être calculés par une instruction de récurrence de la forme : « *compteur : = compteur +1* ». La variable d'accumulation quant à elle permet l'actualisation des calculs par l'instruction de récurrence suivante : « *somme : = somme + compteur* » .

Règle$\{\text{REP}\} \Rightarrow \{\text{TST}\} : 0.73$ avec $Supp(\{\text{REP}\}) = 0.56$ et $Supp(\{\text{TST}\}) = 0.65$.
Si un apprenant a réussi à la notion de structure répétitive, il réussira ,peut-être, aussi la notion test conditionnelle ou les opérations booléennes.

La notion d'itération est, en informatique, une notion de base constituée pour permettre la construction des algorithmes qui nécessitent des répétitions de séquences d'instructions identiques au sein des boucles (Laborde, 1985). Ces auteurs distinguent trois parties qui interviennent dans la construction d'une itération : un avertisseur d'itération, une condition et un corps de l'itération. L'avertisseur de l'itération est une expression qui annonce une itération. La condition, qui comme son nom l'indique, conditionne une fois vérifiée, l'exécution de la partie de la boucle à répéter et, enfin, le corps de l'itération qui est délimitée et qui sera répétée une fois la condition vérifiée. Il se situe entre la condition et la fin de la boucle.
Implicitement chez les apprenants, en Scratch, la boucle `REPETER n FOIS` comprenant l'initialisation des variables, la mise à jour des variables et un compteur. C'est-à-dire, aucune expression booléenne conditionnant l'arrêt du boule n'est requière en Scratch, seule la connaissance du nombre de répétitions est nécessaire pour construire l'algorithme. Explicitement il comprenant l'initialisation des variables, la mise à jour des variables, une condition d'arrêt. En Algobox, comme on a trouvé en langage evolué, par exemple en java, la structure `REPETER n` pourra se traduire par `FOR (INT i=1, i<k, i++)`.

La structure répétée ne présente pas le même ordre de difficulté en Scratch et en Algorithme, cette variation favorise encore le choix de langage Scratch au collège pour initié l'apprentissage de l'algorithme et de la programmation.

Règle$\{\text{TST}\} \Longleftrightarrow \{\text{SIAL}\}$: 0.59 avec S$upp(\{\text{TST}\})$ = 0.65 et S$upp(\{\text{SIAL}\})$ = 0.65 . *Les apprenants réussiront la structure de branchement simple si et seulement sils réussissent au test conditionnel ou aux opérations booléen.*

L'instruction TEST recouvre une action complexe, elle comprend à la fois l'évaluation d'une condition et une décision de branchement dont l'effet est la sortie de la boucle. Le caractère *ad hoc* du robot apparait dans la définition de cette instruction qui dispense de la construction de la formulation logique de la condition.

L'utilisation de l'instruction IF pour arrêter la boucle a été dans un premier temps dissociée de l'instruction `GOTO` (celle-ci occupait une autre ligne dans leurs programmes) et liée à l'instruction `STOP` (`IF` Condition `THEN STOP`). La liaison de ces deux instructions n'a pu être effectuée dans un deuxième temps (`IF` condition `THEN GOTO`) que grâce à l'aide apportée par l'enseignant.
Règle$\{\text{SINN}\} \Rightarrow \{\text{SIAL}\}$: 1.0 avec S$upp(\{\text{SINN}\})$ = 0.55 et S$upp(\{\text{SIAL}\})$ = 0.65 , Règle$\{\text{SINN}\} \Rightarrow \{\text{TST}\}$: 0.79 avec S$upp(\{\text{SINN}\})$ = 0.55 et S$upp(\{\text{TST}\})$ = 0.65 . *Les apprenants qui se sont approprié la structure branchement avec alternative se sont aussi approprié le test conditionnel et la structure de branchement simple.*

Selon ces trois règles, il existe une équivalence significative entre le test conditionnel et la structure conditionnelle simple `SI..ALORS`. Tous les groupes qui ont utilisé avec succès le test de contrôle ont également réussi la structure `SI..ALORS`. La structure répétitive `TANT QUE` semble un peu difficile pour les étudiants, elle est étroitement liée à la structure conditionnelle et aux tests conditionnels.

Règle$\{\text{TNQ}\} \Rightarrow \{\text{SIAL}\}$: 1.0. avec S$upp(\{\text{TNQ}\})$ = 0.43 et S$upp(\{\text{SIAL}\})$ = 0.65 et Règle$\{\text{TNQ}\} \Rightarrow \{\text{TST}\}$: 0.91 avec S$upp(\{\text{TNQ}\})$ = 0.43 et S$upp(\{\text{TST}\})$ = 0.65 .

Dans la gestion des variables, deux informations importantes doivent être coordonnées : une information sur les mises à jour des variables et une information sur la valeur de la variable qui correspond à la valeur de réalisation de la condition d'arrêt (Samurçay, 1985) : « *un accord implicitement admis sur la valeur de la variable consiste à arriver à rendre au moins une fois l'expression booléenne VRAIE au sein de la condition* ».

Parmi les structures itératives, la boucle, `REPETER JUSQU'A` est vue comme celle qui est plus abordable par les élèves novices en informatique. Elle serait plus proche de leurs représentations initiales que les autres boucles (Mejias-Dayoub, 1985).

Il existe deux différences fondamentales entre les boucles `REPETER` et `TANT QUE`. Elles sont à la fois structurelles qu'algorithmiques (Laborde, 1985). La différence de structure est liée à la délimitation du corps de l'itération. Dans la boucle `REPETER`, le corps de l'itération est délimite par les mots *Répéter* et *Jusqu'à ce que*, alors que dans `TANT QUE` le corps de l'itération est délimite par les mots *Faire* et *Fin tant que*.

La deuxième différence liée à leur signification algorithmique est due au fait que le corps de l'itération est au moins exécute une fois dans la boucle `Répéter jusqu'à` être ce qu'il ne puisse plus être exécuté même une fois dans `TANT QUE`. Il suffit que la condition qui autorise cette exécution ne soit pas vérifiée. Si l'effet de la boucle `REPETER` peut-être obtenu à partir de la boucle Tant que par changement de place de la condition, `REPETER` est considérée comme la boucle la plus générale par rapport a `TANT QUE`, soulignent (Laborde, 1985). De plus, le mot jusqu'à ce qu'annonce la condition qui, une fois réalisée arrête la répétition du corps de l'itération alors que la boucle `REPETER` est souvent présente sous forme implicite ou accompagnée de tournures linguistiques (Duchâteau, 2002).

Une structure de contrôle est complexe. Elle comprend beaucoup d'éléments : avertisseur, délimiteur, condition, variable, corps d'itération, compteur, etc. Ces derniers entretiennent entre eux des relations fortes pour son fonctionnement. Si un avertisseur d'itération annonce le début d'une boucle, le corps d'itération qui est un ensemble d'instructions est exécuté avec la véracité de la condition : le résultat du test de la condition donne une orientation à prendre. (Laborde, 1985), soulignent l'importance de l'instruction test dans une boucle en précisant que c'est grâce à elle qu'elle existe : l'instruction test joue deux actions importantes. D'une part, elle évalue la condition et d'autre part, décide le branchement à faire : sortir ou pas de la boucle, arrêter ou continuer l'exécution (Laborde, 1985).

5.8 Discussions.

Comment aider des élèves déconcentrés à l'initiation de l'informatique débranchée à l'école ?
Aider et traiter dans l'angle psychologique, mais comment ?

L'informatique occupe de plus en plus de place dans les programmes du primaire à la terminale de l'enseignement initial. Cependant, elle est plutôt insérée dans d'autres enseignements comme les mathématiques, la technologie et parfois elle est enseignée par des professeurs d'autres disciplines. Le manque de formation de ces enseignants à cette discipline produit des contraintes à l'introduction massive dans le système scolaire.

Comment aider les élèves à participer au cours informatique ?

En fait, plusieurs approches permettront de faciliter une acculturation à la science informatique, mais cela mérite d'être étudiée, entre autre, par l'intermédiaire de la « *programmation créative* » qui favorise spécifiquement d'inventer des jeux, de raconter des histoires et de faire travailler aux enfants plusieurs compétences Romero & Valllerand (2016). D'ailleurs, Quinson (2013) a défini l'informatique débranchée comme des sciences manuelles du numérique. En effet, c'est un autre accès vers la pensée informatique et elle sera utilisable à tout âge et dans des contextes très divers.

Nous connaissons actuellement un ou des obstacles à la familiarisation les élèves à l'informatique à l'occasion des activités sans ordinateur. Les connaissances de quatre piliers de la science informatique, à savoir : les algorithmes, les données/informations, les langages de programmation et les machines sont indispensables. Donc, pour implanter les algorithmes chez les élèves, la disposition de langages de programmation est importante et doit être précise et sans ambigüité.

Quand il s'agit des activités débranchées, elles sont souvent imaginées, créées et mise à dis-

position des supports pédagogiques pour réaliser des activités en classe (Groupe IREM Grenoble, 2017), IREM Clermont-Ferrand). Nous avons constaté qu'à Madagascar, rechercher les supports utilisés. Alayrangues et al. (2017) ont noté que la construction de dessins, à la main, etc. constitue un jeu d'instructions limités (tourne, avance, dessine) qui sont parfois représentées ces recherches par une autre approche possible qui passe par l'analyse d'un jeu. En outre (algorythmique) a avancé une certaine approche basée sur la physique, les formés ont besoin de se déplacer en leur permettant d'illustrer certains algorithmes de tri. IREM Clermont-Ferrand a aussi trouvé une autre méthode de jeu intéressant comme le codage des images. Nous pouvons fixer les images binaires *via* un tableau de 0 et 1 ainsi que de se poser la question de la compression de telles images. Ensuite, nous pouvons étendre l'activité à la représentation des images en couleur et aussi faire des activités autour de la stéganographie qui peuvent consister à dissimuler une information dans une autre information (Alayrangues et al., 2017). Une autre démarche aussi est possible, mais souvent inattendue, faire des activités débranchées autour du fonctionnement des machines, ainsi que nous pouvons organiser un jeu de rôle qui encouragera les participants à se familiariser avec les différents composants d'un ordinateur. Prenons l'exemple de l'activité Gobot ou Gobelets (IREM Clermont-Ferrand), qui a mis à disposition un langage simplifié dont les instructions étaient exprimées par des cartes représentant des flèches. Il est aussi possible d'aborder la notion d'automate qui suggère une activité « *jeu du labyrinthe* » (IREM Clermont-Ferrand). Tous ces exemples peuvent illustrer la notion de langages qui correspondent aux activités algorithmiques, dans les activités débranchées.

Comment les influencer ?

Avant tout, le jeu a et occupe une place majeure lors de développement physique, social, psychique de l'enfant. Becquet (2010). Cet auteur souligne que « *le développement cognitif* » se réalise par une adaptation active de l'enfant à son environnement qui repose sur la mise en œuvre de trois mécanismes : l'assimilation, l'accommodation et l'équilibration. L'enfant va, par assimilation des schèmes de comportement rudimentaires, appréhender des objets de plus en plus variés, modifier ces schèmes pour les accommoder aux propriétés de ces objets, et former ainsi des schèmes plus complexes, plus nombreux et plus mobiles au fil du temps.

En Occident, dès sa conception, l'adulte communique avec le bébé. Dès la naissance, les parents montrent, initient en jouant avec l'enfant à différents jeux, ensuite au fur et à mesure, l'enfant participe puis initie lui-même toutes sortes de scénarios de jeux. En grandissant, l'enfant prend et pratique des jeux d'exploration et d'exercice. C'est à partir des jeux fabriqués et inventés, jeux symboliques, jeux à règles (comme jeux de société, jeux collectifs de plein air, etc.) que l'enfant entre dans d'autres formes d'interactions sociales. A ce stade ceci consiste à accepter des règles communes et commencer à maîtriser son propre désir et de vivre en relation avec les autres.

Par conséquent, son développement est un processus à particulièrement respecter selon son âge. L'enfant nécessite de pratiquer habituellement ces jeux pour son bon développement physiologique, neurologique, psycho-socioaffectif et cognitif. Avec ce jeu, l'enfant s'appuie efficacement sur l'aspect pédagogique. L'enfant pose également les fondations sur lesquelles s'appuieront plus tard des apprentissages disciplinaires. Et Becquet (2010) a dit que, dans la société Tsimihety , comme dans l'ensemble de la société malgache, le terme de famille, loin de se limiter comme

en Occident à un modèle restrictif père mère-enfants, inclue selon une forme pyramidale élargie les grands-parents, les oncles, les tantes, les cousins, les cousines et valorise plus que tout la solidarité entre chacun de ses membres. D'où l'importance de cette structure à conseiller, guider et éventuellement prise en charge par un autre membre de sa famille en dehors de ses parents, comme tante, oncle, grands-parents, etc. ce qui distingue fondamentalement du modèle occidental où l'on a davantage recours à l'intervention de services sociaux et institutionnalisés (écoles, crèches, assistantes maternelles, centres aérés) et des organismes de protection de l'enfance, le rôle de chaque membre de la famille est clairement défini dans un maillage solide qui couvre et encadre chaque enfant et adulte en formation. Dans cette circonstance, quand il s'agit de jouer, l'enfant a l'habitude de pratiquer du jeu à l'école, ce qui facilite une réduction des écarts et l'égalité des chances (Eduscol. Septembre 2015). Le jeu permet à l'enfant à communiquer avec les autres et bâtir des liens forts d'amitié. D'où l'intérêt d'un climat social positif qui conditionne fortement l'engagement actif de l'enfant dans les apprentissages. Des liens entre les pairs et avec les adultes se renouent à partir du jeu coopératif, et par l'intermédiaire du vécu et des épreuves partagées que le jeu renforce le lien entre l'enfant et l'adulte.

En fait, dans l'aspect affectif, le plaisir doit être suscité, le jeu combine la richesse des expériences vécues à des émotions positives. Cet ancrage social favorise la mémorisation et son réinvestissement au service de nouvelles expériences de jeu ou d'apprentissage. Le jeu développe la culture de l'enfant par le biais d'expériences diversifiées, vécues dans le plaisir. Les catégories de jeux à savoir (jeux d'exploration, jeux symboliques, jeux de construction, jeux à règles) fondent un rôle déterminant dans le développement de l'enfant et constituent un réel socle pour ses capacités à apprendre. Ces catégories de jeux construisent l'enfant à être autonome, capable de choisir et prendre d'initiatives (Eduscol. Septembre 2015.) Pour que l'enfant bénéficie d'un développement physiologique, psychologique, psycho-socioaffectif et cognitif, à l'efficacité de son apprentissage. Ceci nécessite d'intégrer avec le monde réel.

La famille aussi est impliquée, par la mise en place des moments d'échange avec le jeu. Ceci peut prendre du temps et les changements du comportement des familles comptent sur le long terme. Avec les passés de parents d'élèves avec l'école, ils ont une perception négative de l'école et il faut réhabiliter ce regard négatif par des jeux intergénérationnels. Et aussi les formations denseignants en informatique et des équipes enseignantes sont aussi proposées autour du jeu libre et du jeu structuré.

En l'idée, comment les aider à intégrer la pensée informatique par jeu ? Alayrangues et al. (2017) Dans l'apprentissage de l'informatique débranchée, l'utilisation d'objets « *concrets* » et complètement « *déconnectés* » comme des bâtonnets, des allumettes, des cartes, des jetons, des perles ...) pourront s'affranchir de la machine et de la technicité de sa programmation. Déjà, l'élève peut mieux appréhender les grands principes de la science elle-même. Nous pourrons avoir recours, d'une manière ludique, à des problèmes complexes (comme la recherche d'informations, tri de données, cryptographie.) et leur résolution via la conception d'algorithmes. L'objectif est de créer le travail en groupe et amène les participants à imaginer leur propre solution ou à une construction commune. Cette manière ludique favorise la mise en œuvre d'une démarche scientifique avec les enfants et adolescents.

Démotivation globale

Letierce-Trotta (1999) a souligné que l'assiduité scolaire a été constatée comme signe de bienêtre et l'abandon scolaire, a été traité sous un aspect quantitatif : régulier ou occasionnel, et sous un aspect qualitatif comme sécher les cours, retards et absences, Choquet & Ledoux (1998). Selon ces derniers, les origines du manque d'assiduité sont particulièrement d'abord l'insatisfaction scolaire, ensuite, la situation familiale et enfin, le mode de vie, surtout de la perception négative de la vie familiale, qui influent sur l'absentéisme scolaire. Selon les analyses psychologiques, ce terme absentéisme est conçu comme un signe d'un malaise global de l'adolescent, qui manifesterait par exemple son mal de vivre à travers son mal d'école Verba (1993, 2006). Nous y rajoutons que certaines origines de l'absentéisme sont la fragilité psychologique propre à l'adolescence, les facteurs socioéconomiques, la désorganisation de la structure familiale, les facteurs ethnoculturels. Pommereau (2013) a évoqué aue l'absence scolaire est une fuite de la réalité, signe de souffrance de l'adolescent. C'est peut-être le cas à Madagascar, notamment dans notre zone d'études montre ou la majorité des familles n'ont pas beaucoup de moyens.

Devant ce manque de moyens ou manque d'encadrement social, ou sociétal, les jeunes pensent à un autre avenir dans les multimédias pour la majorité des garçons et des filles. Ils s'orientent plus vers l'échange en communication « *correspondance* », pour trouver leur avenir. Selon l'étude de Letierce-Trotta (1999), elle nous montre que la pensée courante chez les adolescents qualifie deux choses : soit l'école est le moyen pour accéder dans le futur pour avoir des professions intéressantes et bien rémunérées, et éviter le chômage, soit l'école ne sert à rien. Mais certains y vont pourtant ils se sentent obligés, pour ne pas trainer dans la rue. Ces réalités nous interpellent également dans notre cas ici à Antsiranana. L'école joue-t-elle son rôle ? Les comportements de ces élèves en informatique nous posent des questions. Pourquoi sont-ils démotivés ou déconcentrés lors de cours. En fait, nous reprenons l'étude de Letierce-Trotta (1999) que l'absentéisme est un phénomène aux aspects et aux causes multiples et est un symptôme d'une situation complexe. Ce lien avec le malêtre des adolescents a été souligné par plusieurs auteurs. Ils ont avancé des solutions. Ainsi, il faut se prémunir des comportements à risque sachant que le bienêtre des élèves joue un rôle principal pour éviter ces fléaux et aussi les enseignants devront avoir pleinement conscience dans leurs missions, en tant qu'éducateur et formateur.

Démotivation à la pensée numérique

Bruillard & Baron (2006) ont traité les problématiques liées aux usages des TIC chez les jeunes, et nous allons prendre uniquement ce qui concerne le désintéressement des élèves envers l'informatique. Ils constatent un changement du rapport des jeunes à l'informatique. Les jeunes considèrent l'informatique comme un simple outil de communication et de production. Cela limite et reste un frein pour découvrir les potentialités des machines offertes par leur programmation.

MotivationsNijimbere (2015) montre que beaucoup de jeunes apprennent des technologies informatiques en dehors du cadre scolaire, souvent comme à la maison et/ou avec des pairs. Ils y voient deux raisons motivantes : des initiatives connues pour être plus actives d'une part et d'autre part, des gens qui y sont impliqués lors de l'apprentissage sont accessibles et peuvent donner des informations complémentaires si besoin. Tort et al. (2013) rapportent que quand l'encadrement est assuré par des enseignants spécialistes ceci suscite l'augmentation du nombre d'élèves à partici-

per aux cours. Des initiatives ont été prises pour favoriser l'intégration à l'enseignement et l'apprentissage de l'informatique chez les jeunes que ce soit à des concours ou à des compétitions internationales, ou enfin à l'usage de logiciels (Dagienė & Futschek, 2008). Faire naitre chez les jeunes collégiens et lycéens la pensée informatique peut passer par l'organisation d'une compétition internationale ouverte aux jeunes, par conséquent, l'algorithmique et la programmation ont du succès. (Février *et al.*, 2008) a exprimé que ces concours avaient favorisé aussi des rencontres pour les plus talentueux et avaient facilité des échanges de diverses expériences scientifiques et culturelles. (Burton & Hiron, 2008) : « *D'olympiades* » ont surgi une capacité de programmer. Ensuite, elles ont permis de continuer de développer les notions algorithmiques comme « *des types de données, des fonctions, la logique booléenne et les opérateurs binaires, les tableaux et les structures* », et cela a influencé l'élève à les utiliser pour se familiariser avec les principes de la programmation, pour tout langage utilisé.

5.9 Conclusion partielle

Ainsi, nous avons présenté dans ce chapitre les résultats de nos recherches dans le domaine didactique. Nous avons répertorié les formations que nous avons données : formation en informatique dispensée aux professeurs d'informatique, formation en algorithmique et environnement Scratch et formation en informatique dispensée aux apprenants.

L'enseignement de l'informatique étant encore jeune à Madagascar, certains des résultats de nos études font apparaitre les mêmes problèmes que ceux qui ont été observés dans d'autres pays lors du processus d'introduction de l'enseignement de l'informatique. En effet, nous avons constaté l'absence de la discipline informatique dans le cursus officiel, l'absence ou le manque d'enseignants formés en informatique, la difficulté de reconversion des professeurs de mathématiques en professeurs d'informatique.

De plus, le progrès et l'avancement de la technologie provoquent des obstacles assez importants pour les apprenants. Ils sont attirés par l'aspect technique et pratique de ces nouvelles technologies, ils ignorent ou détestent donc les enseignements théoriques ou déconnectés. Ceci montre combien l'enseignement et l'apprentissage des algorithmes et de la programmation ont besoin de nouvelles techniques sur l'appropriation, l'évaluation des connaissances et sur la conduite en classe.

Pour y remédier, nous proposons des séquences d'enseignements plus ludiques basées sur le langage de programmation visuel avec Scratch. Nous avons testé aussi un nouveau mode d'évaluation des connaissances basé sur des courbes d'évaluation et un apprentissage accompagné dans le cadre d'une méthode de travail collaboratif. Tout cela contribue à améliorer le processus d'enseignement et d'apprentissage de l'informatique au collège.

Enfin, les résultats que nous avons trouvés par l'analyse empirique des observations comportementales des apprenants sont scientifiquement justifiés par l'analyse statistique implicative avec notre outil ASI-MGK dans la dernière section. Dans le dernier chapitre, nous présenterons d'autres résultats de recherche, analysés avec le même outil, mais dans des domaines différents.

Chapitre 6

Application de l'ASI-MGK dans d'autres domaines.

Sommaire

Dans le chapitre 5 nous avons utilisé l'ASI-MGK dans le domaine didactique sur la fouille de connaissances cachées sur l'enseignement/apprentissage de l'informatique en algorithmique. Dans ce dernier chapitre nous montrons l'intérêt et la puissance de cet outil nouvellement construit toujours sur le *data mining* dans divers domaines. Ce sont les applications directes de notre outil d'analyse à l'analyse de données réelles sur une grande variété de sujets en collaboration avec d'autres chercheurs.

6.1 Domaine d'enseignement universitaire.

6.1.1 Des stratégies de ripostes à base de plantes traditionnelles face à la pandémie de covid-19 chez des étudiants universitaires malgaches

Travail de recherche présenté au *11ème Colloque international sur l'Analyse Statistique Implicative 3-6 novembre 2021.*

Auteurs : Christian RAZANATSOAVINA, Bakys Bruno RALAHADY, Andrianasy Angelo DJISTERA, Jeanne RAVAOSOLO et Jean-Claude LABERCHE.

Résumé : L'objectif de cet article est de montrer de différentes ripostes pratiquées chez de jeunes universitaires à travers un traitement traditionnel de plantes médicinales et conventionnels (Covid Organics (CVO) et vitamine C) pour contrecarrer la pandémie de covid-19, pouvant atteindre tout âge, tout sexe et toutes conditions physiques ; surtout que ces étudiants s'exposent quotidiennement à la pandémie covid-19. Face à une absence d'un traitement officiel et efficace contre cette maladie, la population estudiantine troublée dans leurs études a recouru majoritairement à des recettes ancestrales à base de tisanes et de bains de vapeur même si de son côté le gouvernement malgache a proposé des traitements médicaux normalisés. Ils l'ont fait d'autant plus que les bâtiments et des logements universitaires sont surbookés et ne disposent pas suffisamment de toilettes, de bornes fontaines en un mot des infrastructures sanitaires et d'hygiènes permettant de mettre en uvre les gestes barrières indispensables.

Ce sont ces difficultés diverses et variables que nous avons voulu mettre en lumière. Sont ainsi rapportées dans cet article : 1) L'acquisition de données au moyen d'un questionnaire proposé à 200 étudiants de l'université d'Antsiranana, 2) L'analyse des résultats permettant a) de connaitre les ripostes traditionnelles à base de plantes médicinales et conventionnelles faites et b) les inquiétudes liées à leurs environnements sanitaires et d'hygiènes.

Pour disposer de données précises et normées, un questionnaire semi-directif a été utilisé. Les résultats de cette analyse réalisée selon la méthode ASI sentent bien les ressentis du terrain. D'abord, la majorité des étudiants ont opté plus la médecine traditionnelle à bases de plantes non conventionnelles que pour les traitements Covid Organics et vitamine C. Mais ces étudiants font plus de confiance aux plantes médicinales qu'aux traitements conventionnels comme Covid Organics et vitamines C. Ce questionnaire a montré aussi que les étudiants s'estiment en majorité en bonne santé et qu'ils n'ont pas confiance dans les vaccins, mais si avec prudence ils ont pris quand même plus de vitamine C que le Covid Organics (Razanatsoavina et al., 2021a).

6.1.2 Les étudiants de la filière universitaire agronomique de Mandritsara pensent-ils diversifier les cultures pour assurer le développement de leur région

Travail de recherche présentée aux *Journées scientifiques de l'École Supérieure Polytechnique d'Antananarivo « Ensemble face aux objectifs du développement durable » 10 et 11 mars 2021- IAC Vontovorona - Antananarivo.*

Auteurs : Christian RAZANATSOAVINA, Bakys Bruno RALAHADY, Andrianasy Angelo DJISTERA et Jean-Claude LABERCHE.

Résumé : Dans le district de Mandritsara, Région Sofia, quasiment toutes les productions agricoles ont de faibles rendements. Elles n'assurent même pas l'autoconsommation. Cela constitue un obstacle à toute tentative de promotion agro-industrielle. Cette étude cherche à comprendre le ressenti des futurs techniciens supérieurs agricoles vis-à-vis du riz, symbole indéfectible des traditions malgaches, et de la vanille, du girofle, du café, du maïs, de l'arachide, qui pourraient dynamiser le territoire. Une enquête est réalisée, auprès de 200 étudiants en agronomie, de plus de 18 ans, filles et garçons confondus. Le but ultime est de saisir l'évolution des attitudes et des comportements des étudiants.

L'étude comporte trois phases : la mise en uvre d'un questionnaire ; l'application d'une Analyse Statistique Implicative (ASI) et l'interprétation des résultats. Les étudiants sont questionnés sur leurs attitudes face au développement de la culture du riz en tant que culture de rente ou pensent-ils passer à des monocultures plus spécialisées. Les résultats précisent que la culture du riz est choisie comme le principal moteur du développement régional et que les étudiants sont prêts à diversifier la production agricole.

En conclusion, ayant vocation à créer des entreprises agricoles, ces étudiants respectent les symboles, mais ils considèrent aussi la diversité agricole comme un élément déterminant dans le développement du secteur agro-industriel. Une adaptation des programmes pédagogiques devrait contribuer au développement socioéconomique de cette région qui a de fortes potentialités agricoles (Razanatsoavina et al., 2021b).

6.1.3 Changement radical des méthodes d'enseignements universitaires vers numérique face à la pandémie de covid-19 a Madagascar

Travail de recherche présenté au *11ème Colloque international sur l'Analyse Statistique Implicative 3-6 novembre 2021.*

Auteurs : Bakys Bruno RALAHADY, Christian RAZANATSOAVINA, Andrianasy Angelo DJISTERA, Jean-Claude LABERCHE et André TOTOHASINA

Résumé : La pandémie de covid-19 génère une crise profonde du secteur de l'éducation, dans les Universités de Madagascar. Les modes de transmissions pédagogiques ont changé radicalement en numérique face à cette pandémie, sans prendre en compte la non-application complète du système de LMD dans le système éducatif dans ce pays. L'objectif de cet article est de montrer et d'étudier les réactions et adaptations brutales chez les étudiants face aux obligations de changements de méthode d'apprentissage lors de la pandémie de covid-19.

Sont ainsi rapportées dans cet article : 1) Les études d'acquisition de données au moyen d'un questionnaire proposé aux 200 étudiants, 2) L'analyse des résultats et leurs implications sur la

gestion des affaires universitaires.

Les résultats précisent une analyse des réalités du terrain. D'abord, la majorité des étudiants ont dépensé pour une connexion internet et ont été obligés de fréquenter des cybercafés, pour chercher des documents ; ils ont également changé leurs stratégies d'apprentissage, de regroupements entre eux, car la majorité des étudiants n'ont pas non plus les moyens pour assister à un cours ou une conférence en ligne. En conséquence, la majorité des étudiants ont constaté la baisse de leur niveau et estiment insuffisante leur formation. Malgré tout, la majorité des étudiants n'ont pas l'intention d'abandonner leurs études même si ils rencontrent des difficultés.

L'accélération de l'introduction de la nouvelle technologie (TIC) et un changement des modes de transmissions notamment lors de cette pandémie du covid-19 a amplifié des troubles scolaires et des lacunes chez les étudiants. Ils sont motivés, mais frustrés et moins rassurés ; en effet, dans cette nouvelle perspective, tous les acteurs de l'enseignement malgache doivent revoir comment prévoir un modèle d'apprentissage universitaire simple et adapté à une éventuelle perturbation imprévue, comme une troisième vague de coronavirus ou une autre catastrophe naturelle afin d'éviter au mieux ou limiter au pire une rupture pédagogique (Ralahady et al., 2021).

6.2 Domaine agronomique et socioéconomique.

6.2.1 Formation agronomique, écosystème et progrès socioéconomique

Travail de recherche publié dans la Revue des Sciences, de Technologies et de l'Environnement Volume 4 (RSTE4)
Auteurs : Christian RAZANATSOAVINA, Andrianasy Angelo DJISTERA, BEARINIAINA Harrivel, Bakys Bruno RALAHADY et Jean-Claude Laberche.

Résumé : Le travail décrit l'attitude vis-à-vis de l'environnement socioéconomique des étudiants en formation agronomique de l'Université à Mandritsara. L'agronomie y est enseignée de manière systémique en se basant sur le cadre conceptuel analytique proposé par l'« Intergouvernmental Science-Policy Platform on Biodiversity and Ecosystem Services » (IPBES). Dans ce cadre, les facteurs liés aux sociétés humaines sont pris en compte. Ces étudiants reçoivent une formation technique qui favorise leur projet de retour dans un territoire rural riche et naturel. Nous cherchons à identifier les contraintes qu'ils ressentent 1) au moyen d'un questionnaire proposé aux 200 étudiants (garçons et filles de plus de 18 ans) en cursus licence première et deuxième années d'agronomie, 2) par l'analyse des données en utilisant la méthode de l'analyse statistique implicative (ASI).

Les résultats d'analyse avec ASI-MGK confirmer que les jeunes qui viennent à Mandritsara pour y étudier comptent bien retourner dans leurs villages d'origine pour devenir chef d'exploitation agricole. Ils souhaitent abandonner une agriculture ancestrale de subsistance au profit d'une agriculture de production avec les objectifs de devenir exportateur. Ceci est particulièrement certain pour la majorité des garçons. Par contre, pour les filles, plusieurs d'entre elles ne souhaitent pas devenir agricultrices, mais envisagent plutôt de travailler dans le para agricole soit comme fonctionnaire soit dans des organismes de vulgarisation. Mais pour le faire, ils ou elles devront maitriser leur environnement socio familial qui parfois inhibe leurs projets.

6.2.2 Influence d'une formation universitaire en agronomie sur le ressenti des traditions par les étudiants en premier cycle universitaire : cas du district de Mandritsara à Madagascar

Travail de recherche présenté au *10ème Colloque international sur l'Analyse Statistique Implicative 2-5 octobre 2019*

Auteurs : Christian RAZANATSOAVINA, Bakys Bruno RALAHADY et Jean-Claude LABERCHE.

Résumé : Le contexte des recherches présentées dans cet article concerne des analyses qui mesurent des ressentis de futurs techniciens supérieurs à Madagascar. L'étude a été menée, dans un centre universitaire agronomique de la région Sofia, district de Mandritsara auprès uniquement 200 étudiants de L1 et L2 en agronomie, plus de 18 ans (garçons et filles). Le but est d'appréhender l'évolution du ressenti des traditions chez ces futurs techniciens supérieurs agricoles depuis leur entrée en formation supérieure ; en effet, ces étudiants reçoivent une formation devant leur permettre pour changer leurs comportements non seulement techniques, mais aussi sociologiques vis-à-vis du monde agricole, ces derniers étant le principal frein reconnu, au développement rural. Pour ce faire, les étudiants ont été interrogés sur les quatre grands axes majeurs de leur formation, à savoir : leurs attitudes personnelles sur le pouvoir et l'influence des Anciens ; l'utilisation d'une station expérimentale agricole comme support pratique à leur formation ; croyance en leur pouvoir décisionnel particulièrement sur leurs capacités à prendre en main le développement agricole et les modifications dans leur pouvoir décisionnel. Cette étude utilisait un questionnaire semi-directif, composé de questions ouvertes et de questions fermées. Les résultats ont été traités avec comme outil statistique l'Analyse Statistique Implicative. Les résultats de cette analyse montrent que la plupart des étudiants interrogés sont encore sous le pouvoir et l'influence des Anciens et manifestent encore peu de pouvoir décisionnel pour transformer et changer le monde agricole. Néanmoins, ils croient en eux pour produire le meilleur. Cette première étude qui avait pour cible uniquement les L1 et L2 se poursuivra en L3 pour connaitre l'évolution complète d'une cohorte d'étudiants en Bachelor. Cette étude sera renouvelée aussi dans les années à venir au fur et à mesure de la mise en place définitive des structures de cette formation (Razanatsoavina et al., 2019).

6.2.3 Ressenti des contraintes liées a la sécurisation financière d'agriculteurs de la Sofia (Madagascar) vis-à-vis de projets d'amélioration des ressources en eau d'irrigation.

Travail de recherche publié dans la Revue des Sciences, de Technologies et de l'Environnement Volume 4 (RSTE4)

Auteurs : BEARINIAINA Harrivel, RASOANANDRASANA Emilienne, Christian RAZANATSOAVINA, Bakys Bruno RALAHADY et Jean-Claude Laberche,

Résumé : L'objectif de cette étude est de connaitre le ressenti, quant aux contraintes liées à la sécurisation financière chez des agriculteurs de la région de Sofia à Madagascar dans laquelle des projets d'améliorations hydrauliques sont en cours d'études. Classiquement, pour assurer le développement économique et social de régions essentiellement tournées vers une agriculture d'autoconsommation et souffrant de problèmes récurrents de sécheresse, des projets aménagements

hydrauliques sont lancés. C'est effectivement le cas du District de Bealanana de la Région Sofia, à Madagascar.

Alors que de tels projets devraient réjouir voire enthousiasmer les agriculteurs locaux, il est apparu lors des travaux préparatoires sur le terrain, que plusieurs agriculteurs exprimassent une certaine méfiance ou réticence vis-à-vis de ces projets. Généralement, il ne s'agit pas de problèmes techniques, comme on aurait pu à première vue le penser, mais de problèmes lies et aux difficultés a investir. Ces projets laissent souvent les agriculteurs désorientes et peu enclins a innover et il est apparu rapidement que cette méfiance pouvait limiter voire inhiber l'expression des progrès attendus. Ce sont ces contraintes apparemment multiformes et multifactorielles que nous avons voulu connaitre.

Sont ainsi rapportées dans cet article 1) Les études d'acquisition de données concernant ces contraintes au moyen d'un questionnaire proposent a 75 agriculteurs dont 30 de Ambatoriaha, 25 de Bealanana et 20 de Ambatosia ; 2) L'analyse des données par la méthode de l'analyse statistique implicative (ASI) ; 3) Les résultats originaux de cette étude et les implications que ceux-ci peuvent engendrer dans la mise en œuvre des projets d'aménagements. Cette étude a utilisé un questionnaire semi-directif, composé de questions ouvertes et de questions fermées.

Les implications entre Items montrent que la plupart des Chefs d'exploitation attendent des organisations professionnelles d'abord un meilleur accès au crédit et un appui technique comme des aides en nature (semences et des engrais) pour développer l'agriculture irriguée. De plus sont apparues des contraintes liées a la sécurité foncière de leurs champs/, En définitive les agriculteurs perpétuent des techniques traditionnelles et ils ne participent pas activement ni aux gros investissements sur l'aménagement hydro agricole ni aux activités des Associations des Usagers de l'Eau. Les aménagements hydrauliques lourds ne les rendent pas plus dynamiques. Tous ces motifs perpétuent des récoltes médiocres et rendent l'accroissement des productions agricoles impossible (Beariniaina et al., 2021).

6.2.4 Aspiration des agriculteurs vis-à-vis de l'aménagement hydraulique dans les paysages Ambatoriaha, Bealanana et Ambatosia.

Travail de recherche présenté au *Colloque international Toamasina Madagascar, 27 avril 2020 au 30 avril 2020 « Vision scientifique multidimensionnelle, au service de la recherche et du développement. »*

Auteurs : BEARINIAINA Harrivel, RASOANANDRASANA Emilienne, RAZANATSOAVINA Christian, RALAHADY Bakys Bruno et LABERCHE Jean-Claude.

Résumé : Des situations menées et présentées dans cet article se focalisent des analyses qui traitent des aspirations des agricultures dans le district de Bealanana, région de la Sofia auprès de 75 agriculteurs, plus de 18 ans, hommes et femmes dont 30 Ambatoriaha, 25 Bealanana et 20 Ambatosia. Le but est d'analyser les aspirations des agriculteurs à Madagascar vis-à-vis de l'aménagement hydraulique. Pour ce faire, les agriculteurs ont été interrogés sur les cinq grands axes majeurs de leur métier, à savoir : leurs sources d'eau permanentes ou saisonnières, leurs attitudes personnelles vis-à-vis de l'Association des Usagers de l'Eau, leurs connaissances sur l'importance de l'eau, leurs moyens de production lient a l'irrigation et leurs connaissances et relations avec les partenaires techniques et financiers. Cette étude utilisait un questionnaire semi-directif, composé de ques-

tions ouvertes et de questions fermées.

Les résultats ont été traités avec comme outils statistiques l'Analyse Statistique Implicative. Les résultats de cette analyse montrent que la plupart des Chefs d'exploitation sont des hommes. Nous constatons qu' ils ne participent pas activement aux Associations des Usagers de l'Eau et rendent moins dynamiques ces associations. En quatrième trimètre, les petites sources de l'eau dans ces zones se tarissent vite. Et avec une pratique traditionnelle de labour, un manque d'accès d'eau pour irriguer les champs a conduit aux mauvaises récoltes. Mêmes si des formations ou des encadrements techniques, des appuis financiers et matériels avec de nouvelles infrastructures modernes, mènes par des institutions publiques ou prives, sont présent solidaire, résistent a modifier leurs anciennes méthodes en profitant cet aménagement hydraulique (Beariniaina et al., 2020).

6.3 Domaine socio écologique.

6.3.1 Impacts des activités anthropiques et désengagements sociaux aux infrastructures routières : cas PK 318- pont Anjingo (Sofia)-RN6

Article présenté dans les Journées « *Dialogue et savoir-faire : construction des connaissances autour des enjeux et défis écologiques à Madagascar et dans l'Océan Indien* » *Université d'Antsiranana fin novembre 2020.*

Auteurs : Clermont MANANTSOA , O A. RAVONINJATOVO, Émilienne RASOANANDRASANA, Christian RAZANATSOAVINA et Bakys Bruno RALAHADY.

Résumé : Cet article décrit l'état de lieux des infrastructures routières détériorées par le changement climatique, sur le pont Anjingo, PK 318, RN6 axe région Sofia-Diana. Les impacts socio-environnementaux sont parmi des facteurs principaux qui amplifient ces problèmes de détérioration. Dans ce cadre, l'acte de vandalisme de la population riveraine et les usagers ainsi que les comportements de dirigeants via la politique publique et technique sont aussi pris en compte. Ces infrastructures routières bien entretenues permettent à une bonne circulation des biens et des services dans ces régions riches en produits agricoles et en biodiversités. Nous cherchons à identifier les contraintes que les infrastructures subissent 1) au moyen d'une étude sur le terrain sur le PK190 -PK581, 2) par l'analyse des données en utilisant la méthode de l'analyse statistique explicative (ASI) et enfin 3) l'exploitation des résultats originaux et les implications que ceux-ci peuvent engendrer dans la gestion de ces patrimoines. Nous avons posé des questions sur l'état des infrastructures routières :

- Quelles sont des détériorations sur ce pont Anjingo ?
- Quels sont les environnements autour de ce pont ?
- Est-ce que le lieu est une zone sensible à l'inondation ?

Une observation directe fut utilisée. Les résultats, analysés par ASI, précisent : le pont est quasiment détruit. Autour de ce pont n'existe pas d'une végétation qui protège cet endroit et lors de cyclone, ces zones ont une formation d'ilot et méandre suite à l'ensablement du lit de la rivière, ce qui entraine le changement du lit.

Cette étude a permis de réorienter la politique publique en vue de concevoir un développement économique régional et national durable. En plus, elle donne des renseignements sur la

situation environnementale et le comportement de la population riveraine et nous permettra de contribuer à la protection de l'environnement et à la pérennisation des infrastructures (Manantsoa et al., 2020).

6.3.2 Synergie socio écologique : gage des infrastructures résilientes aux stress environnementaux

Article publié dans les *Journées scientifiques de l'École Supérieure Polytechnique d'Antananarivo « Ensemble face aux objectifs du développement durable » 10 et 11 mars 2021- IAC Vontovorona - Antananarivo*

Auteurs : Clermont MANANTSOA , O A. RAVONINJATOVO, Émilienne RASOANANDRASANA, Christian RAZANATSOAVINA et Bakys Bruno RALAHADY.

Résumé : Madagascar, seulement 10% des routes goudronnées étaient en bon état, 28% moyen et 64% en mauvais état. Cet article présente les contraintes de l'infrastructure et son interdépendance avec la biodiversité et analyse des goulots d'étranglement et leurs causes exactes. L'étude met en évidence l'analyse de performance environnementale en combinaison avec l'analyse de la performance liée au comportement et au processus administratif à travers sept grands projets routiers du Ministère des Travaux Publics.

La méthodologie adoptée consiste à entreprendre i) une étude sur le terrain sur RN4 et RN6, ii) une analyse des données en utilisant la méthode d'analyse statistique explicative par le biais de la Méthodologie d'Approche de Recherche Participative et la méthode d'évaluation rapide couplée avec l'approche communautaire, institutionnelle, intersectorielle, ainsi que l'analyse Succès-Forces-Potentialités- Opportunités. Les résultats issus de ce travail de recherche ont montré par le biais du résultat de la variance d'importance, du résultat d'analyse statistique et du tableau d'interrelation que la performance de dudit ministère est douteuse au niveau du suivi de processus et sa performance environnementale est faible, car le pourcentage inférieur à 50%. Ce qui prouve la vulnérabilité du réseau routier qui est l'artère de développement socioéconomique suite à la péjoration climatique.

En bref, ce travail de recherche permettra d'envisager une nouvelle politique de gestion des dépendances vertes routières relative aux Objectifs du développement durables 9,13 et 15 qui précisent la nécessité des infrastructures résilientes et l'urgence de la lutte contre les changements climatiques et leurs répercussions (Manantsoa et al., 2021).

6.4 Domaine informatique et mathématiques appliquées.

6.4.1 Time series homogenization : case of monthly temperature series of the northern part of Madagascar

Travail de recherche présenté au CARI 2018 - Colloque africain sur la recherche en informatique et mathématiques appliquées, Oct. 2018, Stellenbosch, South Africa. 2018.

Auteurs : Ralahady Bruno Bakys et Totohasina André.

Résumé : La technique statistique pour la détection des sauts dans les séries de températures basée sur le modèle de régression est favorable pour homogénéiser les données climatiques de la

partie nord de Madagascar. Ainsi, nous allons présenter les résultats de l'homogénéisation des séries des températures maximales et minimales correspondant à la station climatique d'Antsiranana. L'homogénéisation des séries de températures est réalisée aux échelles mensuelles et quotidiennes (Ralahady & Totohasina, 2018).

6.4.2 Un nouveau logiciel de traitement de données basé sur les graphes implicatifs

Poster présenté au *Colloque international Toamasina Madagascar, 27 avril 2020 au 30 avril 2020 « Vision scientifique multidimensionnelle, au service de la recherche et du développement. »*
Auteurs : TOVOHERY Josoa Michel, RALAHADY Bruno Bakys, TOTOHASINA André et FENO Daniel Rajaonasy.
Objectif : Présenter le nouvel outil d'analyse statistique, qui est un logiciel de graphe implicatif permettant d'extraire des règles d'association du type « Si X, alors Y » dans un grand volume de données avec une mesure de qualité très performante nommée M_{GK} (Tovohery et al., 2020).

6.5 Conclusion partielle.

Dans ce chapitre, nous avons montré la pertinence de la mesure d'intérêt des règles d'association M_{GK} et la puissance de l'outil d'analyse statistique ASI-MGK à travers des recherches appliquées dans différents domaines quant à l'utilisation de données réelles et originales. Plusieurs outils d'analyse nécessitent beaucoup de connaissances et de temps pour fonctionner, tandis qu'avec ASI-MGK, l'outil est autonome sur plusieurs plateformes sans avoir besoin d'une installation, facile à utiliser. Il a une interface simple, la saisie des données se fait en quelques clics et les résultats sont faciles à lire.

Quant à la recherche, tout d'abord, l'outil entérine des observations empiriques ; nous avons justifié scientifiquement nos attentes, plus précisément les hypothèses de nos recherches. Et puis notre outil explore des connaissances cachées dans les données : nous avons obtenu des résultats valides inattendus lors du traitement des études d'application pour lesquelles nous avions été sollicités d'intervenir. Ceci confirme que ces connaissances peuvent donner de nouvelles perspectives à notre recherche et peuvent également élargir notre champ d'études dans d'autres domaines. Nous avons constaté que certains résultats, comme dans l'analyse de l'impact du covid19 chez les étudiants, l'intégration de chercheurs en biochimie ou en pharmacologie pourrait conduire à des résultats originaux très instructifs. En agronomie également, notre analyse statistique originale a rendu plus pertinente l'analyse effectuée par nos collègues sociologues et économistes. Néanmoins le travail fondamental sur l'ASI-MGK n'est pas terminé, la poursuite et l'amélioration de ces études se concentreront sur l'augmentation des effectifs de la population d'étude et l'élargissement de la zone d'étude. Ainsi, travailler en analyse statistique implicative, notamment avec l'ASI-MGK, ouvre la voie à de jeunes chercheurs dans une recherche pluridisciplinaire. Il est indéniable que l'analyse et l'interprétation des résultats font appel à plusieurs chercheurs de divers domaines.

Encore un bel exemple de travail collaboratif à l'instar de ce que nous avons proposé pour l'enseignement de l'informatique au collège.

Conclusions et Perspectives

Devant les idées tantôt divergentes et tantôt convergentes concernant l'introduction des sciences informatiques au collège, certains affirment que le mode débranché est suffisant et les autres affirment que sans être connectée, l'informatique perd son sens. Établir une ou deux hypothèses ne nous a pas suffi. Il nous fallut explorer toutes les idées et fouiller toutes les idées significatives dans cette grande base de données. Utiliser une méthode statistique et sélectionnée ou concevoir un outil puissant s'avère nécessaire.

Dans ce double travail, à la fois disciplinaire et didactique, nous avons présenté dans la première partie de notre travail les théories sur la fouille de données tout en précisant les propriétés pertinentes de la mesure M_{GK}, une mesures certes, mais accompagnée d'algorithmes récents et robustes pour l'extraction des motifs fréquents et l'extraction des règles d'association valide selon ladite mesure.

Alors dans le deuxième chapitre, nous avons contribué à un développement d'un outil d'analyse statistique implicative, en intégrant certaines théories nécessaires et suffisantes pour l'extraction des règles d'association valides selon la mesure M_{GK}. L'outil pourra extraire des règles selon 40 mesures, mais filtrer à la base de la valeur critique définie à partir de Khi-deux de M_{GK}. L'outil est dénommé ASI – MGK, signifiant Analyse statistique Implicative basée sur la mesure M_{GK}. Il est multiplateforme, développé en langage Java, autonome, possédant plusieurs fenêtres de représentation des résultats. Sa fenêtre de visualisation graphique interactive supporte plusieurs séries d'actions d'amélioration de vue.

Dans l'aspect didactique, nous concluons notre travail sur la difficulté d'ordre psychologique et pédagogique quant à l'insertion de l'éducation algorithmique en mode débranché. En effet, introduire les concepts de base en informatique, sans ordinateur ou en mode débranché, démotive les apprenants qui sont déjà assoiffés de se familiariser à l'ordinateur. De plus, préparer une situation d'apprentissage et organiser des activités débranchées nécessite un temps de préparation assez long au préalable et demande une compétence très avancée en pédagogie et en didactique, c'est-à-dire des enseignants bien formés en cette matière. Par conséquent nous avons recommandé d'introduire le cours informatique depuis la classe de sixième, et introduire l'enseignement de l'algorithme en classe de quatrième en utilisant trois modes de représentations telles que : la forme graphique colorée, forme graphique non colorée et la forme textuelle (pseudocode), en motivant l'utilisation de langage visuel Scratch.

En outre, nous avons instauré chez tous les établissements qui ont déjà initié cet enseignement

un curriculum unique et conçue à partir des livres des programmes informatiques et conçu des manuels d'enseignement pour les classes de sixième, cinquième et quatrième. En supplément, des tutoriels numériques en format CD-ROM et en application Android ont été aussi élaborés pour aider les enseignants.

Nous avons expérimenté le temps ou le nombre de séances nécessaires et quelles activités favorables peuvent introduire chaque structure de bases en algorithmique. Ainsi, nous avons établi une taxonomie et l'ordre de priorité de ses structures. Nous avons fourni une nouvelle méthode de conduite de groupe inspirée par la méthode de Jigsaw ou des puzzles. Une méthode visant à donner des coups de pouce au groupe moins avancé par reconstitutions avec les membres de groupes favoris en trois phases. Ceci est tel qu'à la dernière phase les niveaux des groupes seront homogènes.

En fin, une méthode d'évaluation a été empruntée au domaine d'économie ou de la psychologie, une courbe d'évaluation ou courbe d'apprentissage est fonctionnelle. Une évaluation bidimensionnelle pourra évaluer à la fois la connaissance et la performance des apprenants, et identifier ou localiser ses difficultés.

Limites de notre recherche et de notre travail futur.

Sur le plan disciplinaire, lors de cette étude nous avons envisagé d'élaborer un outil autonome, puissant, intégrant toutes les théories d'Analyse de données, en vue de regrouper tous nos travaux de recherche effectué dans notre équipe. Nous avons pu seulement introduire dans cet outil ; une base des règles valides, la validation des règles par usage de la valeur critique de M_{GK}, le graphe implicatif suivi des techniques d'amélioration des vues.

Ainsi, les nouveaux algorithmes des motifs fréquents basés sur l'algorithme Apriori ne sont pas implémentés. Puis, les algorithmes d'extraction des règles valides à l'aide des théories des bases de règle énoncées par H. Ramanantsoa (2016) dans sa thèse ne sont pas encore supportés. Il en est de même pour la classification hiérarchique cohésitive constituant les résultats de recherche de F. Rakotomalala et al. (2019).

Cependant, des nouvelles idées d'amélioration ASI – MGK apparus lors de nos études comme l'implémentation de techniques de segmentation, transformation et de contrôle de données seront approfondies dans le but de les intégrer lors du prétraitement des données, et ce dans le but de traiter d'autres types de données non binaires.

Aussi, pour la visualisation des résultats, tout d'abord une technique d'aide à la décision sera envisagée de rajouter pour aider les utilisateurs à mieux scinder les règles valides concernant les modalités d'études. D'autre part, l'implémentation des algorithmes des graphes connexes et les parcours de graphe s'avèrent être très intéressants à rajouter, pour une autre amélioration de la visualisation par la sélection ou par la répartition des composantes connexes en sous-graphes.

Index

Bibliographie

Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. Dans P. Buneman & S. Jajodia (Eds.), *Proc. of the ACM SIGMOD International Conference on Management of Data*, volume 22 (p. 207–216). Washington,U.S.A. 16, 19, 21

Agrawal, R., Mehta, M., Shafer, J. C., Srikant, R., Arning, A., & Bollinger, T. (1996). The quest data mining system. Dans *KDD*, volume 96 (p. 244–249). 57

Agrawal, R. & Srikant, R. (1994a). Fast algorithms for mining association rules. Dans *Proc. of the 20th VLDB Conference* (p. 487–499). San Diego,Chile. 19, 31, 43

Agrawal, R. & Srikant, R. (1994b). Fast algorithms for mining association rules in large databases. *Proc. of the 20th Intl. Conf. on Very Large Data Bases (VLDB 94)*, 12, 478–499. 34

Ahlberg, C. & Shneiderman, B. (1994). Visual information seeking : tight coupling of dynamic query filters with starfield displays. *CHI'94 : Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press*, (p. 313–317). 59

Alayrangues, S., Peltier, S., & Signac, L. (2017). Informatique débranchée : Construire sa pensée informatique sans ordinateur. Dans *Colloque Mathématiques en Cycle 3 IREM de Poitiers* (p. 216–226). 171, 172

Anderson, J. (2010). *ICT transforming education : A regional guide*. UNESCO. 72

Arsac, J. (1980). *Premières leçons de programmation*. Cedic. 109

Barbut, M. & Monjardet, B. (1970). *Ordre et classification*. Paris : Hachette. 17

Baron, G.-L.and Bruillard, E. (2001). Une didactique de l'informatique ? *Revue Française de Pédagogie*, 135, 163–172. 71

Baron, G.-L. (1987). *La constitution de l'informatique comme discipline scolaire, le cas des lycées*. PhD thesis. 83

Baron, G.-L. & Bruillard, É. (2011). L'informatique et son enseignement dans l'enseignement scolaire général français : enjeux de pouvoir et de savoirs. *Recherches et expertises pour l'enseignement scientifique*, 1, 79–90. 86, 93

Basque, J. (2005). Une réflexion sur les fonctions attribuées aux tic en enseignement universitaire. *Revue internationale des technologies en pédagogie universitaire*, 2(1), 30–41. 8

Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., & Lakhal, L. (2000). Mining frequent patterns with counting inference. *SIGKDD Explorations*, 2, 66–75. 17

Baudé, J. (2010). L'option informatique des lycées dans les années 80 et 90. Récupéré le 19 Novembre 2015 du site de l'EPI : http ://www.epi.asso.fr/revue/histo/h10oi_jb3.htm. 74

Baudé, J. (2013a). La science informatique doit être enseignée dès le secondaire au même titre que la physique ou la biologie. Récupéré le 19 Novembre 2015 du site de l'EPI : http ://www.epi.asso.fr/revue/articles/a1304c.htm. 75

Baudé, J. (2013b). L'informatique dans lenseignement général : plus de 40 ans de présence active de l'epi. pour une complémentarité des approches. *Terminal. Technologie de l'information, culture & société*, (113-114), 53–67. 75

Baudé, J. (2014). L'expérience des "58 lycées". Dans *1024-Bulletin de la société informatique de France*, volume 4 (p. 105–115). [En ligne] http ://www.societe-informatique-de-france.fr/wp-content/uploads/2014/10/1024-4-baude.pdf. 74

Bayardo, R. J. (June 1998). Efficiently mining long patterns from databases. Dans *Proc. of the ACM SIGMOD Conference* (p. 85–93). Washington,U.S.A. 34

Bayardo, R. J., Agrawal, R., & Gunopulos, D. (1999). Constraint-based rule mining in large, dense databases. Dans *Proc. of the 15th ICDE Conference* (p. 188–197). 32

Beariniaina, H., Rasoanandrasana, E., Razanatsoavina, C., Ralahady., B. B., & Laberche, J. (2020). Aspiration des agriculteurs vis-a-vis de l'aménagement hydrauliques dans les paysages ambatoriaha, bealanana et ambatosia. *Colloque international, Vision scientifique multidimensionnelle, au service de la recherche et du développement, Toamasina - Madagascar, 27 à 30 avril 2020.* 181

Beariniaina, H., Rasoanandrasana, E., Razanatsoavina, C., Ralahady, B. B., & Laberche, J. (2021). Ressenti des contraintes liées a la sécurisation financière d'agriculteurs de la sofia, (madagascar) vis-a-vis de projets d'amélioration des ressources en eau d'irrigation. *Revue des Sciences, de Technologies et de l'Environnement (RSTE) volume 4.* 180

Becquet, D. (2010). *Devenir parent au 21e siècle, quelles perceptions et quels enjeux ? : étude comparative France-Madagascar.* PhD thesis, Amiens. 171

Bemarisika, P. (2016). *Extraction de règles d'association selon le couple support-MGK : Graphes implicatifs et Applications en didactique des mathématiques . (Mining association rules using the couple support-MGK : Implicatives graphs and Applications in mathematics education).* PhD thesis, University of Antananarivo, Madagascar. 7, 135

Bertin, J. (1967). *Sémiologie graphique.* 3e édition en 1999 aux Editions de l'Ecole des Hautes Etudes en Sciences Sociales. 60, 61

Birkhoff, G. (1967). *Lattice theory.* 3rd edition, Coll. Publ., XXV. Providence, RI : American Mathematical Society. 17

Bloch, D. (2016). Quelles sciences pour les machines ? petite balade entre l'être et la machine. 81

Boole, G. (1847). *The mathematical analysis of logic*. Philosophical Library. 80

Boole, G. (1854). *An investigation of the laws of thought : on which are founded the mathematical theories of logic and probabilities*. Dover Publications. 154

Brin, S., Motwani, R., & Silverstein, C. (1997a). Beyond market baskets : Generalizing association rules to correlation. Dans *Proc. of the ACM SIGMOD Conference* (p. 265–276). Tucson, Arizona. 49

Brin, S., Motwani, R., Ullman, J. D., & Tsur, S. (1997b). Dynamic itemset counting and implication rules for market basket data. Dans *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, SIGMOD'97* (p. 255–264). New York, NY, USA. 33

Bruillard, É. & Baron, G.-L. (2006). Usages en milieu scolaire : caractérisation, observation et évaluation. 173

Bruillard, É. & Vivet, M. (1994). Concevoir des eiao pour des situations scolaires : approche méthodologique. *Recherche en Didactique des Mathématiques*, 14(1/2), 273–302. 72

Brunk, C., Quelly, J., & Kohavi, R. (1997). Mineset : An integrated system for data mining. *AAAI Press*, (p. 135–138). 57

Burton, B. A. & Hiron, M. (2008). Creating informatics olympiad tasks : exploring the black art. *Olympiads in Informatics*, 2, 16–36. 174

Card, S. K., Mackinlay, J., & Shneiderman, B. (1999). Readings in information visualization : Using vision to think, morgan kaufmann. 58, 61

Casali, A., Cicchetti, R., & Lakhal, L. (2005). A perfect cover of frequent patterns. Dans L. N. in Computer Science (Ed.), *Proc. of the Data Warehousing and Knowledge Discovery (DaWaK) Conference*, volume 3589 (p. 428–437). Copenhagen, Danemark. 20

Chaachoua, H. (2018). T4tel un cadre de référence didactique pour la conception des eiah. 91

Cheung, A. C. & Slavin, R. E. (2013). The effectiveness of educational technology applications for enhancing mathematics achievement in k-12 classrooms : A meta-analysis. Dans *Educational Research Review*, volume 9 (p. 88–113). 72

Chevallard, Y. (1999). L'analyse des pratiques enseignantes en théorie anthropologique du didactique. *Recherches en didactique des mathématiques (Revue)*, 19(2), 221–265. 91

Choquet, M. & Ledoux, S. (1998). *Attentes et comportements des adolescents*. INSERM. 173

Cios, K.J., Pedrycz, W. and Swiniarski, R.W (1998). Data mining methods for knowledge discovery. Dans G. Mineau & B. Ganter (Eds.), *Proc. 9th Int. IEEE Transactions on Neural Networks*, volume 13 (p. 1533–1534). 8

Cockburn, A. & Williams, L. (2000). *The Costs and Benefits of Pair Programming*. 143

Couturier, R. & Gras, R. (2005). Chic : traitement de données avec l'analyse implicative. *Revue des Nouvelles Technologies de l'Information E-3.* Actes des journées Extraction et Gestion des Connaissances (EGC). 58

Culler, E. (1928). Nature of the learning curve. *Psychological Bulletin*, 25, 143–144. 122

Culler, E. & Girden, E. (1951). The learning curve in relation to other psychometric functions. *The American journal of psychology*, 64(3), 327–349. 122

Dagdilelis, V., Balacheff, N., & Capponi, B. (1990). L'apprentissage de l'itération dans deux environnements informatiques. *Aster.* 168

Dagienė, V. & Futschek, G. (2008). Bebras international contest on informatics and computer literacy : Criteria for good tasks. Dans *International conference on informatics in secondary schools-evolution and perspectives* (p. 19–30). : Springer. 174

Dagiene, V., Jevsikova, T., Schulte, C., Sentance, S., & Thota, N. (2013). A comparison of current trends within computer science teaching in school in germany and the uk. Dans *Informatics in Schools : Local Proceedings of the 6th International Conference ISSEP 2013 ; Selected Papers* (p.6̃3). 74

De Souza, C. S. (2005). *The semiotic engineering of human-computer interaction.* MIT press. 113

Dedekind, R. (1900). über zelengungen von zahlen durch ihre grösten gemeinsamen teiler. *Gesammelte Werke*, 2, 103–148. 12

Diethelm, I., Arndt, J., Dnnebier, M., & Syrbe, J. (2013). Informatics in schools : local. Dans *proceedings of the 6th International Conference ISSEP 2013 ; selected papers* Germany : Oldenburg. 71

Dijkstra, E. W. (1968). Go to statement considered harmful [letter to the editor]. *Communications of the ACM*, 11(3), 147–148. 154

Domenach, F. (2002). *Structures latticielles, correspondances de* G*alois contraintes et classification symbolique.* PhD thesis, Université Paris 1 Panthéon-Sorbone, France. 13

Drot-Delange, B. (2013). Enseigner l'informatique débranchée : analyse didactique d'activités. [En ligne] : http ://archivesic.ccsd.cnrs.fr/sic_00955208/document. 75

Duchâteau, C. (2002). Images pour programmer. première partie : Apprendre les concepts de base. édition revue et augmentée. 83, 170

Engelkampk, S., Glaab, & Renner, J. (2016). Constructivist norm research as political practice ż. *European Review of International Studies*, 3(3), 52–62. 127

Escovar, E. L., Biajiz, M., & Vieira, M. T. P. (2005). Ssdm : A semantically similar data mining algorithm. Dans *SBBD* (p. 265–279). : Citeseer. 35

ESPINASSE, B. (2008). *Introduction à la Fouille de données (Data Mining).* Université d'Aix-Marseille. 8

Estes, W. K. (1950). Toward a statistical theory of learning. *Psychological review*, 57(2), 94. 121

Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P., Eds. (1996a). *From data mining to knowledge discovery in databases*. Potland : AI Magazine. 7, 8

Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996b). Knowledge discovery and data mining : towards a unifying framework. Dans *Proceedings of the second International Conference on Knowledge Discovery and Data Mining* (p. 82–88). Portland, OR. 9

Fellows, M. & Parberry, I. (1993). Sigact trying to get children excited about cs. *Computing Research News*, 5(1), 7. 70

Feno, D. (2007). *Mesures de qualité des règles d'association : normalisation et caractérisation de bases*. PhD thesis, Université de La Réunion, France. 15, 135

Feno, D. R., Diatta, J., & Totohasina, A. (2006). Une base pour les règles d'association d'un contexte binaire valides au sens de la mesure de qualité m_{GK}. Dans *Actes du 13ème Rencontre de la Société Francophone de Classification* (p. 105–109). Metz, France. 49

Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. (1992). Knowledge discovery in databases - an overview. volume 13 (p. 57–70). Gte Labs Inc, Distributed Cooperating Learning Syst & Knowledge Discovery Databases Project, Waltham, Ma, 02254. 10

Frege, G. (1879). Begriffsschrift, a formula language, modeled upon that of arithmetic, for pure thought. *From Frege to Gödel : A source book in mathematical logic*, 1931, 1–82. 80

Friedman, J. (1997). What's the connection. Data mining and statistics. 8

Furnas, G. W. (1986). Generalized fisheye views. *CHI'86 : Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press*, (p. 16–23). 59

Gallistel, C. R., Fairhurst, S., & Balsam, P. (2004). The learning curve : implications of a quantitative analysis. *Proceedings of the National Academy of Sciences*, 101(36), 13124–13131. 122

Gańko-Karwowska, M. (2015). Pourquoi l'informatique et la technologie de l'information en tant que matière de formation en pologne ? *Récupéré le 19 Avril 2015 du site de l'EPI*. 76

Ganter, B. & Wille, R. (1999). *Formal concept analysis, Mathematical foundations*. Berlin : Springer Verlag. 14, 17

Gödel, K. (1931). Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für mathematik und physik*, 38(1), 173–198. 80

Goethals, B. & Zaki, M. J. (November 2003). Frequents itemset mining implementations. 9

Gras, R. (2014). Genese et developpement de l'analyse statistique implicative : retrospective historique. *Educ Matem Pesq São Paulo*, 16(3), 645–661. 37

Gras, R. (2015). Prefácio régis gras uso do chic na formaçao de educadores 2015. 37

Guillaume, S. (2000). *Traitement des données volumineuses. Mesures et algorithmes d'extraction des règles d'association et règles ordinales.* PhD thesis, Universté de Nantes, France. 44

Gulliksen, H. (1934). A rational equation of the learning curve based on thorndike's law of effect. *The Journal of General Psychology*, 11(2), 395–434. 122

Hamrouni, T., Yahia, S. B., & Slimani, Y. (2005). Prince : An algorithm for generating rule bases without closure computations. Dans *Proc. of the 7th DaWaK Conference* (p. 346–355). 17

Hamrouni T., D. I., Yahia., S. B., Nguifo, M., & Slimani, Y. (2007). Les itemsets essentiels fermés, une nouvelle représentation concise. (p. 241–252). 17

Hamrouni T., S.BenYahia, e. E. N. (2009). Sweeping the disjunctive search space towards mining new exact concise representations of frequent itemsets. *Data Knowledge Engineering*, 68, 1091–1111. 20

Han, J. (1998). Towards on-line analytical mining in large databases. *SIGMOD Record 27*, no. 1, 97–107. 57, 58

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. Dans W. Chen, J. Naughton, & P. A. Bernstein (Eds.), *2000 ACM SIGMOD Intl. Conference on Management of Data* (p. 1–12). : ACM Press. 32

Hao, M. C., Dayal, U., Hsu, M., Sprenger, T., & Gross, M. H. (2001). Visualization of directed associations in e-commerce transaction data. *Proceedings of VisSym*, (p. 185–192). 58

Hascot, M. & Beaudouin-Lafon, M. (2001). Visualisation interactive d'information. *Information - Interaction - Intelligence (I3)*, no. 1, 77–108. 59, 61

Heathcote, A., Brown, S., & Mewhort, D. (2000). The power law repealed : The case for an exponential law of practice. *Psychonomic bulletin & review*, 7(2), 185–207. 121

Horowitz, E. (1978). *Fundamentals of computer algorithms.* Galgotia publications. 84

Hubwieser, P., Armoni, M., Brinda, T., Dagiene, V., Diethelm, V. I., Giannakos, M. N., Knobelsdorf, M., Magenheim, J., Mittermeir, R., & Schubert, S. (2011). Computer science/informatics in secondary education. Dans *Proceedings of the 16th annual conference reports on Innovation and technology in computer science education-working group reports* (p. 19–38). New York : ACM. 71

Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., & Verkamo, A. I. (1994). Finding interesting rules from large sets of discovered association rules. Dans *Proceedings of the third international conference on Information and knowledge management* (p. 401–407). 53, 58

Knuth, D. (1968). *The Art of Computer Programming*, volume 1. Addison-Wesley, 3rd ed. 84

Kohavi, R., John, G., Long, R., Manley, D., & Pfleger, K. (1994). Mlc++ : A machine learning library in c++. Dans *Proceedings Sixth International Conference on Tools with Artificial Intelligence. TAI 94* (p. 740–743). : IEEE. 56

Kuntzmann, J. (1957). Formation de calculateurs–moyens de calcul–l'atelier arithmétique. 154

Laborde, J. (1985). Projet d'un cahier brouillon informatique de géométrie. *Rapport interne LSD (IMAG). Grenoble.* 168, 169, 170

Lamping, J., Rao, R., & Pirolli, P. (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. *CHI'95 : Proceedings of the SIGCHI conference on Human factors in computing systems*, (p. 401–408). 59

Lang, K., Galanos, R., Goode, J., Seehorn, D., Trees, F., Phillips, P., & Stephenson, C. (2013). Bugs in the system : Computer science teacher certification in the us. Dans *Association for Computing Machinery, Inc. (ACM) and the Computer Science Teachers Association (CSTA).* 74

Leibowitz, N., Baum, B., Enden, G., & Karniel, A. (2010). The exponential learning equation as a function of successful trials results in sigmoid performance. *Journal of Mathematical Psychology*, 54(3), 338–340. 122

Lenca, P., Meyer, P., Vaillant, B., & Lallich, S. (2008). On selecting interestingness measures for association rules : user oriented description and multiple criteria decision aid. Dans *European Journal of Operational Research 184* (p. 610–626). 44

Letierce-Trotta, P. (1999). Collégiens et absentéisme. *Étude des causes dans deux.* 173

Liu, B. & Hsu, W. (1996). Post-analysis of learned rules. *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, (p. 828–834). 53

Liu, B., Hsu, W., Wang, K., & Chen, S. (1999). Visually aided exploration of interesting association rules. *PAKDD'99 : Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, (p. 380–389). 53

Livingstone, I. & Hope, A. (2011). Next gen : transforming the uk into the world's leading talent hub for the video games and visual effects industries. 73

Manantsoa, C., A.Ravoninjatovo, O., Rasoanandrasana, E., Razanatsoavina, C., & Ralahady, B. B. (2020). Impacts des activités anthropiques et désengagements sociaux aux infrastructures routières : cas pk 318- pont anjingo (sofia)-rn6. *Dialogue et savoir-faire : construction des connaissances autour des enjeux et défis écologiques à Madagascar et dans l'Océan Indien. Université d'Antsiranana fin novembre 2020.* 182

Manantsoa, C., A.Ravoninjatovo, O., Rasoanandrasana, E., Razanatsoavina, C., & Ralahady, B. B. (2021). Synergie socio-écologique : gage des infrastructures résilientes aux stress environnementaux. *Journées Scientifiques de l'Ecole Supérieure Polytechnique d'Antananarivo « Ensemble face aux objectifs du développement durable » 10 et 11 mars 2021- IAC Vontovorona - Antananarivo.* 182

Mazur, J. E. & Hastie, R. (1978). Learning as accumulation : A reexamination of the learning curve. *Psychological Bulletin*, 85(6), 1256. 122

Mejias-Dayoub, B. (1985). *Difficultés conceptuelles dans l'écriture d'algorithmes itératifs chez des élèves de collège.* PhD thesis, Université Joseph-Fourier-Grenoble I. 169

Michel, A., Barrauu, L., Dubois, N., & Delzongle, S. (2016). Code et programmation - robots. 85

Modeste, S. (2012). *Enseigner l'algorithme pour quoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l'apprentissage de la preuve ?* PhD thesis, Université de Grenoble. 81, 82

Modeste, S., Gravier, S., & Ouvrier-Buffet, C. (2010). Algorithmique et apprentissage de la preuve. *Repères IREM*, 79, 51–72. 81

Monjardet, B. (2003). The presence of lattice theory in discrete problems of mathematical social sciences. why. *Mathematical Social Sciences*, 46, 103–144. 12

Moreau, J. J. (1987). Une formulation de la dynamique classique. 83, 154

Nakano, Y. & Izutsu, K. (2013). The new course of study and a prospect of information studies education in japan. Dans *Informatics in Schools : Local Proceedings of the 6th International Conference ISSEP 2013–Selected Papers* (p.̃89). 76

Newell, K. M., Liu, Y.-T., & Mayer-Kress, G. (2001). Time scales in motor learning and development. *Psychological review*, 108(1), 57. 122

Nguyen, C. T. & Bessot, A. (2003). La prise en compte des notions de boucle et de variable informatique dans l'enseignement des mathématiques au lycée. *Petit X*, 62. 155

Nijimbere, C. (2015). *L'enseignement de savoirs informatiques pour débutants, du second cycle de la scolarité secondaire scientifique à l'université en France : une étude comparative.* PhD thesis, Sorbonne Paris Cité. 173

Papert, S. (1981). Jaillissement de l'esprit : ordinateurs et apprentissage. 109

Park, J. S., Chen, M. S., & Yu, P. S. (May 1995). An effective hash based algorithm for mining association rules. *Proc. of the 1995 ACM SIGMOD Int. Conf. (SIGMOD'95)*, (p. 175–186). 34

Pasquier, N., Bastide, Y., Taouil, R., & Lakhal, L. (1999a). Closed set based discovery of small covers for association rules. Dans *Proc. 15emes Journees Bases de Donnees Avancées, BDA* (p. 361–381). Bordeaux,France. 32

Pasquier, N., Bastide, Y., Taouil, R., & Lakhal, L. (1999b). Closed set based discovery of small covers for association rules. *ADA'99*, (p. 361–381). 33

Pasquier, N., Taouil, R., Bastide, Y., Stumme, G., & Lakhal, L. (2005). Generating a condensed representation for association rules. *Journal of intelligent information systems*, 24(1), 29–60. 20

Peano, G. (1888). Intégration par séries des équations différentielles linéaires. *Mathematische Annalen*, 32(3), 450–456. 80

Pélissier, E. (2002). Pour une histoire de l'informatique dans l'enseignement français, premiers jalons. volume 50 : Revue de l'EPI. 74

Pernia, E. E. (2008). Strategy framework for promoting ict literacy in the asia-pacific region. Dans *Publication of UNESCO Bangkok Communication and Information Unit. Bangkok : Asia and Pacific Regional Bureau for Education* (p. 4–20). 72

Piatetsky-Shapiro, G. (1991a). Discovery, analysis, and representation of strong rules. *Knowlegde Discovery in Databases*, AAAI Press/The MIT Press, 229–248. 9

Piatetsky-Shapiro, G. (1991b). Knowledge discovery in real data bases. *AI Magazine*, 11, 68–70. 10

Pommereau, D. X. (2013). *L'adolescent suicidaire-3ème édition*. Dunod. 173

Preux, P. (2011). *Fouille de données (Notes de cours)*. Université de Lille 3. 8

Quinson, M. (2013). *Computational Science of Computer Systems*. PhD thesis, Université de Lorraine. 170

Rainsford, C. P. & Roddick, J. F. (2000). Visualisation of temporal interval association rules. Dans *Proceedings of the second international conference on intelligent data engineering and automated learning (IDEAL 2000)* (p. 91–96). : Springer-Verlag. 58

Rakotomalala, H. F., Ralahady, B. B., & Totohasina, A. (2019). A novel cohesitive implicative classification based on m_{GK} and application on diagnostic on informatics literacy of students of higher education in madagascar. Dans *Third International Congress on Information and Communication Technology* (p. 161–174). : Springer. 49, 185

Ralahady, B. B., Razanatsoavina, C., Djistera, A. A., Laberch, J. C., & Totohasina, A. (2021). Changement radical des méthodes d'enseignements universitaires vers numérique face a la pandémie de covid-19 a madagascar. *Analyse Statistique Implicative*, 11, 304. 178

Ralahady, B. B. & Totohasina, A. (2018). Time series homogenization. *Proceedings of CARI 2018*, (p.8̃4). 183

Ralahady, B. B. & Totohasina, A. (2019a). Asi-mgk : Implicative statistical analysis tool based on m_{GK}. *IJCST*, 3(1). 135

Ralahady, B. B. & Totohasina, A. (2019b). Experimental study of the valid rules according to the measure m_{GK}. *IJCST*, 3(1). 50, 135

Ralahady, B. B., Totohasina, A., & J., S. (2019). Apports intéressants de l'indice asi-mgk et des courbes d'apprentissages en didactique de l'informatique. *Analyse Statistique Implicative*, 10, 304. 135

Ramanantsoa, H. (2016). *Contributions à l'amélioration de génération des bases des règles d'association* $\mathrm{M_{GK}}$*-valides et applications en didactique des mathématiques*. PhD thesis. 50, 135, 185

Razanatsoavina, C., Ralahady, B. B., & C., L. J. (2021a). Des stratégies de ripostes plus a base de plantes traditionnelles face a la pandémie de covid-19 chez des étudiants universitaires malgache. *Analyse Statistique Implicative*, 11, 304. 176

Razanatsoavina, C., Ralahady, B. B., DJISTERA, A. A., & Laberche, J. C. (2021b). Les étudiants de la filière universitaire agronomique de mandritsara pensent-ils diversifier les cultures pour assurer le développement de leur région. *Journées Scientifiques de l'Ecole Supérieure Polytechnique d'Antananarivo « Ensemble face aux objectifs du développement durable » 10 et 11 mars 2021-IAC Vontovorona - Antananarivo*. 177

Razanatsoavina, C., Ralahady, B. B., & Laberch, J. C. (2019). Influence d'une formation universitaire en agronomie sur le ressenti des traditions par les étudiants en premier cycle universitaire : cas du district de mandritsara a madagascar. *Analyse Statistique Implicative*, 10, 304. 135, 179

Robitaille, M. (2007). Quand un dispositif de développement professionnel sur lapprentissage coopératif devient un lieu de régulation entre pairs. *Régulation des apprentissages en situation scolaire et en formation*, (p. 171–190). 141

Romero, M. & Vallerand, V. (2016). Co-creative activities for 21st century kids. *Guide d'activités technocréatives pour les enfants du 21e siècle*. 170

Ross, D., Goodenough, J., & Irvine, G. (1980). *Software Engineering : Process. Principles, and Goals*. Number Third Edition. New York. NY : Computer Society Press of the IEEE : Tutorial on Software Design Techniques. 39

Salleb, A. (2003). *Recherche de motifs fréquents pour l'extraction de règles d'associaition*. PhD thesis, Université d'Orleans, France. 15

Samurçay, R. (1985). Signification et fonctionnement du concept de variable informatique chez des élèves débutants. *Educational Studies in Mathematics*, 16(2), 143–161. 168, 169

Sarkar, M. & Brown, M. H. (1992). Graphical fisheye views of graphs. *in CHI'92 : Proceedings of the SIGCHI conference on human factors in computing systems, ACM Press*, (p. 83–91). 59

Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., Owens, B. B., Stephenson, C., & Verno, A. (2011a). *CSTA K–12 Computer Science Standards : Revised 2011*. Technical report, New York, NY, USA. 71, 72, 73, 74

Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., Owens, B. B., Stephenson, C., & Verno, A. (2011b). *CSTA K–12 Computer Science Standards : Revised 2011*. ACM. 74

Shakow, D. (1930). Hermann ebbinghaus. *The American Journal of Psychology*, 42(4), 505–518. 118

Shneiderman, B. (1996). The eyes have it : a task by data type taxonomy for information visualization. Dans *Proceedings of IEEE Symposium on Visual Languages VL'96, IEEE Computer Society* (p. 336–343). 59, 62

Stolee, K. T. & Fristoe, T. (2011). Expressing computer science concepts through kodu game lab. Dans *Proceedings of the 42nd ACM technical symposium on Computer science education* (p. 99–104). : ACM. 74

Sturman, L. & Sizmur, J. (2012). International comparison of computing in schools. 71

Syslo, M. & Kwiatkowska, A. (2008). The challenging face of informatics education in poland. Dans *Informatics Education-Supporting Computational Thinking* (p. 1–18). : Springer Berlin Heidelberg. 76

Sysło, M. M. (2011). Outreach to prospective informatics students. Dans *International Conference on Informatics in Schools : Situation, Evolution, and Perspectives* (p. 56–70). : Springer. 76

Sysło, M. M. & Kwiatkowska, A. B. (2013). Informatics for all high school students. Dans *International conference on informatics in schools : Situation, evolution, and perspectives* (p. 43–56). : Springer. 76

Talbi, E.-G. (2000). *Fouille de données (Data Mining) -Un tour d'horizon-*. Laboratoire d'Informatique Fondammentale de Lille. 8

Texier, F. (2002). *L'utilisation pédagogique de l'informatique l'école : entre volontarisme de praticiens et rigueur des sciences de l'éducation. Ebauche d'une archéologie des idées pédagogiques partir des discours*. PhD thesis, Université de Nantes. 75

Thurstone, L. L. (1919). The learning curve equation. *Psychological Monographs*, 26(3), i. 121

Toivonen, H. (September, 1996). Sampling large databases for association rules. *Proceedings of the 22nd International Conference Very large Data Bases (VLDB'96)*, (p. 134–145). 33, 34

Tort, F. (2011). Le concours castor : un outil de promotion de l'enseignement d'informatique. Dans *Sciences et technologies de l'information et de la communication en milieu éducatif : Analyse de pratiques et enjeux didactiques*. (p. 221–230). : Athènes : New Technologies Editions. 75

Tort, F. & Dagiene, V. (2012). Concours castor : découvrir l'informatique autrement. *E-Dossier de l'audiovisuel : l'éducation aux cultures de l'information*. 75

Tort, F., Kummer_hannoun, P., & Beauné, A. (2013). Engagement et motivations des enseignants du secondaire pour la passation d'un concours d'informatique. 173

Totohasina, A. (2003). Normalisation de mesures probabilistes de la qualité des règles. Dans *Proc. SFDS'03, XXXV ième Journées de Statistiques*, volume 2 (p. 985–988). Lyon 2, France. 135, 165

Totohasina, A. (2008). *Contribution à l'étude des mesures de qualité des règles d'association : normalisation sous cinq contraintes et cas de* M_{GK} *: propriété, base composite des règles d'association et extension en vue d'applications en statistique et en sciences physiques*. PhD thesis, Université d'Antsiranana, Madagascar. 21, 135, 165

Totohasina, A., Ralambondrainy, H., & Diatta, J. (2004). Notes sur les mesures probabilistes de la qualité des règles d'association : un algorithme efficace d'extraction des règles d'association implicative. Dans *Proc. of CARI'04* (p. 511–518). Hammamet, Tunisie. 23, 49

Totohasina, A., Ralambondrainy, H., & Diatta, J. (2005). Une vision unificatrice des mesures probabilistes de la qualité des règles d'association booléennes et un algorithme efficace d'extraction des règles d'association implicatives. *Proc. of TAIMA'04*, (p. 375–380). 23

Tovohery, J. M., Ralahady, B. B., Totohasina, A., & FENO, D. (2020). Un nouveau logiciel de traitement de données basé sur les graphes implicatifs. *Colloque international, Vision scientifique multidimensionnelle, au service de la recherche et du développement, Toamasina - Madagascar, 27 au 30 avril 2020*. 183

Trabelsi, M. (2010). Enseignement de l'informatique en tunisie. Récupéré le 19 Novembre 2015 du site de l'EPI : http ://www.epi.asso.fr/revue/articles/a1002f.htm. 77

Vaillant, B. (2006). *Mesurer la qualié des règles d'association : études formelles et expérimentales*. PhD thesis, École Nationale Supérieure des Télécommunications de Bretagne et Université de Bretagne sud, France. 18

Verba, D. (1993). *Le métier d'éducateur de jeunes enfants*. Syros Paris. 173

Verba, D. (2006). *Échec scolaire : travailler avec les familles*. Dunod. 173

Williamson, C. & Shneiderman, B. (1992). The dynamic homefinder : evaluating dynamic queries in a real-estate information exploration system. *SIGIR'92 : Proceedings of the fifteenth annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press*, (p. 338–346). 59

Wills, G. J. (1996). 288 ways to say "this is interesting". *INFOVIS'96 : Proceedings of the 1996 IEEE Symposium on Information Visualization, IEEE Computer Society*, (p. 54–61). 59, 61

Wilson, A., Hainey, T., & Connolly, T. M. (2013). Using scratch with primary school children : An evaluation of games constructed to gauge understanding of programming concepts. Dans *International Journal of Game-Based Learning (IJGBL)*, volume 3 (p. 93–109). 74

Wilson, C. & ACMCS (2010). *Running the Empty : Failure to Teach K-12 Computer Science in the Digital Age*. Association for Computing Machinery. 72, 74

Witten, I. H. & Frank, E. (2002). Data mining : practical machine learning tools and techniques with java implementations. *Acm Sigmod Record*, 31(1), 76–77. 42

Wong, P. C., Whitney, P., & Thomas, J. (1999). Visualizing association rules for text mining. Dans *Proceedings of the 1999 IEEE symposium on information visualization* (p. 120–123). : IEEE Computer Society. 57

Wright, T. P. (1939). American methods of aircraft production. *The Aeronautical Journal*, 43(339), 131–224. 118

Zaki, M. J., Parthasarathy, S., Ogihara, M., & Li, W. (1997). New algorithms for discovery association rules. Dans *Knowledge Discovery and Data Mining* (p. 283–296). 33, 34

ICICT 2018
International Congress & Excellence Awards

Third International Congress on
INFORMATION AND COMMUNICATION TECHNOLOGY (ICICT-2018)

Certificate

This is to certify that

Hery Frederic RAKOTOMALALA(University of Antsiranana B.P.0, Madagascar)

Bruno Bakys RALAHADY(University of Antsiranana B.P.0, Madagascar) Andre TOTOHASINA(University of Antsiranana B.P.0, Madagascar)

has contributed a paper titled

A novel cohesitive implicative classi cation based on MGK and application on diagnostic on informatics literacy of students of higher education in Madagascar

in the Third International Congress on Information and Communication Technology held at Brunel University London during 27 - 28 February 2018.
The paper has been selected for publication in the ICICT 2018 conference proceedings by Springer AISC [ISSN Number - 2194-5357] as per the guidelines issued by Springer.
We wish the authors all the very best for future endeavors.

R. Simon Sherratt	Xin-She Yang	Nilanjan Dey	Amit Joshi
University of Reading	Middlesex University	Techno India College of Engineering	Organising Secretary
United Kingdom	United Kingdom	India	ICICT 2018

GR FOUNDATION · ACTIVATE LEARNING · CITY OF OXFORD COLLEGE · IEEE United Kingdom and Ireland Section · JECRC Foundation · Springer

International Journal of Computer Science and Technology (IJCST)
ISSN: 2975-382N
http://flyccs.com/jounals/IJCST/Home.html

PAPER REVIEW

Chapter Title:	**EXPERIMENTAL STUDY OF THE VALID RULES ACCORDING TO THE MEASURE M_GK**
Authors:	1Bruno Bakys RALAHADY,2André TOTOHASINA

NOTE: Please put **X** to show your selection

Type of this paper					
	Research	**Survey**	**Tutorial**	**Speculative**	**Others**
Your Choice	**X**				

Evaluation:					
	Very Poor	**Poor**	**Average**	**Good**	**Very Good**
Significance of the main idea(s)				**X**	
Originality				**X**	
Technical quality of the paper					**X**
Awareness of related work					**X**
Clarity of presentation				**X**	
Organization of the manuscript				**X**	
References				**X**	
Paper Length				**X**	

International Journal of Computer Science and Technology (IJCST)
ISSN: 2975-382N
http://flyccs.com/jounals/IJCST/Home.html

How comfortable are you in reviewing this paper?					
	Very Confident	**Confident**	**Adequate**	**Not Confident**	**Not my Area**
Your Choice		**X**			

Overall comments and changes that MUST be made before publication:

1. Broader introduction.
2. No algorithm or method.
3. More references.

Suggestions which would improve the quality of the paper but are NOT essential for publication:

Overall Recommendation:					
	Strongly Reject	**Reject**	**Marginally Accept**	**Accept**	**Strongly Accept**
Recommendation				**X**	

International Journal of Computer Science and Technology (IJCST)
ISSN: 2975-382N
http://flyccs.com/jounals/IJCST/Home.html

PAPER REVIEW

Chapter Title:	**ASI-MGK: Implicative Statistical Analysis tool based on M_{GK}**
Authors:	Bruno Bakys RALAHADY[1] and André TOTOHASINA

NOTE: Please put **X** to show your selection

Type of this paper					
	Research	**Survey**	**Tutorial**	**Speculative**	**Others**
Your Choice			**X**		

Evaluation:					
	Very Poor	**Poor**	**Average**	**Good**	**Very Good**
Significance of the main idea(s)				**X**	
Originality				**X**	
Technical quality of the paper				**X**	
Awareness of related work				**X**	
Clarity of presentation				**X**	
Organization of the manuscript				**X**	
References				**X**	
Paper Length				**X**	

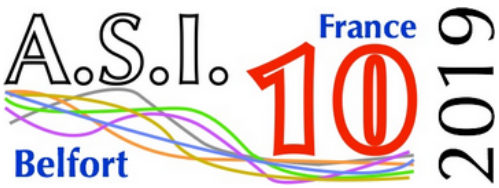
How comfortable are you in reviewing this paper?					
	Very Confident	**Confident**	**Adequate**	**Not Confident**	**Not my Area**
Your Choice		**X**			

International Journal of Computer Science and Technology (IJCST)
ISSN: 2975-382N
http://flyccs.com/jounals/IJCST/Home.html

Overall comments and changes that MUST be made before publication:
• Broader introduction. • No algorithm or method. • Broader conclusions and future works.

Suggestions which would improve the quality of the paper but are NOT essential for publication:

Overall Recommendation:					
	Strongly Reject	**Reject**	**Marginally Accept**	**Accept**	**Strongly Accept**
Recommendation				**X**	



2-5 Octobre 2019
BELFORT Techn'hom
19 avenue du Maréchal Juin - BP 527
90016 Belfort Cedex
Bâtiment D - Génie Civil, Amphithéâtre 4

Lettre d'acceptation

10ème Colloque International sur l'Analyse Statistique Implicative

Cher\Chère(s) Bruno Bakys RALAHADY, André TOTOHASINA, Jean SIMON,

Nous vous confirmons que votre proposition de communication a bien été relue et a été acceptée (selon les modalités détaillées accessibles sur votre compte personnel ASI10) pour être présentée lors du colloque. Le programme sera communiqué sur le site du colloque à partir du 31 août 2019 où vous pourrez connaître le moment de votre communication.

Détails de la communication :

FR-0002 : APPORTS INTERESSANT DE L'INDICE ASI-MGK ET DES COURBES D'APPRENTISSAGES EN DIDACTIQUE DE L'INFORMATIQUES

Attention : Il faudra réaliser une inscription pour participer au colloque et publier la communication dans les actes.

Merci pour votre contribution.

Jean-Claude Régnier, Président du Comité scientifique et de programme
Michel Henry, Vice-Président
Régis Gras, Fondateur
Guy Brousseau, Président d'honneur

Raphael Couturier, Président du comité d'organisation

2-5 Octobre 2019
BELFORT Techn'hom
19 avenue du Maréchal Juin - BP 527
90016 Belfort Cedex
Bâtiment D - Génie Civil, Amphithéâtre 4

Lettre d'acceptation

10ème Colloque International sur l'Analyse Statistique Implicative

Cher\Chère(s) Christian RAZANATSOAVINA, Bruno Bakys RALAHADY, Jean Claude LABERCHE,

Nous vous confirmons que votre proposition de communication a bien été relue et a été acceptée (selon les modalités détaillées accessibles sur votre compte personnel ASI10) pour être présentée lors du colloque. Le programme sera communiqué sur le site du colloque à partir du 31 août 2019 où vous pourrez connaître le moment de votre communication.

Détails de la communication :

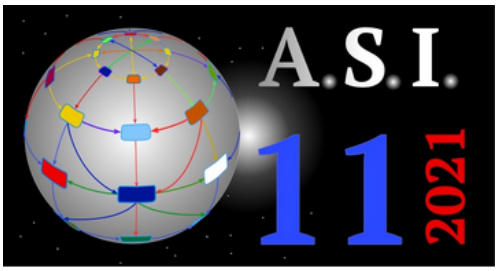
FR-0003 : INFLUENCE D'UNE FORMATION UNIVERSITAIRE EN AGRONOMIE SUR LE RESSENTI DES TRADITIONS PAR LES ETUDIANTS EN PREMIER CYCLE UNIVERSITAIRE: CAS DU DISTRICT DE MANDRITSARA A MADAGASCAR

Attention : Il faudra réaliser une inscription pour participer au colloque et publier la communication dans les actes.

Merci pour votre contribution.

Jean-Claude Régnier, Président du Comité scientifique et de programme
Michel Henry, Vice-Président
Régis Gras, Fondateur
Guy Brousseau, Président d'honneur

Raphael Couturier, Président du comité d'organisation



3-6 Novembre 2021
BELFORT Techn'hom
19 avenue du Maréchal Juin - BP 527
90016 Belfort Cedex
Bâtiment D - Génie Civil, Amphithéâtre 4

Lettre d'acceptation

11ème Colloque International sur l'Analyse Statistique Implicative

Cher\Chère(s) Christian RAZANATSOAVINA, Bakys Bruno RALAHADY, Andrianasy Angelo DJISTERA, Jeanne RAVAOSOLO, Jean-Claude LABERCHE,

Nous vous confirmons que votre proposition de communication a bien été relue et a été acceptée (selon les modalités détaillées accessibles sur votre compte personnel ASI11) pour être présentée lors du colloque. Le programme sera communiqué sur le site du colloque à partir du 15 octobre 2021 où vous pourrez connaître le moment de votre communication et les modalités.

Détails de la communication :

FR-0004 : DES STRATEGIES DE RIPOSTES PLUS A BASE DE PLANTES TRADITIONNELLES FACE A LA PANDEMIE DE COVID-19 CHEZ DES ETUDIANTS UNIVERSITAIRES MALGACHES

Attention : Il faudra réaliser une inscription pour participer au colloque et publier la communication dans les actes.

Merci pour votre contribution.

Jean-Claude Régnier, Président du Comité scientifique et de programme
Antoine Bodin, Vice-Président
Régis Gras, Fondateur
Gérard Vergnaud, Président d'honneur
Raphael Couturier, Président du comité d'organisation

Journées Scientifiques de l'Ecole Supérieure Polytechnique d'Antananarivo
Première Edition, IAC Vontovorona, 10 et 11 mars 2021
« Ensemble face aux Objectifs de Développement Durable »

Poster
N° XXXX

Les étudiants de la filière universitaire agronomique de Mandritsara pensent-ils diversifier les cultures pour assurer le développement de leur région ?

Christian RAZANATSOAVINA[1], Bruno Bakys RALAHADY[1], Andrianasy Angelo DJISTERA[2], Jean-Claude LABERCHE[3]
[1]Ecole Normale Supérieure pour l'Enseignement Technique, Université d'Antsiranana-Madagascar
cmrazanatsoavina@gmail.com, ralahadybru@yahoo.fr
[2]Faculté de Droit, d'Economie, de Gestion et de Mathématiques, Informatique et Applications, Université de Toamasina,
angelo.djistera@yahoo.fr
[3]Professeur des Universités, émérite, DEMESTED group, Thon Duc Thang University Ho Chi Minh City- Vietnam.
Laberche.jean.claude@tdtu.edu.vn

Introduction et Objectifs

Plus de demi-siècle de l'indépendance de Madagascar, la Grande île n'a toujours pas abouti à mettre l'industrie comme principal moteur de la croissance, le riz insuffisant et production en baisse, et centrée sur la famille. De nombreuses Terres arables nombreuses, des variétés de produits de rentes possibles (vanille, girofle, café, arachide, litchis, etc.), énorme problème en diversification, en quantité et en qualité, promotion et politique industrialisation délaissée, moins d'infrastructure, d'innovation, de la recherche scientifique et souvent déconsidère de jeunes techniciens supérieurs et oublié le monde rural qui approvisionne la majorité des matières premières agricoles du secteur industriel de Madagascar.

Le but est de saisir chez ces futurs techniciens supérieurs agricoles, l'évolution de leurs attitudes et de leurs comportements depuis leur entrée en formation supérieure.

Matériels et Méthodes

La recherche a été réalisée, dans le Centre universitaire agronomique à Mandritsara, auprès de 200 étudiants de L1 et L2, de plus de 18 ans, garçons et filles. Questionnaire semi-directif (questions ouvertes et fermées).

Les résultats traités par une Analyse statistique implicative (ASI) qui est un outil statistique, exploitées les données des questionnaires sont transformées en matrices booléennes.

Objectif du travail :
Etudier les relations entre les caractères

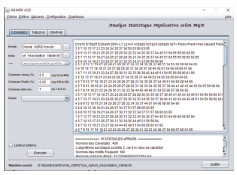
Enquête auprès de la population cible.

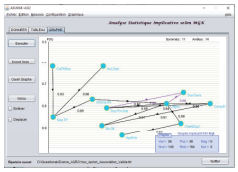
C_1	C_2	...	C_k
$Réponse_{11}$	$Réponse_{12}$	...	$Réponse_{1k}$
$Réponse_{21}$	$Réponse_{22}$	...	$Réponse_{2k}$
...	...	...	...
$Réponse_{n1}$	$Réponse_{n2}$	...	$Réponse_{nk}$

Codage des réponses obtenues en binaire :

$$Code(Réponse_{ij}) = \begin{cases} 1, si\ Réponse_{ij}\ satisfait\ S \\ 0, si\ non \end{cases}$$

C_1	C_2	...	C_k
0	0	...	0
1	1	...	0
...	...	...	...
1	0	...	1





Résultats et Discussions

D'abord la culture du riz est choisie comme premier moteur de développement économique prélude au démarrage industriel. D'ailleurs, le riz est la base alimentaire à Madagascar. Toutefois, la quantité de riz reste insuffisante voire en baisse. L'ouverture culturelle à diversifier des produits, notamment des rentes sont aussi convoités par des jeunes générations. En effet, la majorité des étudiants interrogés sont aussi prêts à développer et diversifier d'autres produits agricoles. Cette ambition générationnelle méritera d'être accompagnée pédagogiquement, techniquement et moyens matériels voire financiers.

Conclusion

Madagascar reste comme un pays symbolique de multi variétés de produits agricoles et la filière industrielle en retard. Des futurs techniciens supérieurs en agronomie ont conscients et prêts à développer le riz et en diversifiant aux autres produits des rentes pour la promotion durable de l'industrie de la région et de Madagascar.

Références bibliographiques

- ✓ Rapport final EPASA-2019 : Evaluation de la production agricole et de la sécurité alimentaire à Madagascar. Enquête.
- ✓ Nationale sur le suivi de l'Objectif du Millénaire pour le Développement à Madagascar : ENSOMD.2012 2013 INSTAT;
- ✓ Rapport 2011, Nations Unies- New-York et Genève. *Le développement économique en Afrique*).
- ✓ Rasoarahona J. – FES (2014). *Etude sur l'état de l'agriculture à Madagascar.*
- ✓ Gafsi, M. (2007). *Exploitations agricoles familiales en Afrique de l'Ouest et du Centre: enjeux, caractéristiques et éléments de gestion*. Editions Quae.
- ✓ Minten, B., Randrianarison, L., & Swinnen, J. F. (2009). Global retail chains and poor farmers: Evidence from Madagascar. *World development*, *37*(11), 1728-1741.
- ✓ RalahadY, B. B., & Totohasina, A. (2019). *Experimental study of the valid rules according to the measure* **GK**.
- ✓ Régnier, J. C., Gras, R., Henry, M., Couturier, R., & Brousseau, G. (2020). Analyse Statistique Implicative.

Remerciements

- ✓ Au Centre universitaire agronomique à Mandritsara et URSFF
- ✓ Aux étudiants de l'Agronomie de Mandritsara
- ✓ Chambre de Commerce et de l'Industrie de la Sofia

Un nouveau logiciel de traitement de données basé sur les graphes implicatifs

TOVOHERY Josoa Michel[1], RALAHADY Bruno Bakys[2], TOTOHASINA André[3] & FENO Daniel Rajaonasy[4]

[1]*Ecole Doctorale Thématique « Science, Culture, Société et Développement » de l'Université de TOAMASINA, josoamicheltovohery@gmail.com*

[2]*ENSET- Université d'Antsiranana, ralahadybru@yahoo.fr*

[3]*ENSET- Université d'Antsiranana, andre.totohasina@gmail.com*

[4]*Faculté de Droit, d'Économie, de Gestion, et de Mathématiques, Informatique et Applications Université de TOAMASINA, fenodaniel2@yahoo.fr*

1. Objectif

Présenter le nouveau logiciel appelé SDD-GI-MGK-TCP, qui est un logiciel de graphe implicatif permettant d'extraire des règles d'association du type « Si X, alors Y » dans un grand volume de données avec une mesure de qualité très performante nommée M_{GK}.

2. Matériels et Méthodes

Dans cette partie, nous allons présenter la manipulation du logiciel SDD-GI-MGK-TCP et son mode de fonctionnement.

2.1. Etapes de travail avec SDD-GI-MGK-TCP

Objectif du travail :

Etudier les relations entre les caractères C1, …, Ck.

C_1	C_2	…	C_k
Réponse11	Réponse12	…	Réponse1k
Réponse21	Réponse22	…	Réponse2k
⋮	⋮	⋮	⋮
Réponse $n1$	Réponse $n2$	…	Réponse nk

Codage des réponses obtenues en binaire :

C_1	C_2	…	C_k
0	1	…	0
1	1	…	0
⋮	⋮	⋮	⋮
1	0	…	1

C_{ij}=1 si la réponse de l'individu i satisfait le souhait S_j sur le caractère C_j (en format .xls).

Lancement du logiciel SDD-GI-MGK-TCP :

1) Importer le fichier .xls ;
2) Paramétrer : MinSup et seuil critique d'acceptation d'une règle ;
3) Exécuter.

2.2. Fonctionnement du logiciel SDD-GI-MGK-TCP

Mesure de qualité utilisée

$$M_{GK}(X \Rightarrow Y) = \begin{cases} M^f_{GK}(X \Rightarrow Y) = \dfrac{P_{X'}(Y') - P(Y')}{1 - P(Y')}, & si\ P_{X'}(Y') \geq P(Y'); \\ M^d_{GK}(X \Rightarrow Y) = \dfrac{P_{X'}(Y') - P(Y')}{P(Y')}, & si\ P_{X'}(Y') < P(Y'). \end{cases}$$

$$M^f_{GK}\_Critique(X \Rightarrow Y) \sqrt{\frac{n_Y(n - n_X)}{n \times n_X \times n_Y}\chi^2(\alpha)}, \qquad si\ P_{X'}(Y') \geq P(Y')$$

Si $P_{X'}(Y') < P(Y')$, alors on étudie la règle $\bar{X} \Rightarrow Y$.

où X' est l'intension du motif X. On utilise seulement M^f_{GK}, car elle est implicative.

Procédure d'extraction des règles d'association

Sélection des motifs fréquents : par l'algorithme « apriori » de Agrawal & Srikant sous la contrainte de MinSup.

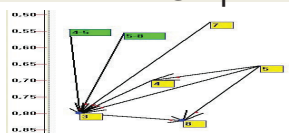
Principe des règles MGK—valides :

1) Si $M^f_{GK}(Y \Rightarrow X) < M^f_{GK}(X \Rightarrow Y)$, alors on retient la règle $X \Rightarrow Y$, respect val. Critique. En cas d'égalité, on a la règle $X \Leftrightarrow Y$.
2) Si M^f_{GK}_critique$(X \Rightarrow Y) < M^f_{GK}(X \Rightarrow Y)$ au seuil b, alors la règle $X \Rightarrow Y$ est validée au niveau de confiance (1-b)100%.

Résultat : Graphe Implicatif valué

3. Résultats

Graphe implicatif valué



Code du graphe implicatif : MAL = 3, FRS = 4, HG = 5, SN = 7 et EPS = 8 Nombre de règles valides : 8

Prémisse	Conséquent	Confiance	MGK	TCP
MAL	EPS	0.8194	Faux	Faux
FRS - HG	MAL	0.8367	Faux	Faux
FRS	MAL	0.8095	Faux	Faux
HG - EPS	MAL	0.8200	Faux	Faux
HG	MAL	0.8305	Faux	Faux
HG	FRS	0.8305	0.4350	Vrai
HG	EPS	0.8474	Faux	Faux
SN	MAL	0.8723	Faux	Faux

Exemple d'interprétation : Si FRS (4), alors MAL (3).

4. Discussion

Comparaison avec le Logiciel CHIC *v.6* et TANAGRA *v.1.4.50*

1) Regroupe les deux représentations de CHIC et Tanagra ;
2) Utilise une mesure de qualité plus performante que les mesures intensité d'implication, confiance et Lift.

5. Conclusion

SDD-GI-MGK-TCP est un logiciel de graphe implicatif permettant aux utilisateurs de vérifier leurs hypothèses en termes de causes à effets. L'introduction du concept de classification cohésive et des règles d'association quantitatives dans ce logiciel seraient notre futur travail.

6. Références

[1] Couturier R., Traitement de l'analyse statistique dans CHIC.

[2] Totohasina A., 2008. Contribution à l'étude des mesures de la qualité des règles d'association : normalisation sous cinq contraintes et cas de MGK : propriétés, bases composites des règles et extension en vue d'applications en statistique et en sciences physiques. HDR spécialité Mathématiques et informatique. Université de Madagascar. https://hal.archives-ouvertes.fr/tel-02481713/document.

[3] Feno R. J., 2007. Mesure de qualité des règles d'association : normalisation et caractérisation des bases. Thèse de Doctorat. Université de La Réunion.

Poster n° 223-P
REVUT & Colloque 2020
Toamasina, avril 2020

ASI-MGK IS AN APPLIED SCIENCE TOOL AVAILABLE TO RESEARCHERS IN ALL DISCIPLINES

RALAHADY Bruno Bakys[1] *, **TOTOHASINA André**[1]

[1]*Ecole Normale Supérieure pour l'Enseignement Technique, Université d'Antsiranana-Madagascar, BP O Antsiranana 201 - Madagascar.*

Keywords : Data analysis, ASI, ASI-MGK, Applied sciences, Confidence threshold.

Introduction

On the occasion of work in didactics carried out on the teaching of data processing in the Malagasy secondary and high schools, we used several multi-factorial experimental methods. To analyze the observed data, we used a non-symmetrical method of data analysis, called implicative statistical analysis (ISA), based on the measure of the intensity of implication. Statistical implicative analysis (ISA) is a non-symmetrical data analysis method designed by Régis Gras based on Gras's Intensity of implication measure [4]. For a large volume of data, Agrawal [1] and his team develop Apriori-type algorithms based on support and confidence deemed even less selective, less relevantand unrecognized the negative association rules. Sylvie Guillaume [5], in his thesis, proposes another more selective implicative quality measure M_{GK}. Totohasina and his teams continued this work, [8] defined these different mathematical properties justifying its relevance and developed its new non-subjective significance threshold. Ralahady, in his thesis [6,7], develops ASI-MGK, an implicative analysis tool based on the support, the M_{GK} and the confidence threshold.

Database

In a transactional database (quantifying the students' knowledge of computer science in their daily school environment) of this didactic research where the different fields correspond to the answers to a questionnaire counting 71 transactions and 50 answers. On the whole we have a Boolean matrix of dimensions 71 x 50.
We want to extract the valid association rules, of type A ⟹B "if a student understands a concept A, then it is very likely that he also understands a concept B", the most relevant.

College students during computer class

Methods

Using the ASI-MGK we performed a successive generation by modifying the minimum support and the confidence threshold.

Quality measure used

Guillaume Kentchaff Measure (M_{GK})

Let X and Y two patterns of a data mining context [6,7,8].

$$M_{GK}(X\Longrightarrow Y) = \begin{cases} \dfrac{P_{X'}(Y') - P(Y')}{1-P(Y')} & \text{If } P_{X'}(Y') \geq P(Y') \\ \dfrac{P_{X'}(Y') - P(Y')}{P(Y')} & \text{If } P_{X'}(Y') < P(Y') \end{cases}$$

$$M_{GK}\text{critic}(X\Longrightarrow Y, \alpha) = \sqrt{\frac{n_{Y'}\cdot(n-n_X)}{n\cdot n_{X'}\cdot(n-n_Y)}}\,\chi_\alpha \quad \text{With X favors Y}$$

If $P_{X'}(Y') < P(Y')$ we study the negative rule $\overline{X} \Rightarrow Y$.

Generation of M_{GK}-valid rules

1) If $M_{GK}(Y\Longrightarrow X) < M_{GK}(X\Longrightarrow Y)$, then we retain the rule X⟹Y, respect val. Critical. In case of a tie, we have the rule X⟹Y.

2) If $M_{GK}\text{critic}(X\Longrightarrow Y, \alpha) < M_{GK}(X\Longrightarrow Y)$ at threshold b, then [7] the rule X⟹Y is validated at the confidence level (1-α)100%.

Pre-processing

Transcription and coding of responses

Codes	Signification	Support
F	Female sex	0.57
M	Male sex	0.43
BVR	Talkative during computer class unplugged	0.60
PTB	Disruptive	0.25
ACT	Actif	0.52
PRT	Participatory	0.52

Boolean variable

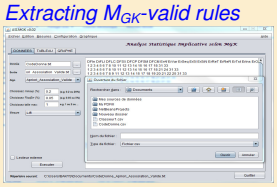
	Q1	Q2	Q3	Q4	⋯	Q16
Stu1	F	Old	Multimedia	17	⋯	Gifted
Stu2	M	Old	Games	16	⋯	Gifted
Stu3	M	New	Games	7	⋯	Weak
Stu4	F	Old	Office	10	⋯	Weak
⋮	⋮	⋮	⋮	⋮	⋮	⋮
Stu75	M	Old	Internet	9	⋯	Weak

ASI-MGK processing

Launching the software

1) Importing the .CSV file;
2) Settings:
   - MinSup
   - Validity threshold
3) Algorithm execution

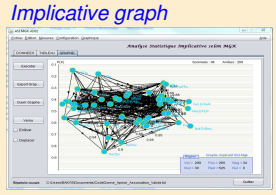
Extracting M_{GK}-valid rules



Boolean matrix

	F	M	BVR	PTB	ACT	PRT	NVE	OBJ	RES	PRS	RTR	PDL
A01	0	1	1	0	0	0	0	0	1	1	1	1
A02	0	1	0	0	1	1	0	1	1	0	0	0
A03	1	0	0	0	0	0	0	1	1	1	1	1
A04	1	0	0	0	0	0	1	1	1	1	1	1
A05	0	1	1	0	0	0	0	0	0	1	1	1
A06	0	1	0	0	1	1	0	1	1	0	0	0
A07	1	0	0	0	1	1	0	1	1	0	0	0
A08	1	0	1	0	0	0	0	1	1	1	1	1
A09	1	0	0	0	0	0	0	1	1	0	0	0
A10	1	0	0	0	1	1	0	1	1	0	0	0
A11	1	0	0	0	1	1	0	1	1	0	0	0
A12	0	1	1	0	0	1	0	1	1	0	0	0
A13	0	1	1	1	1	1	0	0	1	1	0	1
A14	0	1	1	0	1	1	0	1	1	0	0	0
A15	0	1	1	0	1	1	0	1	1	0	0	0
A16	0	1	0	0	1	1	0	1	1	0	0	0
A17	0	1	1	0	1	1	0	1	1	0	0	0
A18	0	1	1	0	1	1	0	1	0	0	0	0

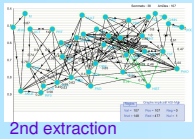
Implicative graph



Results

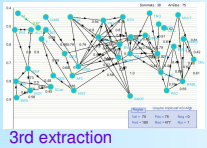
Implicative graph followed by improvement of sight by displacement.

During the first extraction: with minsup = 0.2 and minEur =0.05, we have 280 valid rules.

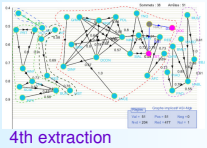


2nd extraction

During the second extraction (*choice of minsup*): By setting minsup = 40%, we have 107 valid rules.



3rd extraction

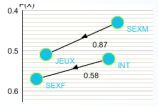


4th extraction

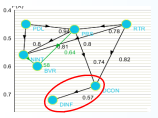
5th extraction

Third to fifth extractions (*choice of minEur):* With minEur = 5.0E-3 we have 79 valid rules, with minEur = 5.0E-4 we have 51 valid rules and with minEur = 5.0E-5, we have 33 valid rules.

Rules interpretation



Rule{SEXM} ⟹{JEUX} : 0.87.
with S*upp*({SEXM}) = 0.43 and S*upp*({JEUX}) = 0.51
Rule{INT} ⟹{SEXF} : 0.58
with S*upp*({INT}) = 0.52 and S*upp*({SEXF}) = 0.57
Here we find that most boys are generally interested in video games. On the other hand, almost all the girls in the class tend to get used to the Internet. We obtained a strong dependence on choice of IT technology or service with gender.



Rule{DCON} ⟹{DINF} : 0.57
with S*upp*({DCON}) = 0.67 and S*upp*({DINF}) = 0.72
Distracting implies hating computers.
Obviously with a percentage of 70%, the students hate computer education without a computer (unplugged) because they are distracted, this is not what they wanted to learn when making contact.
Students feel cheated and hopeless by the switch to unplugged mode.

Conclusions

During these processes, we observed a reduction in the number of valid rules. The less relevant rules disappear as the risk of error is reduced. In addition to the results found for this study, it is very interesting to note that ASI-MGK is an implicative graphing software belonging to the Applied Sciences and that allows the different researchers who use it to verify their hypotheses in terms of cause and effect with a precise risk of error.
It is very interesting to note that this ASI-MGK, as described above has already been used in several scientific works in different disciplines such as :

- Sociology: "*agronomic training, ecosystem and socio-economic progress*"[2].
- Agronomy: *"influence of a university education in agronomy on the feeling of traditions by undergraduate university students: case of the district of Mandritsara in Madagascar"*[3].

This confirms the interest of this technique of statistical analysis for the researchers.

Bibliographic references

[1] AGRAWAL, R., IMIELINSKI, T. and SWAMI, A. (1993). Mining association rules between sets of items in large databases. *In* BUNEMAN, P. et JAJODIA, S., éditeurs : *Proc. of the ACMSIGMOD International Conference onManagement of Data*, volume 22, pages 207–216,Washington,U.S.A.

[2] *C.* RAZANATSOAVINA, B B RALAHADY, J. C. Laberche. and H. BEARINIAINA, (2021). *Formation agronomique, ecosysteme et progres socioeconomique.* Journal of Sciences of Technologies and the Environment *RSTE4, Mahajanga, Madagascar.*

[3] *C.* RAZANATSOAVINA, B B RALAHADY and J. C. Laberche, (2019). Influence of a university education in agronomy on the feeling of traditions by undergraduate university students: case of the district of Mandritsara in Madagascar 10th International Colloquium on Implicative Statistical Analysis (ASI10) October 2-5 2019, Belford, France.

[4] GRAS, R. (2014). Genese et developpement de l'analyse statistique implicative : retrospective historique.*EducMatem Pesq São Paulo*, 16(3):645–661.

[5] GUILLAUME, S. (2000). *Traitement des données volumineuses. Mesures et algorithmes d'extraction des règles d'association et règles ordinales*. Thèse de doctorat, Université de Nantes, France.

[6] RALAHADY, B. B. and TOTOHASINA, A. (2019a). Asi-mgk : Implicative statistical analysis tool based on *m*GK. *IJCST*, 3(1).

[7] RALAHADY, B. B. and TOTOHASINA, A. (2019b). Experimental study of the valid rules according to the measure*m*GK. *IJCST*, 3(1).

[8] TOTOHASINA, A. (2008). *Contribution à l'étude des mesures de qualité des règles d'association : normalisation sous cinq contraintes et cas de MGK : propriété, base composite des règles d'association et extension en vue d'applications en statistique et en sciences physiques.* Thèse de doctorat, Université d'Antsiranana, Madagascar.

THE 3RD INTERNATIONAL CONFERENCE ON APPLIED SCIENCES
ICAS 2022
Ton Duc Thang University, Ho Chi Minh City | 4-5 May, 2022

IMPLICATIVE STATISTICAL ANALYSIS: AN APPLIED SCIENCE TOOL USED IN SOCIOLOGY: AGRICULTURAL STUDENTS OF MANDRITSARA IN MADAGASCAR FACED WITH INNOVATION IN THE LOCAL RURAL WORLD

Christian RAZANATSOAVINA[1], **Bruno Bakys RALAHADY**[1]

[1] *Ecole Normale Supérieure pour l'Enseignement Technique, Université d'Antsiranana-Madagascar, BP O Antsiranana 201 - Madagascar*

Keywords : Data analysis, ASI, ASI-MGK, Applied sciences.

Introduction

This study specifies the attitudes towards the natural, social and economic environment of students trained in agronomy at the University of Mandritsara. These students receive technical training that favors their plans to return to rich rural areas. We want to identify the desires they feel through a questionnaire that is proposed to them and then analyzed.
Mandritsara was created a few years ago where the objective in the agro-environment section is to provide teaching at the License level. Agronomy is taught there in a systemic way based on the analytical conceptual framework proposed by *the " Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services" (IPBES)* [2].

Madagascar map indicates Mandritsara

Mandritsara is a town of 42,000 inhabitants (Region of Sofia), in the northwest of the Big Island. It is in the north of the Malagasy Highlands in a semi-mountainous zone, benefiting from a temperate climate marked by a certain freshness and rare affected by the eye of cyclones.
Economic activities are essentially agricultural, with various crops such as sugar cane, cassava, peas, raffia, peanuts, mangoes, vanilla, cloves, market gardening...as well as cattle and pig farming .

Methods

University and agronomy students

The method of the questionnaire is classical. A semi-directive questionnaire was distributed to 200 boys and girls over 18 years old in their first and second years of agronomy. The future senior technicians are questioned on three points:

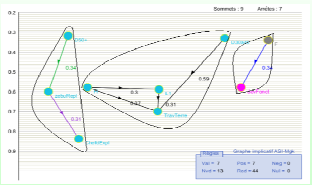
1) Their professional future: do they feel trained to become a farm manager or an agricultural executive.
2) Does the environment of Mandritsara motivate them to work the land?
3) Is leaving home to go to university an obstacle if they want to work in the fields after their studies?

The main points addressed in this questionnaire are:

- the professional choice of future senior technicians in agronomy,
- the grounded feeling of non- Mandritsara students,
- the environment of the study site (university training) and its impact on their motivation to work the land.

Data coding and quality control were performed using Excel spreadsheet office software. From the latter, the characteristics of the sample will be extracted.

The analysis of the data collected begins by counting the numbers of the different items. This is followed by an Implicative Statistical Analysis of the data (ASI-MGK) [3,4]. This new analysis technique has the originality of establishing relationships between items with different distributions.



Presentation of ASI-MGK processing

The general information provided by direct observation of the results of the survey will be clarified by the ASI analysis tool.

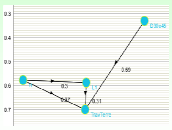
The links between the items through the ASI (threshold of the risk of error $\alpha = 0.025$).

The valid rules obtained between the different items at the risk of error threshold $\alpha = 0.025$ (confidence threshold 1- α = 0.975). It contains the three main points addressed in the questionnaire.

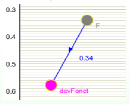
Results

The environment of the study site (university training) and motivation to work the land

We have : Supp({D30a45})=0.33, Supp({H})=0.58, Supp({TravTerre})=0.7, Supp({L1})=0.59 and four valid association rules:
Mgk({D30a45} => {TravTerre})=0.5, Mgk({H} => {L1})=0.3, Mgk({H} => {TravTerre})=0.37and Mgk({L1} => {TravTerre})=0.31



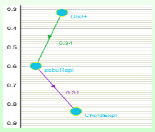
The professional choice of young future senior technicians in agronomy



We have : supp({F})=0.34 , supp({devFonct})=0.58 and only one valid rule MGK({F}->{devFonct})=0.34 which means that the modality {F} favors the modality {devFonct}with an intensity of 0.34

Non-Mandritsara students and their feelings of being grounded

We have : supp({D50+})=0.32, supp({zebuRepl})=0.60, supp({ChefdExpl}) =0.84 and two valid rules; MGK({D50+}=> {zebuRepl})=0.34 which means that the modality {D50+}favors the modality {zebuRepl}with intensity of 0.34 and MGK({zebuRepl}=> {ChefdExpl})=0.31 which means the modality {zebuRepl} favors the modality {ChefdExpl}with intensity of 0.31.



Interpretation of results

All students reject the family structure of the farm based on self-consumption. They are considering more profitable crops than rice, although rice should remain the mainstay of the farm.

The students have a clear professional choice: the girls want to become employees or similar and do not want to return to work in the fields. The boys want to become farm managers.

Students from hard-to-reach areas come to train in agriculture, at the opposite city dwellers are more attracted by the university structure than by the subject matter of the training courses.

Conclusion and perspective

These initial studies, which will be continued to evaluate the evolution of students' aspirations, also provide information to teachers on the means they must develop to best satisfy their students.
In perspective, the protection of the environment (water, soil, natural forests or biodiversity) is urgent and in the face of climate change, university education is inconstantly a tool for success.

Bibliographic references

[1] AGRIDAPE, (2011). *Youth and Farmers. March 2011 - volume 27 n°1.*

[2] BATES P., CHIBA M., KUBE S. & NAKASHIMA D, (2008). *Learning and Knowledge in Indigenous Societies Today* UNESCO - IPBES : Paris, 128pp

[3] RALAHADY B. B. and TOTOHASINA A. (2019) *ASI-MGK: Implicative Statistical Analysis tool based on* M_{GK} in IJCST vol.3 N°.1

[4] RAZANATSOAVINA C., RALAHADY, BB AND LABERCHE, JC (2019). *Influence of a university education in agronomy on the feeling of traditions by undergraduate university students: case of the district of Mandritsara in Madagascar.* Implicative Statistical Analysis, 10: 304

[5] ROS V., X. Yao, M.Villarreal , A. Brizzi , L. Rutten , (2014). *Youth and agriculture: key challenges and concrete solutions.* Joint FAO-CTA-IFAD report, 115 pages (FAO) ISBN 978-92-5-108475-4

THE 3RD INTERNATIONAL CONFERENCE ON APPLIED SCIENCES ICAS 2022

Contribution à l'étude de l'Education algorithmique chez les apprenants du secondaire à l'aide d'analyses statistiques implicatives selon M_{GK}

Résumé :
Constatant la version actuelle du programme scolaire de Madagascar qui ignore le sens véritable de l'informatique, cette thèse, située dans le domaine de Didactiques des Mathématiques et de l'Informatique, propose une stratégie avérée relativement efficace d'intégrer l'éducation algorithmique à travers l'enseignement de la discipline des mathématiques au secondaire à l'aide de ses différentes techniques de résolution de problème. Des expérimentations pédagogiques comprenant des séances d'évaluation suivies d'analyse comparative en présence de population témoin ont été réalisées durant cette thèse pour étudier et identifier les liens implicatifs contenus dans la base de données collectées à cette occasion. Une implémentation optimisée des algorithmes d'extraction des règles d'association a été faite ; ce qui a conduit à la conception et à la réalisation d'un outil informatique d'Analyse Statistique Implicative basé sur la mesure M_{GK} dans le contexte de la fouille de données binaires. Ce qui a permis d'extraire des liens implicatifs ou des règles d'association de type « si comportement (condition) A, alors comportement B ».

Mots-clés
Didactique de l'informatique, éducation algorithmique, expérimentation pédagogique en mathématiques, apprentissage cooéperatif, valeurs critiques de M_{GK}, règles d'association, ASI.

Contribution to the study of Algorithmic Education in secondary school learners using implicative statistical analysis based on M_{GK}

Abstract :
Noting the current version of the school curriculum in Madagascar which ignores the true meaning of computer science, this thesis, located in the field of Didactics of Mathematics and Computer Science, proposes a relatively effective proven strategy to integrate algorithmic education through the teaching of the discipline of mathematics in secondary school using its various problem-solving techniques. Pedagogical experiments including evaluation sessions followed by comparative analysis in the presence of a control population were carried out during this thesis to study and identify the implicative links contained in the database collected from this occasion. An optimized implementation of the association rule extraction algorithms has been made ; which led to the design and implementation of a computer tool for Implicative Statistical Analysis based on the measure M_{GK} in the context of binary data mining. Which made it possible to extract implicative links or association rules of the type "if behavior (condition) A, then behavior B".

Keywords :
Didactics of informatics, algorithmic education, pedagogical experimentation in mathematics, cooperative learning, critical values of M_{GK}, association rules, ASI.

Fandraisan'anjara amin'ny fandalinana ny fampianarana algorithmika amin'ny mpianatra amin'ny ambaratonga faharoa amin'ny fampiasana sava atontan'isa misy fitarihana araka ny M_{GK}

Fintina :
Fahitana ny votoatiny fandaharam-pianarana ankehitriny eto Madagasikara izay tsy miraharaha ny tena dikan'ny siansa momba ny informatika, ity tezy ity, izay ao anatin'ny sehatry ny Didaktikan'ny Matematika sy informatika, dia manolotra paikady mahomby sy azo antoka hampidirana ny fampianarana algoritmika amin'ny alàlan'ny fampianarana ny taranja matematika amin'ny sekoly ambaratonga faharoa mifototra amin'ny fampiasana ireo tetika famahana olana isankarazany. Fanandramana andakilasy izay nisy fotoam-panombanana narahin'ny famakafakana fampitahana amin'ny fisian'ny kilasy vavolombelona, no natao nandritra ity asa fikarohana ity, mba ho fandalinana sy famantarana ireo rohy manjavozavo amin'ny angon-drakitra voaangona tamin'ireo kilasy nokendrena. Nisy ny fampiharana sy fanatsarana ny algoritma momba ny fitrandrahana fitsipiky ny fifandraisana misy fitarihana ; izay nitarika ho amin'ny famolavolana sy ny famokarana ny fitaovana informatika mikasika ny sava atontan'isa mifototra amin'ny refy M_{GK} ao anatin'ny tontolon'ny fitrandrahana angon-drakitra mimari-droa mba hahazoana rohy na fitsipiky ny fifandraisana misy fitarihana « raha toa ka fitondrantena (fepetra) A, dia ny fitondrantena B ».

Teny mampisongadina :
Didaktikan'ny informatika, fampianarana algoritmika, fanandramana andakilasy amin'ny matematika, fianarana miaramientana, sanda fitsikerana M_{GK} farafahakeliny, fitsipiky ny fifandraisana, ASI.

Nombre de tableaux : 23
Nombre de figures : 65
Nombre de pages : 220

Coordonnées de l'auteur :
Tél : +261 34 90 445 05 Email : ralahadybru@yahoo.fr

Directeur : André TOTOHASINA, Professeur Titulaire. Université d'Antsiranana, Madagascar