



**HAL**  
open science

## Mobilité dans les graphes dynamiques

Yoann Pigné

► **To cite this version:**

Yoann Pigné. Mobilité dans les graphes dynamiques : Mémoire d'Habilitation à Diriger des Recherches. Informatique [cs]. Université le Havre Normandie, 2022. tel-03941183

**HAL Id: tel-03941183**

**<https://hal.science/tel-03941183>**

Submitted on 16 Jan 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

# Mobilité dans les graphes dynamiques

Yoann Pigné

le 13 décembre 2022

## Mémoire d'Habilitation à Diriger des Recherches

Section CNU 27 / Discipline Informatique

Université Le Havre Normandie

Présenté devant le jury composé (par ordre alphabétique) de :

### Rapporteurs :

|                      |   |
|----------------------|---|
| Arnaud Casteigts     | Professeur d'université, LABRI, université de Bordeaux                      |
| Frédéric Guidec      | Professeur d'université, IRISA, université Bretagne-Sud                     |
| Christophe Picouleau | Professeur d'université, CEDRIC, Conservatoire national des arts et métiers |

### Examineurs :

|                  |  |
|------------------|--|
| Hocine Cherifi   | Professeur d'université, LIB, université de Bourgogne            |
| César Ducruet    | Directeur de recherche CNRS, EconomiX, université Paris Nanterre |
| Frédéric Guinand | Professeur d'université, LITIS, université Le Havre Normandie    |
| Éric Sanlaville  | Professeur d'université, LITIS, université Le Havre Normandie    |





# Remerciements

Il y a une personne responsable de l'existence de ce mémoire et ce n'est pas moi. Une personne avec l'intention ferme de me faire soutenir cette HDR, alors que je n'avais que des excuses pour ne pas m'y mettre. Éric Sanlaville, merci de m'avoir motivé<sup>a</sup> pour ce projet. Merci de m'avoir relu et conseillé.

Je suis très reconnaissant des retours que m'ont généreusement fourni Arnaud Casteigts, Frédéric Guidec et Christophe Picouleau en acceptant de rapporter sur ce travail. Je sais comme votre temps est précieux et c'est un privilège d'avoir pu vous en subtiliser un peu. Merci également à César Ducruet, Hocine Cherifi, Frédéric Guinand et encore à Éric, d'avoir accepté de faire partie du jury de cette HDR.

Ce travail n'est pas le mien, pas seulement. Il est le fruit de collaborations, de co-encadrements, tous aussi enrichissants les uns que les autres. Je suis très reconnaissant de mes co-auteurs, collègues ou étudiants, qui m'ont beaucoup appris. La liste étant un peu longue et puisqu'il est question de graphes dans ce travail, pourquoi ne pas remercier ces gens et les interactions que nous avons eu dans un véritable réseau, celui de la figure 1.

Merci à L'Équipe, celle avec un « é » majuscule, celle des collègues du laboratoire, pour l'ambiance de travail chaleureuse et conviviale.

Merci à ma famille, Anya, Nahel, Noham et leur maman, pour m'avoir laissé travailler les samedis, les dimanches et les autres jours aussi, pour rédiger ce mémoire.

---

<sup>a</sup>« motivé » n'est pas le bon mot, « botté le ... » est plus approprié.

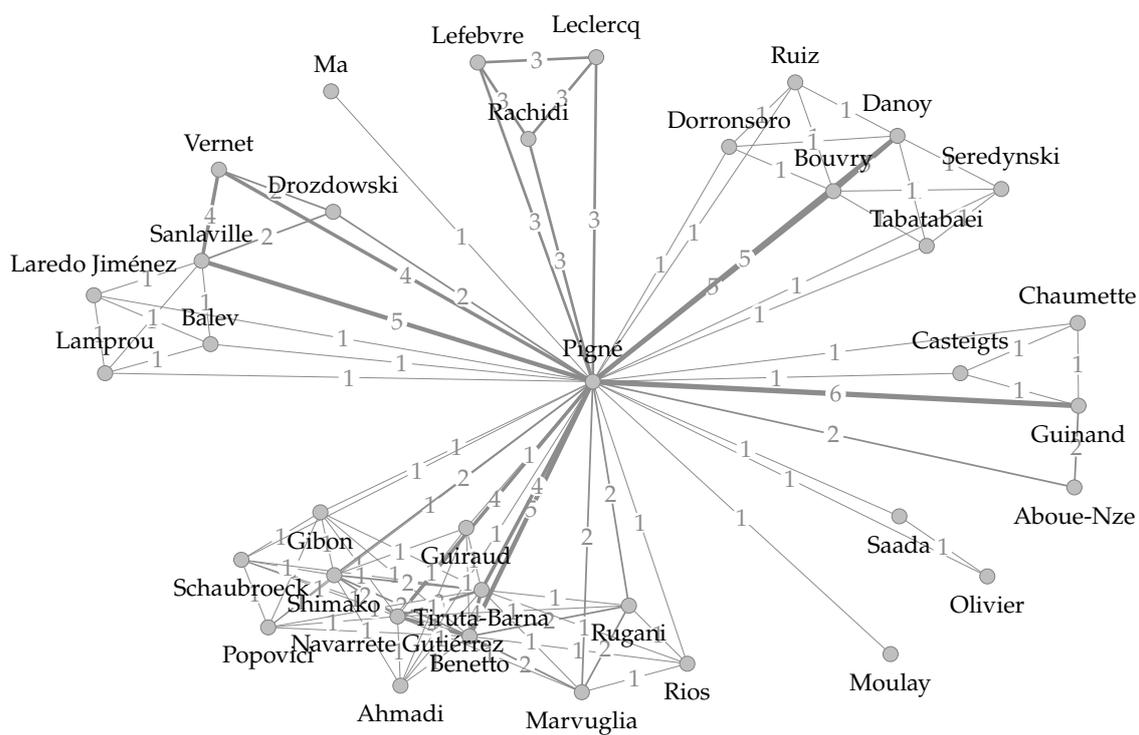


Figure 1. Réseau des co-auteurs avec qui j'ai eu la chance de travailler. Cher-ère-s collègues, par soucis de place seuls vos noms de famille apparaissent, pardonnez-moi cette impolitesse. Cette représentation, très nombriliste, nous laisse aussi voir les différents groupes avec lesquels j'ai pu travailler.

# Sommaire

|   |            |
|---|------------|
| <b>Sommaire</b>   | <b>iii</b> |
| <b>Avant propos</b>   | <b>1</b>   |
| <b>I Mobilité dans les réseaux</b>                          | <b>3</b>   |
| <b>Introduction</b>   | <b>5</b>   |
| <b>1 Mobilité des véhicules</b>                             | <b>7</b>   |
| 1.1 Contexte . . . . .                                      | 7          |
| 1.2 Simulation réaliste de VANETs . . . . .                 | 8          |
| 1.3 Modèles de mobilité réalistes pour les VANETs . . . . . | 10         |
| <b>2 Mobilité avec obstacles</b>                            | <b>13</b>  |
| 2.1 Contexte . . . . .                                      | 13         |
| 2.2 Modéliser des environnements avec obstacles . . . . .   | 13         |
| 2.3 Un modèle de mobilité avec obstacles . . . . .          | 14         |
| <b>3 Mobilité humaine en épidémiologie</b>                  | <b>21</b>  |
| 3.1 Contexte . . . . .                                      | 21         |
| 3.2 Les métapopulations . . . . .                           | 22         |
| 3.3 Modèles de mobilités . . . . .                          | 23         |
| 3.4 Résultats . . . . .                                     | 25         |
| <b>II Analyse de réseaux dynamiques</b>                     | <b>29</b>  |
| <b>Introduction</b>   | <b>31</b>  |
| <b>4 Analyse du cycle de vie</b>                            | <b>33</b>  |
| 4.1 Contexte . . . . .                                      | 33         |
| 4.2 Cycle de vie et réseaux . . . . .                       | 34         |
| 4.3 Analyse statique . . . . .                              | 35         |
| 4.4 Analyse dynamique . . . . .                             | 36         |
| <b>5 Analyse des réseaux maritimes</b>                      | <b>39</b>  |
| 5.1 Contexte . . . . .                                      | 40         |
| 5.2 Modélisation . . . . .                                  | 40         |

|  |  |           |
|--|--|-----------|
| 5.3  | Analyse statique . . . . .                               | 41        |
| 5.4  | Analyse dynamique . . . . .                              | 42        |
| 5.5  | La suite . . . . .                                       | 44        |
| <b>III Algorithmique dans les réseaux dynamiques</b> |  | <b>47</b> |
| <b>Introduction</b>                                  |  | <b>49</b> |
| <b>6</b>   | <b>Le problème des flots de cout minimal</b>             | <b>51</b> |
| 6.1  | Contexte . . . . .                                       | 51        |
| 6.2  | Modèle . . . . .   | 52        |
| 6.3  | Algorithme . . . . .                                     | 52        |
| <b>7</b>   | <b>Le problème de connexité</b>                          | <b>55</b> |
| 7.1  | Contexte . . . . .                                       | 56        |
| 7.2  | Modèle . . . . .   | 56        |
| 7.3  | Algorithme . . . . .                                     | 57        |
| 7.4  | Expérimentations . . . . .                               | 58        |
| 7.5  | Retour sur la génération de réseaux aléatoires . . . . . | 58        |
| <b>8</b>   | <b>GraphStream</b>                                       | <b>63</b> |
| 8.1  | Introduction . . . . .                                   | 64        |
| 8.2  | Principales caractéristiques . . . . .                   | 65        |
| 8.3  | Développement d'un <i>toy problem</i> . . . . .          | 69        |
| 8.4  | Conclusion . . . . .                                     | 73        |
| <b>Conclusion</b>                                    |  | <b>75</b> |
| <b>Références</b>                                    |  | <b>77</b> |

# *Avant propos*

Ce document présente et synthétise l'activité scientifique et les travaux de recherche menés lors de ma carrière dans le monde académique. Je suis enseignant-chercheur, titulaire d'un poste de maître de conférences à l'université Le Havre Normandie. Après des études universitaires en Normandie et une thèse de doctorat soutenue en 2008, sous la direction de Frédéric Guinand, je fus recruté, en 2009, à l'université du Luxembourg, dans l'équipe du Pr. Bouvry. Deux ans plus tard, en 2011, j'intégrais l'université du Havre au poste que j'occupe actuellement.

Ce mémoire, en vue d'obtenir l'Habilitation à Diriger des Recherches, est une occasion assez unique de revenir sur les travaux réalisés, avec un regard différent, plus de recul et peut être une autocritique plus franche. C'est aussi un exercice assez spécial, puisque très franchement narcissique. Je ne me serais jamais autant auto-cité.

Entre la soutenance de thèse, fin 2008, et l'écriture de ce mémoire, fin 2022, quatorze années se sont écoulées. Ce sont les travaux de recherche menés pendant cette période que j'ai choisi de présenter ici. Le document est organisé autour des trois axes principaux que sont la **mobilité**, l'**analyse** et l'**algorithmique** dans le domaine des graphes dynamiques. Les différents travaux prennent assez facilement place dans l'un de ces trois axes. Chaque axe couvre également une fenêtre temporelle qui lui est propre, avec quelques recouvrements.

L'axe **mobilité** (partie I) couvre principalement les travaux de début de période (environs de 2009 à 2015). Un premier chapitre, **Mobilité des véhicules**, revient sur les travaux de mon postdoc au Luxembourg. Le chapitre 2 reprend la thèse de Gaël-Cédric Aboue-Nze sur les obstacles dans les réseaux mobiles ad hoc. Le chapitre 3 discute des résultats obtenus en épidémiologie avec Djamila Moulay.

L'axe **analyse** (partie II) concerne des travaux réalisés en milieu de période (environs de 2014 à 2018). Un premier chapitre, 4, discute de la collaboration, principalement dans le cadre d'un projet ANR, avec des chercheurs en science de l'environnement et concerne l'analyse du cycle de vie. Le chapitre 5 revient sur la collaboration avec César Ducruet autour des réseaux de transport maritimes.

L'axe **algorithmique** (partie III) concerne mes travaux les plus récents (environs de 2018 à 2022). Dans cet axe, les deux chapitres, le premier sur les **flots**, le second sur la **connexité**, reprennent les résultats de la thèse de Mathilde Vernet, co-encadrée avec Éric Sanlaville.

Transversalement à ces trois axes, et transversalement à tout mon travail de recherche, on trouve les graphes et en particulier les **graphes dynamiques**. Cette notion, ce modèle de représentation, est omniprésente dans tout ce qui est présenté ici, des réseaux de véhicules aux calculs de composantes connexes, en passant par

le trafic maritime. La figure A emprunte le formalisme de la gestion de projet pour présenter ces trois axes, comme des *work packages* composés de projets (*tasks*) où les publications sont les *deliverables*. On y voit la distribution temporelle des axes.

Dans le document, chaque axe a sa propre partie. Dans chaque partie figurent des chapitres qui représentent des projets, sanctionnés par une ou plusieurs publications. Ces publications sont rappelées en début de chapitre avec un cadre intitulé « travaux concernés ». La plupart des chapitres contient une section « Contexte », dont le but est de situer le travail en question dans l'espace et le temps, mais aussi de présenter les coauteurs. Enfin certains chapitres possèdent une dernière section, dont le nom varie, qui sert à la fois de conclusion et d'analyse/autocritique.

Enfin, un **dernier chapitre** apparaît à la suite des trois axes. Ce chapitre est dédié à GraphStream, la bibliothèque de manipulation de graphes dynamiques que nous avons développé dans l'équipe, qui m'a suivie (parfois poursuivie) pendant toutes ces années.

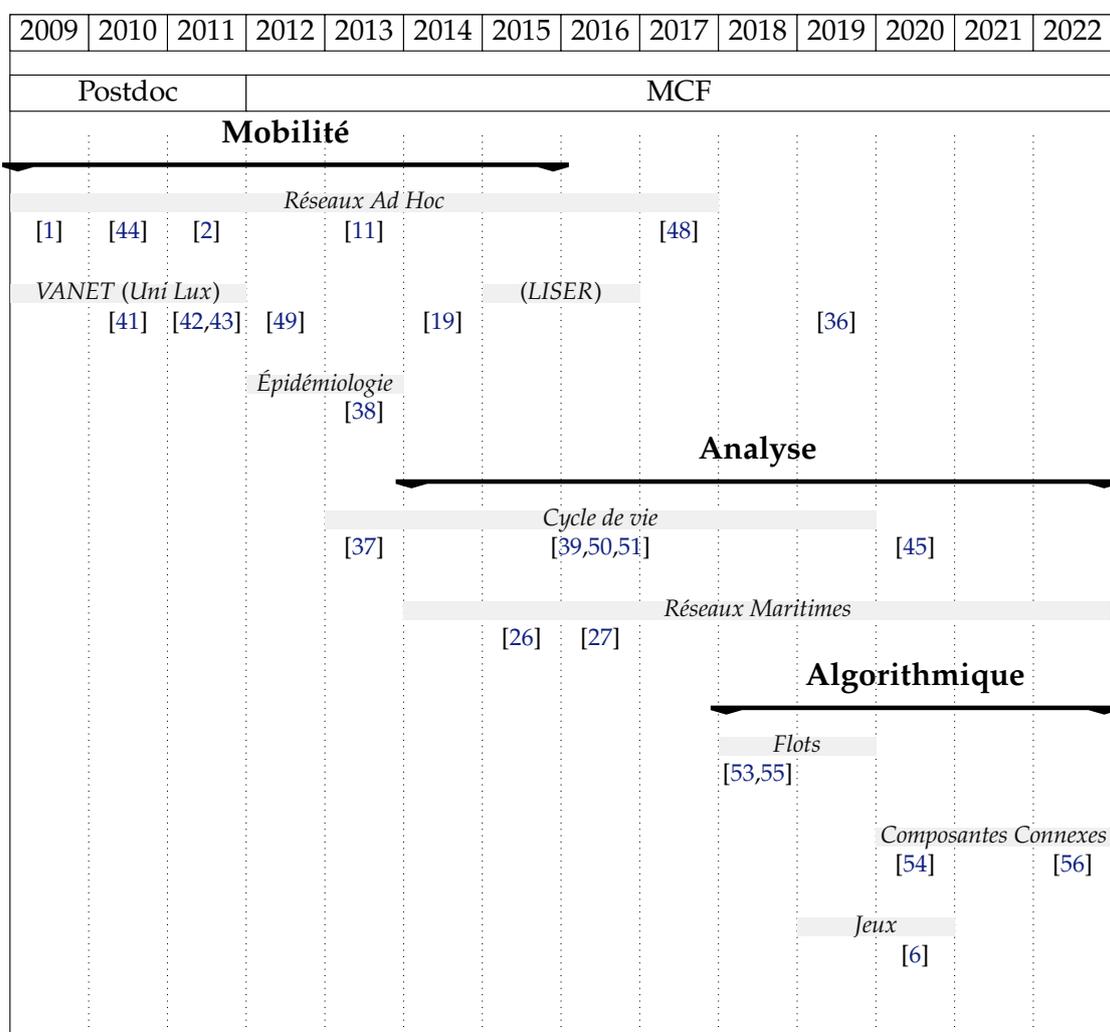


Figure A. Diagramme de Gantt des trois axes de recherche : mobilité, analyse et algorithmique dans les graphes dynamiques, avec les projets et publications associés.

**partie I**

**Mobilité dans les réseaux**



# Introduction

Dans le domaine des graphes dynamiques, la notion de mobilité fait immédiatement penser à la dynamique du réseau lui-même, à ses composants, ses entités, qui se déplacent et donnent sa dynamique au réseau. Nous tentons de différencier ici la dynamique du réseau, de la mobilité des entités qui le composent. Lorsque l'on s'intéresse aux réseaux réels, ceux-ci sont le reflet des observations faites. On voit la dynamique dans de tels réseaux comme la conséquence de l'activité des éléments observés. Ces éléments agissent et interagissent, avec pour conséquence la modification du réseau qui les représente. Dans le cas de la diffusion d'une information entre paires, par exemple, la vitesse et la qualité de la diffusion dans le réseau ne dépend que de l'activité des paires. Dans ce cas, l'activité principale est la mobilité qui permet les interactions entre paires, qui permettent, elles-mêmes, la diffusion. La dynamique résulte donc de l'existence de la mobilité. La figure B montre un autre exemple d'observation du réel, même si ce réel-là reste, pour certains aspects, encore hypothétique, avec des véhicules communiquant par radio, de façon décentralisée, entre eux et avec des unités de bord de route. Tout ce scénario peut s'exprimer d'un point de vue globale avec un graphe qui évolue au fil des liens radios.

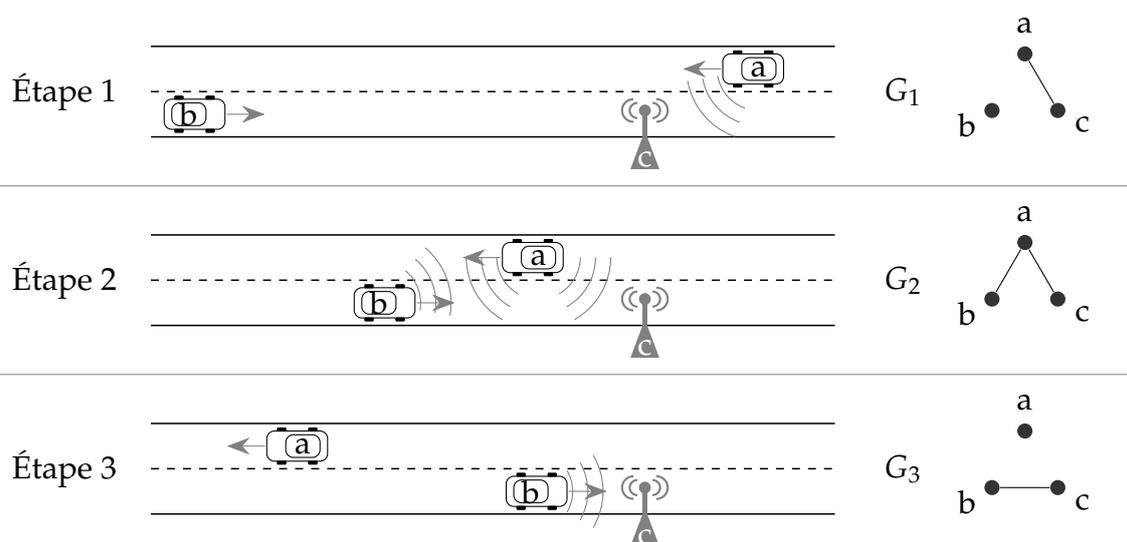


Figure B. Modélisation d'un réseau de télécommunication décentralisé et mobile en un graphe dynamique.

La question de la modélisation et de la simulation des systèmes complexes a toujours été centrale dans mon travail. Dès la fin de la thèse, et dans les premières

années qui l'ont suivie, la notion de mobilité dans ces systèmes est devenue centrale. Pour bien modéliser et simuler le réel à l'aide de graphes dynamiques, il fallait comprendre cette composante du système et donc comprendre la mobilité pour chaque entité du système étudié.

C'est donc la notion de modèle de mobilité des entités du réseau qui nous intéresse et qui, en conséquence, dirige le réseau dynamique. Cette notion est transversale à plusieurs domaines d'application. Les acteurs de cette mobilité sont des usagers de la route quand il s'agit de modéliser les réseaux de communication entre véhicules. Ce sont des entités abstraites et théoriques, quand on s'intéresse directement à des modèles de mobilités synthétiques avec obstacles. Les acteurs de la mobilité sont enfin les humains, leurs habitudes de déplacements journaliers, quand on s'intéresse au réseau qui constitue la propagation d'une maladie à transmission par vecteur.

La volonté peut être d'obtenir un modèle réaliste, qui imite au mieux les motifs réels. Dans ce cas le recours à la simulation réaliste et à l'intégration de données réels, pour rejouer des scénarios réels, est nécessaire. Dans nos travaux, la simulation du trafic routier est un exemple de modélisation réaliste. Elle a constitué l'essentielle de mon activité lors du postdoc à l'université du Luxembourg.

Si le réalisme n'est pas l'objectif, alors la simplicité analytique du modèle peut en être un. Dans nos travaux sur les modèles à obstacles on cherche à simplifier le modèle de mobilité pour le faire ressembler aux modèles de l'état de l'art, tout en intégrant la notion d'obstacle absente des modèles analytiques de l'époque.

# 1 *Mobilité des véhicules*

Travaux concernés : quatre articles en conférences internationales [41–43,49] et un ouvrage [19] paru chez Wiley, issus du postdoc (2009-2011), ainsi qu'un article [36], avec un collègue du LISER, en 2018.

## 1.1 Contexte

Dans les années 2009-2011, période de mon postdoc à l'université du Luxembourg, où le travail présenté ici fut réalisé, la notion de réseau de véhicules équipés de capacités de communication à courte et moyenne distance est très populaire. Ces communications sont opportunistes en fonction du voisinage proche des véhicules et permettent d'envisager des communications de véhicules à véhicules, ou entre des véhicules et des unités en bord de route, elles-mêmes reliées, ou pas, à une infrastructure centralisée. On nomme ces réseaux VANET<sup>a</sup>. Ils permettent la définition d'hypothèses plus ou moins réalistes, afin d'anticiper les évolutions des technologies, des normes, des lois et des besoins.

Dans des scénarios principalement relatifs à la sécurité routière et à la gestion de trafic routier, la problématique mise en évidence est de maximiser l'efficacité du moyen de communication, tant sur ses aspects techniques (choix des bandes passantes et des technologies radio) qu'algorithmiques (protocoles de routage, normes de communication). Or en 2009, les normes de communication dédiées aux applications routières n'étaient pas encore là et l'accès aux infrastructures de type GSM était problématique (couverture et débits insuffisants, coûts élevés).

Pour tester les différentes hypothèses techniques et algorithmiques, la modélisation à l'aide des VANETs s'imposait et le recours à la simulation était nécessaire. Les simulateurs de réseau radio devaient être « réalistes », au sens où ils devaient prendre en compte les aspects physiques de la communication sans fils (atténuation des signaux, réverbérations, etc.). Mais ces derniers ne permettaient pas facilement de simuler de grandes zones telles que des agglomérations. La modélisation du trafic à l'échelle microscopique (comportement du véhicule) était, elle, largement disponible et permettait de simuler les changements de vitesse, les changements de voie, les règles de priorité. À l'échelle macroscopique (un tronçon de route, une agglomération) les techniques existaient (distribution de flots de trafic, matrices origine-destination) mais les modèles réalistes manquaient.

Notre travail au Luxembourg a donc consisté à proposer des solutions de simulation « réaliste » pour les réseaux VANETs, ainsi que des modèles de mobilité

---

<sup>a</sup>*Vehicular Ad hoc Network*

nécessaires aux simulations. Ces propositions ne sont pas celles d'un chercheur en gestion du trafic routier, mais bien celles d'un informaticien, intéressé par les modèles et par leur optimisation. On verra que la problématique d'optimisation est transversale dans ces travaux.

## 1.2 Simulation réaliste de VANETs

La simulation réaliste de VANETs nécessite à la fois de simuler l'environnement routier dans un espace pouvant être large, par exemple à l'échelle d'une agglomération, mais aussi de simuler dans cet espace, des communications radio de plusieurs types. Ces deux aspects sont traditionnellement gérés de façon indépendante, par différents simulateurs. Or l'intérêt ici est de pouvoir intégrer les deux aspects de la simulation. On veut, par exemple, simuler la réalisation d'un itinéraire prévu à l'avance, puis, en cours de simulation, du fait des informations reçues et du contexte local, modifier l'itinéraire et re-router le véhicule concerné. Ce genre de scénario demande une interaction entre la simulation du trafic et la simulation de la communication réseau. Pour offrir cette possibilité il faut coupler deux simulateurs spécialisés. Cela implique que le simulateur de réseau soit informé de la position instantanée des véhicules, pour calculer les signaux radio. À l'inverse, la simulation d'une application de routage, avec capacités de communication radio à bord du véhicule, doit être capable de re-router celui-ci. Une double interaction est donc nécessaire entre les deux simulateurs. En 2010, dans leur domaine, NS3<sup>b</sup> et SUMO<sup>c</sup> étaient les simulateurs de type « Logiciel libre » de référence. Notre contribution [41] fut de proposer une surcouche logicielle permettant l'interconnexion des deux simulateurs. Ce simulateur, OVNIS<sup>d</sup>, proposa un point d'entrée unique pour concevoir des scénarios de mobilité avec des véhicules capables de communiquer et prendre des décisions macroscopiques (type re-routage) durant la simulation. L'interaction avec le simulateur de trafic se fait très simplement avec une API dédiée. L'intégration dans le simulateur réseau se fait via l'écriture d'une application ayant accès à des interfaces réseaux que l'on configure également. La Figure 1.1 illustre la double interaction entre les deux parties de la simulation.

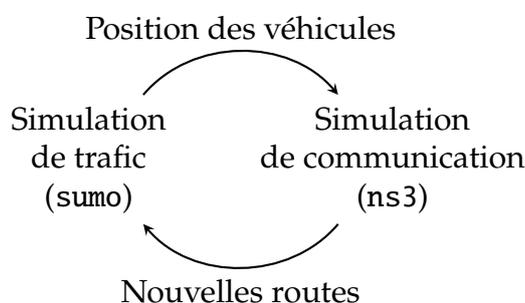


Figure 1.1. Boucles d'interaction entre les simulateurs de réseau et de trafic.

<sup>b</sup><https://www.nsnam.org/>

<sup>c</sup><https://www.eclipse.org/sumo/>

<sup>d</sup>Online Vehicular Network Integrated Simulation: <https://github.com/pigne/ovnis/>

OVNIS propose également une version optimisée de NS3 pour pouvoir fonctionner dans des environnements de simulation étendus (plusieurs dizaines de km<sup>2</sup>). La simulation de la couche physique du modèle réseau de NS3 considère toutes les stations à la fois lors de tous les envois de messages. Quand l'envoi d'un paquet est simulé, un calcul d'affaiblissement du signal est fait entre la station source et toutes les stations de l'environnement afin de déterminer lesquelles sont susceptibles de recevoir le signal et *in fine*, de recevoir la communication. Si l'environnement de simulation est du même ordre de grandeur que la portée des messages radios alors ce modèle fonctionne bien. Mais s'il y a une différence d'ordre de grandeur entre l'environnement simulé et la portée radio, alors les calculs d'affaiblissement de signal entre stations trop distantes les unes des autres deviennent inutiles et peuvent ralentir le simulateur. Quand les environnements simulés grandissent, ces calculs dans la couche physique sont la principale cause de ralentissement du simulateur. Nous avons proposé de décomposer la simulation de la couche physique en un maillage dont la taille de mailles est paramétrique et du même ordre de grandeur que la distance maximale théorique des radios simulées. Cela permet d'isoler les stations dans un voisinage suffisamment large pour que le calcul de l'affaiblissement des signaux et les interférences ne soient pas perturbés, tout en évitant d'interagir avec les stations trop distantes dont on sait qu'elles ne peuvent être atteintes.

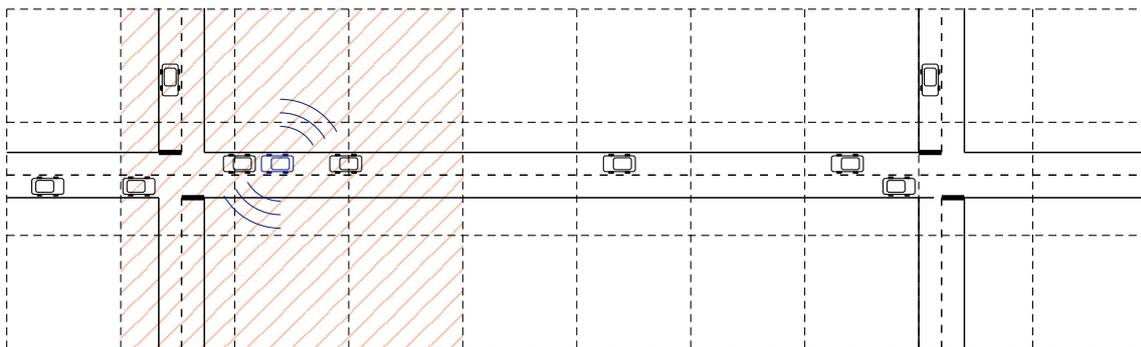


Figure 1.2. Illustration du maillage de l'espace de simulation de façon à restreindre les interactions entre véhicules physiquement proches.

La figure 1.2 illustre l'idée de ce découpage en mailles (le quadrillage en pointillés [---]). Seules les stations se trouvant dans l'une des zones voisines (cellules hachurées [---]) de la station émettrice ( [---] ) sont considérées. Au maximum neuf cellules sont sélectionnées (moins en bordure d'espace de simulation). Il s'agit de la cellule où se trouve la station émettrice, ainsi que des 8 voisines (si elles existent). À l'intérieur de ces cellules la réception des signaux est calculée entre l'émetteur et toutes les stations avec les mêmes contraintes que dans la procédure classique (calcul de l'affaiblissement et des différents modèles physiques). Ainsi pour une taille de maille  $r$  on remarque que la distance maximal accessible dans les cellules est comprise entre  $r$  et  $2\sqrt{2}r$  m.

On note que le choix de la taille des mailles doit être de l'ordre de la portée et de la zone d'interférence des radios simulées. Il faut en effet considérer une zone au-delà du maximum de portée radio car les signaux radios, même s'ils sont

trop faibles pour permettre la réception d'information, participent au calcul du ratio signal sur bruit pour les autres signaux. Ainsi dans les simulations réalisées même si les portées théoriques des radios ne dépassaient pas quelques dizaines à quelques centaines de mètres on préférerait toujours considérer une taille de maille plusieurs fois plus grande que ces valeurs. Par exemple pour la simulation du protocole WIFI 802.11b, réputé inaccessible au-delà de 200 m, des tailles de maille supérieures ou égales à 400 m étaient utilisées.

### 1.3 Modèles de mobilité réalistes pour les VANETs

Dans la suite du postdoc et poursuivant la logique d'obtenir des simulations réalistes, une contribution aux aspects macroscopiques de la simulation de trafic fut proposée. Bien sûr, les équipes de recherche en sciences et technologies des transports proposent de nombreux modèles de mobilités pour la génération de trafic. Notre proposition n'a pas pour ambition de surpasser les modèles existants mais plutôt d'apporter une proposition avec un point de vue différent, avec un modèle générique qui ne soit pas lié à un environnement spécifique, un modèle paramétrique pouvant être optimisé pour les instances considérées.

En effet, le problème est très largement traité. En termes de données, on parle de matrices OD (origine-destination) qui encodent les départs et les arrivées des flux de véhicules. En fonction des domaines, ces matrices sont obtenues par le biais d'enquêtes, de suivis de flottes de véhicules, ou de suivis passifs (comme l'écoute des réseaux cellulaires). Dans [43] le modèle propose de générer des trajets, ou des matrices OD, aléatoires basés sur la définition de zones d'attraction, ainsi que sur quelques jeux de données réelles.

Ces données sources sont de deux natures :

- Des observations fixes à de multiples points du réseau, comme le sont typiquement les données de comptage via des dispositifs de type boucle d'induction, qui compte le trafic sur un tronçon de route donné. Ces données sont parfois librement accessibles, comme c'était le cas en 2010 au Luxembourg et encore aujourd'hui<sup>e</sup>.
- Une carte de l'environnement simulé, avec des zones d'intérêt marquées par une occupation des sols particulière (zones résidentielles, zones commerciales, zones industrielles). Ces données sont largement disponibles dans le projet OpenStreetMap<sup>f</sup>.

Comme son nom ne l'indique pas, le modèle *VehILux* est générique et ne s'applique pas nécessairement au Luxembourg. En fonction des sources de données utilisées il peut être appliqué dans n'importe quelle agglomération. Le cas du Luxembourg est effectivement utilisé pour illustrer ce modèle générique, avec des sources de données propres au pays.

---

<sup>e</sup><https://travaux.public.lu/fr/infos-traffic/comptage.html>

<sup>f</sup><https://www.openstreetmap.org/>

## Un modèle paramétrique générique

Le modèle de génération est basé sur la définition de zones d'attraction dans l'espace de simulation, qui vont diriger le tirage aléatoire des destinations. Les sources aussi sont tirées aléatoirement avec des règles similaires.

On définit des types de zones géographiques qui vont constituer des sources, ou des destinations potentielles pour la génération de matrices OD. Sur un espace donné de simulation, plusieurs instances de ces zones typées seront définies, avec des règles de priorité de sélection paramétriques.

Ainsi une zone d'attraction définie dans l'espace de simulation est caractérisée par :

- son type de zone ;
- sa priorité (une valeur paramétrique qui lui est propre) ;
- sa surface.

Ces trois caractéristiques permettent d'établir une priorité globale dans l'espace de simulation et ainsi de biaiser un tirage aléatoire des zones, pour permettre de sélectionner des positions d'origine et de destination, pour le modèle de mobilité.

La figure 1.3 illustre cette notion de zone d'attraction typée sur une carte de l'agglomération de Luxembourg.

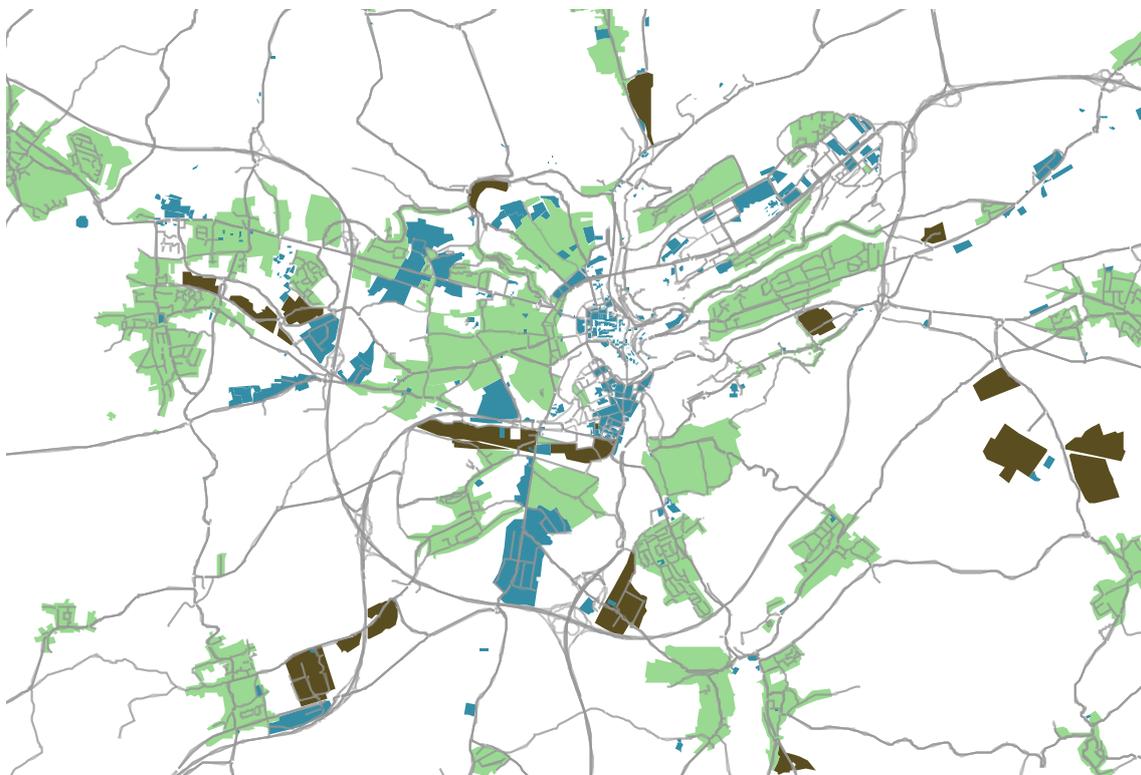


Figure 1.3. Zones d'attraction sur l'agglomération de Luxembourg avec trois types de zones : résidentielles (vert clair), commerciales (bleue moyen) et industrielles (marron foncé). Données OpenStreetMap.

Les flots eux aussi sont soumis à des paramètres (*e.g.* ratio de véhicules provenant de l'extérieur de la zone de simulation) et sont guidés par les données de comptage.

## Optimisation des paramètres du modèle

Les trajets générés peuvent être comparés aux données réelles et le modèle peut être ajusté en modifiant les paramètres des lois de probabilité. En effet, si une partie des données de comptage est utilisée pour diriger le modèle, une autre partie est utilisée pour optimiser ce dernier en ajustant ses paramètres.

Dans [42] on utilise l'analyse de sensibilité pour quantifier l'impact de chaque paramètre sur les sorties du modèle, ainsi que l'impact des paramètres entre eux. On y apprend, sans trop de surprise, que les importances relatives des types de zones influent fortement. On y découvre surtout que le ratio entre les flots venant de l'extérieur de la zone de simulation et le flot interne est le paramètre le plus sensible du modèle.

Vient ensuite l'étape d'optimisation par méthodes meta-heuristiques. Dans [49] trois types d'algorithmes génériques sont expérimentés (*generational GA*, *steady-state GA*, and *cellular GA*). Ces méthodes permettent bien sûr d'améliorer le comportement du modèle au regard des données réelles de contrôle. Mais elles nous montrent surtout les limites du modèle. En effet, si la majorité des points de contrôle (boucle de comptage) est simulée avec précision, certains tronçons, des routes secondaires, restent éloignés de la réalité. L'hypothèse principale est qu'il manque au modèle de calcul de chemin une dimension sociologique, pour prendre en compte les changements de comportement, lors des heures de pointe. En effet, quand le trafic se charge, il devient probablement intéressant d'utiliser, dans une certaine mesure, des routes secondaires, dont la durée de trajet initial a été rejetée par le calcul de plus court chemin basé sur les seuls temps de trajet théoriques, eux même dérivés des vitesses limites légales.

## Retour sur la période VANET

Les travaux présentés dans ce chapitre représentent le travail réalisé durant mon postdoc à l'université du Luxembourg. Le simulateur proposé, ainsi que le modèle de mobilité, ne sont plus tout à fait d'actualité. Les besoins actuels en recherche autour de la mobilité sont spécifiques et nécessitent des applications plus spécialisées (les changements de modalités, la e-mobilité, *etc.*). Ces travaux ont néanmoins été utilisés après mon passage à l'Uni. Le développement du simulateur OVNIS fut continué par Agata Grzybek pour des applications de gestion décentralisée de congestions rapides de trafic. De son côté Sune S. Nielsen s'est appliqué à étendre le modèle VehILux ainsi qu'à continuer le travail d'optimisation sur les paramètres, à l'aide d'algorithme génétique de type co-évolution. Agata et Sune furent tous deux étudiants en thèse avec Pascal Bouvry.

Lors d'une collaboration avec Tay-yu MA du LISER<sup>§</sup>, j'ai réutilisé OVNIS et VehILux pour une application de prédiction de fluidité du trafic à cours terme [36].

---

<sup>§</sup>Luxembourg Institute of Socio-Economic Research

## 2 *Mobilité avec obstacles*

Travaux concernés : [1,2]

### 2.1 Contexte

Ce chapitre présente les principaux résultats de la thèse de Gaël-Cédric Aboue-Nze. Cédric fut encadré, entre 2008 et 2011, par Frédéric Guinand. Bien que n'étant pas co-encadrant de cette thèse, j'ai participé au projet dans son ensemble. La thèse de Cédric commençait à la fin de la mienne. J'ai travaillé sur ce projet en parallèle du mien. Les résultats présentés ici, n'apparaissent pas dans mon manuscrit de thèse. Ma contribution la plus notable porte plus sur la seconde partie du travail, concernant le modèle de mobilité.

### 2.2 Modéliser des environnements avec obstacles

Les réseaux mobiles ad-hoc (MANET) considèrent des terminaux de télécommunication dotés de radios et de protocoles leur permettant d'initier et d'entretenir des communications de pair à pair. De plus ces terminaux possèdent la capacité de se mouvoir, transportés par des humains ou embarqués dans des véhicules. Dans la suite on appelle ces terminaux des *stations*.

Dans ce contexte de MANET, l'évaluation des différents protocoles de communication et de routage passe souvent par l'élaboration de simulations et de modèles de mobilités. Ces modèles sont aussi souvent analysés par le biais des graphes de connexion qu'ils induisent. La notion d'obstacle est également introduite pour modéliser la réalité plus compliquée des environnements réels. Nous avons proposé de modéliser des environnements avec obstacles et d'en proposer une analyse précise en termes de couverture réseau [1]. Ce premier travail a permis d'identifier de façon analytique l'impact des obstacles sur les connexions réseaux et de déduire le degré moyen attendu d'un réseau aléatoire de type MANET.

La figure 2.1 illustre les différentes zones, portant chacune une formulation de la surface de couverture théorique des stations et par conséquent du degré moyen du réseau.

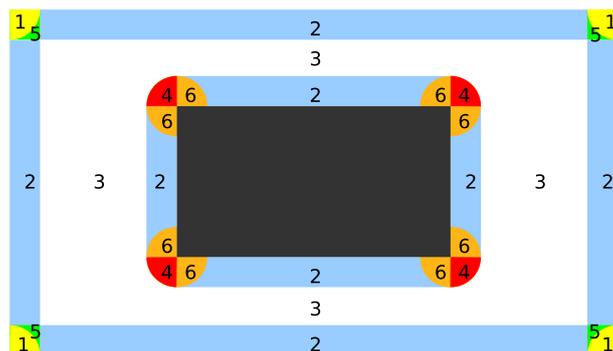


Figure 2.1. Identification des différentes zones autour d'un obstacle, au centre de l'espace, aux bords duquel le degré espéré est différent. Image extraite de la thèse de Gaël-Cédric Aboue-Nze.

## 2.3 Un modèle de mobilité avec obstacles

Une hypothèse, pour calculer la distribution des degrés du graphe de connexion dans une zone avec obstacles, est que la distribution des stations dans l'environnement est entièrement connue. Cette distribution, ou plus généralement la densité des stations, est nécessaire pour calculer le degré à n'importe quel endroit de l'environnement. Dans une hypothèse de non-mobilité, la densité des stations est uniforme dans l'espace libre de l'environnement.

Concernant la mobilité, il est peu probable que les stations mobiles maintiennent une distribution spatiale uniforme dans l'environnement. Le principal avantage du modèle de mobilité aléatoire *Random Waypoint* (RWP) est qu'il fournit une telle densité de stations. En effet, il a été prouvé que RWP<sup>a</sup> possède une distribution de stations stationnaire, ou en d'autres termes, une fonction de densité de probabilité non constante, qui peut être déterminée analytiquement.

Pourtant, RWP ne considère pas les obstacles dans les environnements et pour autant que nous puissions en juger à l'époque, aucune méthode n'avait encore été proposée pour calculer la distribution de densité des stations mobiles, dans un environnement obstrué. D'où la proposition d'une solution de contournement qui consiste en une approximation discrète de la distribution de densité.

Le modèle RSP-O-G (*Random Shortest Path - Obstacle - Grid*) est proposé dans [2]. Compte tenu de la définition d'un environnement obstrué (une zone de simulation et un ensemble d'obstacles dans celle-ci) et de quelques paramètres détaillés par la suite, RSP-O-G est capable de produire différents modèles de mobilité qui respectent les principes de base de RWP :

- sélection aléatoire des destinations,
- déplacement de la source à la destination en empruntant le plus court chemin. Au-delà du modèle de mobilité, RSP-O-G fournit la distribution des stations dans l'environnement choisi.

<sup>a</sup>[8] C. Bettstetter, G. Resta, and P. Santi. 2003. The Node Distribution of the Random Waypoint Mobility Model for Wireless Ad Hoc Networks. *IEEE Trans. Mobile Computing* 2, 3 (September 2003), 257–269.

## Le modèle RSP-O-G

RSP-O-G repose sur un environnement discret. L'espace libre de l'environnement est modélisé par un graphe maillé, une grille incomplète où les mailles ne sont présentes que dans les espaces libres d'obstacles. Les nœuds du graphe représentent toutes les destinations possibles et les stations se déplacent le long des chemins les plus courts entre des paires de nœuds dans ce *graphe de mouvement*. De la taille des mailles de la grille dépendent la taille du graphe, le nombre de destinations et donc la précision du système.

À condition que les plus courts chemins entre toutes les paires de nœuds soient calculés et que le nombre de chemins passant par chaque lien soit donné, une distribution de densité discrète peut alors être proposée sur l'environnement, avec une densité de station estimée sur chaque nœud (et/ou arête) du *graphe de mouvement*.

Pour cela, certaines hypothèses sont considérées.

- Les stations se déplacent le long des arêtes et des nœuds du *graphe de mouvement*.
- La vitesse d'une station sur un chemin est constante sur ce chemin.
- La sélection de vitesse pour chaque nouveau chemin est homogène et considérée comme constante.
- Pendant le parcours continu d'une station le long d'un chemin (nœuds et arêtes), la station est toujours associée à un seul nœud à la fois. Même si sa mobilité est continue ou quasi continue (pas discrets à grain fin), on considère qu'elle évolue discrètement d'un nœud à l'autre le long du chemin.

Soit  $G(V, E)$  le *graphe de mouvement* avec  $V$  l'ensemble des sommets et  $E \in V \times V$  l'ensemble des arêtes non orientées. Soit  $v$  un nœud tel que  $v \in V$ . Un chemin le plus court dans  $G$  est un ensemble minimal  $w$  de nœuds successifs (connectés par des arêtes de  $E$ ) d'un nœud source à une cible. L'ensemble de tous les chemins les plus courts possibles  $W$  entre chaque paire possible de nœuds (source et cible) est également appelé ensemble de chemins les plus courts pour toutes les paires.

$$W = \bigcup_{\{a,b\} \in \mathcal{P}_2(V)} W_{\{a,b\}}, \quad (2.1)$$

où  $W_{\{a,b\}}$  est l'ensemble des plus courts chemins entre les nœuds  $a$  et  $b$  et  $\mathcal{P}_2(V)$  est l'ensemble des 2-combinaisons de  $V$ .

Chercher une densité de station discrète dans  $G$  est identique à trouver, pour chaque nœud  $v \in V$ , la probabilité  $Pr(v)$  qu'une station soit située sur un nœud  $v$ , peu importe le chemin parcouru par cette station.

Cependant cette probabilité est influencée par les deux caractéristiques principales de ce problème particulier :

- Deux plus courts chemins pris au hasard entre deux paires de nœud (source, cible) différents n'auront pas nécessairement la même longueur. Ainsi, étant donné n'importe quel chemin, la probabilité de présence sur un nœud particulier sera inversement proportionnelle à la longueur de ce chemin.

- Plusieurs plus courts chemins peuvent exister entre une paire particulière de nœuds (source, cible). Tous ces chemins ont la même longueur et sont équiprobables entre eux.

Cette probabilité ne peut pas être exprimée directement car elle dépend du chemin sélectionné par la station. En utilisant la loi des probabilités totales,  $Pr(v)$  peut cependant s'exprimer comme suit :

$$Pr(v) = \sum_{w \in W} Pr(v|w) \cdot Pr(w), \quad (2.2)$$

avec  $Pr(v|w)$  la probabilité conditionnelle d'être sur le nœud  $v$  sachant que le chemin  $w \in W$  est sélectionné par la station, et  $Pr(w)$  la probabilité que le chemin  $w$  soit sélectionné par la station parmi tous les autres chemins. D'après (2.1), (2.2) peut aussi s'exprimer comme :

$$Pr(v) = \sum_{\{a,b\} \in \mathcal{P}_2(V)} \sum_{w \in W_{\{a,b\}}} Pr(v|w) \cdot Pr(w). \quad (2.3)$$

$Pr(v|w)$ , sous l'hypothèse de vitesse constante pour les stations est :

$$Pr(v|w) = \frac{\delta_v(w)}{|w|}$$

où

$$\delta_v(w) = \begin{cases} 1 & \text{si } v \in w \\ 0 & \text{sinon} \end{cases}$$

est la mesure de Dirac qui indique si  $v$  appartient à  $w$  et le cardinal  $|w|$  est également considéré comme étant la longueur du chemin  $w$  en termes de nombre de nœuds.

$Pr(w)$  exprime la probabilité de choisir un chemin. Même si, par souci de simplicité, on veut que la sélection d'un chemin soit équiprobable, ce choix est toujours contraint par deux paramètres : le couple de nœuds (source, cible), et le nombre de chemins entre source et cible.

$$\begin{aligned} Pr(w) &= Pr(w|W_{\{a,b\}}) \cdot Pr(\{a,b\}) \\ &= \frac{1}{|W_{\{a,b\}}|} \cdot \frac{1}{\binom{|V|}{2}} \end{aligned}$$

Enfin,  $Pr(v)$ , la probabilité pour une station d'être sur le nœud  $v$  peut être exprimée comme suit :

$$Pr(v) = \sum_{\{a,b\} \in \mathcal{P}_2(V)} \sum_{w \in W_{\{a,b\}}} \frac{\delta_v(w)}{|w|} \cdot \frac{1}{|W_{\{a,b\}}|} \cdot \frac{1}{\binom{|V|}{2}}.$$

### Couverture $\times$ présence $\times$ densité = degré

Si l'on connaît la couverture radio d'une station en tout point de l'environnement ainsi que sa probabilité de présence, lors d'une mobilité suivant le modèle RSP-O-G, alors, pour une densité de stations donnée, on peut estimer une distribution des degrés du réseau de télécommunication dans l'espace de simulation. En d'autres termes, en chaque point de l'espace de simulation, le degré espéré du réseau est le produit de la couverture en ce point, de la probabilité de présence des stations en ce point et de la densité globale du réseau.

La figure 2.2 reprend l'ensemble de la démarche de ce travail partant de l'espace à obstacles, passant par les calcul de couverture et de distribution de présence, pour arriver au degré du réseau escompté en tout point de l'espace. Une autre représentation de la figure 2.2 est tentée, en trois dimensions, dans la figure 2.3.

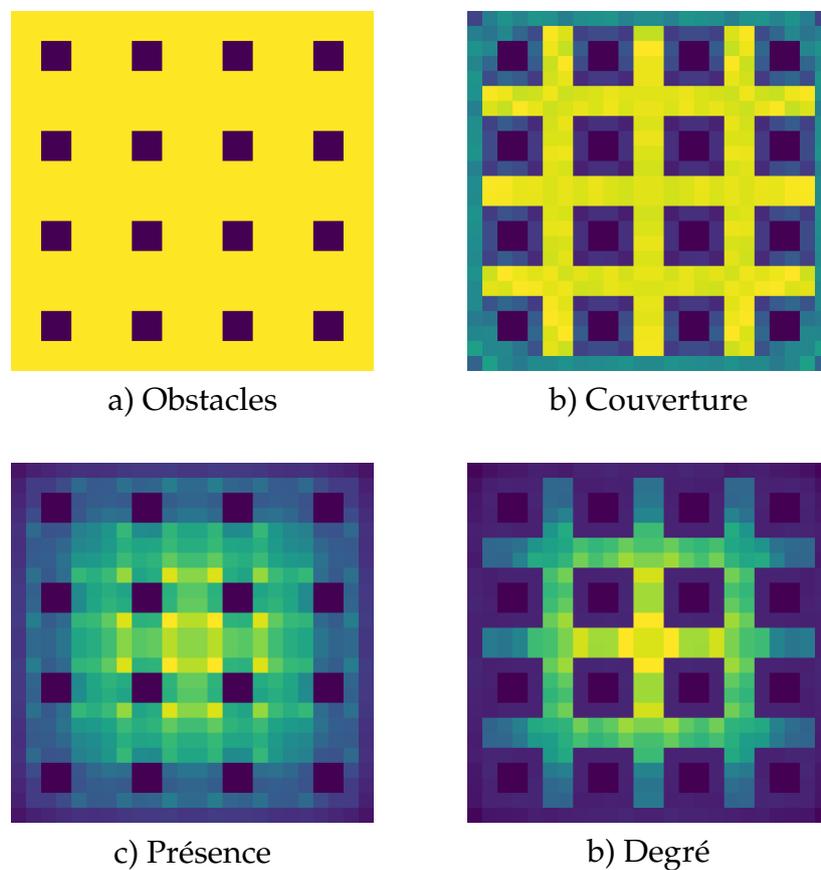


Figure 2.2. Exemple complet du modèle à obstacles. a) environnement de simulation carré avec 16 obstacles (les obstacles sont noirs, l'espace libre est jaune, les bords de la simulation sont également des obstacles). b) Couverture réseau potentiel des stations dans les espaces libres et aux abords des obstacles. c) Distribution des stations (probabilité de présence) selon le modèle de mobilité RSP-O-G. d) Degré escompté du réseau de communication dans l'espace.

Cette illustration résume et conclue ce travail réalisé pour permettre d'intégrer des environnements à obstacles dans le contexte de modélisation et de simulation d'algorithmes de distribués pour les réseaux mobiles ad hoc.

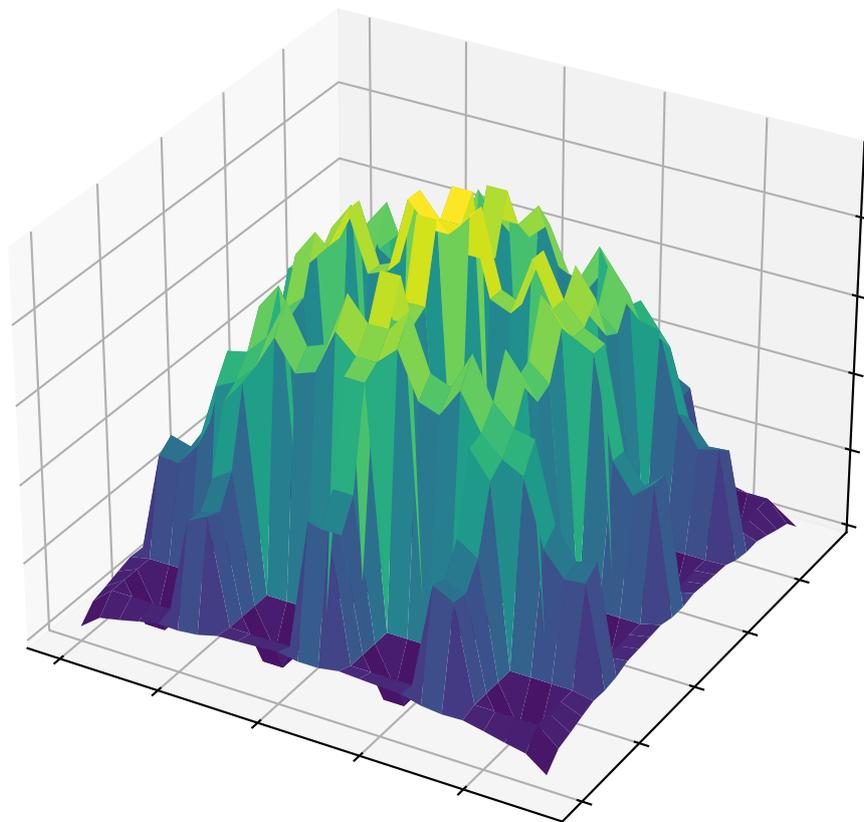


Figure 2.3. Autre représentation, en 3D, de l'exemple de la figure 2.2 pour le degré escompté du réseau de communication dans l'espace.

### Retour sur la distribution des stations

La distribution des stations, telle qu'elle est présentée au-dessus, n'est pas sans rappeler une métrique de graphe qui manipule également les plus courts chemins entre toutes les paires. Il s'agit de la mesure de centralité intermédiaire ou *Betweenness Centrality*. Celle-ci cherche à évaluer la centralité des nœuds (ou des arêtes) dans le graphe, en effectuant pour chaque nœud (ou chaque arête) un ratio entre le nombre de chemins qui passent par ce nœud (cette arête) et le nombre de chemins entre chaque paire du graphe. Une normalisation est ensuite appliquée pour pondérer cette valeur au regard du nombre de 2-combinaisons de  $V$ . Cette valeur n'est pas une distribution. La somme ne vaut pas 1. Aussi, au risque d'une perte de précision et dans un souci de comparaison, on peut effectuer une mise à l'échelle (un *softmax*) de sorte que la somme des valeurs de tous les nœuds (ou les arêtes) vaille 1. On peut ainsi comparer cette centralité mise à l'échelle avec notre distribution des stations. Les deux objets ne sont certes pas identiques. Par définition ils sont différents. Dans RSP-O-G seuls des fractions de chemins (divisés par leur longueur) sont comptés alors que dans la centralité ces chemins sont divisés par le nombre de chemins ayant même source et cible. Pour s'en convaincre la

figure 2.4 montre les deux valeurs sur un graphe simple.

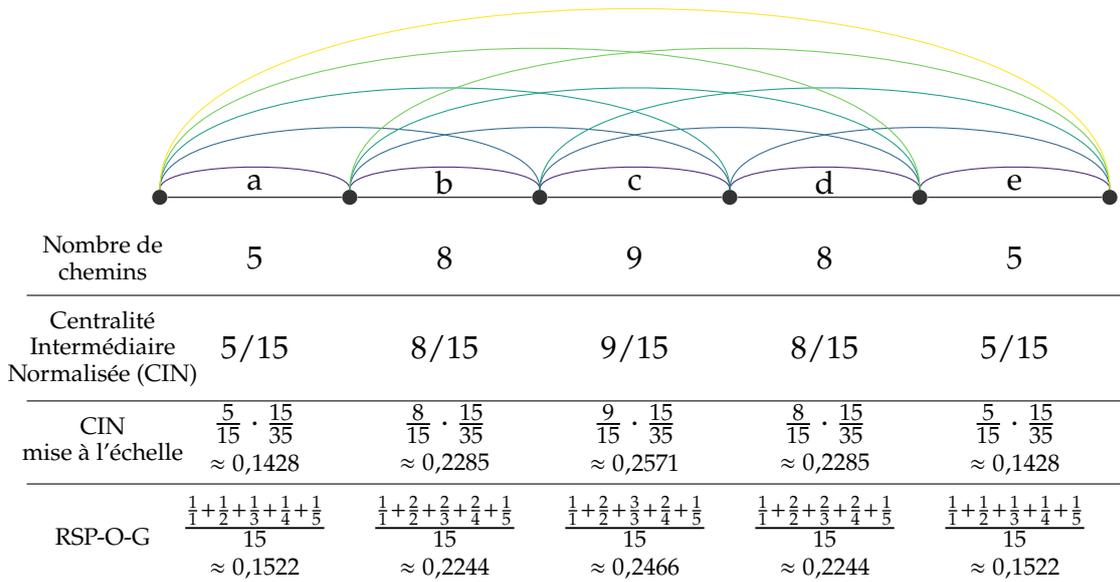


Figure 2.4. Comparaison de la centralité intermédiaire (Betweenness Centrality) avec notre distribution des stations.

On peut également tenter de comparer visuellement les deux objets avec des dégradés de couleurs. Sur la figure 2.5 (sous-figures a et b), la différence entre les deux distributions est à peine visible. Bien sûr, la visualisation effectuée également une mise à l'échelle pour accorder la palette de couleurs aux valeurs, ce qui accroît encore la ressemblance. La centralité intermédiaire n'est pas une distribution statistique sur l'espace (la somme des valeurs ne vaut pas 1). Son but est de qualifier le graphe sur lequel elle est calculée et pas la distribution de stations mobiles, mais la curiosité reste là et on peut se demander si l'usage de cette mesure dans des simulations de type MANET serait vraiment notable.

En termes de complexité calculatoire, les deux algorithmes sont comparables car même si les calculs pour la centralité sont plus simples, les deux approches nécessitent un calcul de plus court chemin entre toutes les paires. Cette étape domine la complexité algorithmique avec un cas au pire en  $O(n^3)$ .

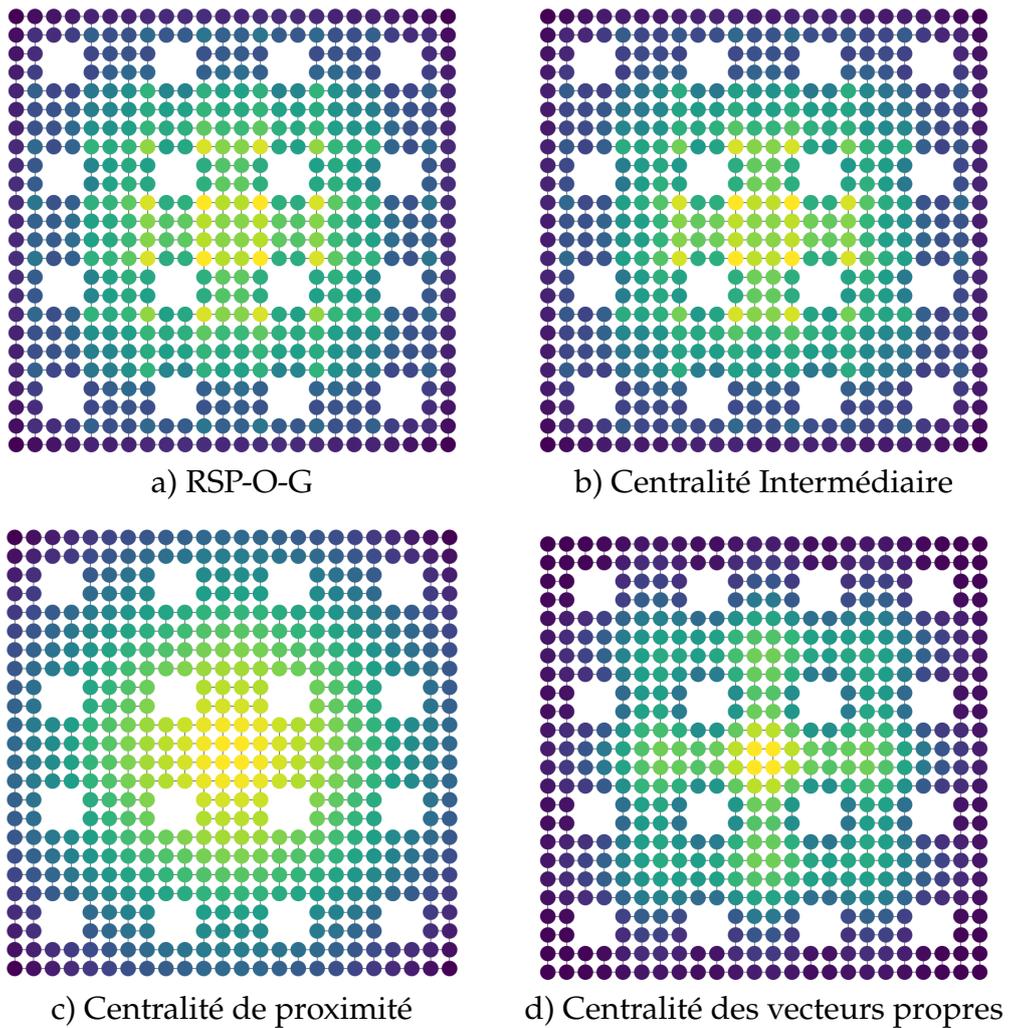


Figure 2.5. Comparaison de la distribution des stations avec des mesures de centralité classiques, sur le graphe de test utilisé dans l'exemple des figures 2.2 et 2.3.

## 3 *Mobilité humaine en épidémiologie*

Travaux concernés : [38]

L'objet de ce travail est l'étude de la diffusion d'un virus, le Chikungunya. Ce travail a pour point de départ un modèle analytique prenant en compte la dynamique du vecteur de la maladie (le moustique tigre) et la dynamique de la diffusion. Le but est d'étudier l'impact de la mobilité humaine mais aussi la mobilité du vecteur en intégrant les différentes mobilités dans le modèle équationnel. Ce travail plutôt novateur à l'époque a consisté à plonger le système dynamique à base de compartiments, dans un réseau représentant l'espace géographique d'étude. Les différentes mobilités étaient modélisées comme des liens du réseau. Le modèle de mobilité se servait d'études de données GSM pour établir des distributions de déplacement dans une population. Cette idée, réappliquée au réseau observé, a permis de reproduire assez fidèlement, de façon numérique, les événements épidémiques de l'île de La Réunion.

### 3.1 Contexte

Mon équipe de recherche, RI2C, a une histoire commune de collaboration scientifique avec les équipes de recherche du laboratoire de mathématiques appliquée de l'université du Havre, le LMAH. Le domaine général des systèmes complexes est souvent le point d'entrée à des projets communs, que ce soit pour l'encadrement d'étudiants (thèses établissement et CIFRE en co-encadrement et co-direction), l'animation de la recherche (institut des systèmes complexe), la vulgarisation scientifique (fête de la science) ou les projets de recherche (régionaux et ANR). Parmi les thèmes de recherche communs, celui du couplage de systèmes dynamiques suivant les liens d'un réseau est souvent d'actualité dans différents domaines d'application (épidémiologie, gestion des phénomènes de panique, synchronisation neuronale). Ce travail est un autre exemple de collaboration entre les deux laboratoires.

Dans sa thèse, Djamila Moulay s'intéressait à modéliser les mécanismes de transmission et de propagation du virus du Chikungunya CHIKV, à l'origine de la maladie infectieuse tropicale du même nom. La modélisation en question consistait en plusieurs systèmes d'équations différentielles ordinaires (EDO) à compartiments. Ces modèles furent étudiés sous plusieurs angles (validité des modèles, estimation des paramètres, réglage des paramètres par contrôle optimal), mais une dernière étape manquant à la thèse était la validation du modèle en le comparant aux données réelles existantes. Il s'agit de décomptes des nouvelles infections, des

rémissions et des décès journaliers concernant un épisode épidémique important, datant de 2005-2006, sur l'île de La Réunion.

Le point de vue global des systèmes dynamiques était un frein à la comparaison réaliste du phénomène réelle et une granularité plus fine du modèle pouvait être atteinte en modélisant le système dynamique sur les sommets d'un réseau dont les liens servaient à modéliser les déplacements humains. Ces liens définissaient un couplage entre les systèmes sur les nœuds. Grâce à cette représentation en réseau, un environnement réaliste, ainsi qu'un modèle de mobilité humaine réaliste pouvait être ajouté au modèle. Notre contribution a donc consisté à aider à la modélisation de cette version étendue du système et à proposer un modèle de mobilité humaine. Quelques lignes de code et de temps GPU<sup>a</sup> plus tard, un modèle de métapopulation capable de se comparer aux données réelles de l'évènement de 2005 était proposé.

## 3.2 Les métapopulations

Les modèles EDO présentés ici sont adaptés et formalisés à l'aide de la théorie de la métapopulation<sup>b</sup>. Celle-ci considère un réseau où les nœuds représentent des habitats réels de l'environnement pour les populations considérées (ici les humains et les moustiques). Dans chacun de ces nœuds, des modèles de transmission et de dynamique de population apparaissent et sont couplés avec des nœuds voisins. Les liens du réseau représentent à la fois le voisinage local d'un nœud et des nœuds plus éloignés qui définissent la mobilité des humains.

Nous nous concentrons sur un cas réel d'épidémie de chikungunya avec l'évènement de 2005-2006 survenu sur l'île de La Réunion, une île française dans l'océan Indien. L'île est modélisée avec un réseau. Puisque nous voulons que ce réseau reflète la densité de la population locale, nous considérons le réseau routier de l'île comme un proxy de la densité humaine, considérant que chaque intersection du réseau routier est un nœud du réseau de métapopulation. La figure 3.1 illustre le processus de création du réseau de métapopulation à partir du réseau routier extrait d'OpenStreetMap. Chaque intersection du réseau routier sert de nœud pour le réseau de métapopulation. Chaque nœud se voit affecter une quantité qui représente un nombre d'habitants. Cette information provient de l'INSEE qui fournit une répartition de la population à l'échelle du kilomètre carré. Les nœuds, dans chaque case de un km<sup>2</sup>, se répartissent la valeur de population donnée par l'INSEE.

La population de moustiques sur chaque nœud doit également être considérée. Il serait erroné de considérer une quantité arbitraire identique sur chaque nœud, sachant que ces nœuds possèdent une distribution dans l'espace qui n'est pas homogène, car reflétant la densité de population humaine et pas celle des moustiques.

---

<sup>a</sup>*Graphical Processing Unit*. Le code numérique pour ces expérimentations fut écrit en CUDA (*Compute Unified Device Architecture*) et exécuté sur des cartes graphiques dédiées au calcul numérique parallèle.

<sup>b</sup>[5] Julien Arino, Jonathan R. Davis, David Hartley, Richard Jordan, Joy M. Miller, and P. van den Driessche. 2005. A multi-species epidemic model with spatial dynamics. *Mathematical Medicine and Biology: A Journal of the IMA* 22, 2 (June 2005), 129–142. DOI : [10.1093/imammb/dqi003](https://doi.org/10.1093/imammb/dqi003)

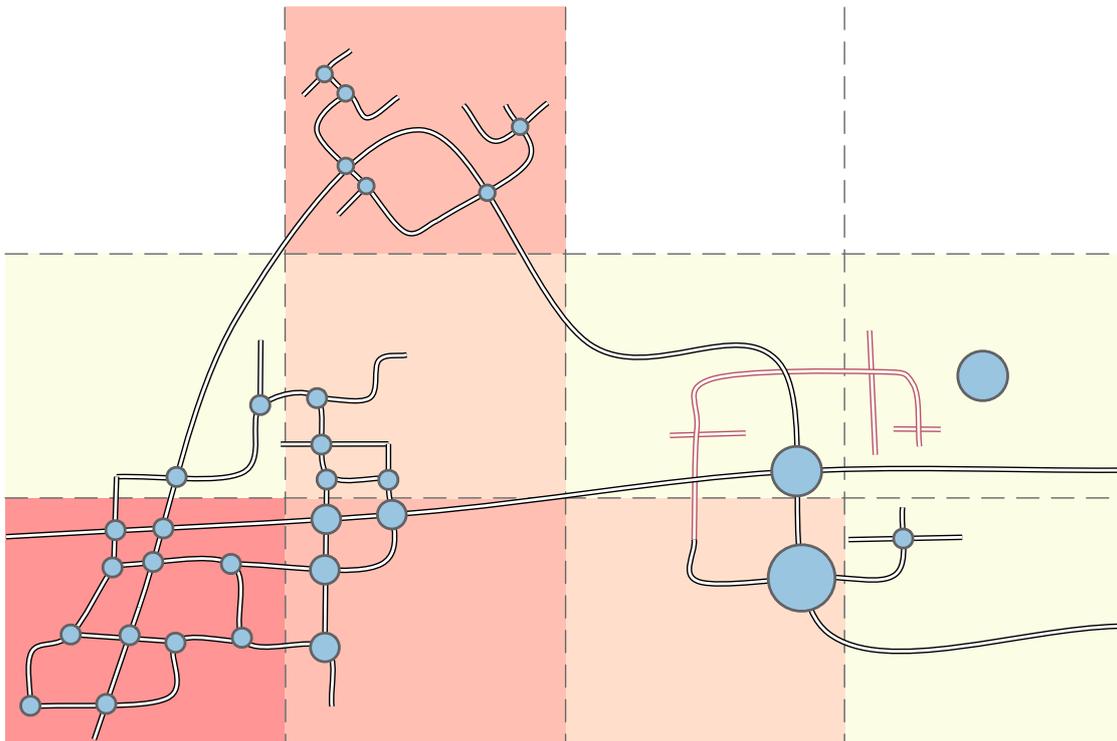


Figure 3.1. Construction du réseau de métapopulation à partir d'OpenStreetMap

Afin de rétablir une homogénéité dans la distribution des nœuds, un calcul de la surface occupée par chaque nœud est opéré à l'aide d'un diagramme de Voronoï. Cette valeur est ensuite majorée par une surface maximum, celle d'un disque de rayon  $d_{max}$  qui représente le rayon maximum d'interaction observé pour le type de moustiques considéré (*Aedes albopictus*). En effet ce modèle ne cherche pas à modéliser toute la population de moustiques, mais uniquement celle susceptible d'interagir avec l'humain. La figure 3.2 illustre la distribution des nœuds du réseau propre à la densité de population ainsi que la surface associée à chaque nœud, via le diagramme de Voronoï, pour répartir les moustiques. La figure 3.3 est un agrandissement de la précédente, centrée sur le quartier Saint-Jacques du chef-lieu Saint-Denis de La Réunion.

### 3.3 Modèles de mobilités

La modélisation de la mobilité humaine fait l'objet de nombreuses études. Depuis des décennies les études de mobilités à l'échelle d'agglomérations existent. Mais c'est depuis que les données de connectivité des téléphones mobiles sont apparues que les études les plus intéressantes ont eu lieu.

Dans un article de 2008<sup>c</sup>, González, Hidalgo et Barabási analysent de tels données provenant d'opérateurs téléphoniques pour proposer principalement deux

<sup>c</sup>[24] Marta C. González, César A. Hidalgo, and Albert-László Barabási. 2008. Understanding individual human mobility patterns. *Nature* 453, 7196 (June 2008), 779–782. DOI: [10.1038/nature06958](https://doi.org/10.1038/nature06958)

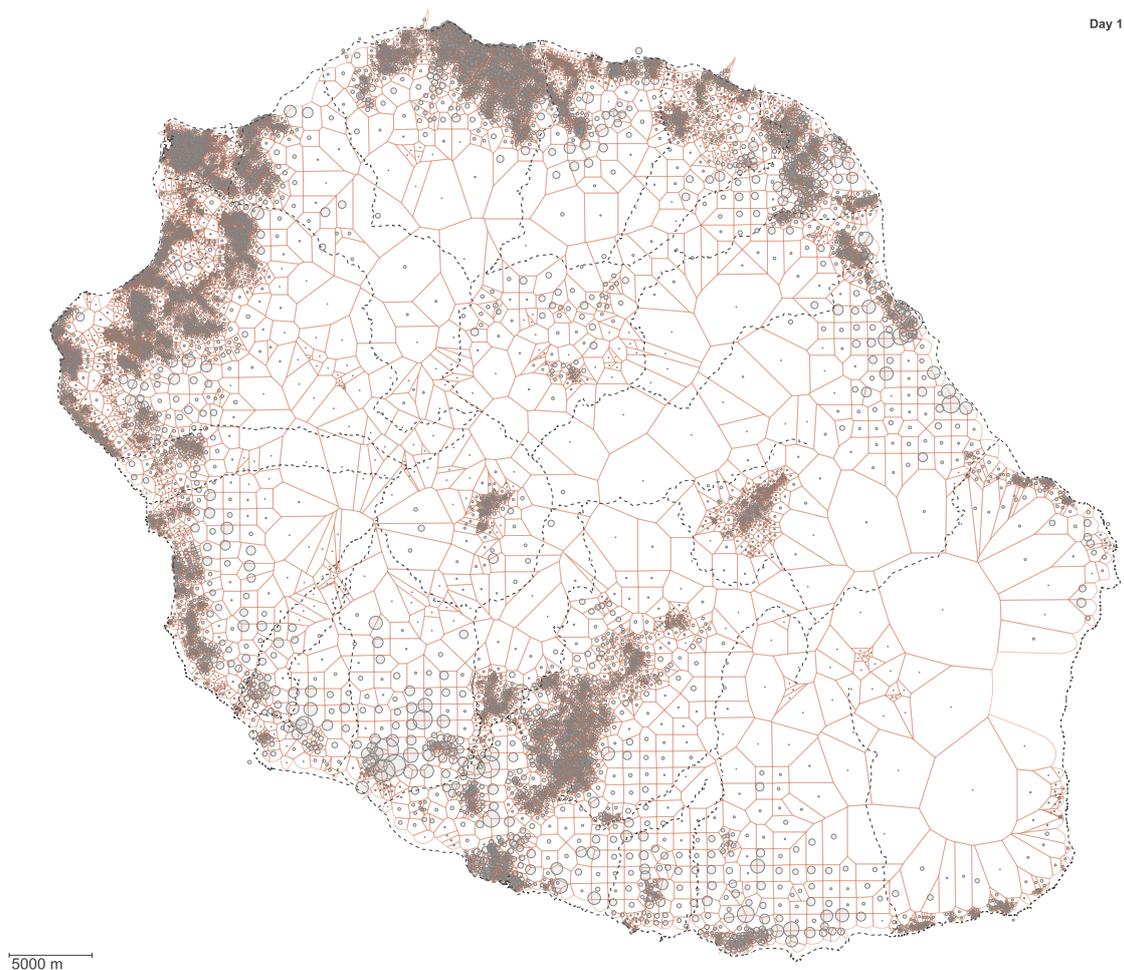


Figure 3.2. Ile de La Réunion. Répartition des populations humaine et de moustiques via un réseau dont les sommets (les points) sont distribués en fonction de la densité humaine et les surfaces associées (les polygones) pour estimer la population de moustiques.

résultats majeurs. Le premier est une estimation de la distribution des distances des déplacements (où rayon de giration centré autour du lieu de résidence). Ils montrent que cette distribution semble suivre une loi de puissance tronquée. La seconde contribution s'intéresse à l'importance des destinations. En classant les destinations favorites des usagers de téléphones mobiles par leur nombre d'apparitions, les auteurs montrent que la fréquence des visites suit une loi de Zipf. Pour aller plus loin, le *survey Human mobility: Models and applications*<sup>d</sup> donne un panorama des différentes études et techniques de modélisation de la mobilité humaine.

Nous nous appuyons sur ces distributions pour créer des schémas de mobilité humaine pour la population de notre modèle. En effet ces résultats nous permettent de tirer aléatoirement un nombre de destinations suivant les distributions données,

<sup>d</sup>[7] Hugo Barbosa, Marc Barthelemy, Gourab Ghoshal, Charlotte R. James, Maxime Lenormand, Thomas Louail, Ronaldo Menezes, José J. Ramasco, Filippo Simini, and Marcello Tomasini. 2018. Human mobility: Models and applications. *Physics Reports* 734, (March 2018), 1–74. DOI: [10.1016/j.physrep.2018.01.001](https://doi.org/10.1016/j.physrep.2018.01.001)



Figure 3.3. Agrandissement de la fig 3.2 centré sur le quartier Saint-Jacques du chef-lieu Saint-Denis de La Réunion. Le réseau routier est également représenté.

avec une distance également donnée par les distributions du modèle. Nous supposons que les individus ne changent d'état de santé que lorsqu'ils sont sur un nœud et non pendant les déplacements.

Dans le modèle équationnel de métapopulation, la mobilité humaine est donnée par une matrice de destinations pour chaque nœud. Elle se représente également sous forme de liens du réseau illustré précédemment. Chaque nœud se verra donc affecter un ensemble de liens de destinations dont le nombre et la distance suivent les distributions discutées. La figure 3.4 illustre les liens de mobilité pour quelques nœuds du graphe représentant l'île de La Réunion.

La mobilité des moustiques aussi est représentée. Plus simplement, un voisinage géographique est défini avec une longueur des liens majorée par la limite  $d_{max}$ , définie précédemment comme la limite d'interaction pour un moustique au long de sa vie. Seuls les nœuds distants de moins de  $d_{max}$  sont connectés et permettent la mobilité des moustiques. Ceux-ci n'empruntent pas les liens de mobilité définis pour les humains.

### 3.4 Résultats

Les résultats montrent l'influence déterminante des mobilités sur la propagation de la maladie. Non seulement la mobilité humaine mais aussi l'interaction vectorielle locale (les moustiques) jouent un rôle important à l'échelle considérée. Les résultats sont ensuite comparés à des données épidémiques réelles concernant l'évènement 2005-2006 à l'échelle de toute l'île de la Réunion et le modèle est validé, comme le

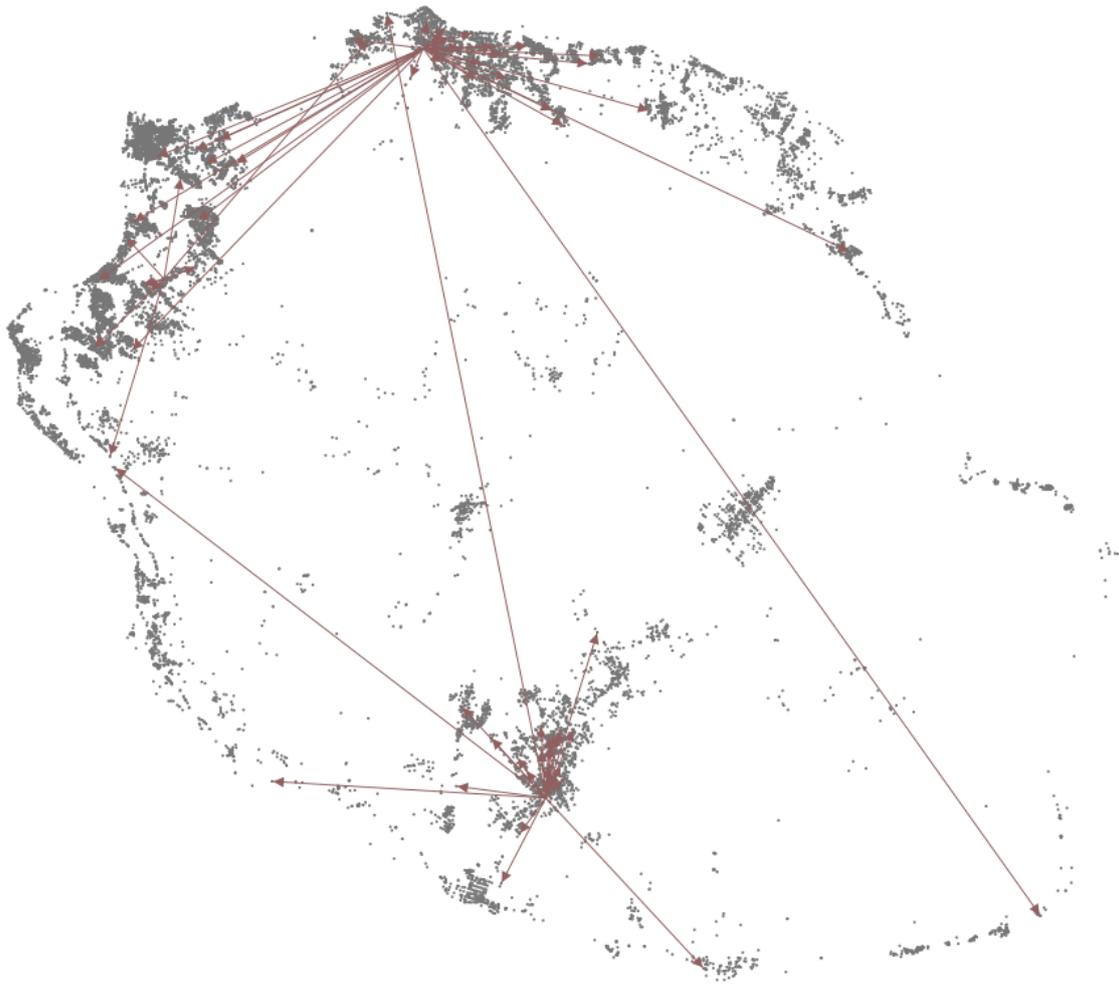


Figure 3.4. Destinations pour le modèle de mobilité humaine du réseau de métapopulation de l'île de la Réunion. Pas soucis de visibilité, seules les destinations d'un échantillon de nœuds est représenté.

montre la figure 3.5.

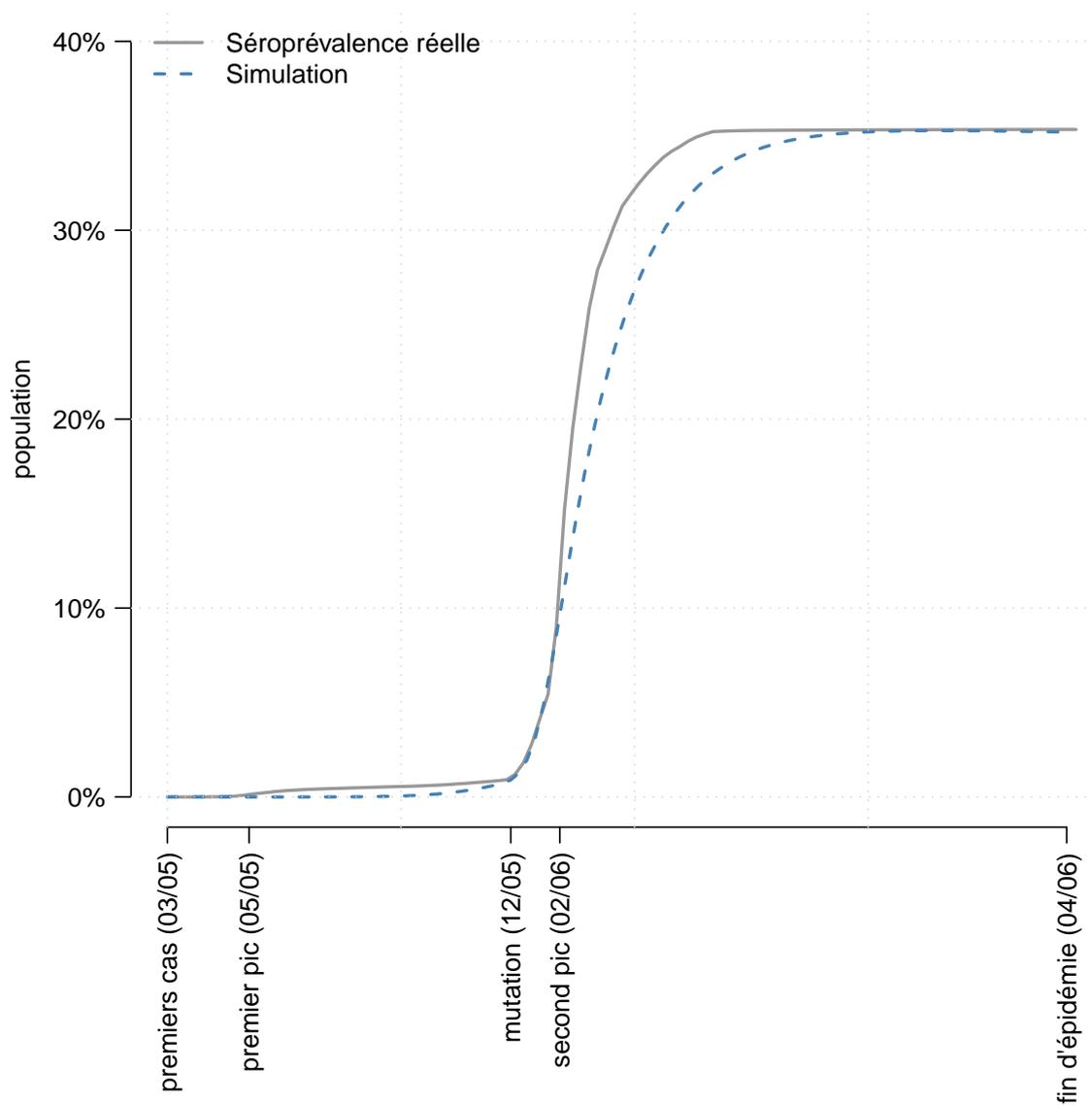


Figure 3.5. Validation du modèles (nombre total de cas infectés) avec les données réelle de l'épisode de La Réunion.



## **partie II**

# **Analyse de réseaux dynamiques**



# *Introduction*

L'analyse de réseaux complexes est une discipline très largement traitée en science des données. Les réseaux considérés sont souvent traités de façon statique même s'ils évoluent naturellement dans le temps. L'analyse s'effectue sur des instantanés du réseau. Les métriques utilisées sont donc propres aux graphes statiques et si elles apportent une compression de la structure du réseau, son évolution, sa dynamique, échappent à ces mesures. Nous sommes convaincus de la nécessité de concevoir des métriques permettant de qualifier la dynamique, au-delà d'évidentes séries temporelles montrant l'évolution des métriques statiques dans le temps.



## 4 *Analyse du cycle de vie*

Travaux concernés : [39] pour l'analyse statique et [45,50,51] pour les aspects dynamiques.

L'analyse du cycle de vie est une discipline proposant de caractériser et de quantifier les conséquences environnementales de l'activité, tout au long de sa vie, d'un système (construction, production, service, procédé). Cette analyse permet, par exemple, d'établir la quantité totale d'énergie électrique nécessaire à la production, au transport, à l'utilisation et enfin, le traitement en fin de vie, de l'ordinateur sur lequel je suis en train d'écrire ces lignes.

Les verrous scientifiques sont souvent liés à des problématiques de résolution de systèmes équationnels linéaires. Ces systèmes matriciels peuvent aussi être vu comme des matrices d'adjacence représentant des graphes que l'on peut tenter d'étudier avec les outils à notre disposition.

### 4.1 Contexte

J'ai tout d'abord été contacté par l'équipe d'Enrico Benetto du LIST<sup>a</sup>, en 2012, pour apporter une expertise et une aide technique pour la réalisation d'un logiciel de calcul de différentes mesures, sur la base d'inventaires de cycles de vie. L'Énergie (avec un  $m$ ) est une unité de mesure qui permet de quantifier globalement l'énergie consommées ou produite par des procédés. Cette unité générique est exprimée en « *solar energy joule* » (sej). Cette technique permet d'exprimer avec une seule unité des productions et des échanges, d'ordinaires quantifiés avec différentes unités. La réalisation de ce logiciel a donné lieu à un dépôt de nom (SCALEM®) et au partage de la propriété intellectuelle associé entre les partenaires du projet.

Ce projet et les résultats générés, ont permis le premier travail d'analyse de réseau [39]. L'analyse est restée statique dans ce travail. Or il est rapidement apparu que la dépendance temporelle entre les procédés devait être prise en compte dans un modèle intégrant la composante temporelle. La notion de réseau temporel devenait centrale dans ce projet. Ce fut principalement l'objet du projet ANR DyPLCA impliquant la même équipe du LIST, l'INSA de Toulouse et le LITIS, de 2014 à 2018. De ce projet, les travaux cités précédemment ([45,50,51]) furent produits. La collaboration s'est arrêté après le projet ANR, principalement à cause de ma difficulté à justifier une collaboration dans un domaine d'application différent de l'informatique, nonobstant l'intérêt de modélisation sous forme de graphes dynamiques que représentent les réseaux d'analyse du cycle de vie.

---

<sup>a</sup>Luxembourg Institute of Science and Technology

## 4.2 Cycle de vie et réseaux

Si un produit  $A$  nécessite l'intervention des produits  $B$  et  $C$  pour être créé, ces mêmes produits nécessitent, à leur tour, d'autres produits pour leur propre création. Cette relation de production constitue un réseau d'autant plus dense que les mêmes procédés se retrouvent rapidement dans les dépendances de la majorité des produits. En effet, tous les procédés industriels, ou presque, ont besoin d'électricité, de métal, d'eau, de transport, etc. On imagine aisément le réseau de dépendance que forment ces produits ainsi que les boucles d'interdépendances (e.g. : l'électricité dans l'usine de métallurgie, le métal dans la centrale électrique).

Ce genre de réseau est utile aux chercheurs en sciences de l'environnement pour calculer des impacts environnementaux puisque chaque production peut être accompagnée d'émissions dans l'environnement. Un parcours dans ce graphe permet, par exemple, de retrouver la quantité de  $\text{CO}_2$  globalement émise pour la production d'un produit donné.

Techniquement les inventaires considérés sont consignés dans une matrice  $A$  où chaque ligne représente un procédé ou produit et où chaque colonne représente les procédés ou produits nécessaires à la création des procédés identifiés en ligne. Cette matrice est pondérée et chaque colonne peut avoir sa propre unité (par exemple, la production d'un litre d'eau douce, par évaporation de l'eau de mer, nécessite *des* kW d'électricité). La matrice peut également être exprimée en utilisant une seule unité, globale pour tous les procédés, comme c'est le cas pour l'énergie.

Il est possible de trouver, dans cette matrice  $A$ , l'inventaire des procédés nécessaires à la production d'un produit donné. La solution à ce problème se trouve dans la résolution d'un système linéaire du type  $A \cdot s = f$ , où  $A$  est donc la matrice de tous les flots possibles entre tous les procédés (on l'appelle aussi matrice technosphère); le vecteur  $f$  est dit le vecteur demande, il représente ce que l'on cherche à produire. Typiquement c'est un vecteur contenant des zéros à tous les indices, sauf à l'indice correspondant au procédé que l'on cherche à observer, où sa valeur vaut 1. Finalement  $s$  est le vecteur solution. Il contient la distribution et la quantité des procédés nécessaires à la production de produit identifié dans  $f$ .

Le vecteur  $s$  permet également d'identifier les impacts environnementaux associés à la production étudiée. En effet une seconde matrice  $B$  décrivant les interventions environnementales de tous les procédés, existe souvent et permet de connaître, toujours via la résolution d'un système linéaire  $B \cdot s = g$ , la liste et la quantité des émissions  $g$  dans l'environnement, pour le procédé considéré. C'est dans ce dernier vecteur  $g$  que l'on trouvera, par exemple, la quantité totale de  $\text{CO}_2$  globalement émise pour la production du produit considéré.

Le calcul matriciel décrit précédemment permet d'obtenir des quantités de matière globales pour chaque procédé. Or, chaque procédé peut intervenir plusieurs fois pour produire un seul produit (par exemple l'électricité ou le transport sont nécessaires à presque tous les procédés). Pour mettre en évidence ces multiples accès, une approche différente est nécessaire. La matrice technosphère  $A$  est carrée et peut être interprétée comme une matrice d'adjacence et donc comme un graphe. Un parcours dans ce graphe, partant du procédé étudié et suivant les liens de dépendances entre les procédés, permet d'établir les différents chemins liant les procédés. Ce parcours s'effectue en remontant les liens de production, c'est à

dire, en inversant les arcs du réseau. Lors de ce parcours les ratios de quantités nécessaires pour chaque procédé sont propagés. On peut ainsi parcourir le réseau et rencontrer plusieurs fois un même procédé, pour des demandes différentes. La figure 4.1 illustre le cas d'un procédé,  $P_1$ , nécessitant l'intervention de deux autres procédés,  $P_2$  et  $P_3$ , eux même dépendants du procédé  $P_4$ . Ce dernier est donc sollicité deux fois (une fois via  $P_2$  et une fois via  $P_3$ ) avec deux quantités différentes.

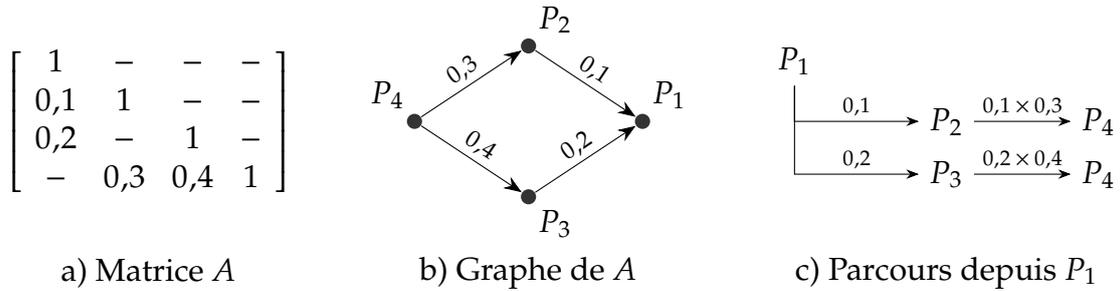


Figure 4.1. Exemple de matrice technosphère interprétée comme un graphe parcouru (dans le sens inverse des arcs) pour découvrir les chemins de production.

Le parcours de graphe décrit ici peut s'apparenter à un parcours classique (largeur d'abord ou profondeur d'abord) à l'exception du fait que les sommets déjà visités peuvent l'être à nouveau. Sans condition d'arrêt, il est possible qu'un tel parcours soit infini. La figure 4.2 illustre un réseau de production contenant une boucle d'interdépendance entre procédés qui pourrait provoquer un parcours infini, sans compter la condition d'arrêt de l'algorithme de parcours. En effet le parcours propage des quantités de ressources (des demandes de production) qui diminuent nécessairement tout au long du parcours. En définissant un seuil, le parcours peut s'arrêter quand la quantité demandée pour un procédé devient inférieure au seuil établi.

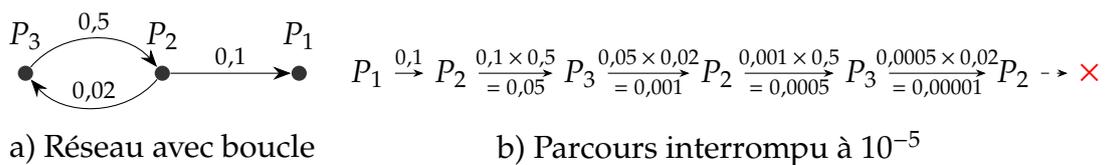


Figure 4.2. a) Exemple d'un réseau de production avec une interdépendance. Les arcs vont dans le sens de production. b) Le réseau crée une boucle dans le parcours. Les flèches indiquent le sens de parcours (inversé par rapport aux arcs du réseau). Le seuil à 0,00001 permet d'arrêter le parcours (X).

### 4.3 Analyse statique

Les graphes générés avec la technique de parcours présentée ci-dessus peuvent être analysés. C'est l'objet de l'article [39] qui étudie un tel réseau, de manière statique.

En plus des valeurs d'énergie pour chaque procédé, le nombre de fois où chaque sommet est visité par l'algorithme de parcours est également analysé. Le nombre de chemins ainsi que les valeurs d'énergie mettent en évidence les procédés concernés par l'étude.

L'étude analysée concerne la production d'eau potable à partir de pompage dans les nappes, à destination d'un réseau de distribution. Les procédés mis en valeur par le décompte du nombre de chemins sont ceux liés à la production de l'eau potable (l'eau elle-même, le sable de filtrage, le chlore, *etc.*) tandis qu'une analyse de la structure du réseau, sans pondération, met en évidence d'autres procédés. En effet les nœuds avec les plus forts degrés, tout comme les nœuds avec les valeurs de centralité intermédiaire les plus fortes, représentent les procédés liés au transport, à l'électricité, à l'acier et aux pesticides. Cette observation peut également être faite avec d'autres analyses. Ainsi l'étude de la production de pellets de bois, ou encore la production agricole d'orge, montrent également une forte centralité des mêmes processus, quand on s'intéresse uniquement à la structure du réseau. Les distributions d'énergie sont d'ailleurs très peu corrélées aux distributions de degrés ou de centralité. Le coefficient de corrélation de Pearson entre les valeurs d'énergie et le degré des nœuds est très proche de zéro ( $-0,003$ ), de même pour le nombre de chemins et le degré ( $0,16$ ), alors que le même coefficient entre le nombre de chemins et l'énergie est élevé ( $0,87$ ) et que la corrélation entre le degré et la centralité intermédiaire (*betweenness centrality*) est également élevé ( $0,88$ ).

La figure 4.3 donne une représentation graphique spatiale du réseau, basée sur un algorithme de force. On y constate que le procédé étudié (Distributed water), qui dispose du plus grand nombre de chemins (par construction), se trouve en périphérie du réseau au centre duquel se trouvent les procédés liés au transport et à l'énergie.

Outre ces observations, ce travail propose une analyse classique de réseau statique. La présence de *hubs* est observée, ainsi qu'une distribution des degrés en loi de puissance. Il n'est pour autant pas possible d'établir la propriété « petit monde » pour ce réseau.

## 4.4 Analyse dynamique

Nous nous sommes ensuite intéressés à la dépendance temporelle entre ces procédés : les temps de fabrication, les temps de transport, les productions saisonnières, *etc.* En pondérant les interactions entre procédés avec des durées, on introduit une notion de parcours respectant des durées dans le réseau. L'impact environnemental n'est plus une simple valeur numérique, mais une série temporelle qui retrace les émissions générées par les différents intervenants pour permettre la production du produit observé.

La figure 4.4 montre, à partir de l'exemple de la figure 4.2, les dates de départs de production qui peuvent être calculées de la même façon que précédemment, en

---

<sup>b</sup>[28] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *J. Stat. Mech.* 2008, 10 (October 2008), P10008. DOI : [10.1088/1742-5468/2008/10/P10008](https://doi.org/10.1088/1742-5468/2008/10/P10008)

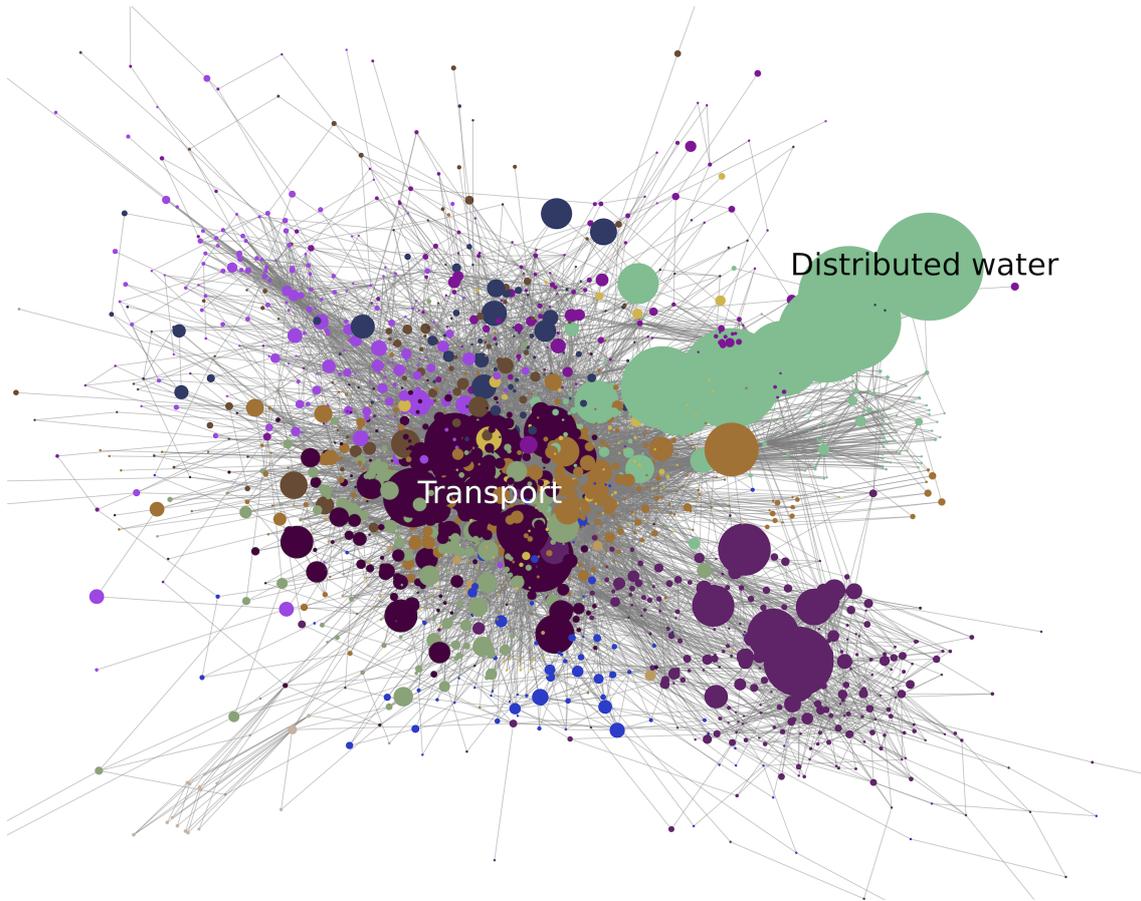


Figure 4.3. Réseau de l'étude de production d'eau potable. Placement des nœuds par algorithme de force (Fruchterman-Reingold). Taille des nœuds proportionnelle au nombre de chemins passant sur le nœud. L'étiquette noir (Distributed Water) indique le nœud avec le plus grand nombre de chemins (plus de 16 millions) et la plus forte énergie (34706545 MseJ), c'est le procédé étudié. L'étiquette blanche (Transport) indique le nœud avec le plus fort degré (282). Les couleurs sont le résultat d'une recherche de clusters par la méthode de Louvain<sup>b</sup>. La meilleure modularité trouvée est 0,52 pour 77 clusters.

remontant le réseau de production. Là aussi les boucles introduisent un parcours infini, que l'on limite avec une date maximale autorisée dans le passé.

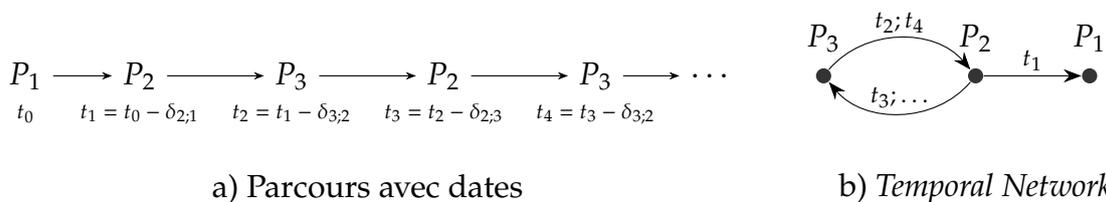


Figure 4.4. a) Parcours calculant les dates de début de production. b) Réseau pondéré par des estampilles de temps.

Cette représentation est similaire à la représentation des *Temporal Networks*

popularisés par Petter Holm et Jara Saramäki<sup>c</sup>

Ce travail fut l'objet de l'ANR DyPLCA dans laquelle je représentais l'université du Havre. J'ai été beaucoup occupé par ce projet, par les aspects techniques, puisque toutes les idées du projet ont fait l'objet de développements logiciels sous forme d'application Web et de service Web. Ce travail est validé dans plusieurs publications de bon niveau en science de l'environnement. La comparaison avec les *Temporal Networks* n'est pas allée plus loin que ce qui est présenté là.

---

<sup>c</sup>[28] Petter Holme and Jari Saramäki. 2012. Temporal networks. *Physics Reports* 519, 3 (October 2012), 97–125. DOI : [10.1016/j.physrep.2012.03.001](https://doi.org/10.1016/j.physrep.2012.03.001)

## 5 Analyse des réseaux maritimes

Travaux concernés : [26,27]

« – Il a perdu, messieurs, reprit Andrew Stuart, il a cent fois perdu ! Vous savez, d’ailleurs, que le *China* – le seul paquebot de New York qu’il pût prendre pour venir à Liverpool en temps utile – est arrivé hier. Or, voici la liste des passagers, publiée par la *Shipping Gazette*, et le nom de Phileas Fogg n’y figure pas. »  
Jules Verne, *Le Tour du monde en quatre-vingts jours* (1870)

L’une des particularités du Havre et de son front de mer, est qu’en s’y baladant, on est certain de croiser, parfois de près, l’un de ces géants des mers de quatre cents mètres de long, plus haut que la majorité des immeubles de la ville, rempli de colonnes de conteneurs. Le commerce international, pour le transport de conteneurs, est l’une des activités principales de la ville. Elle en modèle le tissu social, économique et géographique. Ces boîtes contiennent une bonne partie des produits de consommation importés dans le pays. Les trajets des navires, les ports qu’ils visitent, les routes qu’ils empruntent, sont autant d’informations qui constituent le réseau maritime mondial. Étudier ce réseau, en connaître les propriétés ainsi que les évolutions, constitue le travail de nombreux chercheurs, dans plusieurs disciplines, dont l’informatique.

Le premier écueil dans ce travail est la reconstitution du réseau à partir de données réelles. En effet l’accès aux données de départs et d’arrivées des navires dans les ports n’est pas aussi globalement disponible que peut l’être le suivi de l’aviation commerciale<sup>a</sup>. Connaître, dans tous les ports la liste des arrivées et des départs de navires est possible mais n’est pas automatique. Certains opérateurs comme les assureurs possèdent ces informations. L’utilisation de données automatiques AIS<sup>b</sup> est une autre piste très utilisée.

---

<sup>a</sup>ADS-B (*Automatic dependent surveillance-broadcast*) est un protocole numérique reposant sur une technologie radio déjà implantée sur tous les avions commerciaux. Il permet un suivi précis du trafic aérien commercial.

<sup>b</sup>AIS (*Automatic Identification System*) est un protocole de communication pour le suivi automatique des navires. Il est obligatoire pour les navires dont la jauge brut dépasse 500 GT (environ 6550 m<sup>3</sup>) et pour les navires de pêche de plus de 24 m de long (dans l’Union Européenne).

## 5.1 Contexte

En 2015 nous avons eu l’opportunité de collaborer avec César Ducruet, géographe et chercheur CNRS. César est spécialisé dans l’étude des villes portuaires et dans l’analyse du réseau de transport maritime.

Afin de constituer ce réseau, César n’a pas eu recours à la *Shipping Gazette*, comme Andrew Stuart, mais à la *Lloyd’s List*, une revue proposant des informations sur les arrivées de navires dans les ports du monde depuis le XVIII<sup>e</sup> siècle. Il en a tiré plusieurs analyses, d’abord locales, avec le cas de la Corée du Sud<sup>c</sup>, puis à l’échelle globale<sup>d</sup>, en augmentant la période d’observation.

César nous a proposée, avec Frédéric Guinand, de tenter d’apporter une analyse orientée vers la dynamique et les aspects temporels. Nous avons d’abord travaillé sur deux années complètes de données, puis sur une plus large période (40 ans), avec quelques mois de données par année. Notre contribution fut certes modeste mais très instructive.

J’ai ensuite voulu poursuivre les investigations dans ce réseau maritime en tentant de contourner le problème de l’accès aux données avec une source gratuite et plus pérenne d’information. C’est l’objectif que s’est fixé mon collègue Claude Duvallet, en installant une antenne AIS<sup>e</sup> sur le toit d’un des bâtiments de l’université, afin d’adhérer au réseau de partage de données AISHUB<sup>f</sup> et ainsi récupérer le flux des données partagées, avec potentiellement une couverture mondiale. J’ai donc investi mon temps dans ce projet de récupération de données en temps réel et de reconstruction du réseau macroscopique à partir de données on ne peut plus microscopiques.

## 5.2 Modélisation

Les données de la *Lloyd’s List* montrent les départs et les arrivées de navires pour tous les ports du monde. Le format est celui d’une base de données à trois tables. Une pour identifier précisément les ports, une pour obtenir des détails sur les navires et la plus importante, la table des mouvements qui, pour chaque enregistrement, indique l’identifiant d’un navire, la date, l’identifiant du port et le type de mouvement (arrivée ou départ). La structure du second jeu de données (les quarante années) est sensiblement la même. On peut construire plusieurs réseaux dynamiques à partir de ces informations.

---

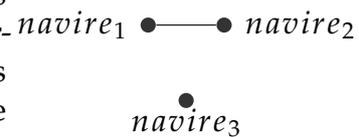
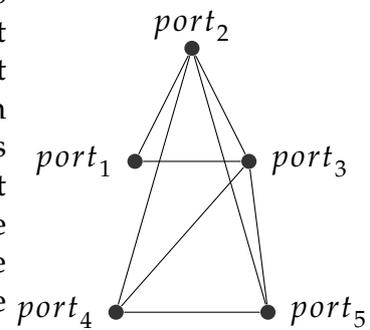
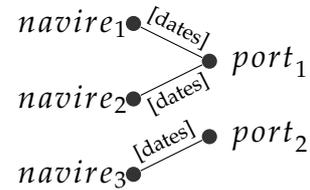
<sup>c</sup>[21] Antoine Fremont and Cesar Ducruet. 2005. THE EMERGENCE OF A MEGA-PORT - FROM THE GLOBAL TO THE LOCAL, THE CASE OF BUSAN\*. *Tijdschrift voor Econ & Soc Geog* 96, 4 (September 2005), 421–432. DOI : [10.1111/j.1467-9663.2005.00473.x](https://doi.org/10.1111/j.1467-9663.2005.00473.x)

<sup>d</sup>[20] César Ducruet. 2017. Multilayer dynamics of complex spatial networks: The case of global maritime flows (1977–2008). *Journal of Transport Geography* 60, (April 2017), 47–58. DOI : [10.1016/j.jtrangeo.2017.02.007](https://doi.org/10.1016/j.jtrangeo.2017.02.007)

<sup>e</sup>AIS (*Automatic Identification System*) est un protocole de communication pour le suivi automatique des navires. Il est obligatoire pour les navires dont la jauge brut dépasse 500 GT (environ 6550 m<sup>3</sup>) et pour les navires de pêche de plus de 24 m de long (dans l’Union Européenne).

<sup>f</sup><https://www.aishub.net>

- D'abord, et pour conserver un maximum d'information, on peut construire un réseau bipartite, avec un sous-ensemble de nœuds, représentant les ports et un autre sous-ensemble, représentant les navires. Les liens n'existent qu'entre navires et ports et représentent les visites des uns dans les autres. On peut pondérer les arcs avec une liste de dates correspondant à ces visites. L'intérêt de ce genre de réseau est de mettre en évidence les affinités de navires avec les ports, où de leurs armateurs avec certains pays, ou régions du monde.
- On peut, et c'est probablement la représentation la plus naturelle et la plus utile, représenter les ports comme des nœuds et les liaisons maritimes (trajets d'un port à l'autre par un navire) comme des arêtes. Ces arêtes peuvent être orientées (ce sont alors des arcs) et datées. Dans une variante on peut relier l'ensemble des ports constituant le trajet d'un navire plutôt que de n'avoir qu'un lien entre les escales  $n$  et  $n + 1$ . Les navires réalisent, la plupart du temps, des trajets prédéfinis et cycliques avec de multiples escales. Les pondérations sur ce genre de réseau peuvent toujours être des dates, ou tout autre information relative aux navires (type de chargement, tonnage, etc.). Ce réseau permet toutes sortes d'analyses structurelles, sur l'importance relatives et l'évolution des ports.
- Enfin, on peut vouloir n'observer que les navires (représentés par les nœuds) et les relier quand ils partagent un port. L'intérêt de cette approche est moins évident. Elle ferait probablement apparaître, comme dans la version bipartite, les différents armateurs.



### 5.3 Analyse statique

Nous avons considéré le réseau avec le point de vue des ports. Nous avons, dans un premier temps, mené une analyse statique en agrégeant le réseau sur une année entière. Une arête, dans le réseau, signifiait qu'au moins un navire avait fait la liaison, entre les deux ports concernés, dans l'année. Nous avons montré dans [27] que le réseau possédait la propriété « petit monde » et qu'il répondait à une distribution des degrés en loi de puissance. Plusieurs lois sont testées pour représenter cette distribution. L'exposant des lois est estimé avec la méthode de maximum de vraisemblance (*MLE*) et le seuil de comparaison (la valeur minimale de  $x$  pour laquelle on commence à comparer les lois avec les données) est donné par la distance de Kolmogorov-Smirnov. La figure 5.1 montre la comparaison. Un

autre test (du rapport de vraisemblance<sup>§</sup>) permet de confirmer qu'il existe bien une loi de puissance pour ces données, ce qui permet de qualifier le graphe d'invariant d'échelle. Cependant une distribution log-normal semble être un meilleur candidat pour corroborer la distribution des degrés du réseau.

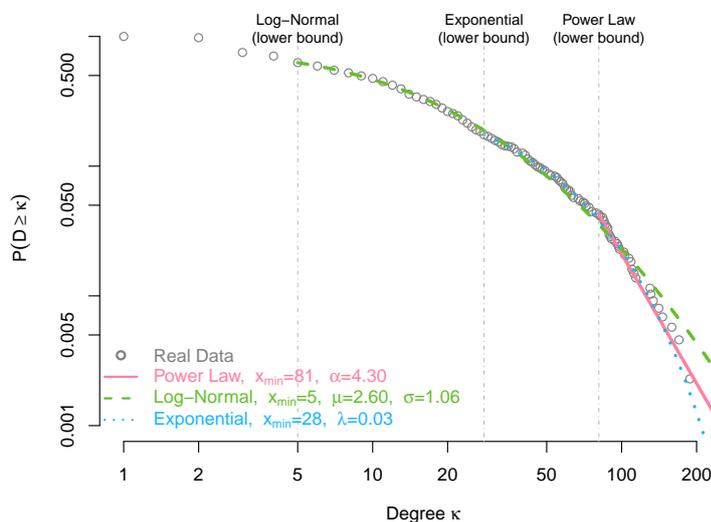


Figure 5.1. Fonction de répartition de la distribution de degrés du réseau maritime agrégé (année 1996) estimée avec plusieurs lois aléatoires.

## 5.4 Analyse dynamique

Nous avons ensuite tenté de limiter l'effet d'agrégation du réseau en considérant des fenêtres de temps glissantes et en analysant les réseaux générés par cette fenêtre. On obtient principalement des résultats sous forme de séries temporelles qui décrivent l'évolution de métriques statiques dans le temps. Cette approche permet d'observer des événements plus ponctuels.

Ces séries peuvent par exemple faire apparaître des particularités atypiques comme le montre la figure 5.2 a) où l'on voit un déficit des départs de navires les dimanches. Ce phénomène s'explique t'il par le fait que le dimanche est majoritairement un jour chômé dans le monde, y compris pour les autorités portuaires ou pour les marins ? Une autre explication est un biais introduit lors de la saisie des données. On peut faire une comparaison, quelque peu sinistre, avec les données concernant les décès hospitaliers dus à la pandémie de Covid. On y retrouve cet effet « dimanche » qu'on sait être un biais dans le protocole de remontée des informations.

Dans un second temps, nous nous sommes intéressés à la faisabilité des trajets aller-retour entre les différents ports, tout en respectant les temps de trajets. Ici les trajets de tous les navires enregistrés constituent des dates de départ possibles. On peut comparer cette approche au routage des paquets sur un réseau informatique,

<sup>§</sup>[14] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (November 2009), 661–703. DOI: [10.1137/070710111](https://doi.org/10.1137/070710111)

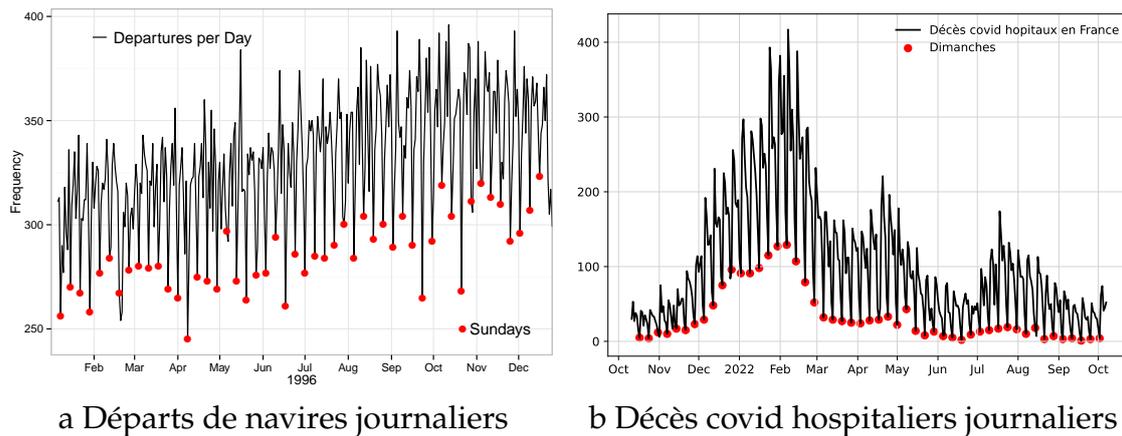


Figure 5.2. a) Périodicité hebdomadaire observée dans les données de départs de navires dans le monde entier en 1996 et comparaison avec b) Les déclarations de décès dus à la Covid dans les hôpitaux français.

où le paquet est un conteneur, ou une marchandise, les routeurs sont les ports et les câbles sont les routes maritimes. Les trajets respectent les temps de trajets imposés par la vitesse des navires. De plus une carence d'une journée minimum est appliquée sur les ports pour prendre en compte les temps de transbordements et autres impondérables. Si les nœuds sont les ports et si les liens sont datés par les départs des navires et pondérés par les temps de traversés alors on dispose d'un réseau de type *temporal network* pour lequel on sait calculer des plus courts chemins respectant les délais<sup>h</sup>. Les trajets au plus tôt (départ le plus tôt possible) sont considérés et comparés à des plus courts chemins classiques, sans contraintes de précedence, uniquement pondérés par les temps de trajets.

On appelle horizon, le nombre moyen de destinations accessibles, aller/retour, en respectant les délais, pour un port donné. L'horizon est d'autant plus petit pour un port que les contraintes temporelles sont fortes (peu de départs, durées de trajets importantes).

On découvre une forte disparité dans les horizons des ports, quand les temps de trajets sont respectés. La figure 5.3 illustre cette disparité ainsi que la différence entre l'approche atemporelle (pondération sans respecter les dates) et l'approche temporelle.

En moyenne l'horizon atemporelle est de 820 nœuds, ce qui correspond presque à la taille de la composante géante du réseau (835), alors qu'en respectant les dates, l'horizon n'est en moyenne que de 460, avec un fort écart type ( $\pm 130$ ). Il en ressort une grande disparité de l'accès au réseau pour certains ports.

<sup>h</sup>[10] B. Bui Xuan, A. Ferreira, and A. Jarry. 2003. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science* 14, 02 (April 2003), 267–285. DOI: [10.1142/S0129054103001728](https://doi.org/10.1142/S0129054103001728)

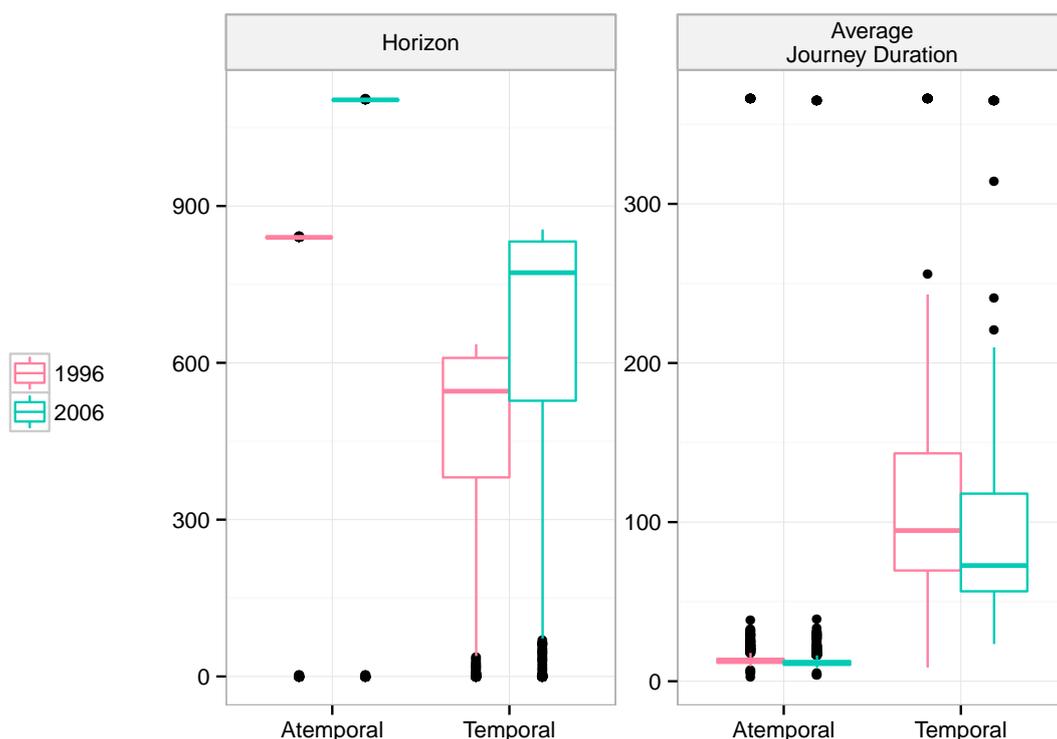


Figure 5.3. À gauche, horizon (nombre de ports accessibles aller/retour) en moyenne pour chaque port, en respectant les trajets et en les ignorant. À droite, temps de trajets aller/retour en moyenne pour chaque port, vers toutes les destinations (versions avec et sans respect des départs). Sur les chandelles, les 3 barres horizontales indiquent la médiane, le premier quartile (Q1) et le troisième quartile (Q3). Les lignes verticales couvrent un espace de  $Q1 - 1,5IQR$  à  $Q3 + 1,5IQR$ , avec  $IQR$  l'espace inter quartile (espace entre Q1 et Q3). Les points extérieurs sont en dehors de cet intervalle.

## 5.5 La suite

À l'époque, ce travail s'est arrêté là et la réflexion sur la disponibilité des données m'a amené à m'intéresser aux sources de données AIS, plus disponibles et plus normalisées. En réalité, le travail de reconstitution du réseau est également très important pour l'AIS. De plus les zones d'ombre (manques de couverture) sont problématiques à l'approche.

À l'heure de la rédaction de ces lignes, je suis impliqué dans un projet de reconstitution du réseau à partir de telles données. Les résultats sont toujours attendus afin de valider l'observation de plusieurs phénomènes (Brexit, nouvelles routes de la soie, pandémie de covid, etc.).

Mes travaux autour des réseaux maritimes, passés et récents, n'ont menés qu'à peu de publications. On peut identifier au moins deux causes à cela. Il y a, tout d'abord, un coût technique important pour traiter, transformer et interpréter les données. Parmi les informations erronées ou incomplètes, on peut citer les arrivées datées avant les départs, les navires avec un identifiant non reconnu, les trajets

théoriquement courts qui durent des mois, etc. Tout ce bruit dans les données impose un travail minutieux et chronophage de traitement, qui laisse moins de temps pour un travail de recherche plus valorisant. Il nous apparaît, ensuite, que la recherche conduite dans un cadre multidisciplinaire est, pour de multiples raisons, plus difficile valorisée.



## **partie III**

# **Algorithmique dans les réseaux dynamiques**



# Introduction

La notion de graphe dynamique peut assez naturellement se définir comme un ensemble de nœuds et d’arcs qui évoluent dans le temps. Néanmoins, on trouve de nombreuses dénominations dans la littérature (*time-varying graphs*<sup>i</sup>, *evolving graphs*<sup>j</sup>, *dynamic networks*<sup>k</sup>, *temporal networks*<sup>l</sup>, *link streams*<sup>m</sup>, etc.). Cette variété s’explique par l’universalité du modèle de graphe qui permet la représentation d’une multitude de problèmes, dans des domaines d’application tout aussi variés. Ces modèles appartiennent non seulement à des domaines d’application différents, mais également à des disciplines différentes (algorithmique distribuée, physique statistique, télécommunications).

Les définitions elles-mêmes ne considèrent pas les mêmes critères. Il existe une combinatoire des modèles possibles de graphes dynamiques, en fonction des critères que l’on se fixe. Par exemple, la représentation du temps peut être continue ou discrète, la présence des nœuds peut être fixe ou variable, etc. La figure C illustre la combinatoire des modèles de graphes dynamiques possibles.

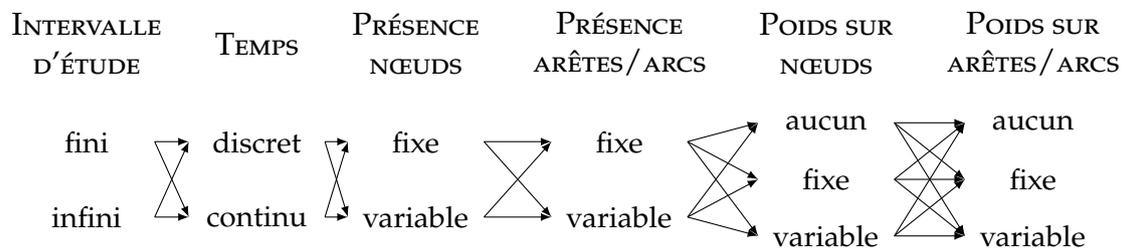


Figure C. Combinatoire des modèles de graphes dynamiques possibles.

<sup>i</sup>[12] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. 2012. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems* 27, 5 (October 2012), 387–408. DOI : [10.1080/17445760.2012.668546](https://doi.org/10.1080/17445760.2012.668546)

<sup>j</sup>[9] Sandeep Bhadra and Afonso Ferreira. 2003. Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks. In *International conference on ad-hoc networks and wireless*, 259–270. DOI : [10.1007/978-3-540-39611-6\\_23](https://doi.org/10.1007/978-3-540-39611-6_23)

<sup>k</sup>[10] B. Bui Xuan, A. Ferreira, and A. Jarry. 2003. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science* 14, 02 (April 2003), 267–285. DOI : [10.1142/S0129054103001728](https://doi.org/10.1142/S0129054103001728)

<sup>l</sup>[31] David Kempe, Jon Kleinberg, and Amit Kumar. 2002. Connectivity and Inference Problems for Temporal Networks. *Journal of Computer and System Sciences* 64, 4 (June 2002), 820–842. DOI : [10.1006/jcss.2002.1829](https://doi.org/10.1006/jcss.2002.1829)

<sup>m</sup>[33] Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. 2018. Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining* 8, 1 (October 2018), 61. DOI : [10.1007/s13278-018-0537-7](https://doi.org/10.1007/s13278-018-0537-7)

Dans les travaux qui suivent, le modèle utilisé considère : un intervalle d'étude fini, le temps discret, des nœuds toujours présents et une présence d'arrête variable. Les poids sur les arcs et nœuds pourront exister en fonction des variantes. Dans ce contexte, on peut définir simplement un graphe dynamique  $G$  comme une succession de graphes statiques :  $G = (G_i)_{i \in \mathcal{T}}$ , où :  $\mathcal{T} = \{1, \dots, T\}$  est l'intervalle d'étude,  $T$  est l'horizon de temps,  $G_i = (V, E_i)$  est un t-graphe,  $V$  l'ensemble des sommets du graphe et  $E_i$  l'ensemble des arêtes à l'étape  $i$ .

La figure D montre un graphe dynamique constitué de quatre t-graphes.

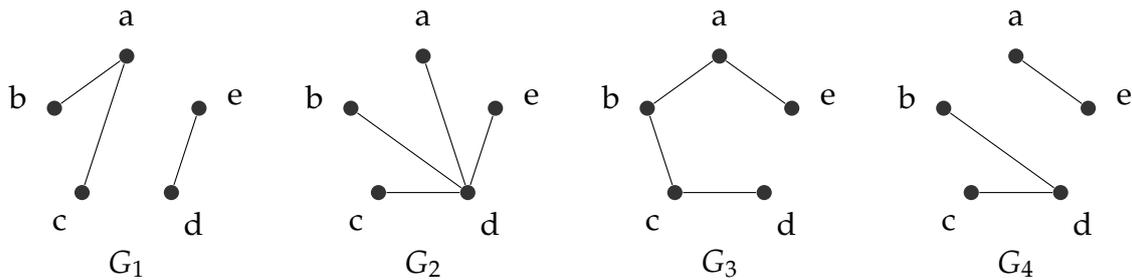


Figure D. Exemple de graphe dynamique avec quatre étapes discrètes.

Une autre représentation de ce modèle de graphe, plus concise, consiste à représenter les nœuds, qui sont toujours présents, ainsi que l'intersection de toutes les arêtes des t-graphes. Ensuite un étiquetage des arêtes, avec une liste d'étapes, permet d'identifier à quelles étapes, une arête donnée existe. La figure E illustre la représentation compacte de la liste de t-graphes de la figure D.

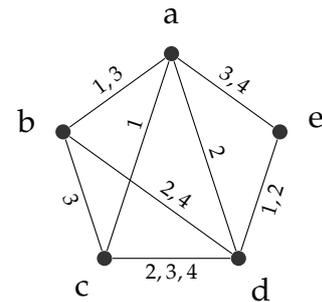


Figure E. Représentation compacte du graphe dynamique avec quatre étapes discrètes de la figure D. Les arêtes sont pondérées par la liste des indices des étapes où elles existent.

# 6 Le problème des flots de cout minimal

Travaux concernés : [53,54]

On s'intéresse au problème de flot de cout minimal, dans un graphe dynamique, avec couts éventuellement infinis et capacités variables. On ne considère pas de temps de traversée sur les arcs et il n'y a pas de stockage. La figure 6.1 illustre le modèle de graphe dynamique considéré ici.

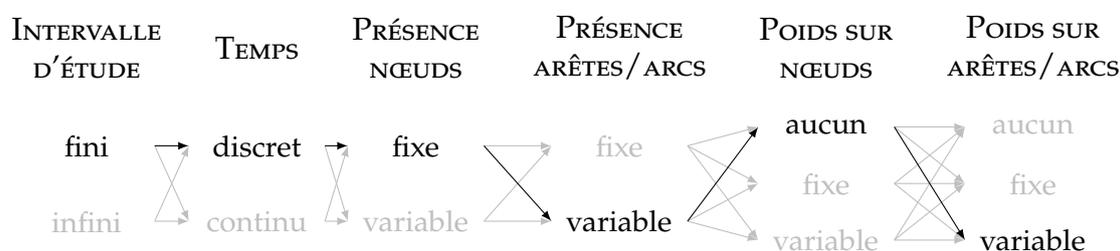


Figure 6.1. Modèle de graphe dynamique considéré pour le problème de flots de cout minimal.

Une solution exacte existe pour ce problème. Elle passe par l'utilisation du « graphe développé », une structure qui consiste à créer un graphe statique composé de l'ensemble des t-graphes. Dans cette structure tous les sommets de  $G$  sont représentés  $T$  fois (une fois pour chaque étape de temps). Enfin, une méta-source et un méta-puits sont ajoutés et reliés aux véritables sources et puits, avec des liens de capacité infinie et de cout nul. L'objectif du travail était de développer un algorithme exact, n'ayant pas recours au graphe développé, jugé trop coûteux en mémoire.

## 6.1 Contexte

Ce travail est le fruit d'une collaboration pérenne avec le professeur Maciej Drozdowski de l'institut de technologie de Poznan, en Pologne, dans le cadre de deux projets successifs. Ces projets, de type *Polonium*, sont des Partenariats Hubert Curien (PHC) franco-polonais. En France c'est par le ministère des affaires étrangères que le projet est mis en œuvre. Le professeur Drozdowski fut le porteur, coté polonais et Eric Sanlaville assura la fonction similaire, côté français. Le projet permit les échanges de chercheurs permanents et doctorants lors de plusieurs séjours. J'ai pu me rendre à Poznan, en 2018, dans ce cadre. J'étais accompagné de Mathilde Vernet, doctorante au LITIS de 2017 à 2020, que j'ai co-encadré avec Eric Sanlaville, son directeur.

Le travail présenté ici est né lors de ces échanges. Ce chapitre revient rapidement sur les résultats obtenus, qui furent publiés dans [53] et qui sont également disponibles dans le manuscrit de Mathilde<sup>a</sup>, puisqu'ils constituent une partie de son travail de thèse.

## 6.2 Modèle

Considérons l'exemple de la figure 6.2 où un réseau dynamique à quatre sommets et trois étapes de temps, a ses arcs pondérés par des couples  $(u, c)$ , avec  $u$  la capacité des arcs et  $c$  leur cout unitaire.

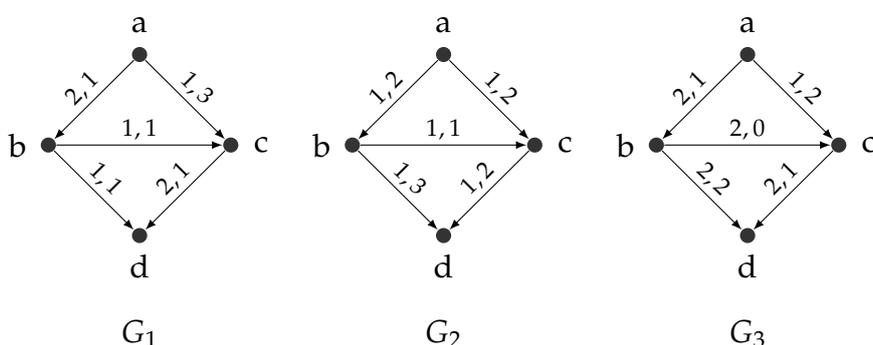


Figure 6.2. Exemple de graphe dynamique a trois étapes de temps, avec des arcs pondérés par des capacités et des couts unitaires.

Un graphe développé peut être construit en ajoutant une source unique ainsi qu'un puits unique, en ajoutant des arcs de capacité infinie et de cout nul, entre le puits unique et les puits à chaque étape. De même pour les sources que l'on connecte à la source unique. La figure 6.3 illustre cette construction sur l'exemple précédent.

Ce graphe étendu présente l'avantage d'être statique. On peut y appliquer les algorithmes classiques, pour y trouver les flots recherchés. Mais il est aussi plus volumineux que le graphe dynamique initial. Outre ses deux sommets, source et puits, il contient également  $2T$  arcs, avec  $T$  la taille de l'intervalle d'étude (ou le nombre d'étapes). L'idée est donc de tenter de se passer du graphe étendu et d'utiliser le graphe dynamique pour résoudre le problème des flots de cout minimal, afin de réduire la complexité du calcul.

## 6.3 Algorithme

L'algorithme classique, pour résoudre le problème de flot de cout minimal, utilise les plus courts chemins pondérés par les couts, pour trouver des chemins de cout minimal et successivement mettre à jour les capacités du réseau.

<sup>a</sup>[52] Mathilde Vernet. 2020. Modèles et algorithmes pour les graphes dynamiques. Thèse. Normandie Université. <https://tel.archives-ouvertes.fr/tel-03030851>

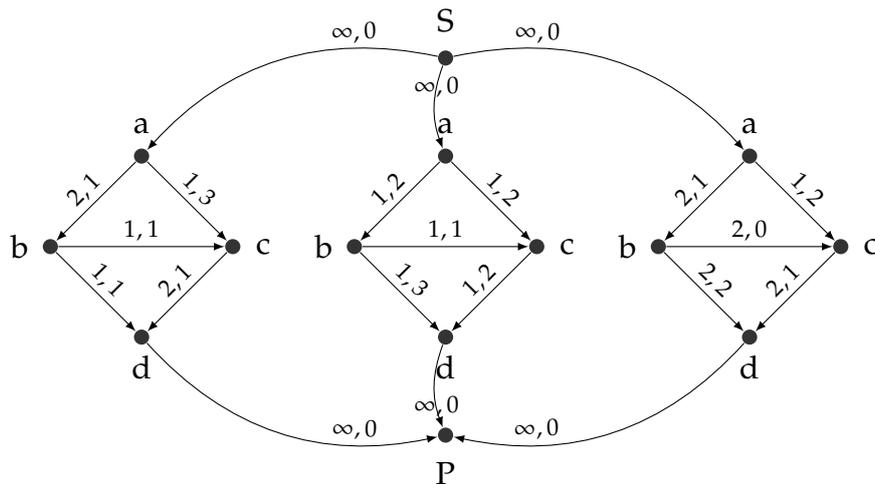


Figure 6.3. Graphe étendu à partir de l'exemple de la figure 6.2.

Le fonctionnement général de l'algorithme et sans entrer dans les détails de la structure sous-jacente utilisée<sup>b</sup>, peut se présenter ainsi :

- Tant que toute la quantité  $U$  de flot n'est pas arrivée au puits :
  1. Calculer les plus courts chemins, en termes de coût, entre la source et le puits.
  2. Trouver la plus grande quantité pouvant passer sur le plus court chemin (capacité minimale parmi les arcs du chemin) et diminuer d'autant les capacités des arcs du chemin.

L'algorithme *Successive Shortest Path* (SSP), s'il utilise l'algorithme de Dijkstra pour trouver les plus courts chemins, possède une complexité calculatoire de l'ordre de  $O(U \cdot (m + n \cdot \log(n)))$ , avec  $U$  la quantité de flot demandée,  $m$  le nombre d'arcs dans le graphe et  $n$  le nombre de nœuds. Si on considère le graphe dynamique, alors SSP fonctionne sur la version étendue du graphe avec une complexité de l'ordre de  $O(U \cdot T \cdot (m + n \cdot \log(n \cdot T)))$  avec  $T$  le nombre d'étapes.

On propose une utilisation de SSP n'utilisant pas le graphe étendu afin d'améliorer la complexité. L'idée générale est d'exécuter SSP sur chacun des t-graphes et de sélectionner le plus petit des plus courts chemins parmi tous les t-graphes à chaque étape de l'algorithme. En effet si les temps de trajets n'ont pas d'impact sur la dynamique (ou s'ils sont tout simplement nuls) alors chaque t-graphe est indépendant. Le sketch de l'algorithme est le suivant :

1. Calculer les plus courts chemins, en termes de coût, entre la source et le puits de chaque t-graphe.
2. Tant que toute la quantité  $U$  de flot n'est pas arrivée au puits :
  - a. Sélectionner le t-graphe qui contient le plus petit plus court chemin.

<sup>b</sup>*Successive Shortest Path* est présenté dans le livre de référence : [3] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. 2013. *Network Flows: Theory, Algorithms, and Applications*. Pearson.

- b. Trouver, dans ce t-graphe, la plus grande quantité pouvant passer sur le plus court chemin (capacité minimale parmi les arcs du chemin) et diminuer d'autant les capacités des arcs du chemin.
- c. Recalculer le plus court chemin de la source au puits dans le t-graphe précédemment sélectionné.

La figure 6.4 est la solution de flot trouvée par cet algorithme sur le graphe dynamique de l'exemple de la figure 6.2. Deux unités de flots passent au temps 1 ; une par le chemin a,b,d, l'autre par le chemin a,b,c,d. Deux autres unités passent par le temps 3, par le chemin a,b,c,d. Aucun flot ne passe à l'étape 2.

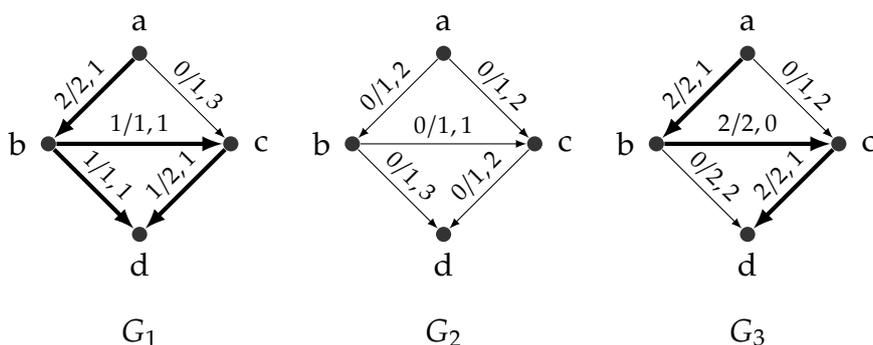


Figure 6.4. Solution de flot avec la méthode SSP dynamique sur le graphe dynamique de la figure 6.2. Les lignes épaisses indiquent où passent les flots, les valeurs  $f/u,c$  indiquent les ratios flot sur capacité, suivit du cout unitaire.

Cet algorithme est exact, polynomial et sa complexité est inférieure à celle de l'algorithme original d'un facteur  $T$ . Elle peut s'exprimer ainsi :  $O((U + T) \cdot (m + n \cdot \log(n)) + U \cdot \log(T))$ . L'algorithme est plus efficace théoriquement que la méthode sur graphe développé et une étude expérimentale sur des graphes synthétiques en prouve l'efficacité pratique.

## 7 *Le problème de connexité*

Travaux concernés : [55,56]

Comme il a été dit dans l'introduction de cette partie, les choix de représentation des graphes dynamiques dans le but de concevoir formellement des algorithmes, sont multiples. Le choix du modèle et les hypothèses que l'on se fixe, déterminent également le type d'algorithme qui peut s'y appliquer. Inversement, si l'on souhaite adapter un algorithme issu de la théorie des graphes classiques, pour les graphes dynamiques, alors les choix dans le modèle et les hypothèses prises formeront un nouveau problème, dont la complexité de résolution déterminera la faisabilité pour de nouveaux algorithmes.

Dans le travail présenté ici, on se positionne dans le cas général, hors de tout contexte applicatif, avec la volonté de proposer une représentation dynamique à un problème initialement statique. La question posée ici est celle du devenir de la notion de composante connexe, dans le cas des graphes dynamiques. Une composante connexe représente, dans un graphe classique, un sous ensemble de nœuds, accessibles deux à deux via un chemin dans le graphe. De plus, cet ensemble se veut maximal, il ne peut exister de nœud accessible par tous les nœuds d'une composante qui ne fasse pas partie de ladite composante.

Dans le contexte dynamique, cette définition doit être adaptée. Plusieurs propositions existent dans la littérature et le critère le plus discriminant est de savoir si les arcs ont des couts temporels associés (des temps de traversée). Le cas échéant, les chemins deviennent des trajets où les durées et l'ordre des dates d'existence des arcs, doivent être respectés. Là encore, plusieurs configurations sont possibles; des trajets aller/retour entre toutes les paires<sup>a</sup>, des trajets d'une longueur bornée<sup>b</sup>, ou des connections sur des fenêtres de temps<sup>c</sup>.

Ici, les temps de trajets sont nuls. Le problème revient donc à identifier les ensembles de nœuds qui restent connectés pendant un intervalle de temps. La figure 7.1 illustre le modèle de graphe considéré. Dans ce cas également la littérature existe et les définitions varient. La notion de connectivité peut être obtenue en

---

<sup>a</sup>[9] Sandeep Bhadra and Afonso Ferreira. 2003. Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks. In International conference on ad-hoc networks and wireless, 259–270. DOI : [10.1007/978-3-540-39611-6\\_23](https://doi.org/10.1007/978-3-540-39611-6_23)

<sup>b</sup>[23] Carlos Gómez-Calzado, Arnaud Casteigts, Alberto Lafuente, and Mikel Larrea. 2015. A Connectivity Model for Agreement in Dynamic Systems. In Euro-Par 2015: Parallel Processing (Lecture Notes in Computer Science), Springer, Berlin, Heidelberg, 333–345. DOI : [10.1007/978-3-662-48096-0\\_26](https://doi.org/10.1007/978-3-662-48096-0_26)

<sup>c</sup>[29] Charles Huyghues-Despointes, Binh-Minh Bui-Xuan, and Clémence Magnien. 2016. Forte  $\Delta$ -connexité dans les flots de liens. <https://hal.archives-ouvertes.fr/hal-01305128>

intersectant les t-graphes<sup>d</sup>. D'autres travaux s'intéressent au temps pendant lequel le graphe reste connexe<sup>e</sup>.

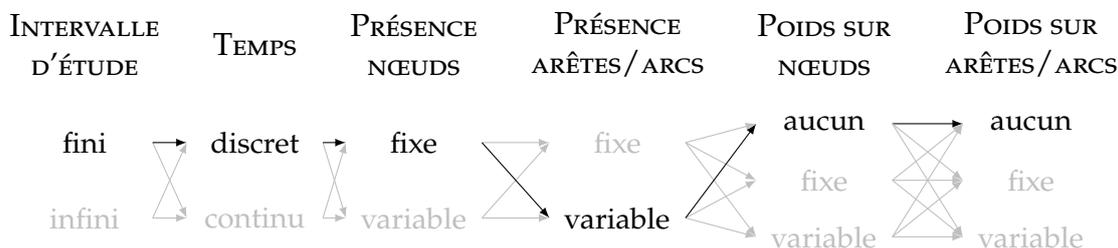


Figure 7.1. Modèle de graphe dynamique considéré pour le problème de composantes connexes.

## 7.1 Contexte

Ce chapitre, comme le précédent, est un des résultats de la thèse de Mathilde Vernet<sup>f</sup>. Ce co-encadrement de thèse et cette équipe de travail constituée de Mathilde, Éric Sanlaville et moi furent bénéfiques et productifs sur de nombreux aspects.

## 7.2 Modèle

Dans ce contexte, une définition de la notion de composante connexe fut proposée. Celle-ci tient compte de l'absence de temps de trajets, en respectant la notion de maximalité propre aux composantes connexes des graphes statiques. Cette définition considère un ensemble maximal de nœuds, mais aussi un nombre d'étapes consécutives maximal, durant lequel, l'ensemble de nœuds existe.

**Composante connexe persistante :** Ensemble  $K$  de sommets de taille  $k$ , connecté pendant  $l$  étapes de temps consécutives :  $p = (K, k, l)$

A cette définition s'ajoute une règle de non-dominance sur les composantes connexes persistantes (PCC). Voici une version simplifiée de cette définition :

**Non-dominance :**  $p=(K,k,l)$  est non-dominée si  $\nexists p' = (K',k',l') \neq p /$

$$\left\{ \begin{array}{l} k' \geq k; l' > l \\ \text{ou } k' > k; l' \geq l \end{array} \right.$$

<sup>d</sup>[13] Arnaud Casteigts, Ralf Klasing, Yessin M. Neggaz, and Joseph G. Peters. 2015. Efficiently Testing T-Interval Connectivity in Dynamic Graphs. In Algorithms and Complexity (Lecture Notes in Computer Science), Springer International Publishing, Cham, 89–100. DOI : [10.1007/978-3-319-18173-8\\_6](https://doi.org/10.1007/978-3-319-18173-8_6)

<sup>e</sup>[4] Eleni C. Akrida and Paul G. Spirakis. 2015. On Verifying and Maintaining Connectivity of Interval Temporal Networks. In Algorithms for Sensor Systems (Lecture Notes in Computer Science), Springer International Publishing, Cham, 142–154. DOI : [10.1007/978-3-319-28472-9\\_11](https://doi.org/10.1007/978-3-319-28472-9_11)

<sup>f</sup>[52] Mathilde Vernet. 2020. Modèles et algorithmes pour les graphes dynamiques. Thèse. Normandie Université. <https://tel.archives-ouvertes.fr/tel-03030851>

Pour une PCC donnée, il ne peut donc exister d'autre PCC avec un nombre supérieur de nœuds et un nombre équivalent d'étapes de temps, ou bien avec un nombre équivalent de nœuds et un nombre d'étapes de temps consécutifs supérieur. Cette règle est en fait nécessaire à la formulation d'un algorithme capable de calculer toutes les PCC en un temps polynomial.

La figure 7.2 montre l'ensemble des PCC non dominées dans le graphe exemple de la figure D. On y trouve 3 PCC :

- la composante  $(\{b, c\}, 2, 4)$  contenant les nœuds  $b$  et  $c$ , valide sur tout l'intervalle d'étude (4 étapes),
- la composante  $(\{a, b, c, d, e\}, 5, 2)$  avec tous les nœuds du graphe et valide aux étapes 2 et 3,
- la composante  $(\{a, b, c\}, 3, 3)$  de taille 3 (avec les nœuds  $a$ ,  $b$  et  $c$ ) et de longueur 3 (étapes 1 à 3).

Il peut exister d'autres composantes, au sens de la définition précédente, mais elles sont dominées par les trois présentées ici. Par exemple il existe la composante  $(\{a, e\}, 2, 3)$  de taille 2 (nœuds  $a$  et  $e$ ) et de longueur 3 (étapes 2 à 4), mais elle est dominée par la composante  $(\{a, b, c\}, 3, 3)$ , car celle-ci possède un nœud de plus pour un nombre d'étapes égal.

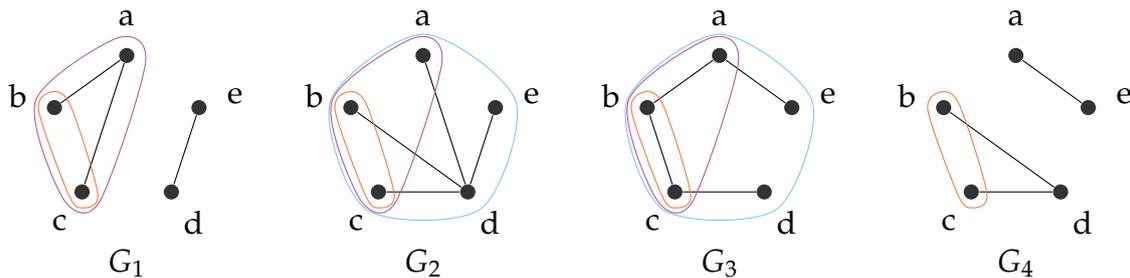


Figure 7.2. Composantes connexes non-dominées depuis le graphe exemple de la figure D.

## 7.3 Algorithme

Ce travail a également permis de proposer un algorithme exact, *online* et polynomial, pour trouver toutes les PCC non-dominées. L'algorithme s'exécute sur le graphe, à chaque étape de temps. Sa complexité est de l'ordre de  $O(n^2T)$  avec  $n$  le nombre de nœuds du graphe et  $T$  l'intervalle d'étude.

Les grandes étapes de l'algorithme sont les suivantes :

- Pour chaque étape de l'intervalle d'étude :
  1. Calculer les composantes connexes classiques sur le t-graphe courant.
  2. Comparer ces composantes avec les PCC en cours à l'étape précédente.
  3. Extraire les PCC terminées à cette étape et conserver celles qui sont toujours en cours.

#### 4. Retirer les PCC dominées.

Les détails de l’algorithme, la preuve de sa correction et de sa complexité, sont disponibles dans la thèse de Mathilde Vernet.

## 7.4 Expérimentations

Afin de montrer l’utilisabilité de l’algorithme, des expérimentations ont été menées.

D’abord, un modèle de graphe dynamique synthétique a été proposé. Basé sur un graphe de départ avec ses propres caractéristiques, l’état des arêtes (présence ou absence) est ensuite décidé en fonction de règles de probabilités données par une chaîne de Markov. Ces expérimentations permettent de confirmer les complexités théoriques affirmées plus haut.

Ensuite, l’algorithme est testé sur des graphes dynamique réels. Hélas, il est difficile de trouver de véritables graphes dynamiques avec des événements distincts d’apparition et de disparition d’arêtes. En revanche les banques de données regorgent de graphes de contacts, dynamiques, où les liens entre entités sont datés mais sans réelle durée. Nous avons donc utilisé plusieurs jeux de données de type graphe de contact, issues de la base SNAP<sup>g</sup>, auxquels des durées d’existence d’arcs furent ajoutés, pour simuler la durée dans le temps des interactions. Les expérimentations permirent de valider la faisabilité et l’utilisabilité de l’approche sur des réseaux réels et conséquents (plusieurs millions de nœuds, plusieurs dizaines de millions d’arcs et, environs 3000 étapes de temps). Les détails des expérimentations sur données réelles sont disponibles dans la publication de ces travaux [56].

## 7.5 Retour sur la génération de réseaux aléatoires

Nous souhaitons ici revenir sur la méthode de génération de graphes dynamiques aléatoires, proposée dans ce travail. Beaucoup de travaux existent autour de la génération aléatoire de graphes. Parmi ceux-là, la randomisation de réseaux réels<sup>h</sup> est une technique qui consiste à partir d’un réseau réel et de modifier/échanger aléatoirement certaines caractéristiques afin d’obtenir un nouveau réseau, aléatoire, qui conserve certaines propriétés et en écarte d’autres. Par exemple l’échange des sources et des destinations entre des paires d’arcs choisies aléatoirement  $((i, j)$  et  $(i', j')$  deviennent  $(i', j)$  et  $(i, j')$ ) permet de conserver la distribution des degrés du graphe mais perturbe complètement les centralités et les trajets. Dans un contexte dynamique aussi la randomisation existe, notamment dans les graphes de contacts, où les dates de contacts peuvent être randomisés, même si cette technique détruit l’effet de *burst* propre à ces réseaux. Ainsi une génération aléatoire avec des lois

<sup>g</sup>Stanford Network Analysis Project <https://snap.stanford.edu/data/>

<sup>h</sup>Une classification des différents travaux sur la randomisation est faite dans [28] Petter Holme and Jari Saramäki. 2012. Temporal networks. Physics Reports 519, 3 (October 2012), 97–125. DOI : 10.1016/j.physrep.2012.03.001

adaptées (des processus de Poisson) permet de retrouver les propriétés recherchées<sup>i</sup>.

Les graphes dynamiques auxquels on s'intéresse ne sont pas des graphes de contact, car il manque à ces derniers la notion de durée de vie des liens. Avec la volonté d'obtenir des graphes dynamiques qui conservent certaines des propriétés d'un graph statique de départ, on propose une méthode inspirée des travaux sur les graphes de contact.

Notre approche est également proche, en de nombreux points, à la notion de *edge-Markovian evolving graphs*<sup>j</sup> proposée par l'équipe romaine de Clementi, Macci, Monti, Pasquale et Silvestri. On revient sur les différences avec cette approche plus tard.

Le modèle propose de générer des graphes dynamiques basées sur un graphe de départ, le graphe sous-jacent. Ce graphe sous-jacent possède ses propres caractéristiques que l'on va tenter de conserver (distribution de degrés, centralités, etc.) Le processus consiste à faire évoluer (apparitions, disparitions) les arêtes du graphe sous-jacent. Seules les arêtes présentes dans le graphe sous-jacent sont considérées pour générer le graphe dynamique. Tous les nœuds du graphe sous-jacent sont présents dans tous les t-graphes générés. Plutôt que de définir les événements d'apparition et de disparition des arêtes à l'aide d'une loi aléatoire on propose la modélisation de l'état de chaque arête à l'aide d'une chaîne de Markov (figure 7.3), qui détermine les états possibles d'une arête (présente ou absente), ainsi que les probabilités de passage d'un état à l'autre.

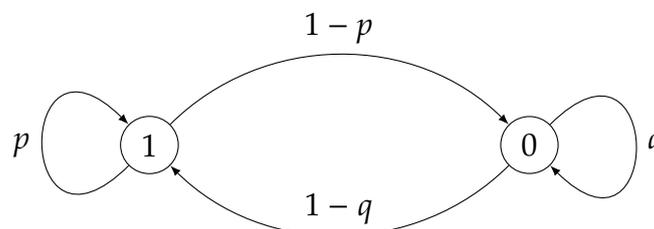


Figure 7.3. Chaîne de Markov décrivant les deux états possibles (1=présence, 0=absence) d'une arête dans le générateur de graphe dynamique et les probabilités de passage.

Une arête présente à l'étape  $t$ , restera présente à l'étape  $t+1$ , avec une probabilité  $p$ . Une arête absente (état 0) à l'étape  $t$ , apparaîtra à l'étape  $t+1$ , avec la probabilité  $1-q$ . Soit  $P$  la matrice de probabilité :

<sup>i</sup>[30] M. Karsai, M. Kivela, R. K. Pan, K. Kaski, J. Kertesz, A.-L. Barabasi, and J. Saramaki. 2011. Small but slow world: How network topology and burstiness slow down spreading. *Phys. Rev. E* 83, 2 (February 2011), 025102. DOI : [10.1103/PhysRevE.83.025102](https://doi.org/10.1103/PhysRevE.83.025102)

<sup>j</sup>[15] [1] Andrea E.F. Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. 2008. Flooding time in edge-Markovian dynamic graphs. In *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing (PODC '08)*, Association for Computing Machinery, New York, NY, USA, 213–222. DOI : [10.1145/1400751.1400781](https://doi.org/10.1145/1400751.1400781)

$$\begin{array}{l}
 \text{présence à } t \\
 \text{absence à } t
 \end{array}
 \begin{pmatrix}
 p & 1-p \\
 1-q & q
 \end{pmatrix}
 \begin{array}{l}
 \text{présence à } t+1 \\
 \text{absence à } t+1
 \end{array}$$

D'après la théorie des chaînes de Markov, il existe un vecteur de probabilité unique  $\pi$  pour une chaîne de Markov satisfaisant  $\pi = \pi \cdot P$ . Dans notre cas le vecteur de probabilité a deux valeurs, pour les deux états de l'arête, 1 (présence) et 0 (absence) :  $\pi = (\pi_1, \pi_0)$ . Ainsi l'égalité de Markov peut s'écrire :

$$(\pi_1, \pi_0) = (\pi_1, \pi_0) \cdot \begin{pmatrix} p & 1-p \\ 1-q & q \end{pmatrix}$$

Ce vecteur de probabilité  $\pi$  représente la distribution stationnaire des états du système. On peut donc définir une valeur de probabilité pour ce vecteur, de façon à obtenir une probabilité de présence pour une arête et par extension pour tout le graphe. On peut ensuite déduire les valeurs des paramètres  $p$  et  $q$  qui maintiennent la distribution stationnaire.

$$\begin{aligned}
 (\pi_1, \pi_0) &= (\pi_1 p + \pi_0(1-q), \pi_1(1-p) + \pi_0 q) \\
 (\pi_1, (1-\pi_1)) &= (\pi_1 p + (1-\pi_1)(1-q), \pi_1(1-p) + (1-\pi_1)q)
 \end{aligned}$$

$$\begin{aligned}
 \pi_1 &= \pi_1 p + (1-\pi_1)(1-q) & ; & & (1-\pi_1) &= \pi_1(1-p) + (1-\pi_1)q \\
 \pi_1 &= \pi_1 p + 1 - q - \pi_1 + \pi_1 q & ; & & (1-\pi_1)q &= 1 - \pi_1 - \pi_1 + p\pi_1 \\
 -\pi_1 p &= -2\pi_1 + 1 - q + \pi_1 q & ; & & (1-\pi_1)q &= -2\pi_1 + p\pi_1 + 1 \\
 p &= 2 - \frac{1}{\pi_1} + \frac{q}{\pi_1} - q & ; & & q &= \frac{2\pi_1 - p\pi_1 - 1}{\pi_1 - 1}
 \end{aligned}$$

Il existe plusieurs couples de solution  $(p, q)$  pour une valeur de  $\pi_1$  donnée. Les valeurs possibles se situent sur une droite comme le montre la figure 7.4.

S'il est vrai que n'importe quel couple  $(p, q)$  sur la droite solution permet d'atteindre la distribution stationnaire et donc le taux de présence moyen des arêtes dans le graphe, il faut noter que plus les valeurs de  $p$  et  $q$  sont élevées, plus lente sera la dynamique du réseaux. En effet, des valeurs élevées pour  $p$  et  $q$  signifient qu'une arête ne changera que peu d'état, puisque  $p$  est la probabilité de rester présente et  $q$  est la probabilité de rester absente, pour une arête. On devine donc, qu'en plus de la probabilité de présence moyenne des arêtes du graphe, ce modèle permet de faire varier la dynamique. Par exemple, une séquence d'états  $s1 = [0, 0, 0, 1, 1, 1, 1, 1, 1, 1]$  signifie qu'une arête est absente aux 3 premières étapes, puis présente aux suivantes. Une seconde séquence,  $s2 = [1, 1, 0, 1, 0, 1, 1, 0, 1, 1]$  signifie qu'une arête, avec ces états, varie plus souvent. Pourtant, les deux séquences contiennent le même nombre d'états. Elles sont identiques en moyenne (arêtes présentes 70% du temps) mais la distribution des états rend l'arête de la séquence  $s2$  plus dynamique. Le choix des valeurs de  $p$  et  $q$  sur la droite de solution (figure 7.4) a une importance sur la dynamique du graphe. Dans les expérimentations présentées ici,

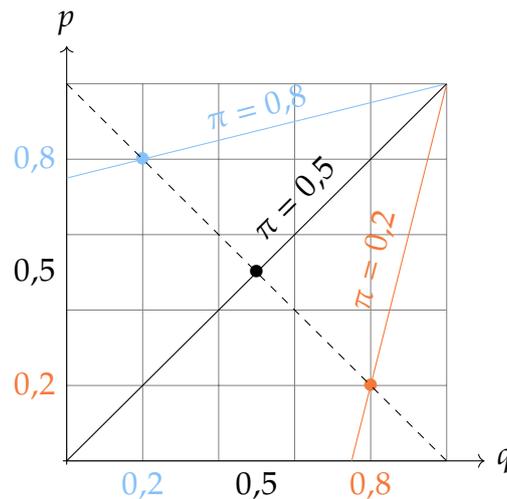


Figure 7.4. Valeurs possibles des probabilités  $p$  et  $q$  de la chaîne de Markov de la figure 7.3, pour des valeurs de  $\pi_1$  (probabilité de présence d'une arête) données. La droite en pointillés illustre le cas particulier où  $p = \pi_1$ .

on s'est intéressé au cas particulier où  $p = \pi_1$ . Dans ce cas les couples de valeurs se trouvent à l'intersection avec la droite d'équation  $1 - q$  (représentée en pointillés sur la figure 7.4).

La méthode est donc un générateur de graphe dynamiques, qui prend en entrée un graphe statique sous-jacent, ainsi que 2 paramètres :

- La *présence* qui est la probabilité asymptotique de présence des arêtes ( $\pi_1$ ).
- La *stabilité* comprise entre 0 et 1 qui permet de choisir un couple de probabilités  $p$  et  $q$  le long de la droite solution pour une *présence* données. Une valeur de 0 sélectionnera les valeurs les plus petites de  $p$  et  $q$ . L'une d'elle sera nécessairement 0. Si  $p$  ou  $q$  valent 0 alors toute arrivée dans l'un des états concernés, à l'étape  $t$ , ne peut être suivi que par un changement d'état à l'étape  $t + 1$ . Une *stabilité* nulle est donc la configuration où le graphe est le plus dynamique. Les arêtes, tout en respectant le taux de présence indiqué, changeront d'état le plus souvent possible. Une *stabilité* de 0,5 peut être alignée sur la valeur particulière  $p = \pi_1$ , pour une dynamique moyenne dans le graphe. Enfin, une *stabilité* à 1 empêche tout changement d'état, le graphe est statique.

En sortie, le générateur peut produire des t-graphes de façon itérative en fonction du besoin. Le nombre d'étapes (l'intervalle d'étude) n'est pas un paramètre du système.

Cette approche ressemble à la notion de *edge-Markovian evolving graphs*, mais elle diffère par l'utilisation du graphe sous-jacent. Dans notre approche, seules les arêtes présentes dans le graphe sous-jacent sont susceptibles d'apparaître ou de disparaître dans les t-graphes. Dans le modèle *edge-Markov*, le graphe  $G_0$ , l'équivalent du sous-jacent, n'est qu'une distribution de départ et toutes les arêtes potentielles du graphe sont susceptibles d'apparaître ou de disparaître. En conséquence, les graphes dynamiques générés par *edge-Markov* ont une propriété forte, que n'ont pas les

notres : il tendent asymptotiquement vers un graphe aléatoire (type Erdős–Rényi). Notre approche est, pour sa part, plus susceptible de conserver les propriétés du graphe sous-jacent (distribution de degrés, centralités, etc.)



## 8 *GraphStream*

GraphStream est né en 2006, durant ma thèse. Avec mon collègue Antoine Dutot, lui aussi thésard à l'époque, nous partagions les mêmes problématiques de modélisation du vivant avec des graphes qui bougent. Nous avons tous les deux, de notre côté, implémenté des modèles de graphe, avec des capacités de visualisation et quelques algorithmes standards. Après quelques discussions au coin café et un *brain storming* pour trouver un nom de projet qui en jette, je jetais à la poubelle ma bibliothèque de graphe, nous refactorions le code d'Antoine, créions le projet sur SourceForge (l'ancêtre de GitHub) et GraphStream était. Le regard bienveillant mais un peu amusé de nos encadrants respectifs nous encourageait à continuer, mais le projet n'était que balbutiant. Plus tard, Antoine réécrivant le moteur graphique, un nouveau développeur/thésard, Guilhelm Savin, récrivait à peu près tout le reste et moi, entre temps parti en postdoc au Luxembourg, j'essayais de suivre. A mon retour du Luxembourg, GraphStream était devenu le projet étendard de l'équipe. Il était utilisé par les doctorants, enseigné aux étudiants du master et était valorisé dans plusieurs projets scientifiques, qui apportaient des fonds pour recruter des développeurs postdoc et ingénieurs. Mon collègue Stefan Balev apportait sa pierre à l'édifice en optimisant les structures de données et les algorithmes. De mon côté je devenais le VPR du projet et enchainais les démonstrations, les présentations, les tutoriels et les ateliers.

Aujourd'hui le développement de GraphStream s'est considérablement ralenti. La version 2.0, publiée en 2020, grâce au conséquent travail d'Hicham Brahim, est considérée comme stable et ne fait plus l'objet de nouveaux développements. La bibliothèque est pourtant toujours activement utilisée et téléchargée sur les plateformes (*GitHub* et *Maven Central*). À l'heure de l'écriture de ces lignes, en 2022, elle est téléchargée environs 12 mille fois par mois et elle est référencée comme dépendance de presque un millier de projets GitHub libres. Il n'y a probablement plus d'avenir pour GraphStream mais je tenais à lui faire une petite place dans ce mémoire, car cet outil m'a accompagné dans l'ensemble des travaux qui sont présentés ici, des VANETs aux composantes connexes, en passant par les bateaux.

Ce chapitre est une petite introduction à GraphStream, à l'image des nombreux tutoriels que j'ai eu l'occasion de présenter. Ces lignes furent rédigées durant ma thèse, en 2008, vouées à être publiées, puis oubliées. Je les restitue ici dans le style et le point de vue de l'époque. Je me suis simplement assuré que les exemples fonctionnaient avec la version actuelle de la bibliothèque.

## 8.1 Introduction

Dans la littérature, le terme de graphe dynamique n'est pas utilisé de manière consensuelle. D'une part, il est souvent fait mention de graphe dynamique pour des travaux qui s'apparentent aux *Complex Networks* [32], aux graphes évolutifs [10], aux *Space-Time Networks* [40], mais également relevant de domaines aussi variés que l'écologie, la biologie ou les sciences sociales pour l'ensemble desquels on parlera plus fréquemment de réseaux d'interactions. D'autre part, la notion même de graphe dynamique n'est souvent définie que par défaut, pour nommer un objet qui ne constitue pas, à proprement parler, le sujet central d'un travail, ou d'une étude, mais simplement le modèle servant de support implicite à l'étude d'un système particulier.

Il existe un corpus assez important de travaux se rapportant à des méthodes de réoptimisation dans un graphe, lorsque celui-ci varie [16,17,22,47]. Dans ce cas, la dynamique du graphe est vue comme une suite d'évènements qui surviennent à une date donnée. À chaque instant une représentation actuelle et statique du graphe peut être faite mais sera rendu obsolète par le prochain évènement. Cette dernière approche nous semble générique au sens où elle ne sert pas un domaine applicatif particulier.

Bien entendu, il existe un grand nombre de solutions logicielles qui permettent de modéliser et de manipuler des graphes statiques<sup>a</sup>. Parmi ces outils, peu autorisent une approche dynamique et encore plus rares sont ceux annonçant une conception basée sur ce paradigme.

C'est donc assez naturellement que s'est imposé le besoin de concevoir et développer un outil logiciel dédié. Les besoins logiciels concernent la modélisation des problèmes, à l'aide de graphes dynamiques, mais aussi la visualisation des graphes.

Les problèmes concernés par les graphes dynamiques sont nombreux et appartiennent à plusieurs contextes scientifiques, qui partagent les mêmes besoins de modélisation.

Dans l'intention de développer un outil le plus transversal possible aux différentes préoccupations, la volonté principale fut un détachement total de quelque application que ce soit.

Nous proposons donc *GraphStream*, une bibliothèque logicielle dédiée à la modélisation, à la manipulation et à la visualisation de graphes dynamiques. Les deux particularités principales de cet outil sont donc une approche des graphes au sens théorique sans la moindre imprégnation applicative, qui facilite son utilisation dans n'importe quel contexte, ainsi qu'une prise en compte centrale de la notion de dynamique.

Nous proposons ici une présentation de la bibliothèque et de ses principales fonctionnalités. Des exemples simples illustrent notre propos.

---

<sup>a</sup>La liste des bibliothèques de graphe est longue et pour ne pas en oublier on ne tentera pas d'être exhaustif. Citons à minima [NetworkX](#) en python, [JGraphT](#) en java et [igraph](#) en langage R.

## 8.2 Principales caractéristiques

Cette section propose de présenter les principales caractéristiques de la bibliothèque. Le premier point présente l'outil en tant que projet informatique. Intéressons-nous ensuite à la facilité d'appréhension de l'outil, grâce à une approche objet simplifiant l'écriture du code. Puis voyons comment la notion de dynamique est intégrée. Enfin nous donnons un aperçu des fonctionnalités graphiques.

### Le projet *GraphStream*

*GraphStream* est un projet né du besoin commun des différents membres de l'équipe RI2C du LITIS en 2006. Son site Web est <http://graphstream-project.org> et le code du projet est hébergé sur *GitHub*<sup>b</sup>. Il est distribué sous la double licence publique *LGPL*<sup>c</sup> et *CECILL-C*<sup>d</sup>. Le site du projet donne accès au code source, ainsi qu'à la documentation. Les nouvelles versions du projet y sont régulièrement publiées.

L'intégralité du code est écrite en Java, par soucis de portabilité et de facilité d'utilisation. L'architecture logicielle est composée de plusieurs modules :

- Un module de base contenant les classes de base sur les graphes et le système d'évènements par flux.
- Un module algorithmique qui définit des mesures sur les graphes, des générateurs de graphes, et des algorithmes classiques ou adaptés à la dynamique.
- Un module dédié à la visualisation qui fournit divers outils de représentation des données sur des graphes dynamiques.

D'autres fonctionnalités du projet existent sans pour autant être présentées ici. Parmi ces fonctionnalités, citons les différentes implémentations du module de visualisation (Swing, Java-fx, Android), ou encore les capacités d'interconnexion avec d'autres outils et d'autres langages via un protocole de communication générique.

### Un formalisme objet intuitif

L'un des premiers choix fait dans *GraphStream* est celui de la simplicité d'apprentissage. L'utilisation du paradigme objet et de Java aide grandement à cette simplicité. Les tâches simples de manipulation d'un graphe s'articulent autour de l'interface centrale *Graph*. Le listing 8.1 crée un nouveau graphe (ligne 1), puis lance une fenêtre de visualisation qui va produire une projection de celui-ci en le mettant en forme dans le plan (ligne 2). À la ligne 3, des évènements dynamiques d'ajout/suppression/modification de sommets et d'arêtes sont lus à partir d'un

---

<sup>b</sup><https://github.com/graphstream>

<sup>c</sup><https://www.gnu.org/licenses/lgpl-3.0.html>

<sup>d</sup>[https://cecill.info/licences/Licence\\_CeCILL-C\\_V1-fr.html](https://cecill.info/licences/Licence_CeCILL-C_V1-fr.html)

fichier et appliqués au graphe. Enfin, quand tous les évènements sont lus, la dernière ligne affiche l'ordre du graphe (son nombre de sommets).

```
1 Graph graph = new SingleGraph("Mon graphe");
2 graph.display();
3 graph.read("MonFichierDeGraphe.dgs");
4 System.out.printf("Ordre du graphe : %d\n", graph.getNodeCount());
```

Listing 8.1: Création, lecture et visualisation d'un graphe

L'interface `Graph` est centrale, elle définit des méthodes permettant de construire, visualiser, modifier et analyser le graphe. Un graphe est représenté par un objet. On verra dans l'exemple suivant que les sommets et les arêtes sont également des objets. Nous n'entrerons pas dans les détails de ces structures de données, notons seulement que plusieurs implémentations de l'interface `Graph` sont disponibles et disposent de différentes caractéristiques. Concrètement, il est possible de faire de l'économie du temps d'exécution ou de la mémoire. L'implémentation par défaut propose des méthodes d'accès et de modification optimisées pour le temps d'exécution mais parfois gourmandes en mémoire. Une autre implémentation tente de réduire au maximum l'utilisation de la mémoire. Pour donner un ordre d'idée, cette dernière implémentation permet de charger un graphe de plus de 10 millions d'éléments (sommets et arêtes) en mémoire centrale d'un ordinateur de 2Go de mémoire. Le point commun entre les graphes, les sommets et les arêtes est qu'ils implémentent tous une interface nommée `Element` qui les définit comme structures de données pouvant contenir des attributs. Ainsi un graphe, un sommet ou une arête peuvent avoir des attributs de types quelconques. En termes de théorie des graphes on dira que ces graphes sont valués. Ces valeurs sont bien entendu modifiables et font partie de la dynamique du graphe.

Le listing 8.2 montre la création d'un graphe par programmation, les sommets et les arêtes sont ajoutés les uns après les autres. D'abord un graphe est créé, puis trois sommets et trois arêtes reliant ces sommets. Enfin, les trois sommets et les trois arêtes se voient assigner un attribut nommé « *label* ».

## Une gestion native de la dynamique

Le second grand principe de la bibliothèque est la gestion native et implicite de la dynamique. C'est probablement la plus grande originalité du projet. Cette gestion se traduit par la notion de flux d'évènements permettant de matérialiser l'idée de temporalité. Les évènements de modification de graphe forment un flux d'entrée dans le graphe qui est modifié en conséquence. Il existe aussi un flux sortant de ce graphe, pour les mêmes évènements. Ce flux sortant agit sur les clients du graphe dynamique qui doivent leur comportement à la dynamique du graphe. Parmi les applications clientes du graphe dynamique on compte les outils de mesure et les algorithmes dynamiques qui nécessitent ces données événementielles ou encore l'interface graphique qui est donc indépendante du graphe dynamique. On peut aussi imaginer n'importe quelle simulation dont le fonctionnement est rythmé par la dynamique du graphe.

```

1 Graph graph = new SingleGraph( "Triangle" );
2
3 Node A = graph.addNode( "A" );
4 Node B = graph.addNode( "B" );
5 Node C = graph.addNode( "C" );
6
7 Edge AB = graph.addEdge( "AB", "A", "B" );
8 Edge AC = graph.addEdge( "AC", "A", "C" );
9 Edge BC = graph.addEdge( "BC", "B", "C" );
10
11 A.setAttribute( "label", "node A" );
12 B.setAttribute( "label", "node B" );
13 C.setAttribute( "label", "node C" );
14 AB.setAttribute( "label", "(A,B)" );
15 AC.setAttribute( "label", "(A,C)" );
16 BC.setAttribute( "label", "(B,C)" );

```

Listing 8.2: Ajout de sommets, arêtes et attributs

Techniquement cette notion de flux est mise en œuvre à l'aide du paradigme de programmation orienté évènement et de la notion de fonction de rappel que l'on retrouve dans beaucoup de langages en générale et dans la conception des bibliothèques graphiques *AWT* et *Swing* dans Java en particulier. L'idée est qu'un générateur d'évènements appelé « source » se voit enregistrer des écouteurs appelés « puits » (sinks). Quand le générateur a besoin de générer un évènement il invoque pour chacun de ses écouteurs la méthode correspondante à cet évènement avec les informations nécessaires. Ce mécanisme permet à un générateur d'évènements de renseigner plusieurs écouteurs et à un écouteur d'écouter plusieurs générateurs. De plus, moyennant une interface préétablie, ce mécanisme permet de minimiser les interdépendances de code entre les différentes classes. Il y a donc deux phases de génération/écoute. D'abord en amont du graphe, le générateur est un lecteur de fichier de graphe dynamique ou un autre processus répondant à l'interface de générateur d'évènements. Dans ce cas le client est souvent unique, c'est le graphe dynamique. Dans la seconde phase de génération/écoute c'est le graphe dynamique qui devient le générateur des évènements et d'autres applications l'écoutent. Ainsi le graphe est à la fois puits et source d'évènements. On appelle de tels objets « pipes ».

La figure 8.1 montre les générateurs et écouteurs de ces flux. Chaque colonne est génératrice pour sa colonne de droite et à l'écoute de sa colonne de gauche.

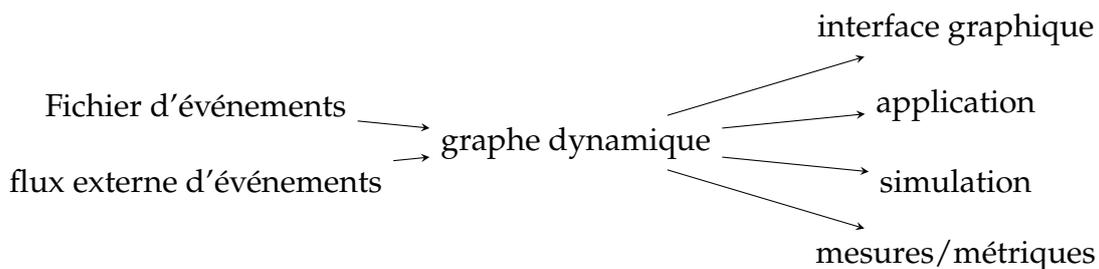


Figure 8.1. Générateurs et écouteurs d'évènements

Les évènements qui mènent au déclenchement de ce processus d'appel aux écouteurs sont les suivants :

- ajout d'une arête ;
- suppression d'une arête ;
- modification d'un attribut d'arête ;
- modification d'un attribut du graphe ;
- ajout d'un sommet ;
- suppression d'un sommet ;
- modification d'un attribut de sommet ;
- début d'une étape.

### Une visualisation stylisée

Une autre fonctionnalité importante du projet est la possibilité de visualiser et de mettre en forme des graphes, on parle de *layout* en anglais. La visualisation des graphes n'est pas une fonctionnalité nouvelle, elle existe dans la plupart des bibliothèques de graphes, néanmoins cette visualisation et surtout son processus de mise en forme (le *layout*) est dynamique et s'adapte aux changements du graphe. La figure 8.2 montre la mise en forme de deux graphes.

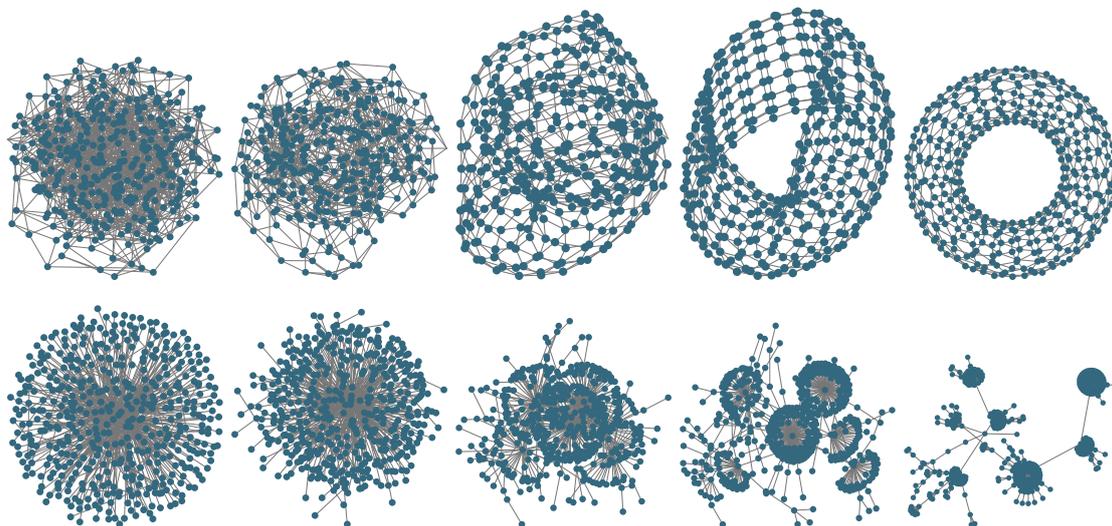


Figure 8.2. Mise en forme de graphe avec *GraphStream*. La première ligne présente plusieurs étapes de la mise en forme d'un graphe torique, la seconde ligne montre plusieurs étapes de la mise en forme d'un graphe aléatoire avec une distribution des degrés en loi de puissance.

De plus *GraphStream* propose une manière originale de personnaliser l'affichage en utilisant la notion de style comme le fait le langage CSS pour les pages *HTML*

et plus largement *XML*. Elle reprend donc l'idée séduisante de la séparation du style et du contenu dans un document. Les feuilles de style contiennent les informations concernant l'apparence graphique alors que les autres données concernant l'affichage et le *layout* du graphe sont séparées. Illustrons le propos à l'aide d'un exemple. Considérons l'extrait de code précédent définissant un graphe à 3 sommets. Définissons maintenant une feuille de style de la même manière que cela pourrait être fait pour une page Web en respectant le formalisme CSS avec des sélecteurs auxquels sont associés des couples attribut/valeur, listing 8.3.

```

1 node {
2     shape: box;
3     size-mode: fit;
4     fill-color: white;
5     stroke-mode: plain;
6     stroke-width: 2px;
7     stroke-color: #33687e;
8     text-style: bold;
9     text-color: black;
10 }
11 edge {
12     size: 2px;
13     shape: cubic-curve;
14     color: #666666;
15     text-align: aside;
16     text-style: bold;
17     text-color: black;
18 }

```

Listing 8.3: Feuille de style

Si cette feuille de style est enregistrée dans un fichier `style.css`, elle peut être affectée à un graphe en y ajoutant un attribut :

```
graph.setAttribute( "ui.stylesheet", "url(style.css)" );
```

La figure 8.3 montre une représentation du graphe construit précédemment, avec et sans l'utilisation de la feuille de style précédente. Notons parmi les modifications graphiques apportées, la modification des couleurs, de l'épaisseur des dessins, la forme incurvée des arêtes, le style des textes ou encore l'apparence en forme de boîtes ajustées au texte des sommets.

### 8.3 Développement d'un *toy problem*

Après avoir présenté les principales fonctionnalités de la bibliothèque, toujours dans le but de présenter l'outil, nous proposons d'illustrer le développement d'une application simple de type simulation distribuée nécessitant un graphe dynamique comme environnement. Pour mettre en œuvre le graphe dynamique, c'est bien entendu *GraphStream* qui est utilisé. Nous allons voir que les fonctionnalités graphiques de l'outil vont au-delà de la simple représentation de graphes et permettent de représenter le fonctionnement de l'application entière.

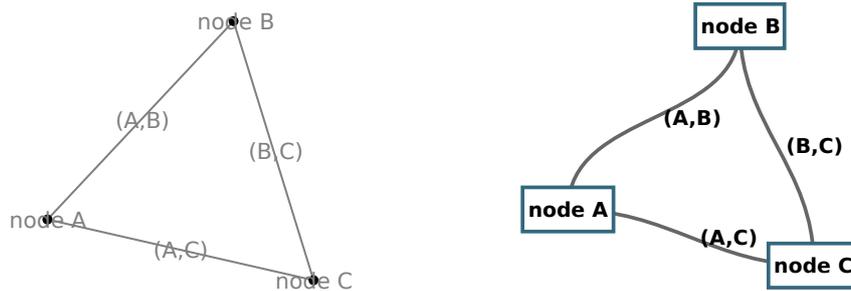


Figure 8.3. Représentation d'un graphe avec (figure de droite) et sans (figure de gauche) l'utilisation d'une feuille de style.

Après avoir détaillé le fonctionnement de l'algorithme que l'on propose de développer ici, nous verrons les quelques points techniques intéressants et importants que sont :

- la technique d'évolution itérative du graphe et de l'algorithme ;
- le mécanisme d'héritage des classes du graphe ;
- le mécanisme de simulation d'application distribué (vision locale) ;
- l'interface graphique.

## Description

Nous proposons d'implémenter à partir d'un modèle simpliste le comportement d'une colonie de fourmis à la recherche de nourriture dans un environnement. L'environnement en question n'est pas une surface continue mais un graphe dynamique, qui peut donc évoluer dans le temps. La colonie de fourmis est localisée sur l'un des sommets du graphe et des sources de nourriture, elles aussi localisées sur des sommets, sont disséminées dans le graphe. Trouver de la nourriture revient à construire un chemin dans le graphe dynamique partant du sommet contenant le nid et arrivant au sommet d'une source de nourriture. Nous souhaitons mimer le comportement localisé et individuel de chaque fourmi en espérant observer l'apparition d'un comportement collectif grâce au mécanisme de stigmergie tel que décrit dans [25].

Notre souhait principal dans cette simulation est de reproduire le comportement individuel et localisé des fourmis. Ce comportement est le suivant : une fourmi se déplace aléatoirement dans son environnement en étant attirée par les pistes de phéromones. Pendant son déplacement elle se souvient de son trajet. Si elle rencontre de la nourriture elle en prélève et rentre au nid en empruntant le chemin qu'elle a fait jusqu'à présent et en déposant des phéromones sur son chemin de retour.

Ce comportement individuel va être implémenté dans un objet indépendant capable de se déplacer dans le graphe dynamique. C'est donc une approche orientée agents mobiles que nous retenons et le fonctionnement des fourmis va être itératif, très semblable au comportement décrit dans [18]. Pour permettre

l'exécution itérative de l'algorithme, les fourmis sont référencées dans un objet (la colonie) qui donne la main à chaque fourmi.

### Itération principale

Le fonctionnement itératif de l'algorithme fourmi est inséré dans le fonctionnement itératif du graphe dynamique. En effet le graphe dynamique vu par *GraphStream* possède un fonctionnement itératif, étape par étape. L'algorithme fourmi doit donc évoluer en même temps que le graphe dynamique. Cette contrainte pose la question de la vitesse relative de l'application par rapport à celle du graphe. L'un des paramètres du programme concerne donc le nombre d'itérations que font les fourmis entre deux étapes du graphe dynamique.

Le listing 8.4 montre l'exemple de l'initialisation d'un graphe, de la lecture itérative d'évènements et de l'exécution de l'algorithme fourmi. Ligne 1, un graphe est créé. Ligne 2, un objet *FileSource* est créé, cet objet est responsable de l'ouverture et de la lecture du fichier contenant les évènements. L'objet *FileSource* est également responsable de la génération des évènements à l'attention du graphe. La ligne 3 enregistre donc le graphe en tant qu'écouteur du *FileSource*. La ligne 4 ordonne au *FileSource* de commencer la lecture de l'entête du fichier. La ligne 6 commence la boucle principale qui itère sur les étapes du graphe. Chaque appel à la fonction *nextStep()* va lire les évènements du graphe séquentiellement jusqu'à la prochaine étape. La ligne 7 initie la boucle qui va lancer un nombre de fois prédéfini l'algorithme fourmi (ligne 8).

```

1 Graph graph = new DefaultGraph("environnement");
2 FileSource fileSource = FileSourceFactory.readerFor( "graphFile.dgs" );
3 fileSource.addSink( graph );
4 fileSource.begin( "graphFile.dgs" );
5
6 while( fileSource.nextStep() )
7     for( int i=0; i < nbIterationPerStep; i++ )
8         colony.go();

```

Listing 8.4: Lecture et exécution itératives

### Héritage

Pour faciliter la conception et spécialiser le graphe aux exigences de l'application, il est possible d'utiliser l'héritage sur les classes de *GraphStream*. Dans notre exemple la spécialisation va servir à implémenter l'idée des phéromones et du mécanisme d'évaporation. En effet, ici les arêtes sont considérées comme les pistes de l'environnement que parcourent les fourmis, c'est donc elles qui reçoivent les dépôts de phéromones. Le graphe quant à lui est l'environnement, c'est donc lui qui est chargé de l'évaporation des pistes de phéromones. Le listing 8.5 donne un exemple de spécialisation des arêtes du graphe.

```

1 public class Piste extends SingleEdge {
2     protected static final double EVAPORATION = 0.1;
3     protected double taux = 0.0;
4
5     public void depotPheromone(double quantite) {
6         taux += quantite;
7     }
8
9     public void evaporation() {
10        taux *= 1 - EVAPORATION;
11    }
12
13    public double getTaux() {
14        return taux;
15    }
16    ...
17 }

```

Listing 8.5: Spécialisation des arêtes

## Simulation du fonctionnement décentralisé

Comme nous l'avons vu, le souhait principal est de mimer le comportement distribué des véritables colonies de fourmis à l'aide d'agents mimant les fourmis. Ces objets doivent comme les fourmis disposer d'une vision localisée de l'environnement. Cela est naturellement fait avec le graphe en ne donnant accès à chaque fourmi qu'à son sommet courant. À partir de ce sommet, une fourmi ne peut accéder à un élément distant du graphe, seul le voisinage direct de son sommet est accessible. Pour manipuler un élément distant elle doit se rendre vers celui-ci en parcourant le graphe. Le listing 8.6 donne un exemple de choix du sommet suivant par une fourmi. Pour simplifier, on suppose que la fourmi choisit l'arête dont le taux de phéromone est maximal, même si en réalité il s'agit d'un choix aléatoire biaisé par le taux. La boucle de la ligne 3 parcourt toutes les arêtes sortantes du sommet courant.

```

1 double tauxMax = 0;
2 Node suivant;
3 for (Piste e : courant.getEachLeavingEdge())
4     if (e.getTaux() > tauxMax) {
5         tauxMax = e.getTaux();
6         suivant = e.getOpposite(courant);
7     }

```

Listing 8.6: Choix du sommet suivant par une fourmi

## Interface graphique

Nous l'avons vu précédemment, *GraphStream* permet la visualisation et la mise en forme de graphes dynamiques. La stylisation permet la modification des couleurs, épaisseurs et autres caractéristiques graphiques des éléments. La figure 8.4 montre

un aperçu de la visualisation du programme. Les sommets et les arêtes sont blancs et la couleur ainsi que l'épaisseur de l'ombre des arêtes varient en fonction de leur quantité de phéromones. Une vidéo de démonstration des autres possibilités de visualisation est disponible à l'adresse <http://youtu.be/XX5rRF6uxow>.

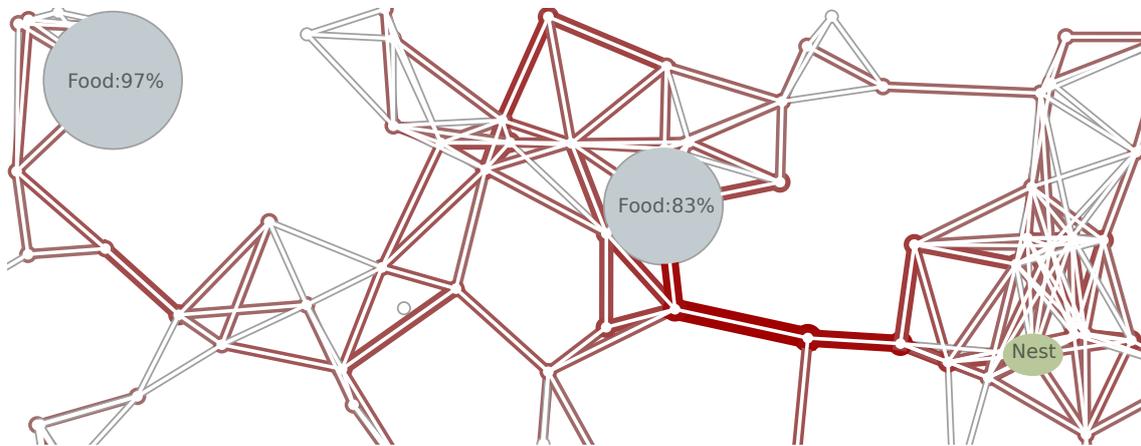


Figure 8.4. Représentation graphique du graphe environnement des fourmis à la recherche de nourriture.

Il est possible d'aller plus loin avec l'affichage et d'utiliser des composants graphiques supplémentaires au graphe (*sprites*). Il est ainsi possible de représenter des objets graphiques dans l'espace du graphe avec ses propres coordonnées ou attaché aux éléments. Dans notre exemple il est possible de visualiser les fourmis en associant un de ces objets graphiques à chaque fourmi. Le listing 8.7 illustre la création d'un de ces objets graphiques. À la ligne 1 un graphe est créé, à la ligne 3 un objet `SpriteManager` est créé pour gérer les composants graphiques additionnels, les *sprites*. La méthode `display()` du graphe affiche le graphe et sa mise en forme. La ligne 6 montre la création d'un tel élément. La ligne 7 permet d'associer l'élément graphique à un élément du graphe (ici à un sommet d'identifiant "node1"). La ligne 8 permet de positionner l'élément par rapport à son sommet d'attache. Les lignes 9-11 ajoutent des attributs à l'élément tout comme il est possible d'ajouter des attributs aux éléments classiques de graphes, ici un `label` et une personnalisation du style avec des clés de style spécifique aux *sprites*.

## 8.4 Conclusion

Devant le besoin d'outils de manipulation de graphes dynamiques indépendants de tout domaine d'application, nous avons proposé `GraphStream`, où le modèle de graphe dynamique considéré repose sur un flux d'évènements constituant la dynamique du graphe. Ce modèle est proche dans sa vision événementielle des applications liées à la ré-optimisation. Le projet propose également une visualisation et une mise en forme des graphes dynamiques. Cette visualisation comprend une stylisation de l'affichage avec des feuilles de style ainsi que la possibilité d'afficher des objets graphiques autres que les noeuds et les arêtes, pouvant être associés à des sémantiques définies par l'utilisateur et ainsi servir des domaines d'application variés.

```
1 Graph graph = new SingleGraph();
2 [...]
3 SpriteManager sprites = new SpriteManager(graph);
4 graph.display();
5 [...]
6 Sprite ant = sprites.addSprite("ant1");
7 sprite.attachToNode("node1");
8 sprite.position(6, 0, (float)((Math.PI * 2) * Math.random()), Style.Units.
    PX);
9 sprite.setAttribute("ui.label", "1");
10 sprite.setAttribute("ui.style",
11     "color:#FFFFFFF00;sprite-shape:text-ellipse;image:url('ant.png');");
```

Listing 8.7: Utilisation de sprites

## Conclusion

Ce manuscrit s'achève et avec lui s'arrête l'introspection qui me fut nécessaire pour pouvoir de nouveau présenter critiquer et apprécier les travaux du passé. Les premières relectures d'anciens papiers amenèrent parfois de l'amusement, parfois de la nostalgie, quelques remords, mais aussi des sueurs froides quand, n'ayant que survolé un résultat oublié et mal expliqué, j'en remettais en cause la véracité avant de me convaincre que non, tout n'est pas faux.

Je ne conclurai pas en détail sur chaque projet, ayant déjà porté quelques analyses çà et là, mais je peux formuler quelques perspectives sur les bases de ce qui a été présenté ici.

J'ai cessé presque toute activité conservant les aspects mobilité et algorithmique distribuée. Des idées, des morceaux d'articles et quelques lignes de codes se cachent encore dans mon disque dur, mais ce train-là est probablement parti.

J'ai récemment repris mes activités dans le domaine de l'analyse des réseaux maritimes. La reconstitution du réseau maritime mondiale à l'aide de données AIS (voir chapitre 5, section 5.1). Mon but est toujours de proposer des métriques adaptées aux graphes dynamiques et qui ne soient pas que de simples séries temporelles pour qualifier les processus observés.

Ma principale activité est maintenant dans le domaine de l'algorithmique où j'espère pouvoir continuer à contribuer aux problématiques de connexité et autres problèmes adaptés des graphes statiques vers le dynamique.

La génération de graphes dynamiques est également d'un grand intérêt pour moi (voir notamment la section 7.5 du chapitre 7). Qu'elle soit purement algorithmique et stochastique, comme dans la thèse de Vincent Bridonneau (co-encadré avec Frédéric Guinand) ou bien qu'elle fasse appel à l'apprentissage automatique dont on connaît l'omniprésence dans tous les champs de la recherche.



## Références

- [1] Gaël-Cédric Aboue-Nze, Frédéric Guinand, et Yoann Pigné. 2009. Impact of Obstacles on the Degree of Mobile Ad Hoc Connection Graphs. In *Proceedings of the 2009 Fifth International Conference on Networking and Services (ICNS '09)*, IEEE Computer Society, Valencia, Spain, 332-337. DOI:<https://doi.org/10.1109/ICNS.2009.36>
- [2] Gaël-Cédric Aboue-Nze, Frédéric Guinand, et Yoann Pigné. 2011. Impact of Square Environment on the Connectivity in Finite Ad Hoc Networks. In *2011 The 14th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, Brest, France, 1-5. Consulté à l'adresse <https://ieeexplore.ieee.org/document/6081589>
- [3] R. K. Ahuja, T. L. Magnanti, et J. B. Orlin. 2013. *Network Flows: Theory, Algorithms, and Applications*. Pearson.
- [4] Eleni C. Akrida et Paul G. Spirakis. 2015. On Verifying and Maintaining Connectivity of Interval Temporal Networks. In *Algorithms for Sensor Systems (Lecture Notes in Computer Science)*, Springer International Publishing, Cham, 142-154. DOI:[https://doi.org/10.1007/978-3-319-28472-9\\_11](https://doi.org/10.1007/978-3-319-28472-9_11)
- [5] Julien Arino, Jonathan R. Davis, David Hartley, Richard Jordan, Joy M. Miller, et P. van den Driessche. 2005. A Multi-Species Epidemic Model with Spatial Dynamics. *Mathematical Medicine and Biology: A Journal of the IMA* 22, 2 (juin 2005), 129-142. DOI:<https://doi.org/10.1093/imammb/dqi003>
- [6] Stefan Balev, Juan Luis Laredo Jiménez, Ioannis Lamprou, Yoann Pigné, et Eric Sanlaville. 2020. Cops and Robbers on Dynamic Graphs: Offline and Online Case. In *Structural Information and Communication Complexity (Lecture Notes in Computer Science)*, Springer International Publishing, Paderborn, Germany, 203-219. DOI:[https://doi.org/10.1007/978-3-030-54921-3\\_12](https://doi.org/10.1007/978-3-030-54921-3_12)
- [7] Hugo Barbosa, Marc Barthélemy, Gourab Ghoshal, Charlotte R. James, Maxime Lenormand, Thomas Louail, Ronaldo Menezes, José J. Ramasco, Filippo Simini, et Marcello Tomasini. 2018. Human Mobility: Models and Applications. *Physics Reports* 734, (mars 2018), 1-74. DOI:<https://doi.org/10.1016/j.physrep.2018.01.001>
- [8] C. Bettstetter, G. Resta, et P. Santi. 2003. The Node Distribution of the Random Waypoint Mobility Model for Wireless Ad Hoc Networks. *IEEE Trans. Mobile Computing* 2, 3 (juillet–septembre 2003), 257-269. Consulté à l'adresse [citeseer.ist.psu.edu/bettstetter03node.html](http://citeseer.ist.psu.edu/bettstetter03node.html)

- [9] Sandeep Bhadra et Afonso Ferreira. 2003. Complexity of Connected Components in Evolving Graphs and the Computation of Multicast Trees in Dynamic Networks. In *International Conference on Ad-Hoc Networks and Wireless*, Springer, 259-270. DOI:[https://doi.org/10.1007/978-3-540-39611-6\\_23](https://doi.org/10.1007/978-3-540-39611-6_23)
- [10] Binh-Minh Bui-Xuan, A. Ferreira, et A. Jarry. 2003. Computing Shortest, Fastest, and Foremost Journeys in Dynamic Networks. *International Journal of Foundations of Computer Science* 14, 02 (avril 2003), 267-285. DOI:<https://doi.org/10.1142/S0129054103001728>
- [11] Arnaud Casteigts, Serge Chaumette, Frédéric Guinand, et Yoann Pigné. 2013. Distributed Maintenance of Anytime Available Spanning Trees in Dynamic Networks. In *Ad-Hoc, Mobile, and Wireless Network (Lecture Notes in Computer Science)*, Springer Berlin Heidelberg, Wroclaw, Poland, 99-110. DOI:[https://doi.org/10.1007/978-3-642-39247-4\\_9](https://doi.org/10.1007/978-3-642-39247-4_9)
- [12] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, et Nicola Santoro. 2012. Time-Varying Graphs and Dynamic Networks. *International Journal of Parallel, Emergent and Distributed Systems* 27, 5 (octobre 2012), 387-408. DOI:<https://doi.org/10.1080/17445760.2012.668546>
- [13] Arnaud Casteigts, Ralf Klasing, Yessin M. Neggaz, et Joseph G. Peters. 2015. Efficiently Testing T-Interval Connectivity in Dynamic Graphs. In *Algorithms and Complexity (Lecture Notes in Computer Science)*, Springer International Publishing, Cham, 89-100. DOI:[https://doi.org/10.1007/978-3-319-18173-8\\_6](https://doi.org/10.1007/978-3-319-18173-8_6)
- [14] Aaron Clauset, Cosma Rohilla Shalizi, et M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (novembre 2009), 661-703. DOI:<https://doi.org/10.1137/070710111>
- [15] Andrea E. F. Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, et Riccardo Silvestri. 2008. Flooding Time in Edge-Markovian Dynamic Graphs. In *Proceedings of the Twenty-Seventh ACM Symposium on Principles of Distributed Computing (PODC '08)*, Association for Computing Machinery, New York, NY, USA, 213-222. DOI:<https://doi.org/10.1145/1400751.1400781>
- [16] Camil Demetrescu et Giuseppe F. Italiano. 2006. Fully Dynamic All Pairs Shortest Paths with Real Edge Weights. *Journal of Computer and System Sciences* 72, 5 (août 2006), 813-837. DOI:<https://doi.org/10.1016/j.jcss.2005.05.005>
- [17] Camil Demetrescu et Giuseppe F. Italiano. 2007. Algorithmic Techniques for Maintaining Shortest Routes in Dynamic Networks. *Electronic Notes in Theoretical Computer Science* 171, 1 (avril 2007), 3-15. DOI:<https://doi.org/10.1016/j.entcs.2006.11.006>
- [18] Marco Dorigo et Thomas Stützle. 2004. *Ant Colony Optimization*. The MIT Press. DOI:<https://doi.org/10.7551/mitpress/1290.001.0001>
- [19] Bernabé Dorronsoro, Patricia Ruiz, Grégoire Danoy, Yoann Pigné, et Pascal Bouvry. 2014. *Evolutionary Algorithms for Mobile Ad Hoc Networks: Bouvry/Evolutionary Algorithms for Mobile Ad Hoc Networks*. John Wiley & Sons, Inc., Hoboken, NJ. DOI:<https://doi.org/10.1002/9781118833209>

- [20] César Ducruet. 2017. Multilayer Dynamics of Complex Spatial Networks: The Case of Global Maritime Flows (1977–2008). *Journal of Transport Geography* 60, (avril 2017), 47-58. DOI:<https://doi.org/10.1016/j.jtrangeo.2017.02.007>
- [21] Antoine Fremont et Cesar Ducruet. 2005. THE EMERGENCE OF A MEGA-PORT - FROM THE GLOBAL TO THE LOCAL, THE CASE OF BUSAN\*. *Tijd voor Econ & Soc Geog* 96, 4 (septembre 2005), 421-432. DOI:<https://doi.org/10.1111/j.1467-9663.2005.00473.x>
- [22] Daniele Frigioni, Alberto Marchetti-Spaccamela, et Umberto Nanni. 1996. Fully Dynamic Output Bounded Single Source Shortest Path Problem. In *Proceedings of the Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '96)*, Society for Industrial and Applied Mathematics, USA, 212-221.
- [23] Carlos Gómez-Calzado, Arnaud Casteigts, Alberto Lafuente, et Mikel Larrea. 2015. A Connectivity Model for Agreement in Dynamic Systems. In *Euro-Par 2015: Parallel Processing (Lecture Notes in Computer Science)*, Springer, Berlin, Heidelberg, 333-345. DOI:[https://doi.org/10.1007/978-3-662-48096-0\\_26](https://doi.org/10.1007/978-3-662-48096-0_26)
- [24] Marta C. González, César A. Hidalgo, et Albert-László Barabási. 2008. Understanding Individual Human Mobility Patterns. *Nature* 453, 7196, 7196 (juin 2008), 779-782. DOI:<https://doi.org/10.1038/nature06958>
- [25] Plerre-P. Grassé. 1959. La reconstruction du nid et les coordinations interindividuelles chez *Bellicositermes natalensis* et *Cubitermes* sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Ins. Soc* 6, 1 (mars 1959), 41-80. DOI:<https://doi.org/10.1007/BF02223791>
- [26] Frédéric Guinand et Yoann Pigné. 2015. Time Considerations for the Study of Complex Maritime Networks. In *Maritime Networks Spatial Structures and Time Dynamics* (César Ducruet). Routledge, 163-189. Consulté à l'adresse <https://www.routledge.com/products/9781138911253>
- [27] Frédéric Guinand et Yoann Pigné. 2016. Reachability in Maritime Transportation Networks. In *Complex Networks: From Theory to Interdisciplinary Applications*, Marseille. Consulté à l'adresse <https://www.cpt.univ-mrs.fr/~barrat/COMPLEXNETS2016/complexnets2016.org/program.html>
- [28] Petter Holme et Jari Saramäki. 2012. Temporal Networks. *Physics Reports* 519, 3 (octobre 2012), 97-125. DOI:<https://doi.org/10.1016/j.physrep.2012.03.001>
- [29] Charles Huyghues-Despointes, Binh-Minh Bui-Xuan, et Clémence Magnien. 2016. Forte Delta-Connexité Dans Les Flots de Liens. Consulté 24 octobre 2022 à l'adresse <https://hal.archives-ouvertes.fr/hal-01305128>

- [30] M. Karsai, M. Kivelä, R. K. Pan, K. Kaski, J. Kertész, A.-L. Barabási, et J. Saramäki. 2011. Small but Slow World: How Network Topology and Burstiness Slow down Spreading. *Phys. Rev. E* 83, 2 (février 2011), 025102. DOI:<https://doi.org/10.1103/PhysRevE.83.025102>
- [31] David Kempe, Jon Kleinberg, et Amit Kumar. 2002. Connectivity and Inference Problems for Temporal Networks. *Journal of Computer and System Sciences* 64, 4 (juin 2002), 820-842. DOI:<https://doi.org/10.1006/jcss.2002.1829>
- [32] Matthieu Latapy et Clemence Magnien. 2007. Measuring Fundamental Properties of Real-World Complex Networks. DOI:<https://doi.org/10.48550/arXiv.cs/0609115>
- [33] Matthieu Latapy, Tiphaine Viard, et Clémence Magnien. 2018. Stream Graphs and Link Streams for the Modeling of Interactions over Time. *Social Network Analysis and Mining* 8, 1 (octobre 2018), 61. DOI:<https://doi.org/10.1007/s13278-018-0537-7>
- [34] Dimitri Lefebvre, Sara Rachidi, Edouard Leclercq, et Yoann Pigné. 2018. Temporal Fault Diagnosis for K-Bounded Non-Markovian SPN. In *5th International Conference on Control, Decision and Information Technologies, CoDIT 2018, Thessaloniki, Greece, April 10-13, 2018*, IEEE, Thessaloniki, Greece, 271-276. DOI:<https://doi.org/10.1109/CoDIT.2018.8394843>
- [35] Dimitri Lefebvre, Sara Rachidi, Edouard Leclercq, et Yoann Pigné. 2018. Diagnosis of Structural and Temporal Faults for K-Bounded Non-Markovian Stochastic Petri Nets. *IEEE Transactions on Systems, Man, and Cybernetics: Systems PP*, (novembre 2018), 1-13. DOI:<https://doi.org/10.1109/TSMC.2018.2875726>
- [36] Tai-Yu Ma et Yoann Pigné. 2019. Bayesian Dynamic Linear Model with Adaptive Parameter Estimation for Short-Term Travel Speed Prediction. *Journal of Advanced Transportation* 2019, (2019), 10. DOI:<https://doi.org/10.1155/2019/5314520>
- [37] A. Marvuglia, B. Rugani, G. Rios, Y. Pigné, E. Benetto, et L. Tiruta-Barna. 2013. Using Graph Search Algorithms for a Rigorous Application of Emergy Algebra Rules. *Metallurgical Research & Technology* 110, 1 (2013), 8. DOI:<https://doi.org/10.1051/metal/2013050>
- [38] Djamila Moulay et Yoann Pigné. 2013. A Metapopulation Model for Chikungunya Including Populations Mobility on a Large-Scale Network. *Journal of Theoretical Biology* 318, (février 2013), 129-139. DOI:<https://doi.org/10.1016/j.jtbi.2012.11.008>
- [39] Tomás Navarrete Gutiérrez, Benedetto Rugani, Yoann Pigné, Antonino Marvuglia, et Enrico Benetto. 2016. On the Complexity of Life Cycle Inventory Networks: Role of Life Cycle Processes with Network Analysis. *Journal of Industrial Ecology* 20, 5 (octobre 2016), 1094-1107. DOI:<https://doi.org/10.1111/jiec.12338>

- [40] Stefano Pallottino et Maria Grazia Scutellà. 1998. Shortest Path Algorithms In Transportation Models: Classical and Innovative Aspects. In *Equilibrium and Advanced Transportation Modelling*, Patrice Marcotte et Sang Nguyen (éd.). Springer US, Boston, MA, 245-281. DOI:[https://doi.org/10.1007/978-1-4615-5757-9\\_11](https://doi.org/10.1007/978-1-4615-5757-9_11)
- [41] Yoann Pigné, Grégoire Danoy, et Pascal Bouvry. 2010. A Platform for Realistic Online Vehicular Network Management. In *IEEE International Workshop on Management of Emerging Networks and Services*, IEEE Computer Society, Miami, FL, USA, 615-619. DOI:<https://doi.org/10.1109/GLOCOMW.2010.5700389>
- [42] Yoann Pigné, Grégoire Danoy, et Pascal Bouvry. 2011. Sensitivity Analysis for a Realistic Vehicular Mobility Model. In *Proceedings of the First ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet '11)*, ACM, Miami, FL, USA, 31-38. DOI:<https://doi.org/10.1145/2069000.2069007>
- [43] Yoann Pigné, Grégoire Danoy, et Pascal Bouvry. 2011. A Vehicular Mobility Model Based on Real Traffic Counting Data. In *Proceedings of the Third International Conference on Communication Technologies for Vehicles (Nets4Cars/Nets4Trains'11)*, Springer-Verlag, Berlin, Heidelberg, 131-142. DOI:[https://doi.org/10.1007/978-3-642-19786-4\\_12](https://doi.org/10.1007/978-3-642-19786-4_12)
- [44] Yoann Pigné et Frédéric Guinand. 2010. Short and Robust Communication Paths in Dynamic Wireless Networks. In *Swarm Intelligence (Lecture Notes in Computer Science)*, Springer Berlin Heidelberg, Brussels, Belgium, 520-527. DOI:[https://doi.org/10.1007/978-3-642-15461-4\\_52](https://doi.org/10.1007/978-3-642-15461-4_52)
- [45] Yoann Pigné, Tomás Navarrete Navarrete Gutiérrez, Thomas Gibon, Thomas Schaubroeck, Emil Popovici, Allan Hayato Shimako, Enrico Benetto, et Ligia Tiruta-Barna. 2020. A Tool to Operationalize Dynamic LCA, Including Time Differentiation on the Complete Background Database. *Int J Life Cycle Assess* 25, 2 (février 2020), 267-279. DOI:<https://doi.org/10.1007/s11367-019-01696-6>
- [46] Sara Rachidi, Edouard Leclercq, Yoann Pigné, et Dimitri Lefebvre. 2018. Moving Average Control Chart for the Detection and Isolation of Temporal Faults in Stochastic Petri Nets. In *23rd IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2018, Torino, Italy, September 4-7, 2018*, IEEE, Torino, Italy, 493-498. DOI:<https://doi.org/10.1109/ETFA.2018.8502633>
- [47] G. Ramalingam et Thomas Reps. 1996. An Incremental Algorithm for a Generalization of the Shortest-Path Problem. *Journal of Algorithms* 21, 2 (septembre 1996), 267-305. DOI:<https://doi.org/10.1006/jagm.1996.0046>
- [48] Riadh Saada, Yoann Pigné, et Damien Olivier. 2017. New Selection Strategies of Actor's Substitute in DARA for Connectivity Restoration in WSAWs. In *Ad Hoc Networks (Lecture Notes of the Institute for Computer Sciences, Social Informatics et Telecommunications Engineering)*, Springer International Publishing, Ottawa, Canada, 38-49. DOI:[https://doi.org/10.1007/978-3-319-51204-4\\_4](https://doi.org/10.1007/978-3-319-51204-4_4)

- [49] M. Seredynski, G. Danoy, M. Tabatabaei, P. Bouvry, et Y. Pigné. 2012. Generation of Realistic Mobility for VANETs Using Genetic Algorithms. In *2012 IEEE Congress on Evolutionary Computation (CEC)*, Brisbane, QLD, Australia, 1-8. DOI:<https://doi.org/10.1109/CEC.2012.6252987>
- [50] Allan Hayato Shimako, Ligia Tiruta-Barna, Yoann Pigné, Enrico Benetto, Tomás Navarrete Gutiérrez, Pascal Guiraud, et Aras Ahmadi. 2016. Environmental Assessment of Bioenergy Production from Microalgae Based Systems. *Journal of Cleaner Production* 139, (décembre 2016), 51-60. DOI:<https://doi.org/10.1016/j.jclepro.2016.08.003>
- [51] Ligia Tiruta-Barna, Yoann Pigné, Tomás Navarrete Gutiérrez, et Enrico Benetto. 2016. Framework and Computational Tool for the Consideration of Time Dependency in Life Cycle Inventory: Proof of Concept. *Journal of Cleaner Production* 116, (mars 2016), 198-206. DOI:<https://doi.org/10.1016/j.jclepro.2015.12.049>
- [52] Mathilde Vernet. 2020. Modèles et Algorithmes Pour Les Graphes Dynamiques. Theses. Normandie Université. Consulté 12 octobre 2022 à l'adresse <https://tel.archives-ouvertes.fr/tel-03030851>
- [53] Mathilde Vernet, Maciej Drozdowski, Yoann Pigné, et Eric Sanlaville. 2018. Successive Shortest Path Algorithm for Flows in Dynamic Graphs. In *16th Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, Paris, France, 51-54. Consulté à l'adresse [http://ctw18.lipn.univ-paris13.fr/CTW18\\_Proceedings.pdf](http://ctw18.lipn.univ-paris13.fr/CTW18_Proceedings.pdf)
- [54] Mathilde Vernet, Maciej Drozdowski, Yoann Pigné, et Eric Sanlaville. 2020. A Theoretical and Experimental Study of a New Algorithm for Minimum Cost Flow in Dynamic Graphs. *Discrete Applied Mathematics* (janvier 2020). DOI:<https://doi.org/10.1016/j.dam.2019.12.012>
- [55] Mathilde Vernet, Yoann Pigné, et Eric Sanlaville. 2018. Algorithmique Dans Les Graphes Dynamiques : Le Cas Des Flots. Lorient, France. Consulté à l'adresse <https://hal.archives-ouvertes.fr/hal-02115413>
- [56] Mathilde Vernet, Yoann Pigné, et Éric Sanlaville. 2022. A Study of Connectivity on Dynamic Graphs: Computing Persistent Connected Components. *4OR-Q J Oper Res* (avril 2022). DOI:<https://doi.org/10.1007/s10288-022-00507-3>