Résolution de problèmes multicritères (durée/sécurité) pour la conception de plans d'évacuation de personnes

Ismaila Abderhamane Ndiaye

UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS

École Doctorale MIPTIS

Laboratoire d'Informatique : EA 6300, OC ERL CNRS 6305

THÈSE présentée par :

NDIAYE Ismaïla Abderhamane

soutenue le : 3 Mars 2016

pour obtenir le grade de : Docteur de l'université François - Rabelais de Tours

Discipline / Spécialité : Informatique

Résolution de problèmes multicritères (durée/sécurité) pour la conception de plans d'évacuation de personnes

THÈSE DIRIGÉE PAR :

NÉRON Emmanuel	Professeur, Université François-Rabelais de Tours

RAPPORTEURS :

BRIAND Cyril	Professeur, Université Paul Sabatier Toulouse III
LABADIE Nacima	HDR, Université de Technologie de Troyes

JURY :

BRIAND Cyril	Professeur, Université Paul Sabatier Toulouse III
CARLIER Jacques	Professeur, Université de Technologie de Compiègne
JOUGLET Antoine	HDR, Université de Technologie de Compiègne
LABADIE Nacima	HDR, Université de Technologie de Troyes
MENDOZA Jorge Ernesto	Maître de conférences, Université François Rabelais de Tours
NÉRON Emmanuel	Professeur, Université François Rabelais de Tours

Remerciements

Les travaux réalisés dans cette thèse sont effectués au sein du Laboratoire d'Informatique (EA 6300, OC ERL CNRS 6305) de l'Université François Rabelais de Tours. Ils s'intègrent dans l'appel à projet de l'Agence Nationale de la Recherche portant la référence ANR-11-SECU-002-01 projet DSS_EVAC_LOGISTIQUE (Appel CSOSG 2011).

Je voudrais commencer par remercier mon directeur de recherche Emmanuel Néron pour son encadrement de qualité, son calme et sa bonne humeur durant ces trois années de thèse. Malgré ses responsabilités administratives grandissantes, il a toujours su se rendre disponible afin de suivre l'avancement de mes travaux, me conseiller si nécessaire et me mettre dans les meilleures conditions possibles de réussite.

Ensuite, j'aimerais remercier l'ensemble du personnel de Polytech'Tours et du Laboratoire d'Informatique avec qui j'ai partagé de très bons moments autant sur le plan professionnel qu'humain.
En France, il est de coutume de dire que "les petits ruisseaux font les grandes rivières" et j'ajouterai qu'il en va de même pour les rencontres et expériences humaines qui jalonnent et forgent nos parcours. Aussi, j'aimerais plus particulièrement associer à ces remerciements l'ancien directeur de Polytech Tours, Christian Proust, pour sa disponibilité, sa bonne humeur et ses conseils tout au long de ces années. Le directeur du département informatique de Polytech Tours Pierre Gaucher sans qui je n'aurais pas pris goût à l'enseignement. Le directeur du Laboratoire d'Informatique Jean-Charles Billault pour son humanisme et sa pédagogie qui ont su susciter mon intérêt pour la recherche opérationnelle. Le directeur d'équipe d'ordonnancement et Conduite Vincent T'kindt avec qui j'ai fait mes premières armes sur des problèmes d'ordonnancement d'atelier. Emmanuel Dewaele, ancien employé de la Compagnie des Mobilités qui m'a permis de me spécialiser dans les systèmes d'informations géographiques, spécialisation qui a mené à cette thèse sur le routage de masse. Nicolas Monmarché pour son aide précieuse dans l'avancement de mes recherches.
A toutes ces personnes ayant directement ou indirectement contribuées à mon parcours, je souhaite leur exprimer ma profonde reconnaissance.

De même, je souhaite remercier mes collègues doctorants du Laboratoire d'Informatique pour la bonne humeur et les discussions scientifiques ou non que nous avons pu avoir avec une mention particulière au trio Nhat Vinh VO, Mohamed Cissé sans oublier mon amie et complice Kaouthar (Boukebab) Deghdak avec qui j'ai passé des moments inoubliables

durant ces trois années de thèse.

En outre, ce travail de thèse ne serait pas aussi abouti sans l'aide des étudiants du département d'informatique de Polytech Tours. En effet, ces derniers m'ont permis d'avancer plus rapidement dans mes recherches à travers des projets tutorés. Parmi eux, des contributions singulières ont été effectuées par Anaïs Linot et Hamza Ayoub. Je leur exprime ma gratitude pour l'aide précieuse qu'ils m'ont apportée.

Enfin, j'aimerais dédier ce travail de thèse aux membres de ma famille sans qui cette aventure n'aurait pas été possible. A mes oncles Mbaye Guéye et Ady Diaw pour leurs assistances et leurs conseils de haute portée. A mes chères petites sœurs Ndeye Astou et Ramatoulaye Ndiaye mes complices de tous les jours pour leur joie de vivre et l'émulation dans nos parcours respectifs. A feu mon père Omar Malick Ndiaye qui m'a inculqué les notions de vertus et du sens d'un travail bien accompli. Enfin et non des moindres, j'aimerais adresser un vibrant hommage à ma chère et tendre mère, Fatou Aïdara, pour son amour sans failles, son soutien indéfectible, et qui depuis ma tendre jeunesse n'a ménagé aucun effort pour mes études. Je lui en suis éternellement reconnaissant.

Résumé

Les travaux présentés dans cette thèse visent à proposer des méthodes de routage d'une population de masse à travers un réseau perturbé dont les données varient dans le temps pour l'aide à la conception de plan d'évacuation. Ce problème s'illustre parfaitement en cas de catastrophe d'origine humaine ou naturelle où les populations (potentiellement) impactées par ces sinistres doivent quitter leur lieux de vie pour une période pouvant aller d'un à plusieurs jours.

Dans la littérature, ces routages de masse sont souvent modélisés comme des problèmes de flots dynamiques dont l'objectif est de minimiser la durée globale du transfert des individus depuis un certain nombre de points de départs dangereux vers des points d'arrivée sûrs. Toutefois, peu de travaux prennent en compte la notion de sécurité durant ce routage et encore moins le déploiement d'agents (policiers, sapeur-pompiers, ambulanciers,...) pouvant sécuriser et/ou faciliter le déplacement des personnes. Dans ce cadre, la notion de sécurité peut être vue comme un élément ayant une influence sur la qualité de vie des personnes dont nous organisons le déplacement. En effet, elle peut être liée à un nuage radioactif, un feu de forêt, un tsunami, un tremblement de terre ou une inondation pouvant rendre les chemins empruntés dangereux et moins praticables pour les individus. Ainsi nous avons un problème où deux critères potentiellement conflictuels sont à optimiser afin de garantir que les individus aient le moins de dommages possibles dus à l'évacuation. Pour minimiser la durée de l'évacuation, il faudra prendre un ensemble de plus courts chemins alors que l'optimisation de la sécurité consistera à faire des détours pour prendre des chemins éventuellement plus longs mais aussi plus sûrs.

Pour résoudre ce problème d'optimisation, une approche itérative permettant de résoudre différents niveaux de complexité est proposée. Notre première contribution porte sur l'amélioration de la complexité des algorithmes concernant le problème d'optimisation de la durée moyenne d'évacuation puis par extension, nous prenons en compte la notion de sécurité durant ce routage grâce à une approche lexicographique. Notre seconde contribution porte sur la modélisation de la notion de sécurité comme un problème de flot généralisé avec pertes. Enfin, notre dernière contribution porte sur le déploiement des forces de secours dans un contexte de flot généralisé avec perte où ces dernières seront positionnées sur les zones critiques.

Ce travail s'inscrit dans le cadre du projet Franco-Allemand DSS_EVAC_LOGISTIC qui a bénéficié d'une aide de l'Agence National de la Recherche portant la référence ANR-11-SECU-002-01projet DSS_EVAC_LOGISTIQUE (Appel CSOSG 2011).

Mots clés : réseau dynamique, évacuation, transbordement, modèle microscopique, modèle macroscopique, flot généralisé, multicritère, durée, sécurité, routage.

Abstract

The work presented in this thesis aims to propose methods for routing a mass population through a disturbed network whose data vary over time. This problem can be raised by disasters due to humans or natural events where people (potentially) affected have to leave their living places for a period of one to several days.

In the literature, mass routing are often modeled as dynamic flow problems whose objective is to minimize the overall duration of the evacuation process from a set of gathering points towards another set of shelter locations. However few papers take into account the concept of safety during this routing nor deploying task forces that can secure or facilitate this process. In this context, the safety can be seen as a danger affecting the quality of life of people we organize the trip. So, the safety can be seen as a danger that influence the health of the people we are trying to evacuate. Indeed, hazardous event can be related to a radioactive cloud, a fire, a tsunami, an earthquake or a flooding which make some of paths becoming dangerous or less usable by evacuees. So we have an optimization problem where two conflicting criteria are to be optimized to guarantee that the individuals will have least damage possible during the evacuation. Then, during the evacuation, it will be necessary to have a set of shorter paths(ways) while the optimization of the safety(security) will consist in making detours to take longer but safer paths.

To solve this optimization problem, an iterative approach allowing to solve various levels of complexity is proposed. Our first contribution concerns the improvement of the complexity of the algorithms regarding the optimization the average duration of the evacuation, then by extension, we take into account the notion of safety by applying a lexicographical approach. Our second contribution concerns the modelling of the notion of safety as a generalized flow with losses problem. Finally, our last contribution concerns the deployment of the task forces over critical points of the network to improve their safety.

This research has been supported by the French National Agency of Research ANR-11-SECU-002-01- project DSS_EVAC_LOGISTIQUE.

Keywords : dynamic network, evacuation, transshipment, macroscopic model, microscopic model, generelized flow, multicriteria, duration, safety, routing.

Table des matières

Liste des tableaux

Table des figures

Introduction générale

Depuis les années 90, les problèmes d'évacuation d'infrastructures telles que les avions, les tunnels, les bateaux, les bâtiments et les stades ont largement été étudiés dans la littérature [Hamacher et Tjandra, 2002]. De manière générale, il faut router (i.e. acheminer) un certain nombre d'individus depuis différents points de départ vers des points d'arrivée potentiels sachant que ces derniers ont une capacité limitée d'accueil. Aussi, ce routage s'effectue à travers un réseau dynamique dont l'une des premières modélisations a été proposée par [Ford et Fulkerson, 1962] à l'aide d'un graphe étendu dans le temps. Les sommets de ce graphe sont des intersections ou des points d'intérêt alors que les arcs, quant à eux, peuvent être vus comme des couloirs ou des routes ayant des attributs (temps de traversée, capacité, sécurité,...) et reliant deux emplacements adjacents de l'infrastructure étudiée. De manière classique, l'objectif de ces problèmes d'évacuation est la minimisation de la durée de ce routage mais sans prendre en compte des notions comme le risque encouru par les personnes dont il faut organiser le déplacement des points dangereux vers les zones sûres. Afin de résoudre ces problèmes, deux types d'approches sont classiquement utilisées : les modèles macroscopiques et les modèles microscopiques.

Dans le cadre du projet ANR DSS_Evac_Logistique dans lequel s'inscrit cette thèse, nous nous intéressons à l'évacuation de masse en cas d'inondation, d'incendie, de tremblements de terre ou de catastrophes d'origine humaines où les données terrain varient dans le temps. Les personnes doivent quitter leurs lieux de vie pour une période allant de quelques jours à plusieurs mois et l'objectif de ce projet est de proposer des modèles de la chaîne complète des événements en vue de proposer un outil d'aide à la construction de plans d'évacuation. Dans cette thèse, considérant comme résolu le problème de localisation des points de départ et des points d'arrivée, nous devons développer un ensemble d'approches macroscopiques qui prennent en compte les notions de durée et de sécurité durant l'évacuation d'une zone impactée. Cette thèse se subdivise en cinq chapitres.

Dans le chapitre 1, nous présentons les motivations du projet DSS_Evac_Logistique. Nous explicitons les domaines de compétence de chacun des acteurs ainsi que les interactions entre les différentes entités prenant part au projet. Enfin nous présentons plus en détail le problème d'évacuation macroscopique abordé durant cette thèse à travers les données qui seront prises en compte, les contraintes considérées ainsi que les types de critères à optimiser.

Dans le chapitre 2, l'état de l'art des problèmes d'évacuation est présenté à travers les ap-

proches macroscopiques et microscopiques sous-jacentes. Pour l'approche macroscopique, nous étudions dans un premier temps comment un problème d'évacuation peut être modélisé en un problème de flot dynamique. Par la suite nous étudions les différents types de politiques d'évacuation qui peuvent être appliquées. Ces dernières permettent d'avoir plusieurs informations comme le nombre de personnes pouvant être évacué en un temps donné ou connaissant le nombre de personnes à évacuer quel est le temps requis pour les mettre en sécurité. Toujours dans cette partie nous verrons comment la politique d'évacuation peut influer sur la diversité des plans d'évacuation proposés mais aussi comment prendre en compte le critère de sécurité. Pour l'approche microscopique, nous présentons les différents types d'éléments pouvant être pris en compte durant l'évacuation afin de rendre celle-ci réaliste. Ainsi, nous verrons dans un premier temps l'impact que les aménagements d'un lieu peuvent avoir sur le comportement des individus durant l'évacuation. Puis, nous verrons l'impact de l'âge mais aussi des attributs physiques quant aux aptitudes des personnes à suivre les consignes mais aussi à se mettre à l'abri. Ensuite, nous étudierons les différents types de lieux auxquels cette approche peut être appliquée. Enfin nous présenterons un panel de méthodes existantes et permettant à la simulation de l'évacuation de se rapprocher le plus possible du comportement humain en temps de crise. A la suite de l'état de l'art, les deux chapitres suivants traitent de deux classes de problèmes d'évacuation.

Le chapitre 3 porte sur l'évacuation de personnes sans pertes. Dans cette partie, nous partons de l'hypothèse que toutes les personnes doivent être sauvées. Ainsi, la notion de sécurité n'influe pas sur la vie des individus et peut être vue comme le niveau de satisfaction des évacués ou les dommages et intérêts que nous aurons à leur payer pour les avoir soumis à un sur-risque durant l'évacuation. Dans un premier temps, nous proposons une approche monocritère permettant de minimiser la durée moyenne d'évacuation tout en améliorant la complexité des meilleures méthodes de résolution connues pour ce problème. Par la suite nous étendons cette approche afin de prendre en compte la notion de sécurité grâce à une approche lexicographique qui, pour deux solutions partielles identiques en terme de durée moyenne d'évacuation, choisira la plus sûre. Cette extension permet d'avoir un optimum de Pareto faible en terme de sécurité. Sachant que quelle que soit la politique d'évacuation adoptée le critère de la durée aura la même valeur, nous nous intéressons par la suite à ce qui se passe si nous donnons plus de temps que nécessaire aux individus pour évacuer les lieux. Cela permet la construction d'un front de Pareto approché dégradant la durée moyenne d'évacuation au profit de la sécurité. Dans le cadre de nos expérimentations, nous utilisons des instances réelles de la ville de Nice. Ces dernières permettent de modéliser des scénarios-catastrophes d'ampleurs et de tailles variables impactant tout ou partie de la ville. Contrairement à ce qui pouvait être traité dans la littérature en terme de taille d'instance, nos expérimentations montrent que notre approche n'est ni sensible à la taille des réseaux de grande taille considérés, ni au nombre de points de rassemblement et centres de secours. En outre du point de vue de l'optimisation multicritère, les résultats expérimentaux montrent tout d'abord que la prise en compte de la notion de sécurité n'est pas préjudiciable en temps de calcul tout en donnant des résultats équivalents en terme de durée de l'évacuation et meilleurs en terme de sécurité. Dans un second temps, ils illustrent comment le contournement des zones dangereuses influe sur la sécurité du plan d'évacuation proposé mais aussi sur la durée nécessaire afin de mettre en œuvre celle-ci.

Le chapitre 4 porte sur l'évacuation de personnes avec pertes. Dans cette partie, nous partons de l'hypothèse que potentiellement toutes les personnes ne réussiront pas à atteindre un des centres de secours sans dommage et qu'elles pourraient nécessiter une prise en charge ou une aide. Ainsi, dans cette partie nous aurons comme objectif de maximiser le nombre de personnes pouvant atteindre les centres de secours avant une certaine date T sans pour autant nécessiter d'aides extérieures. Pour cela nous rendons plus contraignant les modèles décrits dans le chapitre 3 afin de s'assurer que toute personne blessée durant l'évacuation devra attendre sur place que l'on vienne lui porter secours. Par la suite nous prenons en compte le fait qu'un nombre limité de forces de secours puissent être déployées sur le réseau afin d'aider au bon déroulement de l'évacuation. Ces derniers auront pour rôle de sécuriser certaines portions critiques du réseau afin que les personnes ne les traversent pas si elles sont trop dangereuses ou en les rendant plus sûr si leurs niveaux de dommages ne sont pas importants. Une extension de cette approche permet de prendre en compte le routage de ces derniers depuis leurs lieux de cantonnement (i.e. caserne de pompiers, gendarmerie, hôpitaux, ...) vers les zones à sécuriser. Dans le cadre d'une collaboration avec un collègue membre du projet DSS_Evac_Logistique, nous avons aussi étudié le problème où les niveaux de dommage du réseau n'étaient pas connus avec certitude. Ce dernier point nous a amené à intégrer à notre approche la notion de robustesse des plans d'évacuation proposés afin de s'assurer qu'ils restent réalisables et efficaces même dans le pire des scénarios. Même si l'ensemble des problèmes d'optimisation présentés dans cette partie ont été prouvés $NP\text{-}Difficiles$ au sens fort, nous avons pu grâce aux particularités de la nouvelle structure de données développée et présentée dans le chapitre 3 les résoudre en pratique pour des données de grande taille comme celle de notre cas d'étude à savoir la ville de Nice. Pour cela nous avons développé une métaheuristique que nous avons testée sur de petites instances dans le but de pouvoir les comparer avec la solution optimale obtenue à l'aide d'un solveur mathématiques. Suivant les problèmes traités les heuristiques ont une déviation moyenne en pourcentage de 4% à 7% par rapport à la solution optimale trouvée par un solveur mathématiques. Nous supposons que cette efficacité est conservée lors de la mise à l'échelle afin de prendre en compte des données réelles de grande taille.

Enfin, dans le chapitre 5, nous présentons les intégrations de nos solutions dans l'outil commun d'aide à la décision développé par les différents acteurs du projet DSS_Evac_Logistique. Ce dernier intègre l'ensemble de la chaîne des événements. Il comprend la création d'un scénario-catastrophe appliqué à une ville, des populations impactées à évacuer, le choix des centres de secours pouvant les accueillir, le routage de ces derniers des points de rassemblement vers les différents points sûrs comme les centres de secours permanents ou temporaires et le transbordement des piétons des centres de secours temporaires vers les centres de secours permanents si ces derniers sont trop éloignés à l'aide de bus. En outre, nous présenterons la simulation de ce processus et sa visualisation à l'aide de l'outil nommé "VisualFlow"

Chapitre 1

Contexte du projet

1.1 Introduction

Le projet DSS_Evac_Logistique est un projet Franco-Allemand de recherche dont l'objectif final est de fournir un logiciel d'aide à la décision multicritère permettant aux services de la sécurité civile de préparer l'évacuation d'une ville de taille moyenne en cas de catastrophe d'origine naturelle ou humaine. Du coté français, le cas d'étude choisi est la ville de Nice. Cette dernière est une grande agglomération se situant près de l'Italie et pour laquelle les risques de tremblements de terre et de glissements de terrain ne sont pas négligeables au regard du passé sismique de la région. Les scénarios de catastrophes naturelles simulés sont des séismes dont l'épicentre se trouve en mer, au large de la ville de Nice. Par voie de conséquence, et suivant la magnitude du séisme considéré, cela peut induire un tsunami qui peut aussi affecter la partie côtière de la ville. Pour ces cas de figure, les plans d'évacuation considérés sont de trois ordres.

- L'évacuation de piétons depuis les zones dangereuses vers les zones sûres ou vers des arrêts de bus. A chaque zone à évacuer est associée une date de début d'évacuation permettant de réguler l'évacuation et de ne pas avoir toute la population dans la rue au même moment. Nous considérons que ces derniers peuvent subir des dommages durant les déplacements dans le réseau.
- L'évacuation de personnes par voitures personnelles qui est une extension de l'évacuation de piétons. En effet il s'agit d'un groupe de personnes non séparables et se déplaçant avec une vitesse accrue sur des routes particulières. Ils peuvent soit se diriger vers un centre de secours, soit à l'extérieur de la ville pour rejoindre les membres de leur famille se trouvant dans d'autres villes non impactées par la catastrophe.
- L'évacuation par bus ou moyen public est une évacuation mise en place permettant aux autorités de réquisitionner des moyens de transports en commun pour évacuer les piétons habitant trop loin des centres de secours et ne pouvant pas s'y rendre à pied. Nous supposons que les bus partent d'un dépôt, collectent les personnes qui les attendent aux arrêts convenus puis les amènent aux différents centres de secours.

Du coté allemand, le cas d'étude est la ville de Kaiserslautern. Cette dernière n'est pas soumis à un risque sismique mais à un risque industriel. En effet la base aérienne Ramstein, dont les pistes se situent dans le prolongement de la ville, se situe à une vingtaine

de kilomètres de cette dernière. Cela peut induire un risque d'accident d'avions militaires armés sur la ville. La zone potentiellement impactée par les débris devant être évacuée jusqu'à ce qu'elle soit vérifiée et sécurisée. En outre, l'autre cas de figure étudié par la partie allemande est un problème récurrent en rapport avec la découverte de bombes larguées durant la seconde guerre mondiale et qui n'ont toujours pas explosées. Dans ce cas de figure, une zone de sécurité doit être définie autour de la bombe et toutes les personnes y résidant doivent être évacuées en dehors de cette dernière pour pouvoir procéder à son déminage. Dans les deux cas de figure considérés du coté allemand, seul l'évacuation par bus est considérée en supposant que les personnes se rendent de manière autonome aux arrêts de bus.

1.2 Présentation des tâches et des acteurs

Afin de pouvoir réaliser ce travail, le projet DSS_Evac_Logistique a été subdivisé en tâches aussi appelées "Working Packages" comme l'indique la figure Fig. 1.1. Ce dernier se compose de 8 parties qui interagissent entre elles. Afin d'expliciter le contexte du projet et de nous situer dans ce dernier, nous allons présenter ses différentes parties.

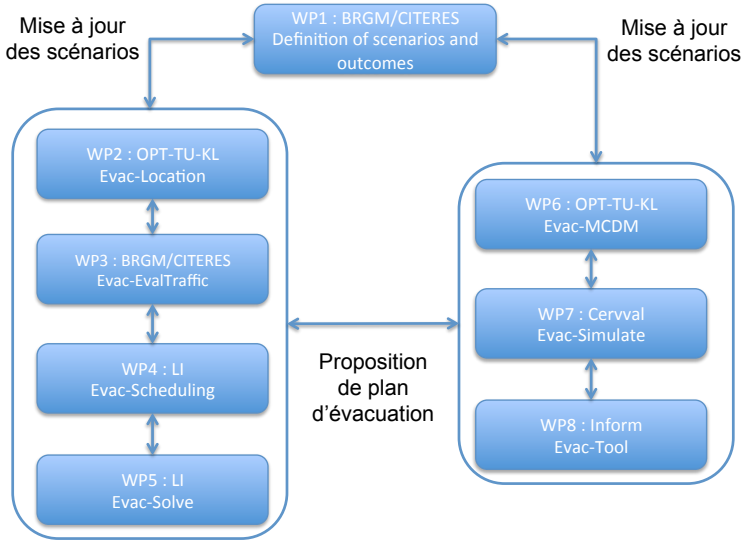


FIGURE 1.1 – Packages du projet DSS_Evac_Logistique

WP1 : Evac-Scenario

Lors d'une catastrophe naturelle comme une inondation, un glissement de terrain ou un tremblement de terre il n'y a pas que les voies de communication qui sont impactées mais aussi les infrastructures de vie. Ainsi l'étude de la vulnérabilité des structures est un aspect aussi important que celle permettant de savoir quelles sont les zones qui seront touchées par le sinistre. En effet, cela permet par exemple de connaitre le nombre de personnes devant

être déplacées ou de savoir si les centres de secours à ouvrir sont toujours utilisables. Le responsable de cette partie est le laboratoire Citeres [1] dont le travail est de deux ordres.

- Fournir une étude détaillée des différentes zones de notre cas d'étude (la ville de Nice).
- Fournir un modèle permettant d'évaluer le comportement des bâtiments lorsqu'ils sont soumis à différents niveaux de contraintes.
- Estimer la répartition de la population à différentes périodes de l'année.

Afin de connaitre les forces appliquées aux bâtiments, le laboratoire Citeres travaille en collaboration avec le BRGM [2] d'Orléans. Dans le cadre du projet, son rôle est de fournir des scénarios-catastrophes simulés comme des glissements de terrain, des tremblements de terre ou encore des tsunamis. Les deux scénarios retenus sont les tremblements de terre de Ligure (Italie) de 1887 ainsi que celui du Vésubie(France) de 1564. Tenant compte des niveaux de dommages engendrés par ces séismes, leur magnitude ont été estimées et les failles relocalisées à différents endroits autour de la ville de Nice afin de générer les scénarios-catastrophes. Par exemple lorsque la faille est placée au large de la ville de Nice cela induit un tsunami pouvant atteindre la côte ainsi que des déformations ou des glissements de terrain. Ces derniers peuvent ainsi associer à chaque partie de la ville un niveau de dommage prenant aussi bien en compte les dommages sur les routes, les bâtiments que l'empiétement des infrastructures effondrées sur les routes. Ainsi en combinant les travaux du laboratoire Citeres et du BRGM, nous obtenons les données suivantes :

- Les populations impactées par un sinistre (nombre, localisation).
- L'impact du sinistre sur le réseau routier.
- L'ensemble des infrastructures utilisables lors de l'évacuation (points de rassemblement, centres de secours, arrêts de bus,...)

WP2 : Evac-Location

Le positionnement des points de rassemblement et des centres de secours est un aspect important du projet, car devant permettre aux évacués de savoir où ils doivent se réunir pour avoir les informations d'évacuation et où ils doivent se rendre pour être pris en charge ou mis en sécurité. Sachant que l'ouverture de chacun de ces emplacements a un coût humain et financier, il faut décider de les positionner à des endroits stratégiques, sûrs et accessibles pour la population. La liste des endroits candidats et exploitables est obtenue grâce au travail réalisé par le laboratoire Citeres et le BRGM dans le "Working Package 1". Le choix des points de rassemblement ainsi que des centres de secours à ouvrir est réalisé par les collègues allemands du laboratoire "AG Optimierung" de l'Université Technique de Kaiserslautern. Ce travail est complété par le laboratoire Citeres par des études de terrain effectuées lors de plusieurs années sur le cas d'étude français (Nice) à l'occasion de stages de groupes.

1. CItés, TERritoires, Environnements et Sociétés UMR 7324 - Université de Tours
2. Bureau de Recherches Géologiques et Minières - Orléans

WP3 : Evac-EvalTraffic

Le "Working package 3" vient en complément du précédent et permet de mettre à jour les informations du réseau par rapport aux informations relatives au scénario-catastrophe et produites par le "Working Package 1". En effet, l'ajout des différentes infrastructures de secours modifie les caractéristiques du réseau en créant potentiellement des ralentissements à leurs abords. La seconde contribution de cette partie est de reconfigurer le réseau routier afin d'améliorer le processus d'évacuation. Cela passe par la fermeture de certaines voies et le changement de direction d'autres pour orienter l'évacuation vers certaines zones ou accroitre la capacité de certaines routes. Le réseau routier ainsi obtenu à la fin de cette étape est celui qui sera utilisé afin de procéder à l'évacuation des personnes dans les prochaines étapes. Pour obtenir ce résultat, les laboratoires d'"AG Optimierung" et de Citeres travaillent en collaboration avec le BRGM.

WP4 : Evac-Scheduling

Une fois que toutes les données sont à disposition, il s'agit maintenant de procéder à l'évacuation des individus. Plusieurs modes d'évacuation sont considérés (i.e. à pied, en voiture personnelle et par bus). Ce "Working Package" ne s'occupera que de l'évacuation par des moyens de transport. Les voitures personnelles ont la possibilité soit de sortir de la ville impactée soit d'aller dans un centre de secours. Nous considérerons que chaque parking de centre de secours a une capacité maximale. Comme toutes les personnes ne disposent pas de voiture personnelle, celles qui se trouvent loin des centres de secours peuvent être évacuées par bus. Pour l'évacuation par bus, ces derniers partent d'un dépôt de bus et doivent collecter les individus qui les attendent aux arrêts de bus. Ainsi, sachant que le nombre de bus est limité et qu'ils doivent suivre des routes prédéfinies (i.e. réseau routier approximé), l'objectif est de déterminer la tournée de bus permettant de minimiser la durée du processus d'évacuation ainsi que le "risque" que le plan d'évacuation proposé ne puisse pas être réalisé. Ce travail est réalisé dans le cadre d'une thèse encadrée au Laboratoire d'Informatiques de l'Université de Tours.

WP5 : Evac-Solve

Les travaux de notre thèse s'intègrent dans ce "Working Package". Notre travail consiste à router les individus des points de rassemblement vers les centres de secours et/ou arrêts de bus. Ce routage se fera à travers un réseau réel dynamique. Durant la modélisation nous considérerons aussi les arrêts de bus comme des centres de secours ayant une capacité maximale d'accueil. Nous devons aussi déterminer les dates auxquelles les vagues successives de personnes arrivent aux arrêts de bus afin que le "Working Package 4" puisse planifier les tournées de bus. Nous utiliserons une approche macroscopique afin que la solution proposée puisse être mise à l'échelle et appliquée à de grande ville. Comme les individus doivent traverser des zones potentiellement dangereuses, nous devons non seulement minimiser la durée de ce routage mais aussi le risque encouru par l'ensemble des évacués.

WP6 : Evac-MCDM

Les deux précédents "Working Packages" fournissent un ensemble de solutions de compromis (durée,sécurité) qui peuvent être en nombre important. L'échantillonnage et la visualisation de ces solutions est un donc un aspect important afin que les décideurs puissent avoir un ensemble représentatif de solutions. Cela devient d'autant plus important lorsque le nombre de critères augmente et dépasse 3 comme dans la partie Allemande avec des critères comme le coût matériel, la satisfaction des évacués ou encore leur temps d'attente moyen avant d'être pris en charge. Le responsable de cette partie est le laboratoire "AG Optimierung" de l'Université Technique de Kaiserslautern.

WP7 : Evac-Simulate

Après la sélection des solutions, il faut pouvoir évaluer les plans d'évacuation et vérifier qu'ils ne mènent pas à des situations aberrantes. Le partenaire industriel en charge de cette partie est l'entreprise française CERVVAL dont un de leurs domaines d'expertise est la simulation d'environnement urbain. Dans le cadre du projet, il simule les interactions entre les transports publics (bus), les automobiles et les piétons. Un modèle multi-agents est ainsi développé où chaque individu a sa propre psychologie ainsi que ses propres objectifs et priorités pouvant entrer en compétition avec ceux des autres. Cela permet notamment de faire dévier les solutions impartialement calculées par les approches macroscopiques pour évaluer leur robustesse. Cette simulation permet aussi de détecter les embouteillages qui peuvent se former dans le réseau car n'étant pas pris en compte durant les différentes étapes de la résolution. En effet, la vitesse de déplacement sur une route à un moment donné dépend de plusieurs facteurs comme les feux tricolores, le nombre de personnes se trouvant déjà sur la route, sa longueur, le nombre de voies ou encore la possibilité ou non de dépasser des personnes.

WP8 : Evac-Tool

Pris séparément les différents "Working Packages" ne sont pas très utiles. Pour assurer leurs interactions nous avons un dernier partenaire industriel qui est la société allemande "INFORM GmbH" dont le rôle est d'unifier les différents "Packages" au sein d'un seul et même outil d'optimisation. Ce dernier porte le nom de "VisualFlow" et dispose de plusieurs fonctionnalités telles que :

- La création de scénarios catastrophes.
- La création d'instances de problèmes d'évacuation.
- La création à la main d'instances permettant aux autorités d'effectuer des entrainements.
- La résolution des problèmes d'évacuation.
- La sélection de solutions pertinentes ainsi que leur visualisation sur la carte de la zone impactée.
- La visualisation du processus d'évacuation.

1.3 Présentation du problème d'évacuation macroscopique de piétons

Dans cette partie, nous faisons un zoom sur le Working Package 5 qui nous concerne durant cette thèse et nous explicitons les différentes données dont nous disposons ainsi que les contraintes auxquelles nous sommes soumis.

Données

Les données dont nous disposons sont les attributs d'un réseau routier auquel un scénario-catastrophe a été appliqué. Dans notre cas de figure, il s'agit de celui de la ville de Nice. Le graphe associé peut ainsi se décomposer en sommets et en arcs. Les sommets représentent des endroits remarquables sur le réseau alors que les arcs permettent de modéliser les routes. Chaque route a une longueur, une vitesse de déplacement autorisée, un niveau de sécurité, ainsi qu'une capacité maximale. Ces données sont issues du travail réalisé par le WP1 et prennent en compte les modifications du réseau routier dues aux dommages des zones impactées. Nous avons aussi deux types de sommets particuliers, les points de rassemblement et les centres de secours. Les points de rassemblement représentent des emplacements d'où partent la population d'une ou plusieurs infrastructures endommagées. Ces points du réseau sont considérés comme relativement sûrs. Les individus y sont informés et y reçoivent les consignes d'évacuation ainsi que leur lieu d'affectation. Les centres de secours quant à eux sont des lieux où les personnes seront prises en charge par les autorités compétentes qui peuvent décider de les déplacer de nouveau vers d'autres centres. Ces centres de secours ont une capacité maximale d'accueil pour les personnes mais aussi pour les véhicules personnels.

Contraintes

Plusieurs contraintes doivent être prises en compte durant le processus d'évacuation. Afin que toute la population ne se retrouve pas au même moment sur les routes, les autorités ont la possibilité de réguler l'évacuation. Cette régulation est effectuée en autorisant le début d'évacuation des différents points de rassemblement à des dates arbitraires. Une fois l'évacuation commencée, il faut s'assurer que le nombre de personnes sur chaque type de route n'excède pas la capacité de cette dernière. Pour éviter les bousculades pour les piétons ainsi que les embouteillages pour les voitures personnelles, nous interdisons le fait que les individus puissent attendre sur les routes. Cette contrainte a pour but d'éviter le sur-risque inhérent au fait d'avoir beaucoup d'individus dans un espace restreint. Nous noterons aussi qu'une fois que les individus auront quitté les points de rassemblement, il leur sera interdit de repasser deux fois par le même point. Dans le cadre de notre approche macroscopique, nous supposerons aussi que les individus de même type auront la même vitesse de déplacement tout au long de l'évacuation. Ces vitesses de déplacement identiques impliquent aussi qu'il leur sera impossible de doubler un groupe de personnes sur la même route.

Critères

Lors d'une évacuation, le premier critère à considérer est la durée d'évacuation qu'il faut minimiser. Suivant la politique d'évacuation que l'on souhaite adopter, ce critère peut être exprimé sous différentes formes. En effet, il peut s'agir de minimiser la durée moyenne d'évacuation, de maximiser le nombre de personnes sortant à chaque instant ou de minimiser la date d'arrivée de la dernière personne dans un centre de secours. Le point commun de ces différentes politiques est qu'elles ont toutes la même valeur concernant le critère de la durée (i.e. durée totale de l'évacuation). Dans le cadre de notre projet, ce critère est lié aux distances parcourues par les individus lors de l'évacuation. Considérant que la vitesse de déplacement des évacués est constante, nous en déduisons ainsi le temps de trajet. En outre, contrairement à la plupart des problèmes d'évacuation étudiés dans la littérature, nous considérons comme second critère à optimiser la notion de **sécurité**. En effet, lors de l'évacuation les personnes peuvent avoir à traverser des zones dangereuses afin de se mettre en sécurité. Ce critère peut être illustré par l'évacuation d'un bâtiment en feu. On peut ainsi être amené à avoir le choix en un chemin court mais avec beaucoup de fumée toxique et un chemin long avec beaucoup moins d'émanation. il ne s'agit donc plus seulement d'évacuer rapidement les personnes mais de s'assurer que ces dernières arrivent à bon port dans la meilleure forme physique possible.

Modélisation et évaluation d'une solution

Dans le cadre d'une évacuation macroscopique, la modélisation d'une solution peut être faite en utilisant des modèles de flots dans les graphes. La différence fondamentale avec les graphes statiques réside dans la prise en compte de l'aspect dynamique du réseau permettant ainsi l'utilisation d'un arc à sa capacité maximale à plusieurs reprises. Ainsi, pour tous les arcs, le nombre de personnes les utilisant à différents moments définit le plan d'évacuation. Pour l'évaluation d'une solution, nous utilisons le plan d'évacuation précédemment établi. Par exemple, s'agissant du critère de la durée d'évacuation, nous pourrons le mesurer en considérant le centre de secours recevant le dernier évacué. De même, il est possible de mesurer la durée moyenne d'évacuation en considérant l'ensemble des centres de secours. Pour l'évaluation de la sécurité associée au plan d'évacuation, nous utilisons les attributs des arcs traversés durant l'évacuation. Ainsi un ensemble d'arcs définissant un chemin, moins ces derniers seront dangereux plus le chemin emprunté par les évacués sera qualifié de sûr. Ramené à l'ensemble des chemins utilisés, cela permet d'attribuer une note de sécurité au plan d'évacuation.

1.4 Conclusion

Nous avons présenté le projet DSS_Evac_Logistique ainsi que l'ensemble des acteurs qui y participent. En premier lieu, nous avons présenté comment les niveaux de vulnérabilité des infrastructures étaient obtenus et comment la génération de scénarios-catastrophes était obtenue. En second lieu, nous avons abordé comment l'optimisation des ressources et le routage des individus étaient effectués. Enfin, dans un dernier temps, nous avons présenté comment ces résultats pouvaient être visualisés et intégrés dans un outil d'aide à la décision

pour les utilisateurs finaux. En outre, nous nous sommes situés dans le cadre de ce projet et avons présenté le problème d'évacuation macroscopique de piéton à travers les données et les contraintes considérées ainsi que la manière de modéliser les plans d'évacuation.

Dans le chapitre suivant, nous présentons un état de l'art portant sur les problèmes d'évacuation. Deux cas de figure sont étudiés, l'approche macroscopique et l'approche microscopique. Dans la première partie, nous nous intéresserons à la modélisation d'un problème d'évacuation ainsi qu'à l'application de plusieurs politiques de routages des individus (approche macroscopique). Dans la seconde partie nous traiterons des différents aspects à prendre en compte pour rendre une évacuation réaliste (approche microscopique).

Chapitre 2

Etat de l'art

2.1 Introduction

Les problèmes d'évacuation et de routage de masse à travers des réseaux dynamiques sont des problèmes largement étudiés dans la littérature étant des déclinaisons de problèmes de flots dans des graphes. Ainsi, nous devons l'une des premières définitions d'un réseau dynamique à [Ford et Fulkerson, 1962] pour lequel il s'agit d'ajouter la notion de temps de traversée sur chaque arc alors que la capacité de ce dernier devient le nombre maximum d'unités de flot que l'on peut y faire passer à chaque unité de temps considérée. Grâce à cette approche simpliste, il est possible de construire un nouveau réseau étendu où l'on duplique les sommets et les arcs du réseau dans le temps ainsi que l'ajout de sommets fictifs sources et puits pour se ramener à un problème de flot classique avec un sommet source et un sommet puits. Le mode de construction de tels réseaux étendus dans le temps peut être trouvé dans la littérature sous le nom de "Time Expanded Network" avec des utilisations diverses et variées [Hamacher et Tjandra, 2002], [Miller-Hooks et Stock Patterson, 2004] et [Fleischer et Skutella, 2007]. L'avantage d'une telle modélisation du problème d'évacuation est de pouvoir utiliser toutes les approches et méthodes déjà développées pour les problèmes de flot comme trouver un flot maximum, un flot maximum à cout minimum ou encore un flot multicommodité. Toutefois la construction de ces graphes étendus mène à faire passer la taille des données d'entrée de polynomiale à pseudo-polynomiale car un nœud ou un arc peuvent être dupliqués pour chaque période de temps. En outre, suivant le niveau de precision souhaité, trois types d'approches sont possibles. L'approche avec le plus petit niveau de précision est l'approche macroscopique qui consiste à considérer la population à router comme un flot homogène dont les individus ont les mêmes attributs [Hamacher et Tjandra, 2002]. Les caractéristiques qu'ils partagent sont principalement d'ordre physique comme la vitesse de déplacement, la corpulence ou l'espace occupé sur un arc. Comme pour le flot, on considérera alors qu'il n'y a pas d'interaction entre les individus s'ils se rencontrent sur un sommet ou une intersection. Comme il s'agit de s'affranchir de certaines contraintes réelles, cette approche fournit une borne inférieure de la durée globale du routage des individus. Cette approche est particulièrement indiquée lorsqu'il s'agit de router des personnes à travers un réseau routier avec des centaines de milliers de sommets et d'arcs.
Pour un plus haut niveau de précision dans le plan d'évacuation mais aussi pour le rendre

plus robuste nous devons considérer séparément chaque individu à router avec ses propres caractéristiques physiques mais aussi gérer les interactions entre ces derniers. Cela mène à appliquer une approche microscopique dont le but est de simuler autant que possible le comportement humain durant le processus d'évacuation avec des éléments perturbateurs comme les embouteillages, les bousculades, le stress ou les incivilités [Klüpfel *et al.*, 2001]. Une telle approche nécessite beaucoup de puissance de calcul et est indiquée lorsqu'il s'agit de l'évacuation de lieux restreints comme des bâtiments pouvant être modélisés par quelques centaines de sommets et d'arcs. Toutefois plus on prend en compte les contraintes du monde réel plus il est envisageable d'avoir une borne supérieure fiable de la durée de l'évacuation. Une approche de compromis combinant partiellement les approches macroscopique et microscopique est possible et permet de limiter les points négatifs des précédentes approches. Aussi nommé approche mésoscopique ou sandwich [Borrmann *et al.*, 2012], il s'agit de mettre en place un processus où l'on augmente la borne inférieure de l'approche macroscopique et où l'on diminue la borne supérieure de l'approche microscopique afin de tendre vers une solution qui reflète un peu plus la réalité tout en permettant une mise à l'échelle dans le cas d'un routage de masse à l'échelle d'une ville.

Dans cet état de l'art, nous présentons plus en détail les approches macroscopiques et microscopiques précédemment citées à travers les sous-problèmes qui leur sont associés. Une attention particulière sera donnée à la première approche étant donné que l'objectif de cette thèse est l'établissement de plans de routage macroscopiques pour l'évacuation de personnes.

2.2 Modèle macroscopique

2.2.1 Dynamic Network Flow

2.2.1.1 Données et Modélisation

On considère un graphe dynamique $G_{dyn} = (V, A, B, C, D, \Delta, T)$ composé de $|V| = N$ sommets et $|A| = M$ arcs. Pour tout sommet $i \in V$ nous noterons Γ_i^- l'ensemble de ses prédecesseurs et Γ_i^+ l'ensemble de ses successeurs. Pour tout arc $((i,j);t) \in A \times \{0,..,T\}$ nous avons une capacité $C^t_{(i,j)}$ et une durée de traversée $D^t_{(i,j)}$. Associée à la donnée $D^t_{(i,j)}$, nous avons la donnée $\Delta^t_{(i,j)}$ qui est égal au moment $\tau \in \{0,..,T\}$ auquel il faut partir du sommet i pour aller au sommet j afin d'y arriver au moment t. Deux objectifs sont à prendre en compte : le sauvetage de tous les individus ainsi que la politique d'évacuation qui dépend du problème d'évacuation étudié. En ce sens, un état de l'art complet décrivant l'ensemble des sous-problèmes et politiques de flots dynamiques a été proposé par [Aronson, 1989]. Ainsi l'un des premiers objectifs que l'on peut associer à un graphe dynamique consiste à transporter un ensemble B_S individus répartis sur K points de départ $k \in \{1,...,K\}$ avec chacun B_{S_k} personnes vers L points d'arrivée $l \in \{1,...,L\}$ ayant une capacité d'accueil de B_{P_l} unités. En outre chaque point de départ S_k aura une date de début d'évacuation $R_{S_k} \in \{0,...,T\}$. Afin de se ramener à un problème de flot classique, il est possible de fédérer les K points de départ avec un sommet source fictif S et les L points d'arrivée avec un sommet puits fictif P.

2.2.1.2 Modèle mathématique

Dans cette partie nous présenterons les contraintes génériques relatives à tous les problèmes d'évacuation. Les fonctions objectifs qui définissent les politiques d'évacuation utilisées seront présentées lors de la présentation de ces différents problèmes d'évacuation.
Soit $F^t_{(i,j)} : A \times \{0,..,T\} \longrightarrow \mathbb{N}$ la fonction de flot d'évacués avec $F^t_{(i,j)}$ la variable indiquant le nombre d'individus empruntant l'arc $(i,j) \in A$ au moment $t \in \{0,..,T\}$.

S.C

$$0 \leq F^t_{(i,j)} \leq C^t_{(i,j)} \qquad \forall (i,j) \in A, \forall (t \in \{0,..,T\} \qquad (2.1)$$

$$F^0_{(S,S_k)} = B_{S_k} \qquad \forall S_k \in \Gamma^+_S \qquad (2.2)$$

$$F^t_{(S,S_k)} = 0 \qquad \forall S_k \in \Gamma^+_S,\ \forall t \in \{1,..,T\} \qquad (2.3)$$

$$\sum_{t=0}^{R_{S_k}-1} \sum_{j \in \Gamma^+_{S_k}} F^t_{(S_k,j)} = 0 \qquad \forall S_k \in \Gamma^+_S \qquad (2.4)$$

$$\sum_{t=R_{S_k}}^{T} \sum_{j \in \Gamma^+_{S_k}} F^t_{(S_k,j)} \leq F^0_{(S,S_k)} \qquad \forall S_k \in \Gamma^+_S \qquad (2.5)$$

$$\sum_{(i,j) \in A} F^{t-D^{\Delta^{(t)}_{(i,j)}}}_{(i,j)} = \sum_{(j,k) \in A} F^t_{(j,k)} \qquad \forall j \in V \setminus \{S,P\}, \forall t \in 0,..,T \qquad (2.6)$$

$$\sum_{t=0}^{T} \sum_{i \in \Gamma^-_{P_l}} F^t_{(i,P_l)} \leq B_{P_l} \qquad \forall P_l \in \Gamma^-_P \qquad (2.7)$$

La contrainte (2.1) est la contrainte classique de capacité indiquant que quelque soit l'arc et quelque soit le moment considéré le nombre total d'individus pouvant entrer dans l'arc n'excède pas sa capacité maximale. Les individus quittent le sommet initial S pour se rendre à leur point de départ S_k à la date 0 et ce trajet nécessite une durée R_{S_k} (2.2), (2.3). Les contraintes (2.4) et (2.5) permettent de s'assurer qu'aucune personne ne quitte son point de rassemblement S_k avant la date de début au plus tôt R_{S_k}. La conservation du flot à chaque sommet est assurée par la contrainte (2.6) alors que le respect des capacités d'accueil des centres de secours est garanti par la contrainte (2.7). Il est aussi à noter que dans le cas général des problèmes d'évacuation, il est possible de permettre l'attente sur des sommets particuliers $i \in A$ en ajoutant des arcs de type (i,i) à l'ensemble A et en leur associant les capacités $C^t_{(i,i)}$ permettant de savoir à tout moment le nombre maximum de personnes pouvant attendre à un point. Aussi, cette attente nécessite une durée $D^t_{(i,i)} = 1$ et un $\Delta^t_{(i,i)} = t - 1$.
Les contraintes ci-dessus sont partagées par la plupart des problèmes de flots dynamiques. Ils permettent notamment de modéliser le routage d'un seul type de commodité dans le réseau dynamique mais peuvent aussi facilement être étendues pour prendre en compte plusieurs commodités (voir [Groß et Skutella, 2015]). Dans notre cas, nous nous limiterons à une seule commodité qui modélisera soit les piétons, soit les voitures personnelles.

2.2.2 Maximum Dynamic Network Flow

La notion de flot dynamique maximum a été introduite par [Ford et Fulkerson, 1958]. L'objectif de ce problème d'évacuation est de maximiser, pour une durée T fixée, le nombre de personnes pouvant atteindre le sommet puits P partant d'un sommet source S. Ci-dessous la fonction objectif associée :

$$\text{Maximiser} \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} F_{(P_l,P)} \tag{2.8}$$

Ce problème de flot est notamment utilisé pour la modélisation de problèmes génériques d'évacuation [Choi *et al.*, 1988a] mais aussi de cas plus particuliers comme ceux de bâtiments en feu [L.G.Chalmet *et al.*, 1982]. Il se révèle très utile pour déterminer une borne inférieure de la durée de l'évacuation pour les sous-problèmes avec des politiques d'évacuation différentes. Une de ses particularités est que la solution optimale peut aussi être obtenue en n'autorisant pas les attentes sur les sommets et sur les arcs [Hoppe et Tardos, 1994]. Ainsi pour résoudre ce problème il est possible de le considérer comme un problème de flot à cout minimum et d'utiliser l'algorithme du flot temporel répété de [Ford et Fulkerson, 1962]. Il suffit alors de trouver un flot max dans le graphe statique, de le décomposer en chaînes Φ_i et de les répéter dans le temps. Chaque chaîne considérée Φ_i nécessite une durée $\Lambda(\Phi_i)$pour aller du sommet S au sommet P. Ainsi, afin d'avoir le flot max dynamique pour une durée limite T fixée, il suffit de répéter la chaîne Φ_i de t allant de 0 à $T - \Lambda(\Phi_i)$. Une illustration de l'algorithme peut être trouvée dans l'état de l'art sur les problèmes d'évacuation de [Hamacher et Tjandra, 2002]. Toutefois cette méthode ne reste valable que lorsque les données, comme la capacité des arcs ou les temps de traversée, ne varient pas dans le temps. Afin de combler ce vide, [Minieka, 1974] propose la première approche permettant de prendre en compte les changements dans le réseau au cours du temps en modifiant l'algorithme du flot temporel répété introduit par [Ford et Fulkerson, 1962]. La modification porte sur la détermination des sommets non utilisables afin d'arriver avant la deadline T et la recherche de K plus courts chemins afin d'envoyer le maximum d'unités possible le long de ces chemins. Le problème de recherche des K plus courts chemins peut être résolu en temps polynomial pour K fixé. Cependant la structure particulière du graphe ne lui permet pas de prendre en compte l'attente sur les sommets ou les arcs.

2.2.3 Earliest Arrival Flow

Introduit par [Gale, 1959], cette approche vient en complément du problème de flot max dynamique introduit par [Ford et Fulkerson, 1958]. Tout d'abord connu sous le nom de Universally Maximum Flow, ce problème a été créé afin de garantir la propriété de maximisation du flot dynamique à chaque moment considéré.

$$\text{Minimiser} \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} (t \times F_{(P_l,P)}) + HV \left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} F^t_{(P_l,P)} \right) \tag{2.9}$$

Lors de l'évacuation d'une zone dangereuse, il y a une question qu'il est légitime de se poser : "Est-ce-que la situation va se dégrader durant l'évacuation" ? En effet cette question est pertinente durant l'évacuation d'infrastructures comme le Titanic ou les tours jumelles du "World Trade Center" pour lesquels personne ne savait combien de temps ils tiendraient avant de sombrer ou s'effondrer. Ce problème trouve aussi des applications à l'échelle de grandes villes en cas de montées des eaux dues à des pluies diluviennes, des tremblements de terre avec répliques ou encore l'évacuation de villes côtières à cause d'un tsunami dont on ignore la hauteur des vagues une fois arrivées sur les côtes. Le point commun de ces catastrophes d'origine humaines ou naturelles est que nous ne savons ni de combien de temps nous disposons pour mettre en sécurité les personnes en danger ni si nous pourrons toutes les sauver. Ainsi la meilleure politique d'évacuation utilisable dans ces circonstances est de maximiser le nombre de personnes mises en sécurité à chaque instant. Une autre formulation moins individualiste de ce problème permet aussi de le voir comme la minimisation de la durée moyenne d'évacuation [Jarvis et Ratliff, 1982].

$$\text{Minimiser } \frac{1}{B_S} \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} (t \times F_{(P_l,P)}) + HV \left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} F^t_{(P_l,P)} \right) \tag{2.10}$$

Des approches et des méthodes similaires basées sur la recherche de plus courts chemins augmentant ont été proposées par [Wilkinson, 1971] et [Minieka, 1973]. Toutefois comme ils utilisaient un réseau étendu dans le temps dépendant de T, ils sont pseudo-polynomiaux. Par la suite un schéma d'approximation polynomial utilisant une décomposition du flot dans le graphe statique a été proposé par [Hoppe et Tardos, 1994]. [Baumann, 2006] propose une nouvelle approche afin de traiter le cas d'une évacuation depuis plusieurs points de départ vers un seul point d'arrivée. Cette approche se base sur la détermination d'un graphe utilisant la politique de maximisation du flot à chaque instant. Une fois ce schéma déterminé, le plan d'évacuation final est obtenu en utilisant l'algorithme développé par [Hoppe et Tardos, 2000] sur le nouveau graphe. Les deux parties de cet algorithme sont polynomiales aussi bien en complexité temporelle que spatiale, aucun des deux n'utilise de graphe étendu dans le temps. Pour autant, lorsque dans le cas général il s'agit de prendre en compte plusieurs sommets d'arrivées comme dans notre cas, *nous n'avons pas connaissance de l'existence d'algorithmes polynomiaux.* En outre, les précédentes méthodes ne prennent pas en compte le changement potentiel des données durant le processus d'évacuation comme une route qui deviendrait partiellement ou momentanément inutilisable à cause de la montée des eaux durant une inondation. L'étude de ce genre de variations de données en appliquant cette politique d'évacuation peut être trouvée dans les travaux de [Ogier, 1988] et [Fleischer, 2001] pour les flots continus ainsi que [Hamacher et Tjandra, 2003] et [Baumann et Köhler, 2007] pour les flots entiers.
Toutefois, en utilisant cette politique d'évacuation, on peut se retrouver avec des problèmes insolubles où l'on ne peut pas évacuer tout le monde même lorsque $T = +\infty$, [Takizawa *et al.*, 2012]. Ce type de problème peut être rencontré lorsqu'il existe plusieurs sommets puits (centres de secours ou arrêts de bus) et que les mauvais groupes de personnes y sont répartis. La figure 2.1 montre un exemple de réseau dynamique avec deux sommets sources avec chacun 2 personnes à évacuer et deux sommets puits avec une capacité d'accueil de 2 chacun. Sur chaque arc (i, j) nous avons les attributs (a, b) qui représentent respectivement

la durée de traversée et la capacité maximale de l'arc. Dans ce cas de figure, la politique

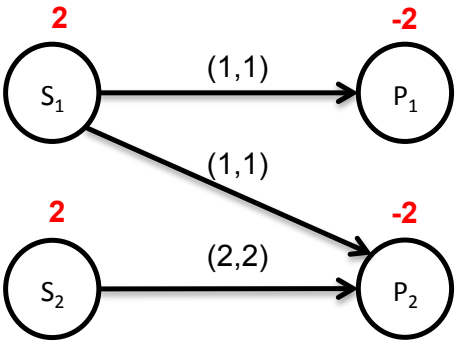


FIGURE 2.1 – Inexistence du "Earliest Arrival Flow"

d'évacuation maximisant le nombre de personnes en sécurité à chaque instant nous oblige à envoyer, à l'instant $t = 0$, 2 unités depuis le sommet S_1 avec une personne en direction de P_1 et une autre personne en direction de P_2. Nous avons ainsi deux personnes en sécurité à l'instant $t = 1$. Toutefois, il nous restera 2 personnes à évacuer depuis le sommet S_2. Ainsi seule une personne pourra être envoyée à l'instant $t = 0$ depuis le sommet S_2 en direction du sommet P_2. Il nous restera donc une personne non sauvée et par voie de conséquence la non-existence de solution en appliquant cette politique d'évacuation. Un autre exemple avec un unique sommet source illustrant le même problème est donné par [Baumann et Skutella, 2009]. Grâce aux travaux de [Schmidt et Skutella, 2014], il est possible de savoir s'il existe un plan d'évacuation faisable en utilisant ce type de politique d'évacuation. Pour cela ils définissent deux types de sous-graphes qui ne doivent pas se trouver dans le réseau dynamique utilisé pour l'évacuation pour s'assurer de l'existence d'une solution lors de l'utilisation de cette politique d'évacuation. Longtemps resté un problème ouvert, ce dernier a été prouvé comme étant NP-Difficile par [Disser et Skutella, 2015] dans le cas général.

2.2.4 Quickest Flow Problem

Il s'agit d'un problème d'évacuation où l'on cherche à router B_S personnes d'un unique point de départ S (bâtiment en feu) vers un unique point d'arrivée P (extérieur du bâtiment). L'objectif associé à ce problème est de savoir quel est le temps minimum nécessaire T pour réaliser ce routage ainsi que le modèle de flot associé.

$$\text{Maximize} \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} F^t_{(P_l,P)} : \text{Recherche dichotomique sur l'horizon } T \tag{2.11}$$

Afin de montrer la différence de politique d'évacuation avec le "Earliest Arrival Flow" nous allons illustrer les deux modes d'évacuation à l'aide du graphe dynamique Fig. 2.2 qui représente l'évacuation d'un bâtiment. Ce dernier montre un graphe dynamique avec 5 personnes devant être évacuées du sommet S qui est la zone dangereuse vers le sommet P

qui est la zone sûre ayant une capacité d'accueil de 5 personnes. Dans cet exemple, nous disposons d'un chemin court mais restreint (ascenseur) et d'un chemin long mais spacieux (escaliers). En appliquant la politique du "Earliest Arrival Flow" il faut obligatoirement

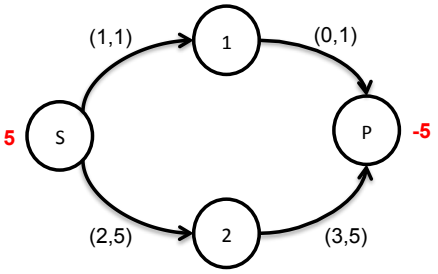


FIGURE 2.2 – Exemple de graphe dynamique

maximiser le nombre de personnes arrivant à chaque moment et donc soit envoyer 5 personnes sur le chemin $(S, 1, P)$ aux moments $t = [0, .., 4]$ ou en envoyer 4 sur le chemin $(S, 1, P)$ au moment $t = [0, .., 3]$ et une personne le long du chemin $(S, 2, P)$ au moment 0. Le "Quickest Flow" quant à lui admet les mêmes solutions que le "Earliest Arrival Flow" avec en plus la possibilité d'envoyer 5 personnes le long du même chemin $(S, 2, P)$ au moment 0. De manière générale cette dernière solution peut être associée à la recherche de flot maximum dynamique et est facile à trouver. En utilisant cette solution pour résoudre ce problème d'évacuation on se retrouve avec la solution illustrée par la figure Fig. 2.3. Toutefois, on peut facilement voir que si quelque chose se passe à la date $t = 4$ comme

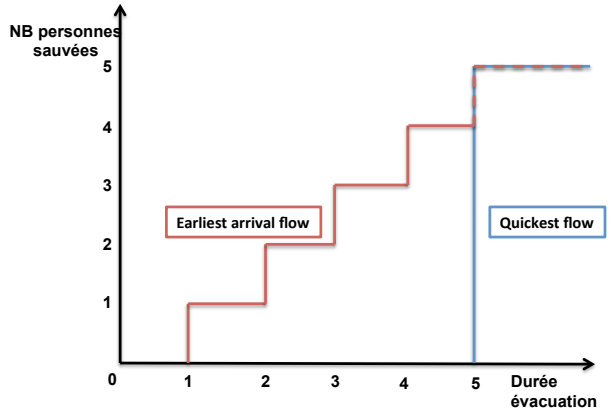


FIGURE 2.3 – Exemple de différences entre le "Earliest Arrival" et le "Quickest" flow

l'effondrement du bâtiment considéré, alors avec cette politique d'évacuation nous pouvons potentiellement perdre toutes les personnes alors que dans l'autre cas nous en aurions au moins sauvé 4.
Connaitre la date probable de fin d'évacuation devient vital afin de mettre en place ou non cette politique. Cette durée peut être facilement obtenue en faisant une recherche dichotomique sur la durée T et en appliquant un des algorithmes de résolution de flot maximum

à coût minimum ou ceux résolvant le problème de flot maximum dynamique. Grâce à B. Klinz, [Hoppe et Tardos, 2000] proposent une méthode permettant de déterminer en temps polynomial la durée T nécessaire pour router les individus quelle que soit la politique d'évacuation et quel que soit le nombre de sommets de départ et d'arrivée. Toutefois cette méthode indique juste la durée minimale nécessaire et non le plan d'évacuation associé. Les premiers algorithmes polynomiaux permettant de déterminer le plan d'évacuation sont l'œuvre de [Burkard *et al.*, 1993]. Ils améliorent la simple méthode de résolution initiale basée sur la recherche dichotomique en résolvant un problème de flot maximum dynamique. En effet, ils proposent plusieurs méthodes permettant de réduire l'écart entre la borne supérieure et la borne inférieure du temps nécessaire T pour réaliser l'évacuation. Toutefois cette approche permet seulement de résoudre les problèmes d'évacuation mono-source mono-puits. [Hoppe et Tardos, 1994] proposent un algorithme polynomial permettant de prendre en compte les cas d'évacuation où les personnes partent d'un nombre fixé de points de départ pour se rendre dans d'un nombre fixé de refuges ou points sûrs. Cette approche se base sur la résolution lexicographique du problème de flot maximum dynamique pour toute paire de sommets source et puits. Cependant, l'une des limites de cette approche est que la capacité totale des centres de secours est égale au nombre total de personnes à sauver. Ainsi, il ne garantit pas que la solution associée au flot sera entière. Ces problèmes sont réglés par les mêmes auteurs [Hoppe et Tardos, 2000] en proposant le premier algorithme polynomial permettant de résoudre le problème d'évacuation multi-sources et multi-puits dont l'objectif est la minimisation de la durée de l'évacuation. Pour cela ils construisent un problème équivalent qui garde les mêmes propriétés que le problème d'évacuation initial. Par la suite, on procède à l'ouverture progressive des points de départ et des centres de secours permettant de minimiser l'écart entre la demande et la capacité d'accueil et respectant la durée minimale d'évacuation T obtenue grâce à la contribution de Klinz (voir [Hoppe et Tardos, 2000]). Cependant, lorsque l'on considère que les données du réseau peuvent varier durant l'évacuation, le problème devient ouvert [Tjandra, 2003]. Par exemple, lorsque l'on considère que des arcs ont des fenêtres de disponibilité (réduction ou augmentation de la capacité) et qu'il est interdit d'attendre sur les sommets ou les arcs alors ce problème d'évacuation devient NP-difficile et non-approximable même pour juste évacuer une seule personne d'un point S vers un point P [Köhler *et al.*, 2008]. En autorisant l'attente sur les sommets, le problème peut être résolu en temps polynomial pour évacuer une personne mais reste ouvert pour un nombre quelconque de personnes. Pour résoudre le cas général où les arcs ne sont pas juste indisponibles pendant une période mais dont l'état évolue dans le temps, [Miller-Hooks et Stock Patterson, 2004] proposent une méthode qui se résume en quatre phases. Dans un premier temps ils transforment le problème d'évacuation quelconque en un problème d'évacuation mono-source et mono-puits à l'aide du réseau dynamique étendu dans le temps (voir [Ford et Fulkerson, 1962]). Par la suite, ils maximisent le nombre de personnes arrivant à tout moment (voir [Wilkinson, 1971]). Puis ils mettent à jour le graphe d'écart associé au graphe étendu dans le temps précédemment construit. S'il reste des personnes à évacuer ils cherchent un autre chemin augmentant arrivant le plus tôt possible afin de le saturer, sinon ils s'arrêtent et retournent la solution trouvée. Cet algorithme est pseudo polynomial car son principal défaut est l'utilisation du réseau étendu dans le temps sur lequel sont effectués des calculs de plus courts chemins et dans le pire des cas, autant de fois qu'il y a de personnes à évacuer. Afin de réduire la taille

de ce graphe dynamique étendu dans le temps, [Fleischer et Skutella, 2007] proposent une approche permettant d'agrandir le pas de discrétisation du réseau tout en garantissant le fait que la solution obtenue ne déviera pas trop de la solution optimale.

2.2.5 Prise en compte de la sécurité

La prise en compte de la notion de sécurité en plus de la durée durant l'évacuation est un domaine récent. Le critère de sécurité peut être modélisé de manière additive ou multiplicative. Dans le premier cas, les problèmes d'évacuation peuvent être vus comme des problèmes de flot à coût minimum dynamique et dans le second cas comme des problèmes de flot généralisé. Toutefois, outre le fait que [Klinz et Woeginger, 2004] ont montré que la version additive est un problème NP-Difficile, cette dernière ne permet pas de bien modéliser les dangers ponctuels situés dans le réseau. Si l'on considère que plus une personne est exposée au danger moins celle-ci a de chance de s'en sortir indemne, alors l'approche multiplicative des niveaux de sécurité est très pertinente. Issus de la théorie des problèmes de flots généralisés fournissant beaucoup d'applications pratiques (voir [Aronson, 1989], [Powell *et al.*, 1995], [Powell *et al.*, 1995]) ce dernier permet de modéliser la notion de sécurité tout au long de l'évacuation. Pour cela, on suppose que l'on dispose d'un ensemble $Q = \{Q^t_{(i,j)} \in \mathbb{R}^+\}$ qui représente le taux de gain ou de perte en traversant l'arc $(i, j) \in A$ au moment $t \in \{0, .., T\}$. En effet, ces modèles de flot permettent de garder en mémoire les incidents qui se sont déroulés plus tôt et impactant un groupe de personnes. Par exemple si une personne se casse le pied en tentant de traverser une zone dangereuse A, elle aura encore plus de mal à surmonter une seconde zone dangereuse B. Dans le cadre d'un graphe simple non dynamique, [Onaga, 2006] montre qu'il est possible de trouver la solution optimale en utilisant les chaînes augmentantes offrant le plus grand gain (i.e. ayant les plus petites pertes). [Groß et Skutella, 2012] présentent une approche basée sur les problèmes de flots généralisés dans un réseau dynamique qui prennent en compte aussi bien la durée (i.e. minimisation de la durée globale d'évacuation) que la notion de sécurité (i.e. minimisation de l'exposition au risque.)Aussi l'approche qu'ils utilisent allie les propriétés du réseau étendu dans le temps développé par [Ford et Fulkerson, 1958, Ford et Fulkerson, 1962], de sa compression (voir [Fleischer et Skutella, 2007]) et les méthodes développées pour les flots généralisés dont notamment celle de [Wayne, 1999]. Cette dernière approche peut être vue comme une résolution par epsilon contraintes où l'on fixe l'horizon de planification T (i.e. la date de fin impérative de l'évacuation)et où l'on minimise les pertes de personnes durant l'évacuation. Toutefois, dans ce dernier cas, il s'agit d'une évacuation avec perte d'individus alors que dans notre cas toutes les personnes doivent atteindre les centres de secours. Il montre aussi que modéliser un problème d'évacuation avec prise en compte de la sécurité en utilisant un flot généralisé rend le problème NP-Difficile au sens fort. Dans le cas particulier où les niveaux de sécurité (multiplicateurs) des arcs $Q^t_{(i,j)} \leq 1$ alors il existe un schéma d'approximation polynomial (FPTAS) permettant de résoudre le problème. Ils montrent aussi qu'en modifiant l'algorithme de résolution "earliest arrival flow" basé sur la recherche de chaînes augmentantes, il est alors possible de trouver les dates d'arrivées des évacués aux centres de secours permettant de maximiser le nombre de personnes indemnes. Le modèle proposé par [Groß et Skutella, 2012] présente aussi d'autres particularités comme le fait de ne pas prendre en compte la variation des données dans le

temps, et le fait que les niveaux de dangers soient proportionnels à la longueur des arcs.

2.2.6 Flots avec multiplicateurs

La première loi de Kirchhoff sur la conservation du flot caractérise les problèmes de flots classiques en partant du principe que du sommet source au sommet puits rien ne se perd et rien ne se crée. Toutefois lorsqu'il s'agit de modéliser des phénomènes de croissance ou de perte les modèles classiques de flot incluant la contrainte de conservation ne suffisent plus. En effet, si l'on considère qu'une partie du flot peut être perdue ou augmentée de volume suivant le milieu traversé alors il s'agit d'un flot généralisé pour lequel les arcs ou les sommets possèdent des coefficients multiplicateurs ou diviseurs. Ces types de problèmes de flot ont été introduits par [Dantzig, 1951] puis étudiés par [Jewell, 1962] et [Sahni, 1974]. Ils permettent de modéliser facilement des problèmes d'optimisation ayant des applications dans la vie réelle. Plusieurs illustrations aussi diverses que variées de ce concept sont données par [Gondran et Minoux, 2009] afin de permettre par exemple la modélisation de marché de change, des problèmes de chargement de machines, de gestion financière de stock ou encore la recherche d'une politique énergétique optimale.

Dans le cas d'une évacuation de masse à la suite d'une catastrophe naturelle, la sécurité des infrastructures de routage utilisées n'est plus garantie et peut mener à une mises en danger des évacués. Aussi la modélisation des points dangereux du réseau peut se faire à l'aide de coefficients (diviseurs) réels compris entre 0 et 1. Ceci correspond par exemple au nombre moyen de personnes impactées ou mise en danger lors de la traversée d'un nœud. Une fois cette transformation effectuée le problème d'évacuation peut être vu comme un problème de flot généralisé dans le temps où l'on cherchera à minimiser les pertes pour aller du sommet initial S au sommet final P. Une approche similaire est proposée par [Groß et Skutella, 2012] pour prendre en compte la durée d'évacuation ainsi qu'une notion de sécurité dans le cas de coefficients multiplicateurs entre 0 et 1 et pour un flot continu.

Dans cette section nous allons tout d'abord présenter la modélisation d'un problème d'évacuation en problème de flot avec multiplicateurs (section 2.2.6.1), par la suite nous détaillerons le modèle mathématique générique associé pour le problème de flot avec multiplicateurs. Enfin nous évoquerons les différentes méthodes de résolution existantes dont nous présenterons les grandes lignes (section 2.2.6.2). Pour plus de détails concernant les méthodes de résolution, nous vous invitons à vous référer aux travaux de thèse de [Wayne, 1999].

2.2.6.1 Données et modélisation de problèmes d'évacuation

Soit $G = (V, A, C, D)$ orienté composé de $|V| = N$ sommets et $|A| = M$ arcs. Chaque arcs dispose d'une durée de traversée $D_{(i,j)}$ ainsi que d'une capacité $C_{(i,j)}$ et chaque sommet possède un niveau de sécurité $Pr_{(i)}$. La figure 2.4 montre un exemple de graphe avec la sécurité sur les sommets. Il est facile de passer à une modélisation avec les niveaux de sécurité sur les arcs en scindant les sommets en deux et inversement en fusionnant des sommets. La figure 2.5 montre un exemple de transformation, de la modélisation par sommet vers une modélisation par arcs, centrée sur le sommet j. Nous pouvons aussi borner

le nombre d'unités B de flot sortant du sommet source S.

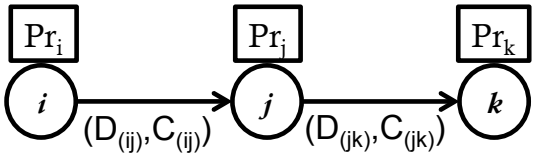


FIGURE 2.4 – Flow généralisé - Modélisation par sommets

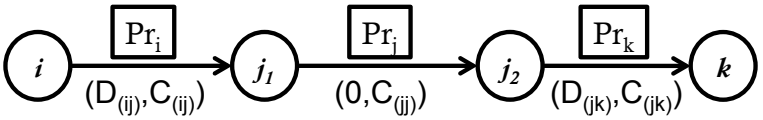


FIGURE 2.5 – Flow généralisé - Modélisation par arcs

2.2.6.2 Modèle mathématique

Ci-dessous le modèle mathématique de base pour le problème de flot avec multiplicateurs.

$$\text{Maximiser} \sum_{i \in \Gamma_P^-} F_{(i,P)} \tag{2.12}$$

Sous contraintes

$$F_{(i,j)} \leq C_{(i,j)} \qquad \forall (i,j) \in A \tag{2.13}$$

$$\sum_{i \in \Gamma_j^-} (F_{(i,j)} \times Pr_{(i,j)}) \geq \sum_{k \in \Gamma_j^+} F_{(j,k)} \qquad \forall j \in V \tag{2.14}$$

$$\sum_{i \in \Gamma_S^+} F(S,i) \leq B \tag{2.15}$$

La contrainte (2.13) est la contrainte classique de limite d'utilisation des arcs. Quel que soit l'arc $(i,j) \in E$ le nombre d'unités sur l'arc est inférieur à la capacité de l'arc. La contrainte (2.14) indique que lorsqu'un flot traverse un arc (i,j) ce dernier le multiplie par un coefficient et la quantité de flot sortant du sommet j est inférieure ou égale à la quantité totale de flot qui l'atteint. Enfin, la contrainte (2.15) permet de le limiter à B unités le nombre total d'unités sortant du sommet source S afin de prendre part au processus de routage. Pour le modèle mathématique prenant en compte un graphe dynamique avec les durées de traversées, se référer aux travaux de [Groß et Skutella, 2012] qui montrent que le problème est NP-Difficile au sens faible pour un flot continu grâce à une réduction depuis le problème PARTITION et propose un schéma d'approximation en temps polynomial (FPTAS) lorsque tous les coefficients $Pr_i \leq 1$. Plusieurs méthodes génériques existent afin de résoudre le problème de flot continu généralisé comme le "Gains scaling" introduit par

[Edmonds et Karp, 1972], la recherche de chaînes améliorantes par [Onaga, 2006] avec des coefficients $Pr_i \leq 1$ ou encore la recherche de chaînes améliorantes de capacité maximale [Goldberg *et al.*, 1989]. Toutefois, la plupart d'entre elles ont une complexité temporelle pseudo-polynomiale car dépendant de l'arc ayant la plus grande capacité dans le graphe. Certains restent utilisables en pratique comme celle développée par [Wayne, 2002] avec une complexité de $O(m^3n^2Log(\max C_{(i,j)}))$ mais pour autant que nous sachions le problème reste ouvert dans le cas général.

Dans cette partie de l'état de l'art nous avons présenté les différentes approches macroscopiques permettant de réaliser l'évacuation d'une population. Ainsi, différentes politiques d'évacuation permettant de prendre en compte les notions de durée et de sécurité lors du routage de personnes ont été présentées. Toutefois, ce type de modélisation ne prend pas en compte les aspects humains qui peuvent faire diverger un plan d'évacuation.

2.3 Modèle microscopique

Les modèles microscopiques permettent de prendre en compte les différents aspects des individus mais aussi leur environnement. Un état de l'art très complet proposé par [Greenwood *et al.*, 2014] permet de classer les sous-problèmes de cette approche en quatre sous-catégories :

- L'impact de l'aménagement des lieux sur le processus d'évacuation.
- L'impact des caractéristiques physiques des évacuées.
- Les différents types de lieux à évacuer.
- La modélisation des interactions entre les personnes.

Même si ces approches sont principalement indiquées dans le cadre de l'évacuation de bâtiments (immeuble, stade, bateau), il n'en demeure pas moins que des similitudes peuvent être trouvées avec l'évacuation de masse d'une ville. Nous allons donc présenter ces différentes sous-parties et préciser les liens qu'elles partagent avec notre problème de routage.

2.3.1 L'impact de l'aménagement des lieux sur le processus d'évacuation

La disposition de certains éléments dans un bâtiment est fait de manière à pouvoir réguler le flot d'individus s'y déplaçant. Par exemple, un escalier de secours permettant d'aller d'un point A vers un point B ou vice-versa est humainement plus fatigant à utiliser dans un sens que dans l'autre (i.e. descendre est plus facile que monter). Aussi la largeur de l'escalier permet d'influer sur le nombre de personnes pouvant y entrer à chaque instant alors que la hauteur des marches permet d'influer sur la vitesse de déplacement [Fujiyama et Tyler, 2011]. Un état de l'art et une étude de plusieurs cas d'évacuation utilisant les escaliers de secours en cas d'incendie peuvent être trouvés dans [Peacock *et al.*, 2010]. Avec ce genre d'aménagement, une étude prenant en compte la vitesse de déplacement est effectuée par [Fujiyama et Tyler, 2011]. Ils montrent que la vitesse de déplacement est fonction de l'âge et du poids des individus qui sont des attributs pouvant constituer des facteurs de ralentissement. En effet la vitesse de déplacement n'est plus linéaire et varie en fonction de l'effort à fournir. Dans notre cas, un parallèle peut être effectué avec l'utilisation de portions de routes pentues avec une forte déclinaison. L'évacuation utilisant les escaliers de

sécurité a l'avantage d'être sûre car utilisant un lieu robuste et pouvant résister au danger potentiel. Cependant, il peut ne pas être suffisant et d'autres solutions qui sont normalement prohibées peuvent être utilisées. Parmi celles-ci, il y a les escalators [Okamoto *et al.*, 2011], et les ascenseurs [Kinsey *et al.*, 2011]. L'escalator peut être utilisé pour reconfigurer le sens de circulation si l'on modifie le sens mais aussi la vitesse de déplacement. Lorsqu'ils ne sont plus fonctionnels, ils redeviennent des escaliers. Le cas des ascenseurs est plus mitigé car pouvant induire un sur-risque lors de l'évacuation. En effet, ils peuvent tomber en panne et emprisonner ses occupants.

En outre, il est difficile de garantir qu'il n'y aura pas l'intervention de facteurs humains comme le stress pouvant induire à des bousculades ou à des sur-capacités d'utilisation. [Kinsey *et al.*, 2011] montrent que même si ces deux modes d'évacuation sont autorisés (i.e. escalators et ascenseurs), les évacués n'envisagent de les utiliser que lorsqu'ils estiment avoir une chance de les utiliser rapidement sans attendre. Pire, son étude montre que près de $\frac{2}{3}$ des personnes n'utiliserons pas ce mode de déplacement même s'il est autorisé du fait du sur-risque potentiel en cas de catastrophe. Dans notre cas cela peut se rapporter à router les évacués vers des arrêts de bus afin qu'ils puissent prendre des navettes vers les centres de secours (voir [Bish, 2011], [Goerigk *et al.*, 2013]). D'autres types d'aménagements comme la largeur des portes (voir [Twarogowska *et al.*, 2014], [Daamen et Hoogendoorn, 2012]) et les intersections (voir [Appert-Rolland *et al.*, 2014],[Bretschneider et Kimms, 2011]) ainsi que la présence d'obstacles sur la route (voir [Olfati-Saber, 2006]) peuvent influer sur le comportement des évacués ainsi que sur leurs déplacements. Ces aménagements induisent de nouveaux problèmes dans le cadre en l'évacuation. En effet, autant pour la modélisation macroscopique une porte peut être représentée par un arc avec une capacité sans interaction entre les individus, autant dans le cadre de la modélisation microscopique c'est un lieu d'interaction voir même un goulot d'étranglement lorsque plusieurs individus cherchent à la traverser alors que leur corpulence ne le permet pas. Par exemple une porte peut être empruntée en même temps par deux enfants alors que ce n'est pas le cas de deux adultes. Ce problème peut aussi être étendu aux personnes en situation de handicap qui ne peuvent pas forcément emprunter les escaliers ou difficilement avec de l'aide. Il en va de même pour les intersections pour lesquelles on suppose qu'il n'y a pas de collisions alors que cet aspect doit être géré dans le cadre d'une approche microscopique.

2.3.2 L'impact des caractéristiques physiques des évacuées

Comme nous avons pu le voir dans la section 2.3.1, tous les évacués ne sont pas égaux face à l'environnement qu'ils doivent traverser. En effet, l'âge, le sexe, l'état physique et mental peuvent avoir une certaine importance afin que l'évacuation puisse se passer dans de bonnes conditions. Nous pouvons noter les cas particuliers concernant les enfants ainsi que les personnes à mobilité réduite. Du fait de leur âge ou de leur condition physique (vieillesse, handicap ou corpulence) certains des individus peuvent avoir certaines difficultés lors de l'évacuation. [Larusdottir et Dederichs, 2011] ont réalisé une étude sur l'évacuation d'une crèche. Dans cette étude, il s'agit de prendre en compte l'évacuation d'enfants avec différents niveaux de motricités allant de bébés de 6 mois à des enfants de 6 ans. En outre, le personnel de la crèche est limité en nombre et deux types de déplacements sont étudiés à

savoir sur sol plat ainsi qu'en utilisant des escaliers. Ils mettent aussi en évidence le fait que les enfants les plus âgés et connaissant les procédures d'évacuation ont tendance à courir durant l'évacuation ce qui peut induire des chutes et des blessures alors que les plus jeunes ont eux besoin de l'aide d'un adulte. Cette étude montre aussi que les enfants sont disciplinés et suivent les instructions d'évacuation car ils prennent les exercices d'évacuation comme un jeu. [Lu, 2011]) présente le cas d'exercices d'évacuation d'écoles en cas de séisme avec l'étude de différentes politiques d'évacuation et une population allant de 3 à 18 ans. Les conclusions de l'étude montrent qu'il n'y a pas de corrélation entre l'âge des individus à évacuer et leur vitesse de déplacement. En effet, plus les élèves étaient âgés et moins ils respectaient les consignes d'évacuation comme le fait de ne pas prendre ses affaires, de ne pas discuter avec ses connaissances ou de respecter les chemins à emprunter. Ces différences de performances au sein de la population peuvent aussi être constatées pour d'autres raisons. Parmi elles, figurent les personnes stressées ou subissant une pression [Daamen et Hoogendoorn, 2012] mais aussi les personnes à mobilité réduite. Dans le premier cas, le stress peut pousser les évacués à prendre des décisions en contradiction avec les consignes alors que dans le second cas il faut mobiliser une ou plusieurs personnes afin d'assurer la survie d'un individu ne pouvant pas se déplacer de manière autonome [Adams et Galea, 2011]. D'autres problèmes tels que les défauts d'aménagements peuvent aussi ralentir l'évacuation car ces derniers dépendent aussi des handicaps [Bengtson *et al.*, 2011]. Ainsi déclencher une évacuation avec seulement une alarme peut laisser de coté les malentendants qui auraient besoin d'un signal visuel. Il en va de même pour les malvoyants qui ne peuvent pas suivre les panneaux indiquant les sorties. Ces problèmes précédemment cités sont étudiés dans le cadre de l'évacuation d'infrastructures comme des bâtiments mais trouvent aussi une résonance dans d'autres types d'évacuation.

2.3.3 Les différents types de lieux à évacuer

L'environnement à évacuer influe fortement sur la durée d'évacuation ainsi que sur le stress engendré par le danger. Nous pouvons ainsi dans un premier temps considérer l'évacuation de couloirs ou de tunnels. Il s'agit soit de faire passer un ensemble individus d'un point A à un point B, soit de considérer qu'ils sont dans un tunnel encombré qu'ils doivent le quitter. [Rupprecht *et al.*, 2011] étudient le cas d'évacuation passant par des couloirs sans obstacles avec l'influence et la corrélation entre la largeur du couloir et la formation de rangées allant dans le même sens pouvant être considérées comme un flot continu de personnes. Il existe aussi le cas où les individus sont déjà dans un tunnel lorsque la situation d'urgence est arrivée et qu'ils doivent laisser leurs moyens de locomotion pour continuer à pied. [Li *et al.*, 2011] étudient le comportement des individus lors de l'évacuation d'une rame de métro en cas de sinistre et doivent regagner la surface. Une des déclinaisons de l'évacuation de tunnels est l'évacuation d'avions au sol en cas d'incendie (voir [Best *et al.*, 2014], [Zhi-Ming *et al.*, 2014]). Dans ce cas de figure, il existe plusieurs sorties possibles, des obstacles à éviter ainsi qu'un ou plusieurs foyers d'incendie qui s'étendent au fil du temps. A la différence d'autres types d'infrastructures où le processus de construction n'est pas guidé par les contraintes de durée d'évacuation, l'évacuation des avions intervient bien plus tôt dans leur conception. En effet lorsqu'ils dépassent une capacité de 44 personnes, ils doivent pouvoir garantir que leur évacuation à pleine charge peut être effectuée en moins de 90

secondes [Administration, 1990]. Ainsi, il s'agit donc de pouvoir simuler l'évacuation de ces derniers, bien avant leur construction, en considérant le plus possible les facteurs humains. En prenant en compte les structures verticales, des environnements comme les bateaux [Klüpfel *et al.*, 2001] et les bâtiments [Wang *et al.*, 2008] peuvent aussi être considérés. Toutefois, plus la structure considérée grandie moins l'approche microscopique permet de prendre en compte la particularité des lieux ainsi que les comportements des individus.

2.3.4 La modélisation des interactions entres les personnes

L'approche microscopique a pour but de simuler le plus possible les interactions entre les personnes à évacuer et leur environnement afin d'avoir un plan d'évacuation qui se rapproche le plus possible de la réalité. Une des particularités de cette approche est qu'elle n'est pas déterministe et est basée sur des données empiriques en rapport avec les évacués. Afin de prendre en compte ces aspects, une des modélisations possibles est de segmenter l'environnement en cellules pouvant contenir une ou plusieurs personnes. [Klüpfel *et al.*, 2001] propose un modèle basé sur les automates cellulaires dans le cadre d'une évacuation de bateau. Ils modélisent chaque cellule comme étant l'espace vitale nécessaire pour accueil une personne de taille moyenne. Chaque cellule ne peut être occupée que par une seule personne et chaque personne peut avoir des caractéristiques propres. En outre, une des particularités de leur approche est le comportement des individus quant à leurs directions ainsi que leurs vitesses de déplacement qui peuvent être influencées par des probabilités associées aux cellules sur lesquelles ils se trouvent. L'ordre de déplacement de chaque individu est obtenu de manière aléatoire et le nombre de cases pouvant être couvertes dépend de la perception de l'individu. Cette perception peut être réduite par les personnes se trouvant sur le chemin ainsi que par les obstacles. L'objectif n'est pas de minimiser la durée de l'évacuation de la population mais de maximiser la vitesse de déplacement des individus vers un point sûr quelconque.
De nombreux travaux utilisant la notion d'automate cellulaire pour l'évacuation de zones restreintes avec obstacles peuvent être trouvés dans la littérature (voir [Zheng *et al.*, 2010], [Alizadeh, 2011]). Du fait de la forme carrée des cellules, l'une des limites de cette approche réside dans l'incapacité de prendre en compte convenablement des obstacles et des virages n'ayant pas des angles droits. [Klüpfel, 2014] propose une segmentation de l'environnement en triangles permettant ainsi de modéliser des courbes ainsi que des lieux comme des ronds-points. Dans la même lignée [Hiyoshi *et al.*, 2014] propose une approche similaire pour modéliser l'environnement en se basant sur le diagramme de voronoï alors que [Borrmann *et al.*, 2012] utilise une modélisation hexagonale permettant de se déplacer dans six directions depuis un point.
Afin de pouvoir simuler l'évacuation d'une grande zone, il faut limiter la précision de l'approche microscopique. [Bretschneider, 2012] propose une modélisation où l'aspect microscopique n'est considéré que sur les intersections et les interactions qui s'y déroulent, alors que sur les arcs une modélisation par flot est effectuée. Ainsi, chaque intersection est modélisée par les différents points d'interactions possibles non partageables. Toutefois la gestion de ces interactions pour des trajectoires personnelles prédéfinies a une problématique similaire au "JobShop" qui est un problème d'ordonnancement difficile. [Borrmann *et al.*, 2012] propose une approche similaire combinant une approche microscopique et ma-

croscopique afin de bénéficier des points forts des deux, tout en limitant leurs points faibles. Aussi appelé approche "sandwich" il s'agit d'effectuer une résolution macroscopique basée sur le problème du "quickest flow" et par la suite effectuer la simulation microscopique de la solution obtenue. Une fois la simulation effectuée, cela permet d'avoir des indications sur les capacités des arcs ainsi que les vitesses de déplacement associées. Ces nouvelles données sont utilisées de nouveau dans le modèle mathématique. Ce processus est répété jusqu'à ce que les solutions retournées par les deux approches soient cohérentes.
D'autres approches microscopiques s'inspirant du règne animal existent. On peut citer [Olfati-Saber, 2006] lorsqu'il s'agit de modéliser la notion de groupe de personnes ayant des caractéristiques similaires à des bancs de poissons ou des nuées d'oiseaux en déplacement dans un environnement alors que [Forcael *et al.*, 2014] propose une approche basée sur une optimisation par colonie de fourmis.

2.4 Conclusion

Dans, ce chapitre nous avons présenté l'état de l'art concernant les problèmes d'évacuation. Comme nous l'avons montré, il existe principalement deux approches complémentaires que sont l'approche macroscopique et l'approche microscopique.
La modélisation des approches macroscopiques passe par l'utilisation de modèles de flot dans les graphes dynamiques afin de prendre en compte l'aspect temporel inhérent aux déplacements des individus. Aussi, suivant la politique d'évacuation adoptée, il est possible de considérer plusieurs critères en rapport avec la durée (i.e. durée moyenne d'évacuation ou date de fin d'évacuation) tout en garantissant que le flot sera maximal (i.e. toute la population sera évacuée). Nous avons aussi présenté différentes manières de prendre en compte la notion de sécurité en la modélisant comme un problème de flot à cout minimum (i.e. critère de sécurité additif) ou un problème de flot généralisé (i.e. critère de sécurité multiplicatif).
Les approches microscopiques passent par l'utilisation d'automates cellulaires ou de systèmes multi-agents qui interagissent entre eux mais aussi avec leur environnement. Aussi, nous avons dans un premier temps présenté l'impact des aménagements sur les individus devant être évacués (i.e. type d'escalier, largeur de couloir, porte, ...). Par la suite, nous avons aussi présenté l'impact des caractéristiques physiques (i.e. âge, sexe, corpulence,...) des personnes sur le processus d'évacuation. Enfin nous avons présenté les types d'environnement dans lesquels cette approche est envisageable ainsi que comment les appliquer.
Dans le chapitre 3 nous nous intéresserons principalement à l'approche macroscopique alors que dans le chapitre 4 nous nous intéresserons, par certains aspects, au rapprochement de l'approche macroscopique et de l'approche microscopique.

Chapitre 3

Routage de personnes sans pertes

3.1 Introduction du problème d'évacuation sans pertes

En matière d'évacuation, la préparation de scénarios d'évacuation ainsi que les exercices de prévention sont les premiers niveaux de sécurité qui peuvent être appliqués par les autorités afin de s'assurer que les différents services publics et la population sont bien sensibilisés aux mesures de sécurité. Ainsi la situation potentiellement dangereuse est clairement identifiée et le but est d'en éloigner les personnes afin de ne pas nuire à leur vie ou santé. Il peut aussi être considéré que l'on dispose d'un délai raisonnable permettant de mettre toutes les personnes à l'abri avant que la situation ne se détériore (i.e. évacuation préventive). Aussi, suivant le type de catastrophes d'origine naturelle ou humaine, cette approche est particulièrement indiquée.
En effet, il est possible de prévoir sur plusieurs jours la montée des eaux d'un fleuve en fonction de la pluviométrie de régions en amont et ainsi évacuer les zones dangereuses avant qu'elles ne soient touchées. Dans une plus large mesure, il est possible de prévoir les zones qui seront impactées par un cyclone tropical plusieurs jours à l'avance et tenter son évacuation avant qu'il n'atteigne les terres comme pour l'ouragan Katrina en Août 2005. Pour les catastrophes d'origine industrielle, cela s'illustre parfaitement notamment en Allemagne avec les bombes de la seconde guerre mondiale n'ayant toujours pas explosées comme pour l'évacuation près du pont de Mülheim à Cologne en 27 Mai 2015. En effet, lorsque l'une d'elles est découverte, un périmètre de sécurité est défini afin que les services de déminage puissent les déplacer en toute sécurité et les faire détonner dans un lieu dégagé. Ainsi, en partant de l'hypothèse qu'elles n'exploseront pas tant qu'aucun élément extérieur n'entrera en leur contact, l'évacuation de la population autour se fera calmement sans danger de mort. De ce fait, la préparation à de tels scénarios-catastrophes permet d'évaluer les risques encourus par la population mais aussi d'évaluer la capacité des forces de secours et des instances publiques à répondre à ce genre de situations critiques.

Ce processus mène à l'optimisation de deux critères que sont la durée d'évacuation et la sécurité. De ce fait, les démineurs ne peuvent commencer leur intervention que lorsque tous les civils auront quitté la zone dangereuse. Il est aussi à noter que pendant l'évacuation les chemins empruntés par les civils doivent autant que possible éviter de passer près de la source de danger au cas où l'hypothèse de départ concernant l'explosion de la bombe

ne serait plus valide. De manière classique, l'hypothèse de non perte de vie humaine est un problème central en matière d'évacuation. Ainsi nous supposons dans cette partie que toute personne évacuée finira par atteindre un centre de secours. Dans un premier temps, nous proposons un ensemble de méthodes exactes d'énumération du front de Pareto.
Par la suite, nous étendons notre approche lexicographique afin d'effectuer une énumération approchée.

3.2 Problème lexicographique

3.2.1 Introduction

L'évacuation de piétons lorsque des bâtiments sont en feu, des bateaux en train de sombrer ou en cas de catastrophes naturelles sont des problèmes bien connus. Les individus doivent quitter les zones dangereuses, aussi rapidement que possible, afin de rallier des emplacements sécurisés. Un grand nombre d'articles d'états de l'art en rapport avec le routage dans les réseaux dynamiques peuvent être trouvés dans la littérature comme [Aronson, 1989], [Ford et Fulkerson, 1962], [Hamacher et Tjandra, 2002] et [Powell *et al.*, 1995]. Suivant le niveau de précision souhaité, trois types de modélisation sont possibles et permettent de modéliser différents aspects d'un routage de masse. Il s'agit des modèles microscopiques, macroscopiques et mésoscopiques.
Les modèles microscopiques sont utilisés quand les attributs relatifs aux personnes à router sont connus. Ces attributs peuvent être des informations comme l'âge de la personne, sa vitesse de déplacement ou encore sa corpulence. Les personnes peuvent interagir entre elles en communiquant des informations utiles mais aussi néfastes comme leur stress ou encore se bousculer lorsqu'ils se trouvent dans un endroit confiné. Ainsi ces méthodes permettent de prendre en compte le plus d'aspects possible et permettent de produire un plan d'évacuation qui se rapproche de la réalité. Toutefois, même si ces approches permettent d'obtenir une durée d'évacuation proche de la réalité, elles requièrent beaucoup de puissance de calcul afin de simuler les différents aspects de l'évacuation ainsi que les interactions entre les évacués. En outre elles ne sont pas adaptées à l'évacuation de masse d'une grande ville.
Les modèles macroscopiques permettent d'avoir une vue globale du trafic dans le réseau considéré. En effet, ils permettent de considérer les individus comme un flot homogène ayant les mêmes attributs. Aussi, il n'y a pas non plus d'interactions entre les individus comme les bousculades ou les croisements de personnes aux intersections [Bretschneider, 2012]. Les méthodes basées sur une approche macroscopique donnent une bonne estimation de la durée minimale nécessaire pour router l'ensemble de la population et sont applicables à l'ensemble des infrastructures de grande taille allant d'un stade à une ville entière.
Enfin, aussi connu sous le nom d'approche sandwich [Hamacher *et al.*, 2011a], les méthodes mésoscopiques permettent d'unifier les approches microscopiques et macroscopiques. Ainsi, leur objectif principal est de minimiser l'écart entre les deux classes de méthodes dans le but d'avoir un plan d'évacuation aussi réaliste qu'applicable sur de grandes zones géographiques.
Plus de détails concernant ces méthodes peuvent être retrouvés dans la littérature comme dans [Borrmann *et al.*, 2012], [Hamacher et Tjandra, 2002] et [Klüpfel *et al.*, 2001].

Dans cette section, nous nous intéressons à l'évacuation macroscopique et multi-critère de personnes en cas de catastrophes naturelles comme des tremblements de terre, des glissements de terrain, des inondations ou encore en cas d'incendie de grande envergure. Pour ces cas de figure, il ne s'agit plus juste de minimiser la durée d'évacuation mais aussi de minimiser le risque encouru par la population durant l'évacuation à partir de données de sécurité modélisant le niveau d'endommagement des infrastructures. En effet, les dommages directs ou indirects comme l'effondrement d'infrastructures et la destruction du réseau routier influent sur les durées de trajet, la largeur des routes mais aussi leur sécurité. Dans ce contexte, le plus court chemin allant d'un point de rassemblement vers un centre de secours n'est potentiellement ni le chemin le plus rapide ni le chemin le plus sûr pour évacuer les individus. Notre objectif sera donc de produire un plan d'évacuation minimisant tout d'abord la durée totale d'évacuation et dans un second temps la sécurité associée au plan d'évacuation proposé.
Notons que suivant la politique d'évacuation adoptée, la prise en compte d'un second critère mène à une solution approchée concernant ce dernier alors que le critère de la durée globale de l'évacuation reste optimal. Cette partie est organisée comme suit :

La section 3.2.2 présente les méthodes de résolution génériques liées aux problèmes d'évacuation alors que section 3.2.3 donne une vue globale de notre problème d'évacuation. Nous présentons et illustrons notre nouvelle méthode de résolution dans les sections 3.2.4, 3.2.5 et 3.2.6. Enfin, dans la section 3.2.7, nous présentons les résultats obtenus en utilisant notre approche pour l'évacuation de la ville de Nice et nous analysons ces derniers dans la section 3.2.8.

3.2.2 Problèmes connexes

Dans cette partie, nous rappelons quelques problèmes d'évacuation utilisant une approche macroscopique basée sur les modèles de flots dynamiques sans pertes et nous montrons comment ils peuvent être utilisés pour résoudre notre problème de routage de masse lors d'une évacuation.

3.2.2.1 Earliest arrival flow

Aussi connu sous le nom de "Universal Maximum Flow", le "Earliest Arrival Flow" est un problème de routage en cas d'évacuation dont l'objectif est de maximiser le nombre de personnes pouvant être mises en sécurité à chaque instant donné $t \in H$. Ce problème d'évacuation est particulièrement utile car toute solution pour le Earliest Arrival Flow est aussi une solution pour le problème de flot maximum dynamique dont l'objectif est de maximiser le nombre de personnes pouvant être mises en sécurité avant une date limite T fixée. Cependant, dans le cas général, la réciproque n'est pas vraie. Afin de résoudre ce problème d'évacuation, [Hamacher et Tjandra, 2003] proposent un algorithme permettant de résoudre le problem du "Earliest Arrival Flow" avec une compléxité de $O(NM^2T^3 \max(C^t_{(i,j)}))$. Ce dernier permet de prendre en compte la variation des données dans le temps ainsi que la possibilité pour les évacués de patienter sur des zones de stationnement à capacité limitée. Pour cela, ils utilisent une méthode basée sur la recherche de chaînes augmentantes de plus

courts chemins. Cela revient aussi à trouver dans un graphe d'écart le plus court chemin arrivant à une date t_a fixée à un des centres de secours, tout en prenant en compte les capacités résiduelles des arcs. La prise en compte de plusieurs points de rassemblement et de plusieurs centres de secours est possible en ajoutant un sommet source initial, ainsi qu'un sommet puits final. Les arcs reliant ces sommets particuliers au reste du réseau dynamique ont une durée de trajet nulle et ainsi n'influent pas sur la topologie du graphe dynamique. Dans le cas où les données ne varient pas dans le temps, [Baumann, 2007] prouve qu'il est possible d'avoir un algorithme polynomial facilitant la résolution de ce problème. Ce dernier se base sur une approximation du graphe qui réduit le nombre d'arcs du réseau en gardant ceux qui sont susceptibles être utilisé. Utilisant le graphe ainsi construit comme base, [Baumann et Skutella, 2009] proposent une approche permettant de calculer le modèle de flot associé au graphe simplifié. Cela nécessite une transformation du problème de "Earliest Arrival Flow" en un problème de transbordement qui une fois effectué permet d'utiliser l'algorithme polynomial de [Hoppe et Tardos, 2000] qui permet de minimiser la durée d'évacuation.
Toutefois, comme nous l'avons dit dans l'état de l'art, le "Earliest Arrival Flow" n'existe pas pour tous les types de graphes dynamiques. Afin de s'assurer de son existence, [Schmidt et Skutella, 2014] ont caractérisé les deux types de sous-graphes ne devant pas se trouver dans le réseau utilisé. Aussi, de manière générale, il faut s'assurer que les graphes construits n'utilisent pas d'arcs de durées nulles (hors ceux associés aux sommets source et puits) et que l'on puisse rallier tout centre de secours depuis chaque point de rassemblement. Du fait de la définition de la fonction objectif communément utilisée, la complexité de ce problème d'évacuation est restée ouverte depuis des années. En 1982, [Jarvis et Ratliff, 1982] ont prouvé que minimiser le flot pondéré par les dates d'arrivées est équivalent à la minimisation de la durée moyenne de l'évacuation de toute la population. Se basant sur cette définition de la fonction objectif, [Disser et Skutella, 2015] ont prouvé que le "Earliest Arrival Flow" est un problème NP-Difficile. Au-delà de ce point, ils prouvent aussi que leur algorithme de résolution fait partie de la classe "NP-Mighty" en montrant que tout problème d'optimisation dans NP peut être résolu grâce à la politique d'évacuation basée sur le "Earliest Arrival Flow".
Tout en ayant la même date de fin d'évacuation, il est possible de définir un problème d'évacuation beaucoup moins contraint et dont le "Earliest Arrival Flow" est une solution particulière. Ce dernier se nomme le "Quickest Flow" et nous le présentons dans la section suivante.

3.2.2.2 Quickest flow

Le "Quickest Flow" ou flot le plus rapide est un problème d'évacuation où l'on cherche principalement à minimiser la durée globale de l'évacuation. Dans le meilleur des cas la durée moyenne d'évacuation est équivalente à celle du "Earliest Arrival Flow" et dans le pire des cas elle peut être égale à la date de fin d'évacuation ou l'horizon de planification T. Ainsi, on dispose d'un grand éventail de solutions possibles. Tout comme le "Earliest Arrival Flow", ce problème dispose de quatre variantes en fonction du nombre de points de rassemblement et de centre de secours. Mais toutes peuvent être transformées en un problème avec un unique sommet initial et un unique sommet final. Pour le cas classique

avec un sommet source et un sommet puits, plusieurs algorithmes polynomiaux ont été créés par [Burkard *et al.*, 1993] en utilisant le principe de la répétition du flot maximum à coût minimum dans un graphe statique. Ainsi, le flot correspondant est décomposé en un ensemble de chemins qui sont triés en fonction de leur cout (durée). Suivant l'ordre croissant des coûts (durées des chemins), ils sont utilisés autant que possible en s'assurant d'atteindre le sommet puits au plus tard à la date T. Ce processus est répété tant qu'il reste des individus à router depuis le sommet initial. Ce problème peut être généralisé quand plusieurs points de départ et d'arrivée sont considérés. Lors de cette généralisation, ce type de "Quickest Flow" devient un problème de transbordement nommé "Quickest Transshipment Flow". Dans ce dernier cas, nous avons plusieurs points de rassemblement depuis lesquels un certain nombre de personnes doivent être évacuées et un ensemble de centres de secours avec une capacité maximale d'accueil pour chacun d'eux. La fonction objectif reste la même à savoir la minimisation de la durée globale de l'évacuation. Pour autant que nous sachions, ces problèmes demeurent ouverts lorsque l'on considère que les données peuvent varier dans le temps. [Miller-Hooks et Stock Patterson, 2004] présentent une procédure permettant de réduire les quatre déclinaisons du "Quickest Flow" au cas classique avec un sommet source et un sommet puits en utilisant un graphe étendu dans le temps. En utilisant la propriété suivant laquelle tout "Earliest Arrival Flow" est aussi un "Quickest Flow", [Tjandra, 2003] crée un algorithme pseudo-polynomial se basant sur la recherche de chaînes augmentantes avec une complexité de $O(BT(M + NTLogN))$. Lorsque l'on limite les variations des données en n'autorisant que la possibilité d'avoir des fenêtres de temps de disponibilité sur les arcs le problème devient NP-Difficile et devient même non approximable [Köhler *et al.*, 2008].
Si les données ne varient pas dans le temps, [Hoppe et Tardos, 2000] propose un algorithme polynomial utilisant une décomposition par chaîne ainsi que leur répétition dans le temps. Ils montrent aussi que leur algorithme reste valide si l'on autorise l'attente dans les zones de stationnement.
En 2007, [Fleischer et Skutella, 2007] ont montré qu'il existe une instance du problème de "Quickest Flow" pour laquelle l'écart entre l'approche autorisant l'attente et celle l'interdisant est de $\frac{4}{3}$. Ils ont aussi prouvé que l'écart entre les deux approches pouvait au plus aller du simple au double. En 2014, [Groß et Skutella, 2015] fournissent une instance pour laquelle cet écart est atteint.
Cependant dans plusieurs cas pratiques, comme dans les télécommunications, les réseaux électriques ou en cas d'évacuation, le fait de pouvoir stationner n'est pas souhaité ou peut être interdit. En effet, cela nécessite une infrastructure de stockage qui a un coût non négligeable tout au long du réseau et peut être dangereux dans le cas d'individus stressés à cause des bousculades.

3.2.2.3 Sécurité durant l'évacuation

Comme nous l'avons déjà dit, l'objectif principal durant une évacuation est de minimiser la durée globale d'évacuation. Toutefois depuis quelques années, de nouveaux objectifs tels que la maximisation de la sécurité associée à un plan d'évacuation ou la minimisation de l'exposition des évacués aux dangers potentiels sont proposés dans la littérature. [Opasa-

non et Miller-Hooks, 2009] présent le problème permettant de trouver le plan d'évacuation le plus sûr. Ce dernier est très utile en cas d'évacuation d'urgence d'un bâtiment ou d'une zone géographique. L'objectif de ce problème d'évacuation est de trouver un ensemble de chemins maximisant la sécurité de chaque évacué. Cela revient aussi à minimiser le danger encouru par l'évacué devant prendre le plus de risque. La modélisation actuelle de l'approche proposée utilise un réseau étendu dans le temps et dont les données varient, ce qui limite son application à des zones géographiques de taille réduite. Toutefois, l'approche proposée est une méthode exacte ayant une complexité pseudo-polynomial de $O(BN^3T^2)$. [Lämmel *et al.*, 2011] quant à eux proposent une méthode microscopique à deux niveaux dans le but de résoudre le problème de minimisation de l'exposition des évacués aux dangers potentiels. Pour cela, deux types de chemins sont définis : des chemins longs mais extrêmement sûrs et des chemins courts mais dangereux qui ne seront utilisés que si les chemins sûrs sont indisponibles. Dans ce cas critique, seuls les mouvements qui permettent de s'éloigner des zones dangereuses sont possibles. Plus de détails concernant ces déplacements réduisant l'exposition au danger peuvent être trouvés dans la littérature [Hamacher et Tjandra, 2002]. Certains travaux ont permis d'allier les critères de sécurité et de durée à travers une évacuation multi-objectif. Pour cela, une des approches est de modéliser le problème d'évacuation comme un problème de flot maximum à cout minimum où la durée est modélisée par le flot lui-même et la sécurité l'exposition des évacués aux dangers (voir [Yuan et Han, 2010] et [Coutinho-Rodrigues *et al.*, 2012]).
Nous pouvons aussi considérer que les zones dangereuses peuvent être mouvantes comme un gaz ou nuage toxique, la montée des eaux dues à une inondation ou un feu de forêt progressant vers une ville. Dans ce cadre, [Göttlich *et al.*, 2011] proposent deux modèles mathématiques qui permettent de prendre en compte deux aspects de l'évacuation. Dans un premier temps, ils modélisent l'évolution et la dissipation d'un gaz dangereux qui définit les zones dangereuses ainsi que le niveau de danger auquel les évacués sont confrontés en fonction de la concentration du gaz. Dans un second temps, ils proposent un ensemble de fonctions objectifs qui prennent en compte les notions de sécurité et de durée. Par exemple, l'une des fonctions objectif optimisant la sécurité minimise l'exposition maximale aux dangers alors qu'un autre minimise l'exposition globale aux dangers de tous les évacués. D'un autre côté, le critère de la durée est optimisé en résolvant un problème de "Quickest Flow" dans un réseau étendu dans le temps.

Nous avons présenté un ensemble de problèmes et de méthodes de résolution en rapport avec l'évacuation sans pertes, c'est-à-dire où toute personne quittant un point de rassemblement arrivera à un centre de secours. Nous avons aussi présenté comment la prise en compte de la sécurité durant l'évacuation peut être effectuée. Dans la section suivante, nous présentons plus en détail le problème multicritère que nous allons résoudre.

3.2.3 Description du problème

3.2.3.1 Données

Nous considérons un réseau dynamique $G = (V, E, C, D, Q, \Delta, T)$ sur un horizon de planification T divisé en un ensemble de points temporels $H = \{0, \ldots, T\}$. Soient $V =$

$\{S_1, \ldots, S_K\} \cup \{P_1, \ldots, P_L\} \cup \{U_1, \ldots, U_N\}$ l'ensemble des sommets du réseau et $E \subseteq V \times V$ l'ensemble des M arcs du réseau. Pour chaque arc $(i, j) \in E$ et pour chaque point temporel $t \in H$ nous associons une capacité $C^t_{(i,j)}$, une durée de trajet $D^t_{(i,j)}$ et une note de sécurité $Q^t_{(i,j)}$ associés au trajet de i vers j au moment t. Les durées de trajet sur les arcs définissent le type de réseau que nous avons ainsi que les types de déplacement qui sont autorisés. Dans le cadre de notre approche macroscopique, nous considérons qu'il est impossible pour un individu d'effectuer un dépassement comme les solutions correspondantes associées peuvent être irréalisables à cause d'une sur utilisation des arcs (voir section 3.2.3.5). La sécurité est une valeur continue $\in [0.0, 1.0]$ (Resp $[Dangereux, \ldots, Sans\text{-}risque]$) qui mesure le niveau de dangerosité sur l'arc (i, j). Il peut être vu comme la probabilité pour une personne de traverser l'arc sans subir aucun dommage. Nous avons aussi $\Delta = \{\Delta^\tau_{(i,j)} \in \{0, .., T\}\}$ qui représente la date $t \in \{0, .., T\}$ à laquelle il faut partir du sommet i pour arriver au sommet j à la date τ sachant que l'arc $(i, j) \in E$ et $\tau = t + D^t_{(i,j)}$. Soient les ensembles de sommets $\{S_1, \ldots, S_K\}$, $\{P_1, \ldots, P_L\}$ et $\{U_1, \ldots, U_N\}$ représentant K point de rassemblement, L centres de secours et N sommets intermédiaires définissant le réseau. Sur chaque point de rassemblement S_k se trouvent B_{S_k} personnes à évacuer, dont l'évacuation débutera au plus tôt à la date Rs_k. A ces points de rassemblement S_k nous ajoutons un point de rassemblement fictif S représentant le super sommet source de notre graphe. Chaque centre de secours P_l peut accueillir au plus A_{P_l} individus. A ces centres de secours P_l nous ajoutons un centre de secours fictif P représentant le super sommet puits de notre graphe. Nous supposons aussi que tout centre de secours peut être atteint depuis tout point de rassemblement et que l'on dispose d'assez d'espace dans l'ensemble des centres de secours pour accueillir l'ensemble des évacués :

$$B = \sum_{k=1}^{K} B_{S_k} \leq \sum_{l=1}^{L} A_{P_l}.$$

3.2.3.2 Variables

Soient les variables définies ci-dessous :
$F^t_{(i,j)} \in \mathbb{N}$: Le nombre d'individus entrant dans l'arc (i, j) au moment t.
$S^t_{(i,j)} \in \mathbb{R}^+$: Le nombre fractionnaire d'individus indemnes entrant dans l'arc (i, j) au moment t.
$X^t_{(i,j,k)} \in \{0, 1\}$: La variable booléenne indiquant si l'arc (i, j) est utilisé par l'évacué k au moment t.

3.2.3.3 Contraintes

S.C

$$F^t_{(i,j)} = \sum_{k\in 1}^{B} X^t_{(i,j,k)} \qquad \forall (i,j) \in E, \forall t \in \{0,..,T\} \quad (3.1)$$

$$0 \leq F^t_{(i,j)} \leq C^t_{(i,j)} \qquad \forall (i,j) \in E, \forall t \in \{0,..,T\} \quad (3.2)$$

$$0 \leq S^t_{(i,j)} \leq F^t_{(i,j)} \qquad \forall (i,j) \in E, \forall t \in \{0,..,T\} \quad (3.3)$$

$$F^0_{(S,S_k)} \leq B_{S_k} \qquad \forall S_k \in \Gamma^+_S \quad (3.4)$$

$$F^t_{(S,S_k)} = 0 \qquad \forall S_k \in \Gamma^+_S,\ \forall t \in \{1,..,T\} \quad (3.5)$$

$$\sum_{t=0}^{R_{S_k}-1} \sum_{j\in\Gamma^+_{S_k}} F^t_{(S_k,j)} = 0 \qquad \forall S_k \in \Gamma^+_S \quad (3.6)$$

$$\sum_{t=R_{S_k}}^{T} \sum_{j\in\Gamma^+_{S_k}} F^t_{(S_k,j)} \leq F^0_{(S,S_k)} \qquad \forall S_k \in \Gamma^+_S \quad (3.7)$$

$$\sum_{i\in\Gamma^-_j} F^{t-D^{\Delta^{(t)}_{(i,j)}}}_{(i,j)} = \sum_{k\in\Gamma^+_j} F^t_{(j,k)} \qquad \forall j \in V \setminus \{S,P\}, \forall t \in \{0,..,T\} \quad (3.8)$$

$$\sum_{i\in\Gamma^-_j} S^{t-D^{\Delta^{(t)}_{(i,j)}}}_{(i,j)} \cdot Q^{t-D^{\Delta^{(t)}_{(i,j)}}}_{(i,j)} = \sum_{k\in\Gamma^+_j} S^t_{(j,k)} \qquad \forall j \in V \setminus \{S,P\}, \forall t \in \{0,..,T\} \quad (3.9)$$

$$\sum_{t=0}^{T} \sum_{i\in\Gamma^-_{P_l}} F^t_{(i,P_l)} \leq A_{P_l} \qquad \forall P_l \in \Gamma^-_P \quad (3.10)$$

$$\sum_{t=0}^{T} \sum_{i\in\Gamma^-_j} X^t_{(i,j,k)} \leq 1 \qquad \forall k \in \{1,..,B\}, \forall j \in \{U_1,..,U_N\} \quad (3.11)$$

$$\sum_{h\in\Gamma^-_i} X^t_{(h,i,k)} = \sum_{j\in\Gamma^+_i} X^{t+D^t_{(h,i)}}_{(i,j,k)} \qquad \forall i \in V \setminus \{S,P\}, \forall t \in \{0,..,T\},$$
$$\forall k \in \{1,..,B\} \quad (3.12)$$

Ci-dessous, un ensemble de contraintes qui permettent de définir le problème d'évacuation considéré :

— Le flot entrant dans l'arc (i,j) au moment t est égal au nombre d'évacués entrant dans cet arc au moment t (3.1).

— Le nombre d'évacués entrant dans l'arc (i,j) au moment t est inférieur ou égal à la capacité de l'arc (3.2).

— le nombre fractionnaire d'individus indemnes entrant dans l'arc (i,j) au moment

t est inférieur ou égal au nombre total de personnes entrant dans l'arc au même moment (3.3).
— Tous les évacués doivent quitter le super sommet source S à la date $t = 0$ (3.4), (3.5).
— L'évacuation de chaque point de rassemblement S_k ne peut débuter avant la date Rs_k (3.6).
— B_{S_k} personnes doivent être évacuées du point de rassemblement S_k (3.7).
— Le nombre total d'évacués arrivant au sommet j au moment t est égal au nombre de personnes quittant ce somment à ce même moment (3.8).
— le nombre fractionnaire d'évacués atteignant le sommet j au moment t est égal au nombre fractionnaire d'individus le quittant à ce même moment (3.9). La baisse du niveau de sécurité le long des chemins fait référence au principe de flot généralisé (voir [Wayne, 1999]).
— Chaque centre de secours P_l peut accueillir au plus A_l évacués (3.10).
— Un évacué ne peut se retrouver deux fois sur le même sommet durant son périple. (3.11)
— Un évacué a besoin d'une durée de $D^t_{i,j}$ pour aller du sommet i au sommet j à la date t (3.12).

3.2.3.4 Fonctions objectif

Dans cette partie, nous décrivons les fonctions objectifs qui peuvent être considérées avec ce modèle mathématique.

$$\text{Maximize} \sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} F^t_{(P_l,P)} : \text{Recherche dichotomique sur l'horizon } T \tag{3.13}$$

$$\text{Minimize } \frac{1}{B}\left(\sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} \left(t \cdot F^t_{(P_l,P)}\right)\right) + HV\left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} F^t_{(P_l,P)}\right) \tag{3.14}$$

$$\text{Minimize } \left(\sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} \left(t \cdot F^t_{(P_l,P)}\right)\right) + HV\left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} F^t_{(P_l,P)}\right) \tag{3.15}$$

$$\text{Minimize } \frac{1}{B}\left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} S^t_{(P_l,P)}\right) \tag{3.16}$$

$$\begin{aligned}\text{Minimize } &\left(\sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} \left(t \cdot F^t_{(P_l,P)}\right)\right) + HV\left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} F^t_{(P_l,P)}\right) \\ &+ \left(\frac{1}{B}\left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma^-_P} S^t_{(P_l,P)}\right)\right)\end{aligned} \tag{3.17}$$

La première fonction objectif permet d'avoir la valeur correspondant à l'horizon de planification T. Cette valeur peut être obtenue en minimisant la date d'arrivée du dernier évacué dans un des centres de secours (3.13). Si les données varient en fonction du temps, cette valeur peut être obtenue en effectuant une recherche dichotomique sur la durée d'évacuation en appliquant un des algorithmes de recherche de flot maximum à cout minimum sur un graphe étendu dans le temps. Sinon si les données ne varient pas dans le temps, cette valeur peut être obtenue en utilisant la propriété de Klinz décrite dans [Hoppe et Tardos, 2000] qui permet d'obtenir cette valeur en temps polynomial sans pour autant calculer le plan d'évacuation associé.
La seconde et la troisième fonction objectif sont liées au type de politique d'évacuation que l'on veut mettre en place. La durée d'évacuation correspondante T est obtenue grace à (3.13). Respectivement, ces fonctions objectif minimisent la durée moyenne d'évacuation des individus ainsi que la somme pondérée du flot arrivant aux centres de secours aux différents moments [Jarvis et Ratliff, 1982]. Ces politiques d'évacuations sont présentées par (3.14) et (3.15).

Dans notre modèle, nous autorisons les évacués à rester au niveau des points de rassemblement $S_k \in \Gamma_S^+$ du début à la fin de l'évacuation. Aussi, nous pénalisons les fonctions objectif (3.14) et (3.15) avec $HV \left(B - \sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} F^t_{(P_l,P)} \right)$ lorsque qu'une personne n'a pas pu atteindre un des centres de secours avant la fin de l'évacuation. HV (i.e. High Value) étant une constante de grande valeur ($HV = (T+1) \cdot B$ est suffisant). De ce fait, la pénalisation est égale à zero si toutes les personnes atteignent les centres de secours avant la fin de l'évacuation. La quatrième fonction objectif est liée à la notion de sécurité durant l'évacuation et permet de mesurer le nombre fractionnaire d'évacués pouvant atteindre les centres de secours en étant indemne (3.16). Cette valeur varie de 1.0 à 0.0 avec 0.0 signifiant que tous les évacués ont pu éviter les zones dangereuses alors que la valeur 1.0 signifiera que toute la population a été exposée aux dangers durant l'évacuation dans le but d'atteindre les centres de secours. Il est possible combiner les fonctions objectif (3.13), (3.15) et (3.16) dans une seule qui est définie par (3.17). Cette nouvelle fonction objectif donne une solution qui est un optimum de Pareto extrême avec la meilleure durée d'évacuation possible ainsi que la meilleure sécurité possible. Les valeurs correspondantes peuvent être respectivement obtenues en récupérant la partie entière et la partie décimale de la fonction objectif unifiée. Tout au long de cette partie Lexicographique, nous faisons référence à la fonction objectif (3.17) pour le problème Lex((Q|S) Flow) [1].

3.2.3.5 Variation des données dans le temps

Les données relatives au réseau routier peuvent varier dans le temps. Celles-ci peuvent aller d'un état statique sans modification à un état où les données changent à chaque instant $t \in \{0, \ldots, T\}$. Dans ce cas extrême, il est nécessaire de construire l'ensemble du réseau étendu dans le temps. Quel que soit le cas de figure, les données relatives à la capacité et à la sécurité peuvent varier sans aucune restriction mais ce n'est pas le

1. $Lex(Quickest|Safest)\ Flow$

cas concernant les données relatives aux durées de trajet. En effet, les durées de trajet permettent de définir le réseau dynamique ainsi que les types de mouvements qui sont utilisés. Ces restrictions peuvent être illustrées en considérant un couloir à une voie ainsi que le réseau dynamique associé. La figure. 3.1 présente une instance pour laquelle B_{S_1} personnes doivent être évacuées depuis le sommet source S_1 à partir de la date R_{S_1} vers le sommet puits P_1 ayant une capacité d'accueil de A_1 personnes. La capacité et la sécurité de l'arc (S_1, P_1) ne varient pas durant l'évacuation et ont respectivement une valeur de 1 et de 1.0. Maintenant, supposons que la durée de trajet $D_{(S_1,P_1)}$ puisse varier dans le temps. Comme décrit dans la figure. 3.2, $D_{(S_1,P_1)}$ est égale à 2 unités de temps à partir de la date $t = 0$, elle augmente à 3 unités de temps à partir de la date $t = 2$ et diminue à 1 unité de temps à partir de la date $t = 3$ jusqu'à la fin de l'évacuation. Le réseau étendu correspondant montre les types de mouvements qui sont possibles, les moments auxquels un sommet particulier peut être atteint et quand nous devons quitter ses prédécesseurs pour l'atteindre à un moment donné (voir $\Delta^t_{(i,j)}$). Par exemple, $D^0_{(S_1,P_1)}$ (*i.e.* $D_{(S_{10},P_{12})}$) $= 2$ alors que $\Delta^2_{(S_1,P_1)}$ (*i.e.* $\Delta_{(S_{10},P_{12})}$) $= 0$.
Cependant, cette configuration montre qu'il y a deux cas où des conflits peuvent apparaître et où les évacués peuvent se mettre à doubler ceux qui se trouvaient déjà sur un arc avant eux. Dans le premier cas de figure, les arcs dynamiques (S_{12}, P_{15}) et (S_{13}, P_{14}) permettent de violer la contrainte de capacité maximale de l'arc statique (S, P). Cela veut dire qu'entre les dates $t = 3$ et $t = 4$, le niveau d'utilisation de l'arc (S, P) peut être égal à 2 alors que la capacité maximale est limitée à 1 unité de flot. Ce cas de figure mène à des plans d'évacuation irréalisables en pratique. Le même type de problème peut être détecté avec les arcs dynamiques (S_{12}, P_{15}) et (S_{14}, P_{15}) où les évacués utilisant un arc à des moments différents arrivent au même moment. Ceci mène aussi à des solutions irréalisables comme la capacité maximale de l'arc est dépassée juste avant d'atteindre la sortie de l'arc P_1 tout comme une porte à largeur limitée. Il est aussi possible de considérer le cas inverse où le nombre de personnes apparaissant à la sortie P_1 de l'arc à la date $t = 5$ dans le réseau dynamique excède le nombre maximal fixé par le graphe statique initial. Dans la suite, nous supposons que lorsque les données varient dans le temps, ces deux cas de figure ne seront pas pris en compte. Si toutefois le cas se présente, les arcs ne respectant pas ces propriétés seront rendus inutilisables avec une capacité maximale fixée à 0. Une fois les durées de trajet fixées, les données de capacités et de sécurité peuvent varier sur les arcs sans aucune restriction.

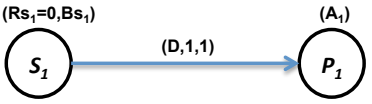


FIGURE 3.1 – Exemple de données variant dans le temps (1)

3.2.3.6 Exemple d'évacuation multicritère

Dans le cas de ce problème d'optimisation, les critères de durée et de sécurité sont conflictuels. La figure 3.3 décrit un exemple simple d'évacuation pour lequel un évacué doit

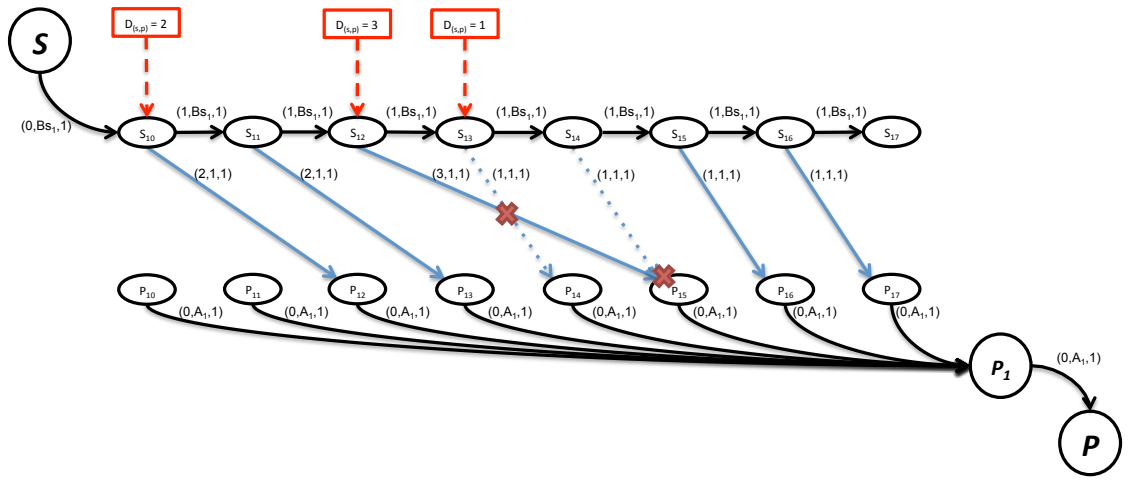


FIGURE 3.2 – Exemple de données variant dans le temps (2)

aller du sommet S au sommet P.
Chaque arc (i, j) du graphe dispose des attributs $(D_{(i,j)}, C_{(i,j)}, Q_{(i,j)})$ (dans cet exemple, les données ne varient pas dans le temps).
Si la personne à évacuer choisit de prendre le chemin $(S, 1, 3, P)$, alors il arrivera à la date 4 (i.e. $1+2+1$) avec une sécurité de 1 (i.e. $1 \times 1 \times 1$). Cependant, si elle choisit de prendre le chemin $(S, 2, 3, P)$, alors elle arrivera à la date 3 (i.e. $1+1+1$) avec une sécurité de 0.5 (i.e. $1 \times 0.5 \times 1$).

Cet exemple montre que réduire la durée d'évacuation peut réduire la sécurité du plan d'évacuation correspondant alors que augmenter la sécurité peut augmenter la durée globale de l'évacuation.

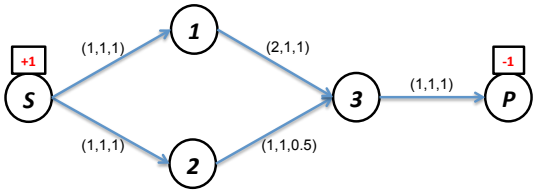


FIGURE 3.3 – Evacuation multicritère avec prise en compte de la durée et de la sécurité

Dans la section suivante, nous ne considérerons que le problème lexicographique Lex((Q|S) Flow) (i.e. minimiser la durée globale de l'évacuation dans un premier temps et lorsque nous avons deux plans d'évacuation en construction à durée identique, nous choisissons celui qui a la meilleure sécurité). Pour que cette approche puisse être appliquée à des instances de taille réelle, nous présentons aussi une nouvelle structure de données permettant une résolution efficace du problème.

3.2.4 Transformation du réseau dynamique

Dans cette section nous présentons notre principale contribution concernant le réseau dynamique.

Le réseau étendu dans le temps, qui a été créé par [Ford et Fulkerson, 1962] en 1962, était l'une des premières modélisations d'un réseau dynamique qui était utilisée pour résoudre le problème de flot maximum dans un réseau avec des capacités et des durées de traversées discrètes sur les arcs. Pour créer un tel réseau dynamique, les sommets et les arcs du réseau doivent être dupliqués dans le temps. Une fois construit, ce réseau permet d'utiliser les algorithmes de flot classiques comme celui de Ford-Fulkerson ou d'Edmond-Karp. Toutefois, ces algorithmes qui dans le cas statique sont polynomiaux deviennent alors pseudo-polynomiaux. La taille du réseau étendu dans le temps lui-même devient un problème à cause de la limitation due à la taille des instances qui peuvent être résolues. Dans cette partie, notre contribution consistera en la proposition d'une nouvelle structure de données permettant d'utiliser le principe du réseau étendu dans le temps sans pour autant le construire explicitement. Pour cela nous synthétisons l'ensemble de ses informations dans un nouveau réseau appelé Réseau Virtuel Dynamique. Nous introduisons aussi le principe de mémoire de décision qui est une matrice creuse dans le but d'obtenir l'état d'utilisation de chaque arc, au lieu d'utiliser directement le graphe résiduel qui aurait nécessité la construction du réseau étendu dans le temps.

3.2.4.1 Du problème de transbordement au problème de "Quickest Flow"

Dans cette partie, nous décrivons comment transformer tout problème de transbordement en un problème classique d'évacuation à savoir le "Quickest Flow". Pour une meilleure compréhension utilisons le réseau étendu dans le temps. Pour cela, nous étendons la procédure créée par [Miller-Hooks et Stock Patterson, 2004] en ajoutant les dates de départ au plus tôt Rs_k de chaque point de rassemblement S_k.
La figure. 3.4 décrit un problème de transbordement pour lequel B_k individus doivent être évacués depuis K points de rassemblement S_k à partie des dates Rs_k vers un ensemble de L centres de secours P_l ayant une capacité d'accueil maximale de A_l individus.

Soit S le sommet source. Comme l'évacuation de chaque point de rassemblement S_k, $k \in \{1, \ldots, K\}$, ne peut commencer au plus tôt qu'à la date Rs_k, nous ajoutons des arcs statiques $(S, S_k^{Rs_k})$ avec $S_k^{Rs_k}$ faisant référence au point de rassemblement S_k à la date Rs_k dans le réseau étendu dans le temps. Cet arc a une durée de trajet de Rs_k, une capacité de B_k et une sécurité de 1.0.
Pour permettre l'attente sur les points de rassemblement, nous ajoutons dans le réseau étendu pour chaque sommet S_k, $\forall t \in \{0, \ldots, T-1\}$ un arc (S_k^t, S_k^{t+1}). Ces arcs ont une durée de trajet de 1, une capacité de B_k, et une sécurité de 1.0. Un exemple de construction du super sommet source peut être retrouvée dans la partie gauche de la figure. 3.5.
Soit P un super sommet puits. Pour chaque centre de secours P_l et $\forall t \in \{0, \ldots, T\}$, nous ajoutons dans le réseau étendu dans le temps un arc (P_l^t, P_l) avec P_l^t représentant le centre de secours P_l à la date t. Ces arcs ont une durée de trajet de 0, une capacité de A_l et une sécurité de 1.0. Pour s'assurer que la capacité limite des centres de secours P_l ne soit pas

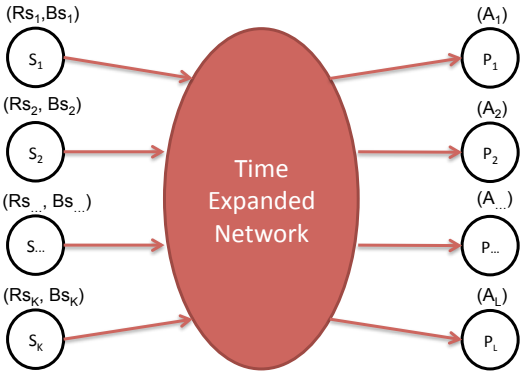


FIGURE 3.4 – Exemple de problème de transbordement

violée, nous ajoutons des arcs statiques (P_l, P) avec une durée de trajet de 0, une capacité de A_l et une sécurité de 1.0. Pour plus de détails concernant cette construction du super sommet puits, se référer à la partie droite de la figure. 3.5.

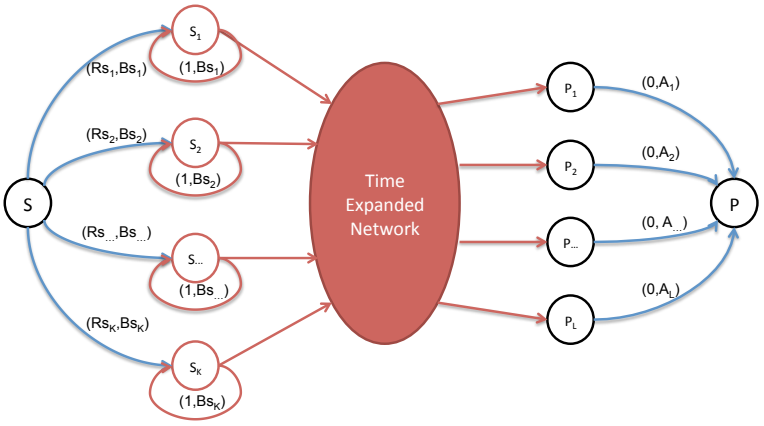


FIGURE 3.5 – Problème classique de Quickest Flow

3.2.4.2 D'un réseau étendu dans le temps au Réseau Virtuel Dynamique

L'utilité du Réseau Virtuel Dynamique est d'éviter d'avoir besoin de construire un réseau étendu dans le temps pour trouver le plus court chemin disponible entre le super sommet source S et le super sommet puits P. Nous commençons par présenter quelques propriétés relatives aux réseaux étendus dans le temps et son extension afin de permettre la prise en compte de données variant dans le temps.

Soit $G_{dyn} = (V, E, C, D, Q, T)$ un graphe dynamique qui ne permet pas l'attente sur les sommets communs appartenant à l'ensemble U. Soient Φ_i une chaîne augmentante i allant

du sommet S au sommet P et $t_{\Phi_i}(X)$ le moment auquel le sommet X est traversé par la chaîne augmentante Φ_i.

Lemme 1. *: Soient $t_{\Phi_i}(X)$ le moment auquel le sommet X est traversé par la chaîne augmentante Φ_i et $V(\Phi_i)$ l'ensemble des sommets utilisés par la chaîne augmentante Φ_i. Soient Φ_1, Φ_2 deux chaînes augmentantes allant du super sommet source S au super sommet puits P.*
$\forall I \in V(\Phi_1) \bigcap V(\Phi_2),\ t_{\Phi_1}(I) \leq t_{\Phi_2}(I) \Longleftrightarrow t_{\Phi_1}(P) \leq t_{\Phi_2}(P)$. Quel que soit le sommet I se trouvant dans l'intersection des sommets composant les chaînes augmentantes Φ_1 et Φ_2, la date à laquelle il est traversé par la chaîne augmentante Φ_1 est inférieure ou égal à la date de traversée par la chaîne augmentante Φ_2 si et seulement si la chaîne Φ_1 atteint le super sommet puits P au moins avant la chaîne Φ_2.

Démonstration. : La chaîne augmentante Φ allant du super sommet source S au super sommet puits P dans le réseau étendu dans le temps est obtenue en appliquant l'algorithme d'étiquetage de Dijkstra. Outre le plus court chemin de S à P, cet algorithme permet aussi d'avoir le plus court chemin du sommet S à tout autre sommet dans l'ensemble U. Si toutes les étiquettes partielles qui appartiennent à la chaîne augmentante Φ_1 ont des valeurs inférieures ou égales à celles associées à la chaîne augmentante Φ_2 alors Φ_1 permettra d'atteindre le super sommet puits P au moins avant Φ_2. Maintenant si nous prenons le raisonnement inverse, supposons que la chaîne augmentante Φ_1 permet d'atteindre le super sommet puits P au moins avant la chaîne Φ_2, alors en utilisant l'algorithme de Dijkstra, toutes les étiquettes appartenant aux sommets communs dans U qui appartiennent à la chaîne augmentante Φ_1 auront une valeur inférieure ou égale aux étiquettes des sommets communs appartenant à Φ_2. □

Corollaire 1. *: Ayant deux chemins définis par Φ_1 et Φ_2, si Φ_1 permet d'atteindre le super sommet puits P avant Φ_2 alors tous les sommets $\in V(\Phi_1) \bigcap V(\Phi_2)$ seront atteints par Φ_1 au moins avant Φ_2.*

Lemme 2. *: Même au cas où les durées de trajet varient dans le temps, le lemme 1 reste valide dans le cadre d'une approche macroscopique en cas d'évacuation.*

Démonstration. : La principale hypothèse en cas d'évacuation macroscopique est que l'ensemble des individus ont des attributs identiques comme la vitesse de déplacement. Supposons que $t_{\Phi_1}(P) \leq t_{\Phi_2}(P)$ et $\exists I \in V(\Phi_1) \bigcap V(\Phi_2)$ tel que $t_{\Phi_1}(I) \leq t_{\Phi_2}(I)$. Suppossons aussi que $\exists J \in \Gamma_I^+ \bigcap V(\Phi_1) \bigcap V(\Phi_2)$ tel que $t_{\Phi_1}(J) > t_{\Phi_2}(J)$. Cela veut dire que les évacués prenant le chemin associé à Φ_2 dépasseront ceux qui ont emprunté l'arc (I, J) avant ceux utilisant le chemin défini Φ_1. Ceci est impossible du fait que les modèles macroscopiques d'évacuation se basent sur le fait que les évacués ont la même vitesse de déplacement. □

Corollaire 2. *: En utilisant le plus court chemin disponible à un moment, même si un groupe* 1 *de personnes quittent dans de mauvaises conditions un point de rassemblement S_k à la date t dans le but d'atteindre le centre de secours P_l et qu'un groupe* 2 *en fait de même à partir de la date $t+\alpha$, $(t+\alpha) \in \{t, \ldots, T\}$ avec de meilleures conditions de déplacement, alors le groupe 2 ne pourra pas arriver plus tôt que le groupe* 1.

Selon les lemmes 1 et 2, pour une date d'arrivée fixée, toutes les étiquettes des sommets communs associés à la chaîne augmentante Φ_i pour aller du super sommet source S au super sommet puits P sont optimales. Comme l'algorithme de plus court chemin de Dijkstra calcule à chaque étape le plus court chemin dont la capacité résiduelle est non nulle et permettant d'aller du super sommet source S à tous les sommets $\in U$ qui sont des sommets statiques. il n'est donc pas nécessaire d'appliquer l'algorithme de Dijkstra sur le réseau étendu dans le temps.

Cependant, il est nécessaire de savoir si un arc dans le réseau étendu modifié est étendu ou non dans le temps [Miller-Hooks et Stock Patterson, 2004]. Le Réseau Virtuel Dynamique est un réseau statique qui a été créé en utilisant le graphe étendu modifié pour lequel, pour chaque arc statique on définit s'il doit être étendu ou pas dans le temps.

Un exemple de Réseau Virtuel Dynamique est proposé sur la figure. 3.6 pour lequel $\forall k \in \{1, \ldots, K\}$ et $\forall l \in \{1, \ldots, L\}$ les arcs $(S, S_k), (P_l, P)$ ne peuvent pas être étendus dans le temps alors que les autres le sont.

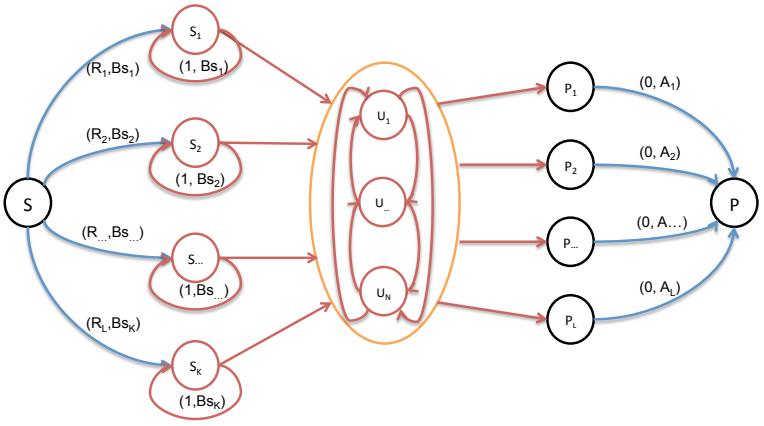


FIGURE 3.6 – Réseau Virtuel Dynamique

3.2.4.3 Du graphe résiduel à la mémoire de décision

Tout comme pour le réseau étendu dans le temps, le graphe résiduel dynamique nécessite la construction d'un graphe étendu dans le temps. Cette construction requiert aussi la duplication de chaque arc dans le temps pour savoir la capacité restante à chaque moment. Normalement, l'obtention d'une telle information peut être effectuée avec une complexité temporelle de $O(1)$ et permet de savoir si un arc est utilisable ou non lors de la recherche du plus court chemin du sommet S au sommet P. Malheureusement tout comme le réseau étendu dans le temps, la complexité spatiale du graphe résiduel dynamique est pseudo-polynomiale avec une taille de MT, ce qui se révèle aussi être un problème. En outre, construire un tel graphe étendu n'a pas de sens dans la mesure où la plupart des arcs resterons inutilisés du début jusqu'à la fin. Pour résoudre ce problème nous introduisons la notion de mémoire de décision qui est une matrice creuse de trois dimensions. Les deux premières dimensions représentent la matrice sommet-arcs du Réseau Virtuel Dynamique.

Alors que la troisième dimension est initialement vide et représente les niveaux d'usage des arcs triés en fonction des dates d'utilisation $t \in \{0, \ldots, T\}$.

Si un arc $(i, j) \in E$ est utilisé à la date t alors son niveau d'usage est enregistré dans la mémoire de décision. Cette nouvelle structure a une double utilité. Elle réduit la taille du graphe résiduel au strict minimum et représente le plan d'évacuation.

Toutefois, dans le pire cas de figure, l'ajout, la récupération ou la mise à jour d'une information requière une recherche dichotomique d'une complexité temporelle de $O(LogT)$ au lieu du $O(1)$ si on avait utilisé un graphe résiduel étendu.

3.2.4.4 Méthode de résolution

Dans cette partie, nous présentons la méthode de résolution et comment nous calculons chaque critère.

Critère de durée [Rosen *et al.*, 1991] ont introduit le principe "Quickest S-P Path" qui correspond au chemin unique le plus rapide pour évacuer un certain nombre de personnes. Contrairement au plus court chemin entre S et P, le "Quickest S-P Path" prend en compte la capacité des arcs durant la recherche. Ils ont aussi prouvé que le plus court chemin entre S et P est équivalent au "Quickest S-P Path" si la capacité résiduelle du chemin est supérieure ou égale au nombre de personnes que l'on veut faire passer dessus. La manière d'obtenir le "Earliest Arrival Flow" à partir d'un ensemble de "Quickest Path" est décrit dans [Ford et Fulkerson, 1962] et [Hamacher et Tjandra, 2002]. [Schmidt et Skutella, 2014] ont montré que le "Earliest Arrival Flow" existe si chaque centre de secours $P_l,\ l \in \{1, \ldots, L\}$ est atteignable depuis l'ensemble des points de rassemblement S_k, $k \in \{1, \ldots, K\}$. Grâce aux lemmes 1 et 2, nous avons prouvé que résoudre un problème de "Earliest Arrival Flow" avec des données variant dans le temps est équivalent à résoudre ce même problème avec un Réseau Virtuel Dynamique. [Tjandra, 2003] montre que la solution optimale du problème "Earliest Arrival Flow" avec des données qui varient dans le temps est aussi optimale pour le problème du "Quickest Flow" avec des données qui varient dans le temps si le dernier évacué atteint un centre de secours avant la fin de l'horizon de planification T. Ainsi, cette approche permet d'avoir un algorithme exact pour le problème de transbordement de personnes et de "Earliest Arrival Flow" avec des données variant dans le temps et avec interdiction de stationner dans le réseau avec pour but d'optimiser la durée d'évacuation.

Critère de sécurité Comme décrit dans la thèse de [Wayne, 1999], nous considérons la sécurité de chaque groupe d'évacués comme étant le produit des sécurités des arcs traversés pour aller du super sommet source S au super sommet puits P (i.e. le nombre fractionnaire d'évacués indemnes). Ainsi, tout en cherchant des chaînes augmentantes arrivant à une certaine date au super sommet puits P, si nous disposons de deux chemins avec des durées partielles identiques, nous choisissons en priorité la chaîne ayant la meilleure note de sécurité. Dans ce cas de figure, l'algorithme multicritère proposé permet d'avoir un optimum de Pareto faible avec la meilleure durée d'évacuation possible. Pour maximiser la sécurité du plan d'évacuation (voir (3.16)), nous maximisons le nombre fractionnaire de

personnes indemnes qui peuvent atteindre les centres de secours. Cette approche pénalise les chemins ayant des arcs dont le niveau de sécurité est proche de 0.0 et favorise ceux dont le niveau de sécurité est proche de 1.0.

3.2.5 Algorithmes

Dans cette partie nous présentons l'algorithme Lex((Q|S) Flow). nous rappelons que ce dernier est une méthode exacte permettant de minimiser la durée pour le problème de transbordement de personnes et le problème du "Earliest Arrival Flow" avec données variant dans le temps et avec interdiction de stationner dans le réseau. En prenant en compte la sécurité, tout en obtenant la meilleure durée possible, cette approche mène a un optimum de Pareto faible pour la sécurité alors que la durée reste optimale.

3.2.5.1 Algorithme : Lex((Q|S) Flow)

Algorithme : Lex((Q|S) Flow)

Donnée(s) :
1) G_{vsn} : Le Réseau Virtuel Dynamique.
2) B : Le nombre total de personnes à évacuer.
3) T : L'horizon de planification.

Début :
4) $t_d \leftarrow 0$ % Date de départ du super sommet source S %
5) $t_a \leftarrow 0$ % Date d'arrivée au super sommet puits P %
6) $F \leftarrow \emptyset$ % Fonction de flot %
7) $FlowMemory \leftarrow \emptyset$ % Mémoire de décision %

8) $F \leftarrow Lex(Shortest_{Forward}|Safest\ S\text{-}P\ path, G_{vsn}, FlowMemory, t_d, \&t_a)$
9) **Tant que** ($B > 0$ et $t_a \leq T$) **Faire**
10) **Si** ($F.valeur = 0$) **Alors**
11) $t_a \leftarrow t_a + 1$
12) **Sinon**
13) $Update(\&FlowMemory, F, t_d)$
14) $B \leftarrow B - \min(B, F.valeur)$
15) **Fin Si**
16) $F \leftarrow Lex(Shortest_{Backward}|Safest\ S\text{-}P\ path, G_{vsn}, FlowMemory, \&t_d, t_a)$
17) **Fin Tant que**

Fin :
Sortie(s) :
18) $FlowMemory$: Le Lex((Q|S) Flow).

Cet algorithme est le cœur de la méthode de résolution. Comme données d'entrée (ligne 1 à 3) nous avons un réseau dynamique virtuel G_{vsn} présenté dans la section 3.2.4, le nombre total de personnes à évacuer B et l'horizon de planification de l'évacuation T. Si les données ne varient pas dans le temps, T peut être obtenu grâce à la propriété de Klinz (voir [Hoppe et Tardos, 2000]), sinon il est possible de le fixer à une valeur arbitrairement assez grande pour permettre l'évacuation de toute la population. Dans la première étape de l'algorithme (ligne 4 à 7), nous initialisons la date de départ t_d, la date d'arrivée t_a, la fonction de flot relative à la chaîne augmentante F et le plan d'évacuation représenté par la mémoire de décision $FlowMemory$. Les dates de départ et d'arrivée sont fixées à 0 alors que la chaîne augmentante est vide au début du processus. Après cette étape, la première chose à faire est de trouver le plus court chemin ayant une capacité résiduelle non nulle grâce à l'algorithme 3.2.5.2 (voir la section 3.2.5.2). Ce chemin permet d'aller du sommet S au sommet P à partir de la date $t_d = 0$ et passant par un réseau non encore utilisé

(voir ligne 8). Si la capacité résiduelle est non nulle alors la date d'arrivée t_a est mise à jour. De la ligne 9 à 17, nous vérifions dans un premier temps que la capacité résiduelle de la chaîne augmentante $F.valeur$ permet d'envoyer des évacués sur ce chemin. Si aucun chemin n'existe, alors nous incrémentons la date d'arrivée souhaitée t_a d'une unité. Dans le cas contraire, il y a une chaîne augmentante qui peut être utilisée pour l'évacuation des personnes.

Pour faire cela, nous mettons à jour la mémoire de décision $FlowMemory$ à la ligne 13 avec l'algorithme 3.2.5.4 (voir la section 3.2.5.4). Après cette étape, nous diminuons le nombre total de personnes restant à évacuer à la ligne 14. A l'étape suivante, nous essayons de trouver une autre chaîne augmentante qui permet d'atteindre le super sommet puits P à la date t_a grâce à l'algorithme 3.2.5.3 (voir section 3.2.5.4). Si nous trouvons une chaîne augmentante de capacité résiduelle non nulle, alors la date de départ t_d est mise à jour dans le but d'atteindre le sommet puits à la date t_a (voir ligne 16).

Une fois de retour à la ligne 9, nous continuons le processus jusqu'à ce que l'ensemble de la population soit évacué ou que nous ayons atteint l'horizon de planification T. Comme données de sortie, l'algorithme produit un plan d'évacuation qui est contenu dans la variable $FlowMemory$. Cette dernière contient l'ensemble des arcs qui sont utilisés durant l'évacuation ainsi que les moments de leurs utilisations. A chaque arc, nous associons trois informations : Le moment d'utilisation, le niveau de sécurité ainsi que que le niveau d'utilisation. La somme pondérée du flot par les dates d'arrivées ainsi que la sécurité du plan d'évacuation peuvent être calculées facilement. La première peut être obtenue en utilisant les arcs $(i, P_l) \in E$ se trouvant dans la variable $FlowMemory$. Pour le niveau de sécurité du plan d'évacuation, il suffit d'ajouter l'ensemble des arcs composant la variable $FlowMemory$ dans un tas de Fibonacci en utilisant comme clé le moment de leur utilisation. Ainsi, jusqu'à ce que le tas soit vide, on récupère l'élément avec la plus petite clé et on propage le nombre de personnes indemnes. A la fin de ce processus, on comptabilise le nombre fractionnaire d'individus indemnes ayant pu atteindre le super sommet source P.

3.2.5.2 Algorithme : Lex(Shortest$_{\text{Forward}}$ | Safest S−P Path)

Algorithme : Lex(Shortest$_{\text{Forward}}$ | Safest S−P Path)
Donnée(s) :
1) G_{vsn} : Le Réseau Virtuel Dynamique.
2) $FlowMemory$: Mémoire de décision.
3) t_d : Date de départ du super sommet source S.
4) t_a : Date d'arrivée au super sommet puits P.
Début :
5) $FiboHeap$: Tas de fibonacci
6) Π : Vecteur de prédécesseurs
7) **Pour chaque** $Vertex \in G_{vsn} \backslash \{S\}$ **Faire**
8) $Vertex.date \leftarrow +\infty$
9) $Vertex.safety \leftarrow 0$
10) $FiboHeap.addVertex(Vertex, Vertex.date + (1 - Vertex.safety))$
11) $\Pi[Vertex] \leftarrow Vertex$
12) **Fin Pour chaque**

13) $S.date \leftarrow t_d$
14) $S.safety \leftarrow 1$
15) $FiboHeap.addVertex(S, S.date + (1 - Vertex.safety))$

16) **Tant que** $isNotEmpty(FiboHeap)$ **Faire**
17) $CurrentVertex \leftarrow FiboHeap.minimum()$
18) $FiboHeap.deleteMinimum()$
19) **Si** $CurrentVertex.date = +\infty$ **Ou** $CurrentVertex = P$ **Alors**
20) *Stop*
21) **Fin Si**
22) **Pour chaque** $NextVertex \in \Gamma^+_{CurrentVertex}$ **Faire**
23) **Si** $isInFiboHeap(NextVertex)$ **Alors**
24) **Si** $isEdgeAvailable(CurrentVertex, NextVertex, FlowMemory)$ **Alors**
25) $NewDuration \leftarrow CurrentVertex.date + D^{CurrentVertex.date}_{(CurrentVertex,NextVertex)}$
25) $NewSafety \leftarrow CurrentVertex.safety \times Q^{CurrentVertex.date}_{(CurrentVertex,NextVertex)}$
27) **Si** $NextVertex.date > NewDuration$ **Alors**
28) $\Pi[NextVertex] \leftarrow CurrentVertex$
29) $NextVertex.date \leftarrow NewDuration$
30) $NextVertex.safety \leftarrow NewSafety$
31) $decreaseFiboKey(NextVertex, NextVertex.date+$
$(1 - NextVertex.safety))$
32) **Sinon Si** $NextVertex.date = NewDuration$
Et $NextVertex.safety < NewSafety$ **Alors**
33) $\Pi[NextVertex] \leftarrow CurrentVertex$
34) $NextVertex.safety \leftarrow NewSafety$
35) $decreaseFiboKey(NextVertex, NextVertex.date+$
$(1 - NextVertex.safety))$
36) **Fin Si**
37) **Fin Si**
38) **Fin Si**
39) **Fin Pour chaque**
40) **Fin Tant que**

41) $F \leftarrow flowBuilder(FlowMemory, \Pi)$
42) $t_a \leftarrow P.date$
Fin :
Sortie(s) :
43) F : *Integral flow function.*
44) t_a : *Arrival date to super sink P.*

Cet algorithme permet de trouver une chaîne augmentante allant du super sommet source S au super sommet puits P. En particulier, cette chaîne augmentante doit permettre de débuter l'évacuation à la date t_d et avoir une capacité non nulle. Comme paramètre d'entrée (ligne 1 à 4) nous avons le Réseau Virtuel Dynamique G_{vsn}, la mémoire de décision qui est vide lors de l'utilisation de cet algorithme, la date t_d de départ depuis le super sommet source S et la date d'arrivée souhaitée $t_a = 0$ au super sommet source P. La première étape de l'algorithme est l'initialisation d'un tas de Fibonacci avec pour objectif de stocker les étiquettes associées à chaque sommet. Nous initialisons aussi la variable Π qui permet de reconstruire la chaîne augmentante une fois le sommet cible trouvé (voir ligne 5 à 6). Après cette étape, nous ajoutons tous les sommets $i \in G_{vsn} \backslash \{S\}$ contenus dans le tas de Fibonacci en utilisant comme clé la somme de la durée et le complément de

la sécurité (voir 7 à 12). De la ligne 13 à 15, nous mettons comme sommet de départ le super sommet source S et nous l'ajoutons au tas de Fibonacci. De la ligne 16 à 40, nous récupérons le sommet avec la plus petite étiquette se trouvant dans le tas de Fibonacci et nous mettons à jour ses successeurs. Si le sommet courant correspond au super sommet P, alors nous avons trouvé une chaîne augmentante non nulle permettant d'aller du sommet S au sommet P. Dans le cas contraire si l'étiquette du sommet courant a une durée infinie, alors il n'existe pas de chaîne augmentante. Dans les deux cas, nous arrêtons l'algorithme. De la ligne 22 à 39 nous mettons à jour les sommets successeurs du sommet courant. Pour faire cela, nous vérifions à la ligne 24 que l'arc défini par le sommet courant et son successeur est disponible au moment auquel nous souhaitons l'utiliser. Cette information peut être obtenue en recherchant l'arc correspondant dans la mémoire de décision. La partie restante de l'algorithme est similaire à l'algorithme d'étiquetage de Dijsktra. Les lignes 41 et 42 permettent de calculer le modèle de flot ainsi que la capacité résiduelle associée à la chaîne augmentante allant du super sommet source S au super sommet puits P si elle existe. Aussi, nous mettons à jour la date d'arrivée t_a avec la date d'arrivée effective au sommet P s'il est atteint. A la fin de l'algorithme, ce dernier donne en sortie le modèle de flot ainsi que la date d'arrivée potentielle au super sommet source P.

3.2.5.3 Algorithme : Lex(Shortest$_{\text{Backward}}$ | Safest S−P Path)

Algorithme : Lex(Shortest$_{\text{Backward}}$ | Safest S−P Path)

Donnée(s) :
1) G_{vsn} : Le Réseau Virtuel Dynamique.
2) $FlowMemory$: Mémoire de décision.
3) t_d : Date de départ du super sommet source S.
4) t_a : Date d'arrivée au super sommet puits P.

Début :
5) $FiboHeap$: Tas de Fibonacci
6) Π : Vecteur de prédécesseurs
7) $CheckResidualEdges$: Booléen

8) **Pour chaque** $Vertex \in G_{vsn} \backslash \{P\}$ **Faire**
9) $Vertex.date \leftarrow +\infty$
10) $Vertex.safety \leftarrow 0$
11) $FiboHeap.addVertex(Vertex, Vertex.date + (1 - Vertex.safety))$
12) $\Pi[Vertex] \leftarrow Vertex$
13) **Fin Pour chaque**

14) $P.date \leftarrow -t_a$
15) $P.safety \leftarrow 1$
16) $FiboHeap.addVertex(P, P.date)$

17) **Tant que** $isNotEmpty(FiboHeap)$ **Faire**
18) $CurrentVertex \leftarrow FiboHeap.minimum()$
19) $FiboHeap.deleteMinimum()$
20) **Si** $CurrentVertex.date > 0$ **Ou** $CurrentVertex = S$ **Alors**
21) *Stop*

22) **Fin Si**
23) $CheckResidualEdges \leftarrow true$

24) **Pour chaque** $PrevVertex \in \Gamma^{-}_{CurrentVertex}$ **Faire**
25) **Si** $isInFiboHeap(PrevVertex)$ **Alors**
26) **Si** $isEdgeAvailable(PrevVertex, CurrentVertex, FlowMemory)$ **Alors**
27) $NewDuration \leftarrow CurrentVertex.date + D_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
28) $NewSafety \leftarrow CurrentVertex.safety \times Q_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
29) **Si** $PrevVertex.date > NewDuration$ **Alors**
30) $\Pi[CurrentVertex] \leftarrow PrevVertex$
31) $PrevVertex.date \leftarrow NewDuration$
32) $PrevVertex.safety \leftarrow NewSafety$
33) $decreaseFiboKey(PrevVertex, PrevVertex.date + (1 - PrevVertex.safety))$
34) $CheckResidualEdges \leftarrow false$
35) **Sinon Si** $PrevVertex.date = NewDuration$
36) **Et** $PrevVertex.safety < NewSafety$ **Alors**
37) $\Pi[CurrentVertex] \leftarrow PrevVertex$
38) $PrevVertex.safety \leftarrow NewSafety$
39) $decreaseFiboKey(PrevVertex, PrevVertex.date + (1 - PrevVertex.safety))$
40) $CheckResidualEdges \leftarrow false$
41) **Fin Si**
42) **Fin Si**
43) **Fin Si**
44) **Fin Pour chaque**

45) **Si** $CheckResidualEdges = true$ **Alors**
46) **Pour chaque** $PrevVertex \in \Gamma^{-}_{CurrentVertex}$ **Faire**
47) **Si** $isInFiboHeap(PrevVertex)$ **Alors**
48) **Si** $isResidualEdgeAvailable(PrevVertex, CurrentVertex, FlowMemory)$ **Alors**
49) $NewDuration \leftarrow CurrentVertex.date - D_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
50) $NewSafety \leftarrow CurrentVertex.safety \times Q_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
50) **Si** $PrevVertex.date > NewDuration$ **Alors**
52) $\Pi[CurrentVertex] \leftarrow PrevVertex$
53) $PrevVertex.date \leftarrow NewDuration$
54) $PrevVertex.safety \leftarrow NewSafety$
55) $decreaseFiboKey(PrevVertex, PrevVertex.date + (1 - PrevVertex.safety))$
56) **Sinon Si** $PrevVertex.date = NewDuration$
57) **Et** $PrevVertex.safety < NewSafety$ **Alors**
58) $\Pi[CurrentVertex] \leftarrow PrevVertex$
59) $PrevVertex.safety \leftarrow NewSafety$
60) $decreaseFiboKey(PrevVertex, PrevVertex.date + (1 - PrevVertex.safety))$
61) **Fin Si**
62) **Fin Si**
63) **Fin Si**
64) **Fin Pour chaque**
65) **Fin Si**

66) **Fin Tant que**

67) $F \leftarrow flowBuilder(FlowMemory, \Pi)$
68) $t_d \leftarrow abs(VertexS.date)$
)**Fin :**
Sortie(s) :
69) F : *Integral flow function.*
70) t_d : *Departure date to super source S.*

Cet algorithme est le complément de l'algorithme 3.2.5.2. Contrairement à ce dernier, il permet de fixer une date t_a impérative au super sommet source P et de trouver une chaîne augmentante de capacité non nulle arrivant à cette date au super sommet puits P en partant du super sommet source S. Tout comme l'algorithme 3.2.5.2 il prend comme paramètre (ligne 1 à 4) le réseau dynamique virtuel G_{vsn}, la mémoire de décision, la date de départ au plus tard $t_d = +\infty$ depuis le super sommet source S et la date d'arrivée impérative t_a au super sommet puits P.
L'une des premières étapes (voir ligne 5 à 7) de l'algorithme consiste à initialiser le tas de Fibonacci dans le but de stocker l'ensemble des sommets qui composent le réseau G_{vsn}. Nous initialisons aussi la variable Π qui permet de construire les chaînes augmentantes permettant d'atteindre le sommet P à une date précise. Enfin, nous avons une variable qui permet de savoir si la chaîne augmentante que nous cherchons nécessite de revenir sur des décisions précédentes. Cette étape d'initialisation terminée, nous ajoutons tous les sommets $i \in G_{vsn} \backslash \{P\}$ dans le tas de Fibonacci en utilisant comme clé la somme de la durée et du complément de la sécurité (voir ligne 8 à 13).
De la ligne 14 à 16, nous prenons comme point de départ le sommet puits P et nous l'ajoutons au tas de Fibonacci avec une valeur de clé faisant de lui l'élément ayant la valeur minimale. De la ligne 16 à 44 nous avons une partie similaire à celle présentée dans l'algorithme 3.2.5.2 de la ligne 16 à 40. Deux différences peuvent toutefois être soulignées. La première est la manière dont nous mettons à jour les prédécesseurs des sommets en cours de traitement. Cependant les arcs peuvent être saturés et nous devons considérer le fait de rediriger les évacués ayant déjà été routés (voir ligne 45 à 65). La seconde différence est la manière de calculer la durée associée aux étiquettes de chaque sommet (voir ligne 49). Ce dernier permet par exemple la redirection d'une partie des évacués ayant emprunté l'arc (PrevVertex,CurrentVertex) à la date $t = NewDuration$.
A la fin de l'algorithme, la ligne 67 permet de calculer le modèle de flot ainsi que la capacité résiduelle associée à la chaîne augmentante partant du super sommet source S vers le super sommet puits P. Si un flot précédemment poussé sur un arc e est révoqué, alors la valeur de flot est égale à la capacité résiduelle de la chaîne augmentante multipliée par -1. Cette approche nous permet aussi de connaitre la date t_d à laquelle nous devons partir du super sommet source S pour espérer arriver au super sommet puits P à la date t_a. Si la date de départ t_d est inférieure ou égale à zero alors il existe une chaîne augmentante valide avec une date de départ effective égale à la valeur absolue de l'étiquette de durée au sommet S, $t_d = abs(S.date)$. Dans le cas contraire, $t_d = +\infty$ (voir ligne 68) et il n'existe plus ne chaîne augmentante permettant d'atteindre le super sommet puits P à la date t_a qui est

fixée.
Comme résultat de cet algorithme, nous retournons le modèle de flot ainsi que la date de départ t_d du super sommet source S.

3.2.5.4 Algorithme : Update FlowMemory

Algorithme : Update FlowMemory

Donnée(s) :
1) $FlowMemory$: Mémoire de décision
2) F : Fonction de Flot.
3) t_d : Date de départ du super sommet source S.

Début :
4) $t \leftarrow t_d$
5) **Pour chaque** edge e $\in F$ **Faire**
6) **Si** $isExpandable(e)$ **Alors**
7) $Index \leftarrow DichoSearch(FlowMemory, e, t)$
8) **Si** $exist(FlowMemory(e, Index), t)$ **Alors**
9) $increase(FlowMemory(e, Index), e.valeur)$
10) **Sinon**
11) $insert(FlowMemory, e, Index, t)$
12) **Fin Si**
13) $t \leftarrow t + D_e^t$
14) **Sinon**
15) **Si** $exist(FlowMemory(e, Index), t)$ **Alors**
16) $increase(FlowMemory(e, 1), e.valeur)$
17) **Sinon**
18) $insert(increase(FlowMemory(e, 1), e.valeur)$
19) **Fin Si**
20) $t \leftarrow t + D_e$
21) **Fin Si**
22) **Fin Pour chaque**

Fin :
Sortie(s) :
23) $FlowMemory$: Le Lex((Q|S) Flow).

L'algorithme de mise à jour de la mémoire décision permet de construire de manière itérative le graphe résiduel du réseau sans pour autant le créer dès le début. Comme paramètres d'entrée, nous avons la mémoire de décision à mettre à jour, le modèle de flot associé à une chaîne augmentante et la date de départ du super sommet source S (voir ligne 1 à 3).
La première étape de l'algorithme à la ligne 4 consiste à initialiser la variable de temps associée au modèle de flot. Par la suite, pour chaque arc e du modèle de flot, nous mettons à jour son niveau d'utilisation s'il existe déjà dans la mémoire de décision (voir ligne 9, respectivement ligne 16) sinon nous l'ajoutons à la bonne place (voir ligne 11, respectivement ligne 18).
Ensuite la variable de temps associée au modèle de flot est mise à jour suivant le type d'arc e considéré (arc étendu dans le temps ou pas, voir Figure 3.6). Si un arc ne peut être étendu dans le temps, alors sa mise à jour ou sa création demande une complexité de $O(1)$. Cependant, quand un arc peut être étendu dans le temps, il est nécessaire de faire une

recherche dichotomique de complexité $O(LogT)$ quelle que soit l'opération souhaitée. Des exemples de mise à jour de la mémoire de décision peuvent être trouvés dans les tableaux ([TableA.3-TableA.11]).

3.2.5.5 Complexité

Comme la mémoire de décision est vide au début de l'algorithme 3.2.5.1, il n'est pas nécessaire de déterminer la capacité résiduelle de chaque arc avant son utilisation par l'algorithme 3.2.5.2. Le calcul du premier modèle de flot nécessite une complexité de $O(M+NLogN)$. L'algorithme 3.2.5.3 nécessite de connaitre la capacité résiduelle de chaque arc avant de pouvoir les utiliser pour évacuer les personnes. Comme nous n'avons pas créé le réseau résiduel qui nous aurait permis d'accéder à l'ensemble de ses informations en $O(1)$, chaque opération sur la mémoire de décision requière une recherche dichotomique de complexité $O(LogT)$. Ainsi dans le pire des cas, la complexité temporelle de l'algorithme 3.2.5.3 est égale à $O(MLogT + NLogN)$.
Comme le Réseau Virtuel Dynamique est composé de N sommet et qu'il n'est pas possible pour les évacués de se retrouver 2 fois sur le même sommet, dans le pire des cas, un chemin utilisé par un évacué contiendra $N-1$ sommets. Tout comme pour l'algorithme 3.2.5.2, l'algorithme 3.2.5.4 met à jour les capacités résiduelles des arcs permettant ainsi avec un cout temporel de $O(LogT)$. Ainsi dans le pire cas, la mise à jour des arcs associés à une chaîne augmentante sur laquelle on veut faire passer des évacuées nécessitera au plus $O((N-1)LogT) = O(NLogT)$ opérations.
Globalement, le pire cas de figure pour l'algorithme consiste à ne trouver que des chaînes augmentantes avec des capacités résiduelles ne permettant de faire passer qu'un seul individu. Ayant B personnes à évacuer, alors la complexité globale de l'algorithme 3.2.5.1 est de $O(B(MlogT + NLogN))$.

Propriété 1. *Les problèmes d'évacuation du "Earliest Arrival Flow" et du "Quickest Flows" avec des données variant dans le temps et sans possibilité de patienter peuvent être résolus avec une complexité temporelle* $O(B(MlogT + NLogN))$.

3.2.6 Illustration

L'exemple suivant permet d'illustrer les algorithmes présentés ci-dessus. Pour pouvoir comparer notre approche, deux solutions sont proposées. La première solution utilise le réseau étendu dans le temps classique alors que la seconde utilise notre Réseau Virtuel Dynamique. Soit $G = (V, E, C, D, Q, \Delta, T)$ le réseau dynamique décrit par la figure 3.7. Dans cet exemple, $Bs_1 = 6$ individus qui doivent être évacués depuis le point de rassemblement S_1 à partir de la date $Rs_1 = 1$ et $Bs_2 = 12$ individus doivent être évacués depuis le point de rassemblement S_2 à partir de la date $Rs_2 = 0$. Les centres de secours P_1 et P_2 ont des capacités d'accueil de $A_1 = 14$ et $A_2 = 5$. Le réseau doit être vide au plus tard à la date $T = 11$ (voir la propriété de Klinz [Hoppe et Tardos, 2000]).

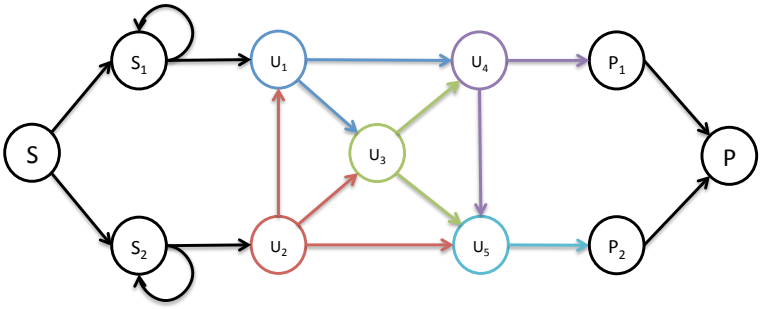


FIGURE 3.7 – Exemple de graphe dynamique

3.2.6.1 Réseau étendu dans le temps

Les données sont fournies dans le tableau Table 3.1. Quand les données ne sont pas définies dans le tableau alors par défaut les valeurs seront égales à : $D^t_{(i,j)} = +\infty$, $\Delta^\tau_{(i,j)} = +\infty$, $C^t_{(i,j)} = 0$ et $Q^t_{(i,j)} = 0.0$.
La figure A.1 permet description complète des données de ce problème à travers un réseau étendu dans le temps. Ce dernier était borné avec un horizon de planification $T = 7$ mais il peut facilement être étendu à $T = 11$.

TABLE 3.1 – Réseau étendu dans le temps

(i,j)	$D_{(i,j)}$	$\Delta_{(i,j)}$	$C_{(i,j)}$	$Q_{(i,j)}$
(U_1, U_3)	1, t≥0	1, $\tau \geq 1$	3, t≥0	1,0 , t≥0
(U_1, U_4)	2, t≥0	2, $\tau \geq 2$	3, t≥0	1,0 , t≥0
(U_2, U_1)	3, t≥0	3, $\tau \geq 3$	4, t≥0	0,7 , t≥0
(U_2, U_3)	1, t≥0	1, $\tau \geq 1$	1, t≥0	1,0 , t≥0
(U_2, U_5)	1, t≥0	1, $\tau \geq 1$	3, t≥0	0,8 , t≥0
(U_3, U_4)	1, t≥0	1, $\tau \geq 1$	2, t≥0	1,0 , t≥0
(U_3, U_5)	2, t≥0	2, $\tau \geq 2$	2, t≥0	1,0 , t≥0
(U_4, U_5)	2, t≥0	2, $\tau \geq 2$	2, t≥0	0,5 , t≥0
(U_4, P_1)	2, t≥0	2, $\tau \geq 2$	4, t≥0	1,0 , t≥0
(U_5, P_2)	1, t≥0	1, $\tau \geq 1$	1, t≥0	1,0 , t≥0
(S_1, S_1)	1, t≥0	1, $\tau \geq 1$	Bs_1,t≥0	1,0 , t≥0
(S_1, U_1)	1, t≥0	1, $\tau \geq 1$	3, t≥0	1,0 , t≥0
(S_2, S_2)	1, t≥0	1, $\tau \geq 1$	Bs_2 , t≥0	1,0 , t≥0
(S_2, U_2)	1, t≥0	1, $\tau \geq 1$	2, t≥0	1,0 , t≥0
(S, S_1)	1, t=0	1, τ=1	Bs_1, t=0	1,0 , t=0
(S, S_2)	0, t=0	0, τ=0	Bs_2, t=0	1,0 , t=0
(P_1, P)	0	0	A_1	1,0
(P_2, P)	0	0	A_2	1,0

3.2.6.2 Résoudre le Lex((Q|S) Flow) avec un réseau étendu dans le temps

Comme décrit dans la littérature [Tjandra, 2003], la solution à un problème de "Earliest Arrival Flow" est équivalent à celui d'un problème de "Quickest Flow" si l'ensemble des évacués sortent du réseau au plus tard à la date T. [Hamacher et Tjandra, 2003] ont démontré que le problème du "Earliest Arrival Flow" peut être résolu en déterminant un ensemble de plus courts chemins augmentants. Pour résoudre le Lex((Q|S) Flow) , lorsque deux chemins ont la même longueur, alors le chemin minimisant le danger sera choisi en priorité par les évacués. Une solution pour l'exemple présenté peut être trouvé dans le tableau Table.A.1. Le nombre fractionnaire d'individus indemnes arrivant dans un centre de secours est de 17. En considérant la sécurité définie par la fonction objectif (3.16), cette instance a une valeur de sécurité de $\frac{1}{18} \times (18 - 17) = \frac{1}{18} = 0,0555$ (Voir tableau A.2). En appliquant la politique d'évacuation décrite par l'équation (3.15), la valeur de la fonction objectif est de $3 + 4 + 10 + 30 + 35 + 8 + 9 + 10 + 11 = 120$ (Voir la somme pondérée des dates d'arrivée aux sommets (P_1, P) et (P_2, P) A.1). La durée moyenne d'évacuation est elle égale à $\frac{120}{18} = 6,6666$ unités de temps. La fonction objectif unifiée (3.17) a quant à elle une valeur égale à : $120 + 0,0555 = 120,0555$.

3.2.6.3 Résoudre le Lex((Q|S) Flow) avec un Réseau Virtuel Dynamique

Dans cette partie nous présentons quatre itérations de l'algorithme permettant de résoudre une instance du problème Lex((Q|S) Flow) tout en utilisant un Réseau Virtuel Dynamique. Le détail concernant les itérations restantes peut être trouvé dans les tableaux [TableA.3 - TableA.11]. La première étape est de déterminer la date d'arrivée au plus tôt des premiers groupes d'évacués.

Comme la mémoire de décision est vide à cette étape, obtenir cette date peut être effectué en déterminant le plus court chemin de capacité non nulle allant du super sommet source S au super sommet puits P grâce à l'algorithme 3.2.5.2. Ce chemin est $(S, S_2, U_2, U_4, P_2, P)$ avec comme dates de départ et d'arrivée $t_d = 0, t_a = 3$, une sécurité de 0.8 ainsi qu'une capacité résiduelle de 1 personne. Après avoir envoyé cet évacué sur ce chemin, nous mettons à jour la mémoire de décision (TableA.3) et nous décrémentons le nombre total de personnes restant à évacuer de $B = 18$ à $B = 17$. A cette étape, nous cherchons s'il existe une autre chaîne augmentante permettant aux évacués d'arriver au moment $t_a = 3$. Pour y arriver, nous utilisons l'algorithme 3.2.5.3 permettant de trouver ce type de chaînes augmentantes. Comme il n'est pas possible d'avoir plus de personnes arrivant à la date $t_a = 3$, nous incrémentons la date d'arrivée souhaitée à t_a à une unité ($t_a = 3 + 1 = 4$). A la nouvelle date d'arrivée souhaitée t_a, l'algorithme 3.2.5.3 permet encore d'obtenir le chemin (S, S_2, U_5, P_2, P) avec une capacité résiduelle d'un individu et avec une sécurité de 0.8. L'évacué est envoyé sur ce chemin et nous mettons à jour la mémoire de décision (TableA.4). Nous réduisons le nombre total d'évacués restant d'une unité ce qui fait passer le nombre d'évacués de $B = 17$ à $B = 16$. La date d'arrivée souhaitée t_a est incrémentée d'une unité ($t_a = 4 + 1 = 5$), comme il n'existe plus de chaîne augmentante arrivant à la date 4. Une fois de plus nous utilisons l'algorithme 3.2.5.3 ce qui nous permet de trouver le chemin $(S, S_2, U_2, U_5, P_2, P)$ avec une capacité résiduelle de 1 personne et une sécurité de 0.8. Nous mettons à jour la mémoire de décision et nous essayons de trouver

un nouveau chemin arrivant à la date $t_a = 5$. Comme seconde chaîne augmentante, nous trouvons $(S, S_2, U_2, U_3, U_4, P_1, P)$ avec comme capacité résiduelle de 1 et une sécurité de $1,0$. Nous mettons à jour la mémoire de décision (tableau A.5) et il n'existe plus de chaîne augmentante permettant d'arriver à la date $t_a = 5$. Alors nous incrémentons d'une unité la date d'arrivée souhaitée et nous continuons ce processus jusqu'à ce que tous les points de rassemblement (i.e S_1 et S_2) soient évacués.

Si l'on considère la fonction objectif unifiée (3.17), sa valeur augmente suivant le nombre d'évacués arrivant aux centres de secours. La partie correspondant à la somme pondérée des dates d'arrivée est associée à la durée (3.15). Elle est respectivement égale à $1 \times 3 = 3$ si $t_a = 3$, $3 + 1 \times 4 = 7$ si $t_a = 4$ et $7 + 2 \times 5 = 17$ si $t_a = 5$. La sécurité définie par la fonction objectif (3.16) est respectivement égale à $\frac{1-0,80}{1} = 0,20$ si $t_a = 3$, $\frac{2-(0,8+0,8)}{2} = 0,20$ si $t_a = 4$ et $\frac{4-(0,8+0,8+0,8+1,0)}{4} = 0,15$ si $t_a = 5$. Une fois ces les valeurs de ces deux fonctions objectif sont unifiés, nous avons la valeur associée au problème lexicographique. Ainsi, la fonction objectif (3.17) a une valeur égale à $3 + 0,2 = 3,02$ si $t_a = 3$, $7 + 0,2 = 7,2$ si $t_a = 4$ et $17 + 0,15 = 17,15$ si $t_a = 5$.

3.2.7 Résultats expérimentaux

Environnent

Nous avons implémenté l'algorithme de résolution du problème Lex((Q|S) Flow) en utilisant le langage C++ avec comme compilateur gcc v. 4.8.2 avec l'option -O3. Toutes les expérimentations ont été effectuées sur une machine virtuelle "VirtualBox" utilisant un des processeurs d'un Intel Core i7-4910QM cadencé à 2.90Ghz. Il avait 8Mo de mémoire cache et 4Go de mémoire RAM avec une fréquence de 1600Mhz. Le système d'exploitation utilisé est Ubuntu 14.04 LTS x64.

Cas d'étude

Afin d'évaluer la méthode de résolution, nous prenons comme cas d'étude la ville de Nice qui est notre cas d'étude dans le cadre du projet DSS_Evac_Logistic. Ce dernier traite de l'évacuation d'une ville de grande taille située dans le sud de la France (Figure 3.8).

En 1618, plusieurs tremblements de terre se sont passés près de Vésubie (France) qui est au nord de Nice. Ces derniers ont causé plusieurs glissements de terrain. Le 23 février 1887, un tremblement de sous-marin avec une magnitude allant de $6,5$ à $6,8$ s'est passé près de la ville de Ligure en Italie. Durant cet événement 635 personnes ont perdu la vie et 555 ont été blessées. En 1979, un grand glissement de terrain s'est déroulé le 16 octobre près de l'aéroport de Nice. A cause des effets du glissement de terrain et du déplacement d'eau, un petit tsunami s'est dirigé quelques secondes plus tard sur les plages de Nice. Le 20 Mai 2012, un autre tremblement de terre s'est déroulé près du nord de l'Italie qui est proche de la ville de Nice avec pas moins de 417 répliques. En janvier 2014, à cause du mauvais temps, plusieurs cas d'éboulement, de glissements de terrain et d'inondations se sont passés près de la ville de Nice. Plusieurs personnes ont ainsi dû être évacuées mais certaines perdirent la vie en essayant de fuir par leur propres moyens. Les 3 et 4 Octobre 2015, des pluies diluviennes

touchant les alpes-maritimes ont mené à des inondations causant une vingtaine de morts. Plus récemment, le 6 Novembre 2015, a eu lieu un tremblement de terre de magnitude 4, 4 et dont l'épicentre se trouvait au parc national du Mercantour au nord de la ville de Nice. Même s'il n'y a pas eu de dégâts notables, les secousses ont toutefois été ressentis jusqu'à notre ville d'étude Nice.

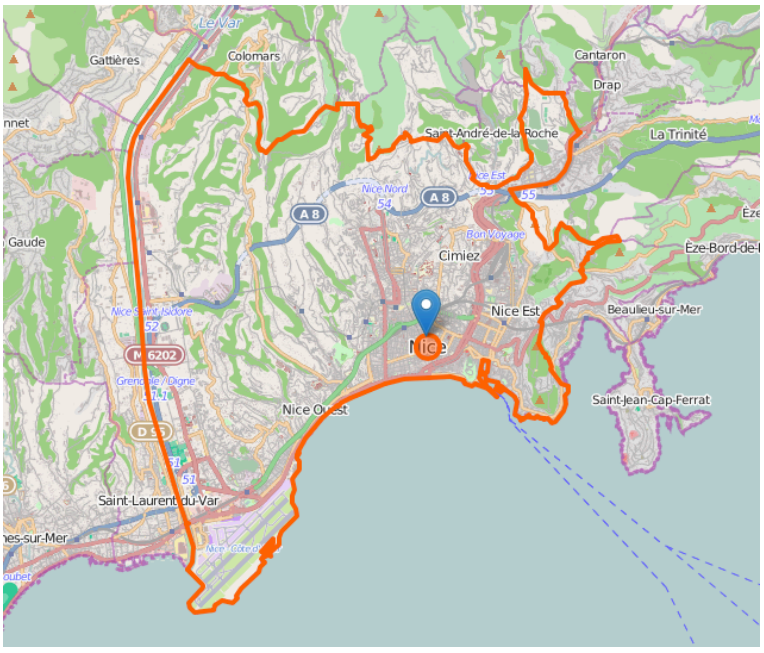


FIGURE 3.8 – La ville de Nice (France)

Afin de pouvoir tester nos algorithmes, nous considérons comme réseau routier celui de la ville de Nice. Ce dernier est défini par un graphe de 46990 sommets, 91412 arcs et 2455, 7km de routes (voies routières et piétonnes extraites des données d'OpenStreetMap [OpenStreetMap, 2015b, OpenStreetMap, 2015a]). Ce réseau représente les données d'entrée de notre algorithme lexicographique 3.2.5.1. A chaque arc est associé un ensemble d'attributs comme la longueur en mètre, le nombre de voies ainsi que le type de route. Comme la longueur de chaque route est en mètre, nous supposons que la vitesse de déplacement des individus est de $1m \cdot s^{-1}$. Cette vitesse de déplacement permet aussi de calculer les durées de trajet sur les arcs. Sans perte de généralité, la vitesse de déplacement des évacués peut être augmentée à une valeur plus élevée. En gardant comme limite la vitesse maximale sur les voies routières, il est alors possible d'effectuer l'évacuation de personnes par voitures personnelles. Le type de routes permet de définir si un piéton peut ou non les emprunter lors de l'évacuation. La largeur de ces dernières, définie par le nombre de voies, permet de savoir combien d'individus peuvent l'utiliser à un moment donné. Ainsi, cette information permet de déterminer la capacité d'une route.
Concernant la sécurité, nous nous référons au travail réalisé dans le BRGM [2] qui est l'un

2. Bureau de recherches géologiques et minières

de nos partenaires durant le projet DSS_Evac_Logistic. Le BRGM, est une institution publique de recherche et développement dans le domaine public, l'aide à la décision et différents domaines des sciences de la terre. Durant le projet, le BRGM réalise des outils d'aide à la décision, des modèles et met à notre disposition son expertise sur le management des risques. Durant le projet, l'une des tâches du BRGM est de simuler et de produire des scénarios lors de catastrophes naturelles comme des tsunamis, des glissements de terrain et leurs impacts sur les bâtiments et les réseaux routiers (voir [Lemoine *et al.*, 2014]). Par exemple, les débris issus de l'écroulement des bâtiments durant les tremblements de terre peuvent empiéter sur les routes et réduire leur capacité. Pour les données de sécurité, ces dernières sont obtenues en calquant le niveau de dommage des infrastructures à celui du réseau de Nice. Une approche similaire permettant de déterminer le niveau de sécurité d'un réseau est présentée par [Göttlich *et al.*, 2011] où à cause de gaz dangereux se déplaçant, les durées de trajet augmentent alors que les capacités des arcs diminuent.

Paramétrage des données

Le paramétrage de nos données s'articule autour de trois paramètres qui permettent de définir un scénario. Il s'agit du type de catastrophe considérée, de la partie de la ville qui est impactée et de la taille de la population à évacuer. Pour le type de catastrophe naturelle, nous considérons deux cas de tremblements de terre, celui du Vésubie et celui de Ligure. Les failles correspondantes ont été déplacées en mer près des côtes Niçoises. Les niveaux de sécurité sont illustrés sur les figures 3.9 et A.2 avec un niveau de sécurité moyen de $99,75\%$ pour le scénario du Vésubie alors que celui de Ligure est de $92,33\%$. L'écart-type associé à la sécurité est respectivement de $0,56\%$ et de $10,46\%$. Pour chacun de ces cas de figure, nous considérons deux types de zones pouvant être impactées par les sinistres. L'évacuation peut être limitée au centre ville (NICE_SMALL) (voir figures 3.10 et A.3) ou l'ensemble de la ville (NICE_LARGE)(voir figures 3.9 et A.2). La taille de la zone impactée permet aussi de déterminer le nombre de personnes que nous devons évacuer. Cette information permet de définir les K points de rassemblement et les L centres de secours. Dans le cas de la zone NICE_SMALL, $K = 512$ alors que $L = 119$ et si nous utilisons la zone NICE_LARGE comme zone impactée $K = 1533$ alors que $L = 242$.

Pour la taille de la population, nous considérons les résidents durant la période hivernale (POP_INSEE) ainsi que celle incluant les vacanciers durant la période estivale (POP_MAX). Les données statistiques concernant la population peuvent être obtenues grâce à [INSEE, 2015]. Afin de savoir combien de personnes doivent être évacuées, nous calquons la carte de répartition de la population avec celle des dommages subis par la ville afin de connaitre les bâtiments qui doivent être évacués. Avec ces trois paramètres, nous créons 8 instances de notre problème d'évacuation comme présenté dans le tableau 3.2. Nous utilisons aussi le Réseau Virtuel Dynamique à la place du réseau étendu afin d'éviter les problèmes dus à la taille des instances solvables. En effet, si nous devions utiliser ce dernier nous aurions du construire un réseau ayant $(N + M) \times T$ éléments avec N le nombre de sommets, M le nombre d'arcs et T l'horizon de planification. Par exemple, même en sous estimant la mémoire nécessaire pour modéliser notre problème en supposant que $T = 5000$ et que toutes les informations concernant un élément peuvent être modélisées avec 10 octets. Dans ce cas, construire un tel réseau étendu afin d'évacuer la ville de Nice

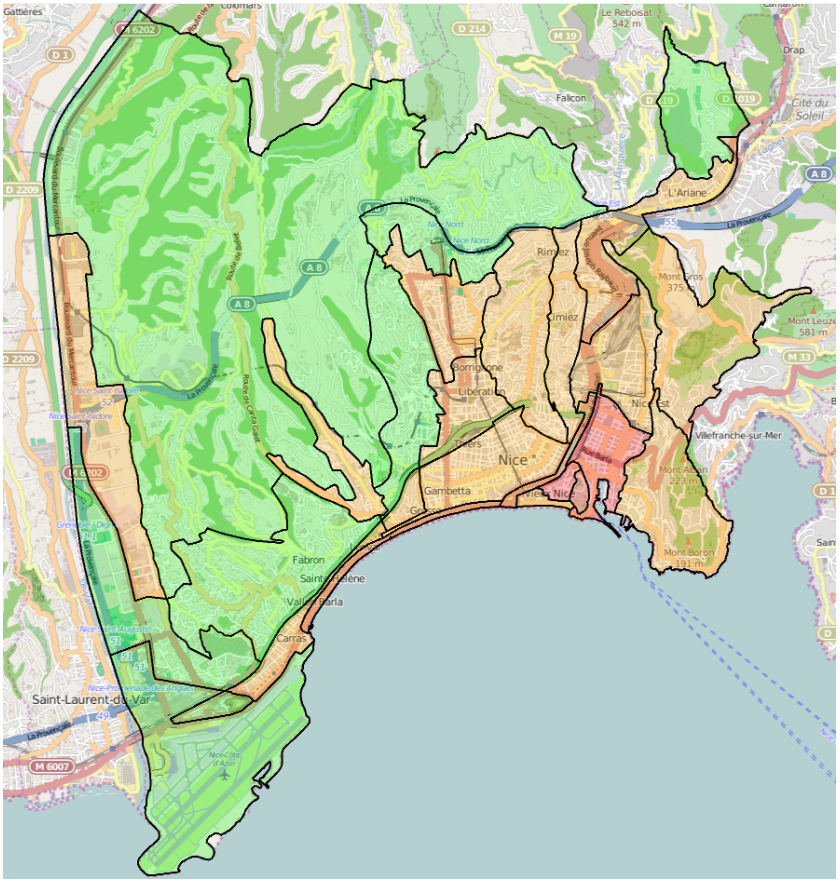


FIGURE 3.9 – Zone impactée par le scénario de Ligure sur l'ensemble de la ville de Nice

nécessiterait $(46990 + 91412) \times 5000 \times 10 octets = 6,44 Go$ de mémoire RAM ce qui est déjà largement au-dessus de la quantité de mémoire disponible pour l'ensemble de notre machine virtuelle de test (i.e. 4Go).

Résultats de tests Lex((Q|S) Flow)

Dans le but de mesurer les performances de notre approche lexicographique, nous effectuons deux types d'expérimentations sur nos instances. Pour la première série d'expérimentations, nous utilisons le critère de la durée pour trouver un ensemble de plus courts chemins sans prendre en compte la sécurité et à la fin nous évaluons le niveau de sécurité associé au plan d'évacuation. Dans la seconde série d'expérimentations, nous prenons en compte le critère de la durée afin de déterminer un ensemble de chaînes augmentantes et lorsque plusieurs chaînes ont la même durée, alors celles avec le meilleur niveau de sécurité seront utilisées en priorité.

Les résultats issus des tests sont présentés dans les tableaux [3.3-3.4] pour les 8 instances définies précédemment dans la section 3.2.7. (voir tableau 3.2)
La colonne "Nombre" de personnes indemnes permet de représenter le nombre fractionnaire

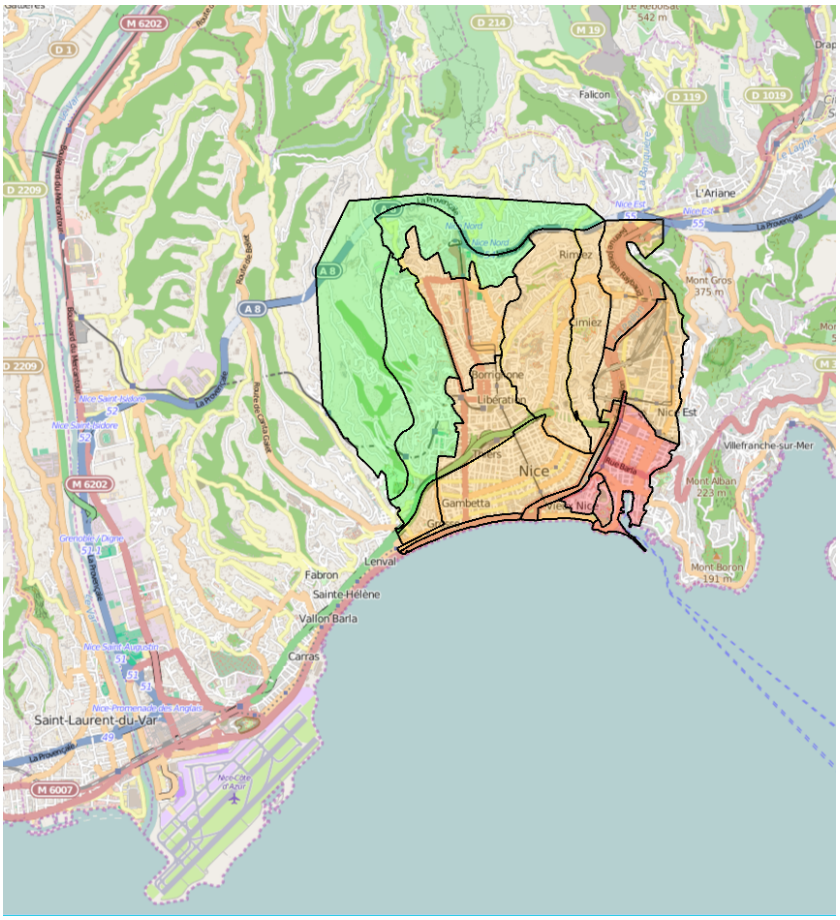


FIGURE 3.10 – Zone impactée par le scénario de Ligure sur le centre ville de Nice

d'évacués ayant atteint un des centres de secours sans subir le moindre dommage durant l'évacuation
(i.e. $\sum_{t=0}^{T} \sum_{P_l \in \Gamma_P^-} S^t_{(P_l,P)}$). Le temps de calcul est représenté en seconde.

Nous pouvons voir que le temps d'exécution nécessaire pour trouver le "Quickest Flow" augmente de manière proportionnelle au nombre de personnes à évacuer B. Cet augmentation correspond bien à la complexité pseudo-polynomiale de l'algorithme 3.2.5.1. En comparant les colonnes correspondantes à la durée d'exécution 3.3 et 3.4 nous pouvons constater que le fait de considérer le critère de sécurité augmente le temps d'exécution de l'algorithme dédié à la résoudre le problème Lex((Q|S) Flow) pour la plupart des instances. Nous pouvons aussi noter que le fait d'augmenter le nombre de points de rassemblement et de centres de secours, à savoir la zone impactée par le sinistre, n'influe pas sur les performances en matière de temps d'exécution. En effet, toutes les instances ont pu être résolues en moins de 3 heures. La sixième colonne du tableau 3.4 montre l'amélioration en nombre d'évacués indemnes. Comparé à une simple approche utilisant un ensemble

TABLE 3.2 – Instances de problèmes d'évacuation

Instance	Zone impactée	Catastrophe naturelle	Taille de la population
1_1_1	NICE_SMALL	Ligure	POP_INSEE
1_1_2	NICE_SMALL	Ligure	POP_MAX
1_2_1	NICE_SMALL	Vesubie	POP_INSEE
1_2_2	NICE_SMALL	Vesubie	POP_MAX
2_1_1	NICE_LARGE	Ligure	POP_INSEE
2_1_2	NICE_LARGE	Ligure	POP_MAX
2_2_1	NICE_LARGE	Vesubie	POP_INSEE
2_2_2	NICE_LARGE	Vesubie	POP_MAX

TABLE 3.3 – Résultats de tests de l'algorithme 3.2.5.1, sans le critère de sécurité (Partie 1).

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
1_1_1	512	119	80.638	5.096,05	18.461	2.860
1_1_2	512	119	113.465	6.984,94	49.287	10.660
1_2_1	512	119	8.013	6.617,83	5.633	266
1_2_2	512	119	11.872	9.801,06	5.721	363
2_1_1	1533	242	94.446	9.090,95	7.791	4.718
2_1_2	1533	242	142.661	12.563,6	14.148	8.870
2_2_1	1533	242	9.972	8.462,84	6.277	563
2_2_2	1533	242	14.444	12.195,1	6.277	682

de chaînes augmentantes, le problème Lex((Q|S) Flow) fournit des solutions dominantes comme présentées dans la septième colonne du tableau 3.4. Dans notre cas d'étude, le gain en matière de sécurité est limité car le réseau utilisé ne donne pas souvent l'occasion d'avoir le choix entre deux chemins de même longueur et de niveaux de sécurité différents. D'un autre côté, la durée de l'évacuation optimale est identique dans les deux cas de figure. En utilisant les instances précédemment définies, nous avons généré 8 instances modifiées qui peuvent être identifiées avec le préfixe "$M\_$". Dans ces instances, nous modifions la longueur des arcs en ne considérant que 10% de leur longueur. Ceci peut aussi être vu comme une augmentation de la vitesse de déplacement à $10ms^{-1}$ (i.e. $36km/h$). Les autres attributs restent inchangés. Les instances ainsi générées conservent la même topologie que le réseau initial de la ville de Nice. Les résultats de tests sur ces nouvelles instances sont décrits dans les tableaux[3.5-3.6].

Les mêmes remarques que celles pour les tableaux [3.3-3.4] peuvent être effectuées pour les instances modifiées. Cependant, lorsque l'on compare les instances normales et les instances modifiées, nous pouvons noter que le fait de réduire la longueur des arcs (i.e. augmentation de la vitesse de déplacement des évacués) réduit la durée de l'évacuation et dans la plupart des cas le temps d'exécution nécessaire pour calculer un plan d'évacuation.

TABLE 3.4 – Résultat de tests de l'algorithme, 3.2.5.1, avec critère de sécurité (Partie 1)

Instance	K	L	B	Personnes indemnes	Amélioration personnes indemnes	Durée de l'évacuation	Temps d'exécution	Augmentation du temps d'exécution
1_1_1	512	119	80.638	5.096,05	0	18.461	2.848	-0,42
1_1_2	512	119	113.465	6.985,35	0,41	49.287	10.645	-0,14
1_2_1	512	119	8.013	6.618,98	1,15	5.633	271	1,84
1_2_2	512	119	11.872	9.810,99	9,93	5.721	354	-2,54
2_1_1	1533	242	94.446	9.091,4	0,45	7.791	4.972	5,10
2_1_2	1533	242	142.661	12.564,5	0,9	14.148	10.606	16,36
2_2_1	1533	242	9.972	8.464,4	1,56	6.277	648	13,11
2_2_2	1533	242	14.444	12.205,7	10,6	6.277	779	12,45

TABLE 3.5 – Résultat de tests de l'algorithme 3.2.5.1 sans critère de sécurité (Partie 2).

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
M_1_1_1	512	119	80.638	4.015,31	15.864	4.402
M_1_1_2	512	119	113.465	5.357,11	48.691	11.957
M_1_2_1	512	119	8.013	7.245,58	571	162
M_1_2_2	512	119	11.872	10.429,2	626	238
M_2_1_1	1533	242	94.446	7.645,45	1.919	4.222
M_2_1_2	1533	242	142.661	10.197,9	8.396	10.260
M_2_2_1	1533	242	9.972	9.176,24	756	312
M_2_2_2	1533	242	14.444	12.852,9	764	422

Dans le cas de l'instance $M\_1\_1\_2$, le temps d'exécution de l'algorithme pour déterminer le plan d'évacuation augmente. Ceci est du au fait que trouver une chaîne augmentante devient de plus en plus difficile lorsque le réseau est presque saturé et que nous devons rediriger le flot de personnes déjà existant. Pour le nombre de personnes indemnes, nous constatons une légère augmentation dans les instances modifiées car le cas dynamique résultant permet de se retrouver plus souvent dans la situation où deux chaînes augmentantes ont la même longueur mais des niveaux de sécurité différents. Ceci montre aussi que notre approche est utile et qu'il existe des classes de graphes pour lesquelles elle est efficace.

3.2.8 Conclusion

Dans cette partie, nous avons donné une approche lexicographique permettant de produire un plan d'évacuation pour le problème Lex((Q|S) Flow). Le but de cette approche est de minimiser la durée globale de l'évacuation tout en prenant en compte la notion de sécurité auquel les évacués sont soumis. En utilisant la méthode développée par [Miller-Hooks et Stock Patterson, 2004] permettant de transformer n'importe quel problème de

TABLE 3.6 – Résultats de test de l'algorithme 3.2.5.1, avec prise en compte du critère de sécurité (Partie 2).

Instance	K	L	B	Personnes indemnes	Amélioration personnes indemnes	Durée de l'évacuation	Temps d'exécution	Augmentation du temps d'exécution
M_1_1_1	512	119	80.638	4.017,74	2,43	15.864	4.427	0,56%
M_1_1_2	512	119	113.465	5357,31	0,2	48.691	11.909	-0,40%
M_1_2_1	512	119	8.013	7.249,29	3,71	571	147	-10,20%
M_1_2_2	512	119	11.872	10.438,4	9,2	626	220	-8,18%
M_2_1_1	1533	242	94.446	7.651,65	6,2	1.919	4.310	2,04%
M_2_1_2	1533	242	142.661	10.200	2,1	8.396	10.380	1,15%
M_2_2_1	1533	242	9.972	9.180,57	4,33	756	334	6,58%
M_2_2_2	1533	242	14.444	12.861,9	9	764	451	6,43%

transbordement en un problème classique de "Quickest Flow", nous avons pu étendre cette dernière afin de pouvoir prendre en compte les dates de début d'évacuation au plus tôt.
Notre première contribution concerne le problème d'évacuation monocritère. Nous avons présenté un algorithme qui permet d'éviter l'utilisation du réseau étendu dans le temps en synthétisant l'ensemble de ses informations dans ce que nous appelons le Réseau Virtuel Dynamique. Cette nouvelle approche permet aussi d'avoir un algorithme pseudo-polynomial exact avec une complexité temporelle de $O(B(MLogT + NLogN))$. Cette complexité est meilleure que celles des meilleurs algorithmes exacts pour les problèmes de transbordement et de $EarliestArrivalFlow$ avec des données dépendantes du temps et une interdiction de stationnement dans le réseau. Pour autant que nous sachions, cette structure de données est efficace et améliore la meilleure complexité connue pour ce problème. Elle permet aussi de résoudre des instances difficiles comme la "1_1_2" et la "M_1_1_2" sans avoir à être limité par les contraintes de taille mémoire associées aux réseaux étendus dans le temps.
Notre seconde contribution porte sur le problème multicritère (i.e. l'ajout du critère de sécurité). De ce fait, nous pouvons maintenant résoudre le problème Lex((Q|S) Flow), où l'ordre lexicographique des deux critères est considéré. Pour autant que nous sachions ce problème était ouvert. Les résultats que nous avons présentés montrent une augmentation en moyenne de moins de $5,72\%$ du temps d'exécution lorsque l'on considère le critère de sécurité. Le nombre de personnes indemnes et la sécurité associée sont améliorés dans toutes les instances traitées de notre problème d'évacuation. Ceci montre que l'approche ainsi proposée est pertinente.

Toutefois la limite de cette approche est qu'elle permet d'avoir qu'une des solutions particulières associées au front de Pareto du problème multicritère. En effet, même si elle est basée sur une approche lexicographique, celle-ci ne peut être utilisée pour énumérer le front de Pareto exact dans la mesure où la valeur optimale de la durée est unique. Aussi, afin de faire croitre le nombre fractionnaire de personnes indemnes atteignant les centres

de secours, il faut accepter l'augmentation de la durée d'évacuation et ainsi considérer des solutions approchées. Dans la section suivante, nous allons étudier comment effectuer l'énumération approchée du Front de Paréto.

3.3 Énumération du front de Pareto

3.3.1 Introduction

Outre la difficulté de calculer un plan d'évacuation maximisant le nombre de personnes indemnes, nous avons montré dans la section 3.3 que les critères de durée et de sécurité sont conflictuels. L'exemple présenté par la figure 3.11 présente un problème d'évacuation où une personne doit être évacuée du super sommet source sommet S au super sommet puits sommet P. Entre ces sommets source et puits, nous avons un ensemble de $N = 4$ sommets communs à traverser. Entre deux sommets consécutifs, nous avons deux chemins possibles ayant chacun une durée de trajet, une capacité ainsi qu'un niveau de sécurité. Dans le cadre du problème multicritère, nous cherchons à minimiser la durée d'évacuation

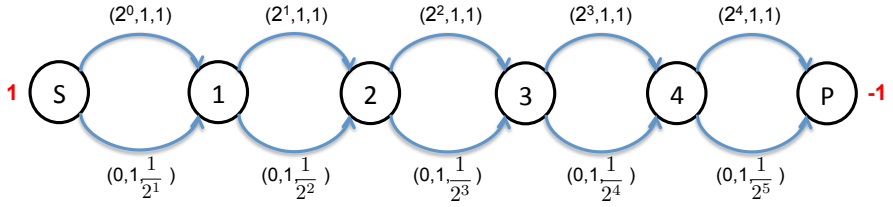


FIGURE 3.11 – Nombre exponentiel de solutions

et à maximiser la sécurité. Cet exemple permet de montrer que le nombre de solutions non dominées est potentiellement exponentiel (i.e. 2^{N+1} solutions pour cet exemple).

Dans cette partie, nous présentons dans un premier temps une méthode exacte permettant d'énumérer le front de Pareto et par la suite nous présentons un ensemble de méthodes heuristiques. Pour cette deuxième partie nous étendons notre approche lexicographique afin de pouvoir maximiser le nombre de personnes indemnes, mais aussi d'énumérer les solutions du front de Pareto.

3.3.2 Description du problème

Les données, variables et contraintes sont identiques à celles présentées dans la section 3.2.3.1. Toutefois, la fonction objectif est différente. En effet, afin de pouvoir déterminer le plan d'évacuation permettant de maximiser le nombre de personnes indemnes atteignant les centres de secours, nous devons relaxer les contraintes liées à la durée de l'évacuation ainsi qu'aux politiques de routage des individus. Pour cela, nous avons besoin de la durée minimale nécessaire pour réaliser l'évacuation de l'ensemble de la population T (3.13). Une fois l'horizon de planification minimale déterminée, cela nous permet de lui associer la

donnée $T^* \geq T$ qui correspond à la nouvelle date d'évacuation au plus tard. Une approche similaire est utilisée par [Groß et Skutella, 2012] afin de borner l'horizon de planification. Ci-dessous nous décrivons l'ensemble des fonctions objectif à considérer :

$$\text{Maximiser} \sum_{t=0}^{T^*} \sum_{P_l \in \Gamma_P^-} F^t_{(P_l,P)} \tag{3.18}$$

$$\text{Maximiser } 1 - \frac{1}{B}\left(B - \sum_{t=0}^{T^*} \sum_{P_l \in \Gamma_P^-} S^t_{(P_l,P)}\right) \tag{3.19}$$

$$\text{Maximiser} \sum_{t=0}^{T^*} \sum_{P_l \in \Gamma_P^-} F^t_{(P_l,P)} + \left(1 - \frac{1}{B}\left(B - \sum_{t=0}^{T^*} \sum_{P_l \in \Gamma_P^-} S^t_{(P_l,P)}\right)\right) \tag{3.20}$$

La fonction objectif (3.18) nous permet de nous assurer que l'ensemble des évacués atteindront les centres de secours avant la fin de l'évacuation fixée à la date $T^* \geq T$. Cette dernière a la particularité de toujours retourner la même valeur optimale qui est de B. Sa raison d'être est de s'assurer que les personnes, quel que soit leur état, atteignent un des centres de secours. La deuxième fonction objectif (3.19) permet de mesurer la proportion de personnes indemnes atteignant les centres de secours. La valeur de cette fonction objectif varie de 0.0 à 1.0 voulant respectivement dire que toutes les personnes ont subi des dommages et tout le monde est indemne. La dernière fonction objectif permet d'unifier les fonctions (3.18) et (3.19). Pour une date T^* fixée, elle permet de maximiser le nombre de personnes indemnes arrivant aux centres de secours. Pour autant que nous sachions ce problème d'évacuation n'a encore jamais été étudié. Dans la partie suivante, nous présentons deux types d'approches permettant de le résoudre. Tout d'abord nous allons présenter une méthode permettant de le résoudre et d'énumérer exactement le front de Pareto. Par la suite nous présenterons un ensemble de méthodes heuristiques permettant d'avoir un front de Pareto approché.

3.3.3 Algorithmes exacts

Afin de résoudre ce problème de manière exacte, nous utilisons le réseau étendu dans le temps. Ce dernier permet de prendre en compte aussi bien les cas où les données varient dans le temps que ceux où elles sont statiques tout au long de l'évacuation.

Lemme 3. *: Soit $G = (V, E, C, D, Q, \Delta, T^*)$ un réseau dynamique défini sur un horizon de planification T^*. La solution optimale du problème d'évacuation sans perte défini par* (3.3.2) *peut être obtenue grâce aux chaînes augmentantes successives de dommages minimaux.*

Démonstration. : Le réseau G a comme particularité d'être défini sur un horizon de planification $T^* \geq T$ permettant de s'assurer qu'il existe au moins un plan d'évacuation permettant à tous les évacués d'atteindre les centres de secours avant la fin de l'évacuation (voir (3.18)). Une fois le réseau dynamique G transformé en réseau étendu dans le temps

(voir [Ford et Fulkerson, 1958] et [Ford et Fulkerson, 1962]) ce dernier devient statique et permet d'utiliser la méthode des chaînes augmentantes à gain maximal (dommage minimal dans notre cas) présentée par [Onaga, 2006]. La fonction de sécurité associée (3.19) est croissante et chaque chaîne augmentante de dommage minimal respecte cet invariant. □

Corollaire 3. *: Si on a assez de temps pour évacuer une population et si on suppose que quel que soit le niveau de dommage subi par les évacués, ils arriveront aux centres de secours, alors la meilleure politique d'évacuation est d'utiliser les chemins les plus sûrs.*

Lemme 4. *: Soit T_{max} la date maximale autorisée pour terminer l'évacuation. Soit $G = (V, E, C, D, Q, \Delta, T_{max})$ un réseau dynamique défini sur un horizon de planification T_{max}. Soit T^{**} l'horizon de planification permettant d'avoir le plus de personnes indemnes aux centres de secours. T^{**} correspond à la date pour laquelle le nombre d'évacués indemnes est égal au nombre d'évacués si $T^* = T_{max}$.*

Démonstration. : Le lemme 3 montre comment déterminer un plan d'évacuation de l'ensemble de la population avec un maximum de sécurité pour une date T^* fixée. La fonction de sécurité (3.19) est croissante et fonction de l'horizon de planification. En effet, plus T^* sera supérieure à T (voir (3.13)) plus il existera de chemins de dommages minimaux. La fonction objectif (3.19) étant croissante, la solution optimale en terme de durée T^{**} est obtenue lorsque la valeur de la fonction (3.19) ne varie plus entre les dates $T^* = T^{**}$ et T_{max}. □

Corollaire 4. *: Le temps nécessaire et suffisant permettant d'avoir le plus de personnes indemnes aux centres de secours peut être obtenu grâce à une recherche dichotomique sur la valeur de T^**

Lemme 5. *: Soit $G = (V, E, C, D, Q, \Delta, T_{max})$ un réseau dynamique défini sur un horizon de planification T^{**}. Le front de Pareto exacte du problème d'évacuation associé à la fonction objectif* (3.3.2) *est entièrement défini entre la date T et la date T^{**} (voir le Lemme4).*

Démonstration. : La fonction objectif (3.19) est croissante en fonction du temps. Sa valeur minimale tout en sauvant l'ensemble de la population est obtenue pour la date T et sa valeur maximale est obtenue pour la date T^{**}. □

Corollaire 5. *: Il existe au plus $(T^{**} - T)$ solutions non dominées permettant d'évacuer l'ensemble de la population.*

Ci-dessous, nous présentons les algorithmes que l'on peut déduire des lemmes 3, 4 et 5.

3.3.3.1 Algorithme de maximisation de la sécurité pour une date fixée

Algorithme : Maximisation de la sécurité pour une date fixée
Donnée(s) : 1) G : Le réseau dynamique

2) B : Le nombre total de personnes à évacuer.
3) $T*$: L'horizon de planification.
Début :
4) G_{Etendu} % Construction du réseau étendu dans le temps avec G et T^*%
5) $F \leftarrow \emptyset$ % Fonction de flot %
6) **Tant que** ($B > 0$) **Faire**
7) Trouver la chaîne augmentante F de dommage minimale
8) Mettre à jour le réseau résiduel de G_{Etendu} avec la chaîne augmentante F
9) Réduire le nombre de personnes restant à évacuer (i.e. $B \leftarrow B - \min(B, F.valeur)$)
10) **Fin Tant que**
Fin :
Sortie(s) :
11) Le réseau résiduel de G_{Etendu}

L'agorithme 3.3.3.1 décrit comment maximiser le nombre personnes indemnes atteignant les centres de secours avant une date fixée. Comme données d'entrée, il reçoit le réseau dynamique associé au problème d'évacuation G, le nombre de personnes à sauver B et l'horizon de planification T^* (ligne 1 à 3). L'une des premières étapes de l'algorithme consiste à construire le réseau étendu G_{Etendu} associé au graphe G avec un horizon de planification T^*. Les complexités temporelle et spatiale pour construire un tel graphe est de $O((N+M)T^*)$. Par la suite, nous mettons la fonction de flot F à son état initial. Tant que toute la population n'a pas été évacuée, nous effectuons une recherche de chaîne augmentante. Les chaînes augmentantes recherchées ont la particularité d'être les chemins les plus sûrs allant du super sommet source S au super sommet puits P (ligne 7 à 10). La recherche de chaque chaîne augmentante dans le réseau étendu peut être effectuée en utilisant l'algorithme de Dijkstra modifié pour prendre en compte la sécurité (voir [Groß et Skutella, 2012]). Ainsi, la recherche de chaque chaîne augmentante nécessite $O(MT^*+NT^*Log(NT^*))$. Aussi dans le pire des cas, où la capacité résiduelle de chaque chaîne augmentante est d'une unité, la complexité de l'algorithme 3.3.3.1 est de $O(B(MT^*+NT^*Log(NT^*)))$. La prochaine étape consiste à étendre cet algorithme afin de déterminer le plan d'évacuation avec le meilleur niveau de sécurité.

3.3.3.2 Algorithme de maximisation de la sécurité

Algorithme : maximisation de la sécurité

Donnée(s) :
1) G : Le réseau dynamique
2) B : Le nombre total de personnes à évacuer.
3) T : L'horizon de planification minimale.
3) T_{max} : L'horizon de planification maximale.
Début :
4) $T*$ % L'horizon de planification.%
5) $Securite$ % Proportion de personnes indemnes %
6) $Securite_{max}$ % Proportion maximale de personnes indemnes %
7) $IndexTemporelMin$ % Horizon de planification minimale%
8) $IndexTemporelMax$ % Horizon de planification maximale%
9) Trouver le nombre maximum de personnes indemnes $Securite_{max}$

avec l'algorithme 3.3.3.1, le réseau G, B évacués et la date T_{max}.
10) $IndexTemporelMin \leftarrow T$
11) $IndexTemporelMax \leftarrow T_{max}$
12) $T* \leftarrow (IndexTemporelMax + IndexTemporelMin)/2$
13) $Securite \leftarrow 0$
14) **Tant que** ($IndexTemporelMin \neq IndexTemporelMax$) **Faire**
15) Déterminer le plan d'évacuation avec l'algorithme 3.3.3.1, le réseau G B évacués et la date T^*
16) Récupérer le niveau de sureté et l'affecter à la variable $Securite$
17) **Si** $Securite = Securite_{max}$ **Alors**
18) $IndexTemporelMax \leftarrow T^*$
19) **Sinon**
20) $IndexTemporelMin \leftarrow T^*$
21) **Fin Si**
22) $T^* \leftarrow (IndexTemporelMax + IndexTemporelMin)/2$
23) **Fin Tant que**
Fin :
Sortie(s) :
24) Le dernier réseau résiduel de G_{Etendu}

L'algorithme 3.3.3.2 permet de déterminer le plan d'évacuation permettant de maximiser le nombre de personnes indemnes arrivant aux centres de secours le plus rapidement possible. En données d'entrée nous avons le réseau dynamique G, le nombre de personnes à évacuer B, l'horizon de planification minimale T et l'horizon de planification maximale T_{max}. Sachant que la fonction de sécurité associée à notre problème est croissante et fonction du temps, nous voulons déterminer la date minimale T^{**} pour laquelle la sécurité est maximale. Pour cela nous avons la variable $Securite$ permettant de connaitre la sureté associée à la date T^* fixée et le meilleur niveau de sureté $Securite_{max}$ obtenu à la date T_{max}. Nous avons aussi les variables $IndexTemporelMin$ et $IndexTemporelMax$ correspondant à notre intervalle temporel de recherche de la date T^*. Cette date est toujours située au milieu de l'intervalle défini par $[IndexTemporelMin, ..., IndexTemporelMax]$ (voir ligne 12).
A partir de la ligne 14, nous réduisons la taille de cet intervalle par une recherche dichotomique. Aussi, tant que les deux bornes de cet intervalle ne correspondent pas à la même date, nous déterminons quel est le niveau de sureté associé à l'horizon de planification T^*. Ce niveau de sureté est sauvegardé dans la variable $Securite$. Si la variable $Securite$ a une valeur inférieure à $Securite_{max}$ alors nous savons que T^{**} se situe dans l'intervalle $[T^*, ..., IndexTemporelMax]$. Nous mettons donc à jour la borne inférieure de l'intervalle en augmentant la valeur de la variable $IndexTemporelMin$. Le cas échéant, si la variable $Securite$ a une valeur égale à $Securite_{max}$ alors nous savons que T^{**} se situe dans l'intervalle $[IndexTemporelMin, ..., T^*]$. Dans ce cas de figure nous diminuons la valeur de la borne supérieure associée à la variable $IndexTemporelMax$.
L'algorithme s'arrête lorsque $IndexTemporelMin = IndexTemporelMax = T^{**}$. Dans le pire des cas, le nombre d'itérations du à la recherche dichotomique est de $O(Log(T_{max}))$. Aussi, la complexité temporelle de l'algorithme 3.3.3.2 est de $O(BLog(T_{max})(MT_{max} + NT_{max}Log(NT_{max})))$. Une fois la date T^{**} obtenue, nous avons maintenant l'ensemble des éléments nécessaires permettant d'énumérer de manière exacte l'ensemble du front de Pareto.

3.3.3.3 Algorithme d'énumération exacte du front de Pareto

Algorithme : Enumération exacte du front de Pareto
Donnée(s) :
1) G : Le réseau dynamique
2) B : Le nombre total de personnes à évacuer.
3) T : L'horizon de planification minimale.
4) T_{max} : L'horizon de planification maximale.
Début :
5) $T*$ % L'horizon de planification.%
6) $T**$ % L'horizon maximale nécessaire pour avoir le maximum de personnes indemnes.%
7) FDP % Variable modélisant le front de Pareto.%
8) Déterminer T^{**} grâce à l'algorithme 3.3.3.2
9) **Pour** T^* allant de T à T^{**} **Faire**
10) Déterminer le plan d'évacuation avec l'algorithme 3.3.3.1, le réseau G, B évacués et la date T^*
11) Récupérer le niveau de sureté et mettre à jour le front de Pareto FDP si nécessaire
12) **Fin Pour**
Fin :
Sortie(s) :
13) Le front de Pareto FDP

L'algorithme 3.3.3.3 est dans la continuité de l'algorithme de maximisation de la sécurité pour un horizon de planification fixée (i.e. L'algorithme 3.3.3.1). Comme données d'entrée, nous disposons du réseau dynamique G, du nombre de personnes à évacuer B, de l'horizon de planification minimale T et de l'horizon de planification maximale T_{max}. A la ligne 7, nous initialisons le front de Pareto permettant de sauvegarder les solutions. A la ligne 8, nous déterminons l'horizon de planification optimale T^{**} à l'aide de l'algorithme 3.3.3.2. Une fois cette date déterminée, nous avons l'intervalle de définition du front de Pareto en terme d'horizon de planification (i.e. $[T, ..., T^{**}]$). Pour chaque horizon de planification $T^* \in [T, ..., T^{**}]$, nous faisons appel à l'algorithme 3.3.3.1 pour maximiser le nombre de personnes indemnes atteignant les centres de secours. Dans le pire des cas, la complexité de l'algorithme 3.3.3.3 est de $O(B(T^{**} - T)(MT^* + NT^*Log(NT^*)))$.
Ainsi, nous avons décrit un algorithme exact permettant de déterminer le front de Pareto de notre problème d'évacuation. Toutefois, ce dernier est fortement dépendant des différents horizons de planification et du fait qu'il faille construire l'ensemble du réseau étendu dans le temps pour pouvoir l'utiliser. Dans la partie suivante, nous allons présenter un algorithme approché permettant de réduire la complexité de cet algorithme.

3.3.4 Algorithmes approchés

Comme illustré précédemment, la résolution exacte du problème d'évacuation avec les algorithmes présentés dans la section 3.3.3 nécessite aussi une complexité pseudo-polynomiale aussi bien du point de vue temporelle que spatiale. Pour le problème Lex((Q|S) Flow), nous avons montré qu'il est possible de réduire aussi bien la complexité temporelle que spatiale avec l'algorithme 3.2.5.1. Cet algorithme a comme particularité d'utiliser les dates d'arrivée afin de déterminer les chaînes augmentantes. Or ces dates d'arrivée sont uniques et sont définies dans l'intervalle $[0, ..., T]$ permettant d'évacuer toute la population.

Toutefois, comme il ne s'agit plus de trouver l'horizon de planification minimale T, mais de maximiser le nombre de personnes indemnes arrivant aux centres de secours, il est alors possible d'utiliser la même approche que celle utilisant le Réseau Virtuel Dynamique.
Aussi, en cherchant une chaîne augmentante avec une date d'arrivée arbitraire, ce dernier ne garantit pas que la chaîne augmentante trouvée aura le meilleur niveau de sécurité. Par voie de conséquence, comme l'invariant défini par [Onaga, 2006] n'est pas respecté, le plan d'évacuation obtenu pour une durée fixée garantit que toutes les personnes seront bien évacuées, mais ne garantit pas que le nombre de personnes indemnes sera optimal. Dans ce cadre, le choix des dates d'arrivée est primordial. Nous présentons dans les sections suivantes deux heuristiques permettant d'obtenir un front de Pareto approché.

3.3.4.1 Algorithme : Safest$_{\text{Backward}}$S−P Path)

Algorithme : Safest$_{\text{Backward}}$S−P Path)

Donnée(s) :
1) G_{vsn} : Le Réseau Virtuel Dynamique.
2) $FlowMemory$: Mémoire de décision.
3) t_d : Date de départ du super sommet source S.
4) t_a : Date d'arrivée au super sommet puits P.

Début :
5) $FiboHeap$: Tas de Fibonacci
6) Π : Vecteur de prédécesseurs
7) $CheckResidualEdges$: Booléen

8) **Pour chaque** $Vertex \in G_{vsn} \backslash \{P\}$ **Faire**
9) $Vertex.date \leftarrow +\infty$
10) $Vertex.safety \leftarrow 0$
11) $FiboHeap.addVertex(Vertex, (1 - Vertex.safety))$
12) $\Pi[Vertex] \leftarrow Vertex$
13) **Fin Pour chaque**

14) $P.date \leftarrow -t_a$
15) $P.safety \leftarrow 1$
16) $FiboHeap.addVertex(P, (1 - P.safety))$

17) **Tant que** $isNotEmpty(FiboHeap)$ **Faire**
18) $CurrentVertex \leftarrow FiboHeap.minimum()$
19) $FiboHeap.deleteMinimum()$
20) **Si** $CurrentVertex.date > 0$ **Ou** $CurrentVertex = S$ **Alors**
21) *Stop*
22) **Fin Si**
23) $CheckResidualEdges \leftarrow true$

24) **Pour chaque** $PrevVertex \in \Gamma^-_{CurrentVertex}$ **Faire**
25) **Si** $isInFiboHeap(PrevVertex)$ **Alors**
26) **Si** $isEdgeAvailable(PrevVertex, CurrentVertex, FlowMemory)$ **Alors**
27) $NewDuration \leftarrow CurrentVertex.date +$
$D^{abs(CurrentVertex.date + \Delta^{CurrentVertex.date}_{(PrevVertex, CurrentVertex)})}_{(PrevVertex, CurrentVertex)}$

28) $NewSafety \leftarrow CurrentVertex.safety \times Q_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
29) **Si** $PrevVertex.Safety < NewSafety$ **Alors**
30) $\Pi[CurrentVertex] \leftarrow PrevVertex$
31) $PrevVertex.date \leftarrow NewDuration$
32) $PrevVertex.safety \leftarrow NewSafety$
33) $decreaseFiboKey(PrevVertex, (1 - PrevVertex.safety))$
34) $CheckResidualEdges \leftarrow false$
35) **Sinon Si** $PrevVertex.Safety = NewSafety$
36) **Et** $PrevVertex.date > NewDuration$ **Alors**
37) $\Pi[CurrentVertex] \leftarrow PrevVertex$
38) $PrevVertex.safety \leftarrow NewSafety$
39) $decreaseFiboKey(PrevVertex, (1 - PrevVertex.safety))$
40) $CheckResidualEdges \leftarrow false$
41) **Fin Si**
42) **Fin Si**
43) **Fin Si**
44) **Fin Pour chaque**

45) **Si** $CheckResidualEdges = true$ **Alors**
46) **Pour chaque** $PrevVertex \in \Gamma^-_{CurrentVertex}$ **Faire**
47) **Si** $isInFiboHeap(PrevVertex)$ **Alors**
48) **Si** $isResidualEdgeAvailable(PrevVertex, CurrentVertex, FlowMemory)$ **Alors**
49) $NewDuration \leftarrow CurrentVertex.date - D_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
50) $NewSafety \leftarrow CurrentVertex.safety \times Q_{(PrevVertex,CurrentVertex)}^{abs(CurrentVertex.date+\Delta_{(PrevVertex,CurrentVertex)}^{CurrentVertex.date})}$
50) **Si** $PrevVertex.Safety < NewSafety$ **Alors**
52) $\Pi[CurrentVertex] \leftarrow PrevVertex$
53) $PrevVertex.date \leftarrow NewDuration$
54) $PrevVertex.safety \leftarrow NewSafety$
55) $decreaseFiboKey(PrevVertex, (1 - PrevVertex.safety))$
56) **Sinon Si** $PrevVertex.Safety = NewSafety$
57) **Et** $PrevVertex.safety < NewSafety$ **Alors**
58) $\Pi[CurrentVertex] \leftarrow PrevVertex$
59) $PrevVertex.safety \leftarrow NewSafety$
60) $decreaseFiboKey(PrevVertex, (1 - PrevVertex.safety))$
61) **Fin Si**
62) **Fin Si**
63) **Fin Si**
64) **Fin Pour chaque**
65) **Fin Si**

66) **Fin Tant que**

67) $F \leftarrow flowBuilder(FlowMemory, \Pi)$
68) $t_d \leftarrow abs(VertexS.date)$
)**Fin :**
Sortie(s) :
69) $F : Integral\ flow\ function.$

70) t_d : *Departure date to super source S.*

Tout d'abord, nous présentons l'algorithme 3.3.4.1 qui est une extension de l'algorithme 3.2.5.3. Son objectif est de trouver une chaîne augmentante de niveau de sécurité maximal arrivant à une date fixée t_a. Aussi, deux différences principales peuvent être notées. Dans un premier temps, l'exploration de sommets à visiter n'est plus guidée par le critère de durée mais intégralement par celui de sécurité. Pour s'assurer de cela, les clés associées au tas de Fibonacci correspondent au niveau de sécurité des sommets (voir lignes 11, 16, 33, 39, 55 et 60). Dans un second temps, la mise à jour des étiquettes de chaque sommet ne s'effectue plus en fonction du critère de durée mais de celui de la sécurité (voir lignes 29, 35, 50 et 56). La complexité temporelle de cet algorithme est identique à celle présentée dans la section 3.2.5.3 soit de $O(MlogT + NLogN)$ avec dans ce cas de figure $T = t_a$. Dans les sections suivantes nous utilisons cet algorithme afin de trouver un ensemble de chaînes augmentante de niveau de sécurité maximal arrivant à une date fixée t_a au super sommet puits P et partant du super source S.

3.3.4.2 Heuristique H1

Algorithme : H1

Donnée(s) :
1) G_{vsn} : Le Réseau Virtuel Dynamique.
2) B : Le nombre total de personnes à évacuer.
3) $T*$: L'horizon de planification.

Début :
4) T_{min} : % Date d'arrivée du premier évacué %
5) $t_d \leftarrow 0$ % Date de départ du super sommet source S %
6) $t_a \leftarrow 0$ % Date d'arrivée au super sommet puits P %
7) $F \leftarrow \emptyset$ % Fonction de flot %
8) $FlowMemory \leftarrow \emptyset$ % Mémoire de décision %
9) $DatesDArrives$ % Vecteur de dates aléatoires d'arrivée. %

10) $F \leftarrow Lex(Shortest_{Forward}|Safest\ S\text{-}P\ path, G_{vsn}, FlowMemory, t_d, \&t_a)$
11) $T_{min} \leftarrow t_a)$
12) Initialisation du vecteur de dates d'arrivée $DatesDArrives$ entre T_{min} et T^*
13) On extrait une date aléatoire du vecteur $DatesDArrives$ et on l'affecte à t_a
14) $F \leftarrow Safest_{\text{Backward}} S\text{-}PPath, G_{vsn}, FlowMemory, \&t_d, t_a)$
15) **Tant que** $(B > 0)$ **Faire**
15) **Si** $(F.valeur = 0)$ **Alors**
16) On extrait une date aléatoire du vecteur $DatesDArrives$ et on l'affecte à t_a
17) **Sinon**
18) $Update(\&FlowMemory, F, t_d)$
19) $B \leftarrow B - \min(B, F.valeur)$
20) **Fin Si**
21) $F \leftarrow Safest_{Backward}\ S\text{-}P\ path, G_{vsn}, FlowMemory, \&t_d, t_a)$
22) **Fin Tant que**

Fin :

Sortie(s) :
18) $FlowMemory$: La mémoire de décision correspondant au plan d'évacuation

L'heuristique présentée par l'algorithme 3.3.4.2 a pour but de trouver un plan d'évacuation maximisant le nombre de personnes indemnes atteignant les centres de secours avant une date fixée T^* tout en garantissant que l'ensemble de la population sera sauvé. Comme données d'entrée, nous avons le Réseau Virtuel Dynamique G_{vsn} qui correspond à notre problème d'évacuation, le nombre de personnes à évacuer B, et l'horizon de planification T^*. Comme variables de l'algorithme nous avons la date T_{min} en dessous de laquelle aucune évacuation n'est possible, t_d la date de départ du super sommet source S, t_a la date d'arrivée au super sommet puits P. Nous avons aussi la variable F qui représente la fonction de flot(i.e. la chaîne augmentante), la mémoire de décision $FlowMemory$ et un ensemble comprenant toutes les dates d'arrivée possibles $DatesDArrives$.
La première partie de l'algorithme (lignes 10 à 11) consiste à déterminer T_{min}. Pour cela nous utilisons l'algorithme 3.2.5.2 qui permet de connaitre la date d'arrivée au plus tôt t_a. A la ligne 12, nous initialisons le vecteur de dates d'arrivée $DatesDArrives$. Une fois le vecteur initialisé, on y extrait une première date d'arrivée et l'on cherche s'il y a une chaîne augmentante de niveau de sécurité maximal (algorithme 3.3.4.1) arrivant à cette date (ligne 13 à 14). Aussi, tant que toutes les personnes n'ont pas été évacuées, et tant que cette date permet de trouver des chaînes augmentantes de niveau sécurité maximale, nous continuons d'utiliser les chaînes augmentantes arrivant à la date t_a. Lorsqu'une chaîne augmentante est trouvée, celle-ci est ajoutée à la mémoire de décision en utilisant l'algorithme présenté à la section 3.2.5.4. Si aucune chaîne augmentante ne permet d'arriver à la date t_a alors nous extrayons aléatoirement une nouvelle date d'arrivée du vecteur $DatesDArrives$ afin d'en faire la nouvelle date d'arrivée. Le processus décrit entre les lignes 15 et 22 est continué jusqu'à ce que l'ensemble de la population de taille B soit évacuée. Comme sortie de cet algorithme, nous avons la mémoire de décision $FlowMemory$ qui correspond au plan d'évacuation. L'avantage de cet algorithme est que à aucun moment, le réseau étendu dans le temps n'est pas construit. La complexité temporelle associée est de $O(B(MlogT^* + NLogN))$. Ayant une meilleure complexité temporelle et spatiale, cet algorithme approché peut être utilisé en lieu et place de l'algorithme 3.3.3.1. Une telle modification permet d'avoir un algorithme approché pour estimer l'horizon de planification minimal permettant de maximiser le nombre de personnes indemnes avec une complexité de $O(BLog(T_{max})(Mlog_{max} + NLogN))$ (voir algorithme 3.3.3.2) mais aussi d'obtenir un front de Pareto approché avec une complexité de $O(B(T^{**} - T)(MlogT^* + NLogN))$ (voir algorithme 3.3.3.3). Toutefois l'un des problèmes de cette heuristique est l'aspect stochastique lié à l'ordre de choix des dates d'arrivée t_a et donc la variabilité des solutions en terme de nombre de personnes indemnes arrivant aux centres de secours. Dans la partie suivante, nous présentons une heuristique de listes de priorités permettant de mieux choisir les dates d'arrivée.

3.3.4.3 Heuristique H2

Algorithme : H2

Donnée(s) :
1) G_{vsn} : Le Réseau Virtuel Dynamique.
2) $T*$: L'horizon de planification.

Début :
3) Normaliser le réseau G_{vsn}
4) $t_d \leftarrow 0$ % Date de départ du super sommet source S %
5) $t_a \leftarrow 0$ % Date d'arrivée au super sommet puits P %
6) $FiboHeap$: Tas de fibonacci
7) $F \leftarrow \emptyset$ % Fonction de flot %
8) $FlowMemory \leftarrow \emptyset$ % Mémoire de décision %

9) $F \leftarrow Lex(Shortest_{Forward}|Safest\ S\text{-}P\ path, G_{vsn}, FlowMemory, t_d, \&t_a)$
10) $F \leftarrow Safest_{\text{Backward}} S\text{-}PPath, G_{vsn}, FlowMemory, \&t_d, t_a)$
11) **Tant que** $(t_a \leq T^*)$ **Faire**
12) **Si** $(F.valeur = 0)$ **Alors**
13) $t_a \leftarrow t_a + 1$
14) **Sinon**
15) $Update(\&FlowMemory, F, t_d)$
16) **Fin Si**
17) $F \leftarrow Safest_{Backward}\ S\text{-}P\ path, G_{vsn}, FlowMemory, \&t_d, t_a)$
18) **Fin Tant que**
19) Ajout des dates d'arrivée de la mémoire de décision $FlowMemory$ au tas de Fibonacci $FiboHeap$

Fin :
Sortie(s) :
20) $FiboHeap$: le tas de Fibonacci associé aux dates d'arrivée les plus sures

Cette heuristique a pour but de déterminer une liste de priorité concernant les dates d'arrivée potentielles. Comme données d'entrée nous avons le Réseau Virtuel Dynamique G_{vsn}, et l'horizon de planification T^*. La premier étape consiste à normaliser le Réseau Virtuel Dynamique G_{vsn}. En effet, ce dernier ne permet pas d'évacuer plus de B individus. Nous relaxons donc les contraintes liées au nombre de personnes à évacuer des points de rassemblement ainsi que celles liées aux centres de secours (voir ligne 3). Contrairement à l'algorithme 3.3.4.2, celui-ci a comme particularité de disposer d'un tas de Fibonacci $FiboHeap$ permettant de trier les dates d'arrivée suivant le nombre de personnes pouvant atteindre un centre de secours à une date donnée. Le but n'est pas donc de déterminer un plan d'évacuation mais de connaitre les dates auxquelles les gens sont le plus susceptibles d'arriver à un des centres de secours. En terme de complexité, l'algorithme 3.3.4.3 effectue dans le pire des cas $O(T^*(MlogT^* + NLogN))$ opérations. Toutefois, une fois utilisé au début de l'algorithme 3.3.4.2, il peut remplacer la méthode aléatoire permettant de choisir l'ordre des dates d'arrivée.

3.3.5 Résultats expérimentaux

Cette partie décrit les résultats expérimentaux permettant d'évaluer les performances de notre algorithme d'énumération du front de Pareto. Toutefois la version exacte est

inutilisable en pratique sur des instances à l'échelle d'une ville du fait de la taille des réseaux étendus dans le temps mais aussi du temps d'exécution nécessaire. Aussi nous présentons les résultats obtenus avec la version approchée utilisant l'algorithme H1 3.3.4.2.

3.3.6 Description des jeux de test générés

Afin d'évaluer notre approche d'énumération approchée du front de Pareto, nous utilisons les mêmes jeux d'instances que ceux utilisés pour évaluer l'algorithme 3.2.5.1 (voir les huit instances de la section 3.2). Ces derniers correspondent à nos réseaux dynamiques. La durée minimale nécessaire afin d'évacuer l'ensemble de la population de chaque instance correspond à la durée d'évacuation T présentée dans le tableau de résultat 3.4 (i.e. $T = 18461$ pour l'instance 1_1_1). Dans le cadre de nos tests, nous définissons l'horizon de planification maximale T_{max} comme étant égale à $T + 1000$ (i.e. $T_{max} = T + 1000 = 19461$ pour l'instance 1_1_1). Ainsi, dans l'intervalle de temps $[T, .., T_{max}]$ nous échantillonnerons le front de Pareto toutes les 50 unités de temps (i.e. $T^* = 18461, 18511, 18561, ..., 19461$ pour l'instance 1_1_1). L'objectif de cette augmentation est de savoir combien de personnes indemnes en plus nous pouvons avoir si nous augmentation la date de fin d'évacuation tout en maîtrisant le nombre potentiel de solutions se trouvant dans l'intervalle $[T, .., T_{max}]$ lors de l'échantillonnage du front de Pareto (i.e. au plus $\frac{1000}{50} + 1 = 21$ solutions).

3.3.7 Résultats de l'énumération approchée du front de Pareto

Dans cette partie, nous présentons les résultats obtenus pour l'énumération du front de Pareto de nos huit instances réelles de la ville de Nice (voir 3.2). Les résultats sont détaillés dans les tableaux [3.7 - 3.14] qui répertorient les solutions non dominées associées à chaque instance.

TABLE 3.7 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 1_1_1

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
1_1_1	512	119	80.638	5.884,83	18.461	1.216
1_1_1	512	119	80.638	6.037,66	18.511	1.190
1_1_1	512	119	80.638	6.038,54	18.561	1.272
1_1_1	512	119	80.638	6.155,3	19.061	1.030
1_1_1	512	119	80.638	6.213,49	19.111	1.561
1_1_1	512	119	80.638	6.288,85	19.361	1.427

La première constatation qui peut être effectuée est que le niveau de sécurité correspondant au nombre de personnes indemnes de toutes les instances est meilleur que celle obtenue en utilisant l'algorithme 3.2.5.1 pour une durée d'évacuation identique T. En effet, ce dernier correspond à une des solutions du problème "Quickest Flow" dont la date de fin est T et pour lequel nous cherchons à maximiser la proportion de personnes indemnes (i.e. le niveau de sécurité du plan d'évacuation). Cette amélioration du niveau de sécurité est du au fait que chaque date d'arrivée possible, nous ne cherchons plus le chemin le plus rapide permettant d'atteindre un centre de secours en partant d'un des points de rassemblement.

TABLE 3.8 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 1_1_2

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
1_1_2	512	119	113.465	8.008,13	49.287	2.430
1_1_2	512	119	113.465	8.010,51	49.737	2.369
1_1_2	512	119	113.465	8.011,93	49.937	2.110
1_1_2	512	119	113.465	8.012,05	49.987	1.975
1_1_2	512	119	113.465	8.012,17	50.037	1.996
1_1_2	512	119	113.465	8.012,29	50.087	1.941
1_1_2	512	119	113.465	8.016,98	50.137	1.895
1_1_2	512	119	113.465	8.017,09	50.187	1.890
1_1_2	512	119	113.465	8.017,21	50.237	1.831
1_1_2	512	119	113.465	8.017.33	50.287	1.859

TABLE 3.9 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 1_2_1

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
1_2_1	512	119	8.013	6.941.32	5.633	167
1_2_1	512	119	8.013	6.943.32	5.683	154
1_2_1	512	119	8.013	6.969.74	5.733	169
1_2_1	512	119	8.013	6.974.98	5.783	166

A la place, nous recherchons la chaîne augmentante ayant le meilleur niveau de sécurité permettant d'arriver à une date fixée T^* à un des centres de secours en partant d'un des points de rassemblement. La seconde constatation est en rapport avec l'évolution du temps d'exécution lorsque l'on augmente la durée de l'évacuation. En effet, ce dernier reste assez stable comparé à ceux obtenus avec l'algorithme 3.2.5.1. Du fait que l'on dispose de plus de temps que nécessaire pour aller du super sommet source S au super sommet puits P, il devient alors plus facile de trouver une chaîne augmentante ne remettant pas en cause les flots précédents qui peut être couteux en temps d'exécution. Enfin l'une des dernières remarques que nous pouvons faire à l'issue de ces expérimentations est le lien qui existe entre le niveau de dommage associé au scénario de catastrophe naturelle et la proportion de personnes indemnes pouvant atteindre les centres de secours. En effet, dans le cadre du scénario de tremblement de terre Vésubie avec un faible niveau d'endommagement des infrastructures, nous pouvons constater que la proportion de personnes indemnes atteignant les centres de secours est élevée car les arcs traversés ont un niveau de sécurité élevé, soit 99,75% en moyenne avec un écart-type de 0,56%. Toutefois, dans le cadre du scénario catastrophe Ligure, cette proportion chute du fait que le niveau de sécurité des arcs traversés par les individus est de 92,33% en moyenne avec un écart-type de 10,46%. Ceci montre l'impact du sinistre considéré sur la proportion de personnes pouvant être évacuées sans dommages.

TABLE 3.10 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 1_2_2

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
1_2_2	512	119	11.872	10.074,7	5.721	304
1_2_2	512	119	11.872	10.088,2	5.771	300
1_2_2	512	119	11.872	10.095,2	5.821	307
1_2_2	512	119	11.872	10.107,3	5.871	311
1_2_2	512	119	11.872	10.107,9	6.071	340
1_2_2	512	119	11.872	10.113,3	6.121	387
1_2_2	512	119	11.872	10.116,1	6.171	396
1_2_2	512	119	11.872	10.117,2	6.221	390
1_2_2	512	119	11.872	10.155,4	6.271	422
1_2_2	512	119	11.872	10158,4	6.421	433
1_2_2	512	119	11.872	10169,0	6.721	388

TABLE 3.11 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 2_1_1

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
2_1_1	1533	242	94.446	9.788,29	7.841	3.077
2_1_1	1533	242	94.446	9.820,46	8.291	2.820
2_1_1	1533	242	94.446	9.835,67	8.391	2.521
2_1_1	1533	242	94.446	9.940,83	8.441	2.340
2_1_1	1533	242	94.446	10.014,8	8.541	2.186
2_1_1	1533	242	94.446	10.137,2	8.641	2.152

3.3.8 Conclusion

Dans cette section nous avons dans un premier temps présenté la structure globale permettant de résoudre de manière exacte le problème d'optimisation de la sécurité durant l'évacuation pour une date fixée. Ce dernier se base sur le fait d'étendre le réseau dynamique dans le temps, en cherchant des chaînes augmentantes de sécurités maximales. Par la suite nous avons proposé une approche permettant d'énumérer de manière exacte le front de Pareto en utilisant une approche $\epsilon - contraintes$ en fixant la durée maximale autorisée pour réaliser l'évacuation. Toutefois, cette approche exacte a deux inconvénients qui sont les complexités temporelle et spatiale qui lui sont associées. Aussi, afin de pouvoir palier à ces deux inconvénients, nous avons proposé une approche heuristique se basant sur le Réseau Virtuel Dynamique. Cette approche heuristique a permis de traiter des instances de grande taille tout en permettant d'optimiser la sécurité du plan d'évacuation comme illustré par les résultats obtenus dans la section 3.3.7. Nous avons ainsi pu montrer que notre approche pouvait être appliquée à des instances réelles de scénario catastrophes appliquées à ville de Nice et permettait de déterminer des plans d'évacuation sûrs pour une durée fixée dans le temps permettant tout aussi bien d'évacuer l'ensemble de la population que de maximiser la proportion d'individus ne subissant pas de dommages durant l'évacuation.

TABLE 3.12 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 2_1_2

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
2_1_2	1533	242	94.446	12.905,5	14.148	4.096
2_1_2	1533	242	94.446	12.920,0	14.198	4.350
2_1_2	1533	242	94.446	12.955,7	14.248	5.812
2_1_2	1533	242	94.446	12.999,4	14.348	4.198
2_1_2	1533	242	94.446	13.056,9	14.398	4.164
2_1_2	1533	242	94.446	13.111,7	15.048	3.254

TABLE 3.13 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 2_2_1

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
2_2_1	1.533	242	9.972	8.705,09	6.277	728
2_2_1	1.533	242	9.972	8.706,29	6.427	694
2_2_1	1.533	242	9.972	8.708,98	6.477	835
2_2_1	1.533	242	9.972	8.733,17	6.527	789
2_2_1	1.533	242	9.972	8.736,31	6.577	781
2_2_1	1.533	242	9.972	8.737,48	6.827	867
2_2_1	1.533	242	9.972	8.754,52	6.877	802
2_2_1	1.533	242	9.972	8.772,76	6.927	428
2_2_1	1.533	242	9.972	8.776,7	6.977	395
2_2_1	1.533	242	9.972	8.787,55	7.277	389

3.4 Conclusion du problème d'évacuation sans pertes

Nous avons, dans cette partie, présenté plusieurs approches permettant d'effectuer le routage sans pertes des individus durant une évacuation dans un réseau dont les données varient dans le temps tout en prenant en compte la contrainte de non stationnement. Dans un premier temps nous avons proposé une approche permettant de résoudre les problèmes d'évacuation correspondant aux politiques du "Quickest Flow" et du "Earliest Arrival Flow". Pour cela, nous avons mis en place une nouvelle structure de données appelée Réseau Virtuel Dynamique et permettant de prendre en compte l'intégralité d'un réseau dynamique étendu dans le temps sans pour autant le construire explicitement. Ce dernier, contrairement au réseau étendu dans le temps, n'a comme seul contrainte que le fait de choisir la date de départ ou d'arrivée d'une chaîne augmentante du super sommet source S au super sommet puit P. En échange de cette contrainte, il permet de ne pas dupliquer les arcs et les sommets dans le temps mais aussi de réduire l'espace de recherches dans le graphe dynamique diminuant ainsi de fait le temps de résolution. Sur cette base, nous avons étendu notre problème d'évacuation en ajoutant le critère de sécurité dans le cadre d'une approche lexicographique Lex((Q|S) Flow) nous permettant sans perte de généralités d'avoir la meilleure politique d'évacuation (i.e. "Earliest Arrival Flow"), la durée minimale nécessaire pour évacuer l'ensemble de la population T ainsi que la prise en compte de la notion de sécurité à savoir l'utilisation des chemins les plus sûrs durant le processus de

TABLE 3.14 – Résultats de tests de l'algorithme 3.3.3.3 appliqué à l'instance 2_2_2

Instance	K	L	B	Personnes indemnes	Durée de l'évacuation	Temps d'exécution
2_2_2	1533	242	14.444	12.469,9	6.277	934
2_2_2	1533	242	14.444	12.520,3	6.477	1.006

routage. Toutefois la solution associée à ce problème lexicographique étant unique, nous avons relaxé la contrainte associée à la politique d'évacuation adoptée en remontant d'un niveau à savoir pour nous intéresser au problème du "Quickest Flow" qui n'impose pas de date d'arrivée impératives tout en ayant la même date de fin d'évacuation. Aussi en connaissant la date de fin d'évacuation au plus tôt, nous avons proposé un algorithme exact permettant de trouver le plan d'évacuation le plus sûr qui lui est associé. En se donnant plus de temps pour router l'ensemble de la population nous avons aussi proposé une approche $\epsilon - Contrainte$ permettant d'énumérer de manière exacte le front de Pareto. Toutefois cette méthode exacte se base sur un réseau étendu dans le temps qui n'est pas utilisable en pratique sur des instances réelles de grande taille. Afin de palier à cette insuffisance, nous avons proposé une approche heuristique utilisant le Réseau Virtuel Dynamique à la place de réseau étendu dans le temps. Comme ce dernier ne possède pas de contraintes sur la politique d'évacuation à adopter, nous avons pu montrer que notre heuristique avait de meilleurs résultats en terme de sécurité que ceux de l'approche lexicographique tout en ayant la même durée d'évacuation sur des instances de taille réelle. En augmentant et en bornant la date de fin d'évacuation, nous avons pu établir le front de Pareto approché associé au problème d'optimisation bicritère durée et sécurité en cas d'évacuation et de routage de masse d'une grande ville dont toutes les personnes atteindront par leur propre moyen un des centres de secours dans un certain état de santé.
Toutefois, il peut arriver que durant une évacuation certaines personnes aient besoin d'aide ou de dispositifs particuliers pour être prises en charge à cause de lésions ou de blessures les empêchant de poursuivre leurs déplacements dans le réseau. Nous étudions ce type d'évacuation dans la partie suivante du manuscrit qui traite de l'évacuation avec pertes dont le but est de minimiser le nombre de personnes soumis aux dangers au point de devoir être prises en charge.

Chapitre 4

Routage de personnes avec pertes

4.1 Introduction du problème d'évacuation avec pertes

Plusieurs études peuvent être trouvées dans la littérature concernant les problèmes d'évacuation. La plupart de ces travaux portent sur l'optimisation du critère lié au temps total nécessaire pour mettre toute la population à l'abri (voir [Hamacher et Tjandra, 2002]). Toutefois, la majeure partie de ces études ne prennent pas en compte le fait que les évacués doivent potentiellement traverser des zones dangereuses ou mortelles durant leur périple afin d'atteindre les zones sûres ou les centres de secours. Ces cas de figure peuvent être illustrés lorsqu'un bâtiment est en feu. Les évacués doivent certes sortir de ce dernier le plus rapidement possible, mais ils doivent le moins possible inhaler des gaz toxiques qui peuvent leur être fatal et les empêcher de poursuivre l'évacuation par manque d'oxygène [Opasanon et Miller-Hooks, 2009]. L'exemple précédent peut être étendu à l'évacuation d'une zone urbaine lorsqu'une catastrophe naturelle frappe cette dernière. Ainsi, les routes qui doivent être utilisées pour évacuer les individus peuvent devenir dangereuses à cause des éboulements, de chutes d'arbres, d'un feu de forêt, d'une inondation ou d'un tsunami [Lämmel *et al.*, 2011]. Dans le but de mesurer le niveau de sécurité d'un plan d'évacuation, ces types de problèmes peuvent être modélisés comme un problème de flot généralisé dynamique. Le critère de durée d'évacuation est alors modélisé par un réseau étendu dans le temps borné alors que le critère de sécurité est modélisé comme étant la proportion de personnes indemnes que l'on peut mettre en lieu sûr avant une date T^* [Groß et Skutella, 2012]. Dans le cadre d'une évacuation où le réseau est perturbé, nous pouvons considérer que chaque arc a un certain niveau de sécurité qui est un nombre réel compris entre 0.0 et 1.0. Ce niveau de sécurité est un scalaire qui module le niveau de santé des évacués utilisant un arc lors de l'évacuation. En conséquence, si 10 personnes tentent de traverser un arc ayant un niveau de sécurité de 0.75 alors seulement "7.5" personnes réussiront à s'en sortir indemnes. Les "2.5" personnes qui auront subi des lésions devront arrêter leur périple car nécessitant des moyens supplémentaires pour poursuivre l'évacuation (médecins, chaises roulantes, atèles, pompiers...). Cependant, cette approche basée sur les flots généralisés, tout comme celle présentée dans le chapitre 3 n'est pas suffisante. En effet, dans l'une ou l'autre des approches, le nombre de personnes indemnes n'est pas un nombre entier alors qu'une personne est indivisible en parties. Dans l'exemple précédent où "7.5" individus

étaient indemnes et pouvaient continuer l'évacuation, il y a dans ce groupe 7 individus et la moitié d'une personne soit $0,5$ qui peuvent continuer l'évacuation. Pire encore, si un autre groupe de personnes indemnes avec la moitié d'une autre personne les rejoint alors les deux moitiés fusionneront et donneront une nouvelle personne indemne. Dans le but d'éviter ces types de contradictions, nous devons prendre en compte le fait que le flot de personnes est strictement entier. En outre, lorsque sur une voie ou une route il y a un obstacle dangereux comme un arbre déraciné, ce dernier n'affecte pas tout l'arc mais bien un point particulier de celui-ci. Il est aussi possible de noter que sur la même voie ou route, nous pouvons avoir plusieurs sinistres devant être surmontés par les individus voulant la traverser. En 1974, [Sahni, 1974] a introduit le problème de flot avec multiplicateurs entiers positifs pour lequel les multiplicateurs $Pr_i \in \mathbb{Z}^+$ se situent non plus sur les arcs mais sur les sommets. Comme durant l'évacuation de la population, la santé des évacués est une fonction décroissante, nous étendons le problème présenté par [Sahni, 1974] afin de prendre en compte cet aspect. En outre, durant le processus d'évacuation, les personnes ne sont la plupart du temps pas livrées à elles-mêmes. En effet, un sous-ensemble d'individus (responsables de sécurité, agents de police, pompiers,...) peut être habilité à organiser l'évacuation afin de la faciliter. Outre le fait de donner des consignes d'évacuations, il est aussi de leur devoir de sécuriser certains passages dangereux comme fermer les ascenseurs, réguler la circulation ou débloquer un chemin rendu inutilisable. Pour cela, nous définissons le problème de flot généralisé entier avec diviseurs (FGED) pour lequel les diviseurs (niveau de sécurité) $Pr_i \in \{x \in \mathbb{R} | 0 \leq x \leq 1\}$ et pour lequel les niveaux de sécurité se situent sur les sommets (Figure 4.1).
Dans ce chapitre, nous traitons différents problèmes d'évacuations qui sont liés à ce problème de flot. Tout d'abord nous étudierons comment maximiser le nombre de personnes entières et indemnes atteignant les centres de secours. Puis, nous verrons comment sécuriser les points critiques du réseau durant l'évacuation afin qu'un plus grand nombre de personnes indemnes arrivent au niveau des centres de secours. Ensuite nous étudierons le déploiement des forces de secours et des interactions qu'ils peuvent avoir avec les évacués. Enfin, dans le cadre d'un travail réalisé en collaboration avec Marc Goerigk qui effectue son post-doctorat au sein de la faculté de mathématiques de l'université technique de Kaiserslautern (Optimization Research Group), nous verrons comment assurer la robustesse de tels plans d'évacuation lorsque les niveaux de dommages ne sont pas connus avec certitude.

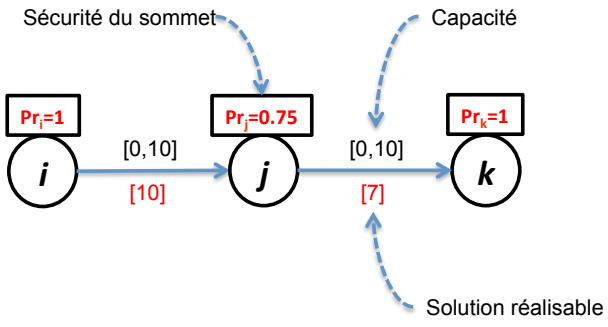


FIGURE 4.1 – Exemple de flot généralisé entier avec diviseurs (FGED)

4.2 Flot généralisé entier avec diviseurs

4.2.1 Description du problème

Dans cette partie, nous présentons la problématique liée à l'évacuation de personnes avec pertes. Afin de décrire cette dernière de manière explicite, le réseau dynamique correspondant au problème d'évacuation sera étendu dans le temps. Cette approche permet ainsi ainsi de se concentrer uniquement l'optimisation du critère de sécurité tout en ne perdant pas la notion de durée. Nous considérons un réseau étendu dans le temps dont les données varient. A ce dernier est associé un ensemble de points de rassemblement (sommets sources) et de centres de secours (sommets puits). Un certain nombre d'évacués doivent se déplacer des points de rassemblement vers des centres de secours ayant une certaine capacité d'accueil. Sur chaque arc du réseau nous avons des attributs de durée de trajet ainsi que du nombre maximum de personnes pouvant y entrer à un moment donné. Sur chaque sommet (réel ou fictif) nous avons un niveau de sécurité qui peut être vu comme la probabilité pour un évacué de le traverser sans subir des dommages. Notre but est de savoir combien d'évacués indemnes au sens strict peuvent effectivement atteindre les centres de secours avant une certaine date tout en restant indemnes. Tout comme pour le problème de transbordement [Hoppe et Tardos, 2000], le réseau dynamique associé à ce problème d'évacuation peut être transformé en graphe en utilisant la procédure proposée par [Miller-Hooks et Stock Patterson, 2004] ou l'extension présentée dans la section 3.2.4.1. Le but d'une telle transformation est d'avoir un super sommet source et un super sommet puits dans le but d'en faire un problème de maximisation de flot.

4.2.1.1 Modèle mathématique

Ci-dessous nous présentons la formulation mathématique du problème de Flot Généralisé Entier avec Diviseur (FGED).
Données :

- $G = (V, E)$ Un réseau étendu dans le temps avec des données qui varient en fonction du temps. Ainsi les attributs de durée sont directement intégrés dans le réseau.
- V : Un ensemble de N sommets.
- S et P : Le super sommet source et le super sommet puits.
- E : Un ensemble de M arcs $\subseteq V \times V$.
- $C_{(i,j)}$: Le nombre maximum de personnes pouvant entrer dans l'arc $(i, j) \in E$.
- Pr_i : La probabilité de traverser sans dommages le sommet i avec $Pr_i \in \{x \in \mathbb{R} | 0 \leq x \leq 1\}$.
- B_S : Le nombre total de personnes à évacuer depuis le super sommet source S.
- B_P : Le nombre maximum de personnes pouvant atteindre le super sommet puits P.
- B_{min} : Le nombre minimum de personnes devant atteindre le super sommet puits P. Cette donnée est lié à la version décisionnelle du problème d'évacuation que nous expliciterons par la suite.

Variable :

- $F(i, j)$: le nombre de personnes empruntant l'arc(i, j) avec $F_{(i,j)} \in \mathbb{N}$.

Contraintes :

$$\text{Maximiser} \sum_{i \in \Gamma_P^-} F_{(i,P)} \tag{4.1}$$

S.C

$$0 \leq F_{(i,j)} \leq C_{(i,j)} \qquad \forall (i,j) \in E \tag{4.2}$$

$$Pr_j \times \sum_{i \in \Gamma_j^-} F_{(i,j)} \geq \sum_{k \in \Gamma_j^+} F_{(j,k)} \qquad \forall j \in V \setminus P \tag{4.3}$$

$$\sum_{i \in \Gamma_S^+} F_{(S,i)} \leq B_S \tag{4.4}$$

Ci-dessous, l'ensemble des contraintes permettant de décrire notre problème central d'évacuation avec pertes.

- Le nombre maximum de personnes entrant dans l'arc $(i,j) \in E$ est inférieur ou égale à la capacité de l'arc considéré (4.2).
- Le nombre total de personnes indemnes sortant d'un sommet $i \in V$ est inférieur ou égal au nombre de personnes y étant entré multiplié par le niveau de sécurité du sommet considéré (4.3).
- Au plus B_S individu pourrons tenter de se mettre à l'abri en quittant le super sommet source S(4.4).

Fonctions objectif : Nous considérons deux types de problème d'évacuation. Lorsqu'il s'agit d'un problème d'optimisation (4.1) nous cherchons à maximiser le nombre d'individus indemnes arrivant au super sommet puits P avant la fin de l'évacuation.
Nous pouvons aussi considérer la version décisionnelle du problème d'évacuation associée à la question suivante : "Ayant un réseau G étendu dans le temps, est-il possible de mettre au moins B_{min} individus indemnes à l'abri ?".

Comme nous pouvons le constater, les durées de trajet ont disparu de la formulation de notre problème d'évacuation. En effet, utilisant un réseau étendu dans le temps, la notion de durée est encodée dans le réseau ainsi construit (voir [Ford et Fulkerson, 1958, Ford et Fulkerson, 1962]). De ce fait, ce réseau étendu peut être considéré comme un réseau statique pour lequel l'objectif est de maximiser le nombre de personnes indemnes qui peuvent atteindre le sommet final en partant du sommet initial. Dans la section suivante, nous étudions la complexité du problème FGED afin de connaitre ses caractéristiques.

4.2.2 Complexité du problème FGED

Notation

Tout d'abord nous introduisons quelques notations supplémentaires.

- $\{_p$: Réduction polynomiale
- $\{_{ps}$: Réduction pseudo-polynomiale
- I_Π : Une instance du problème Π
- $CERT_\Pi$: Un certificat du problème Π

— $L(I_\Pi)$: La longueur de l'instance I_Π.
— $M(I_\Pi)$: Le plus grand nombre d'une instance I_Π

Instance

Une instance du problème FGED est composée d'un graphe orienté $G = (V, E)$ avec N sommets et M arcs, d'un sommet de départ S et d'arrivée P, d'une probabilité $Pr_i \in [0..1] \quad \forall i \in V - \{S, P\}$, d'une capacité $C_{(i,j)} \in \mathbb{N}^* \quad \forall (i,j) \in E$, d'un nombre d'unités B_S partant du sommet S et d'une valeur cible minimale $B_{min} \in [1..B_S]$ devant arriver au sommet P.

Exemple

Soit une instance du problème FGED définie par la figure 4.2 et pour laquelle on essaie de faire voyager 500 personnes du sommet S vers le sommet P.

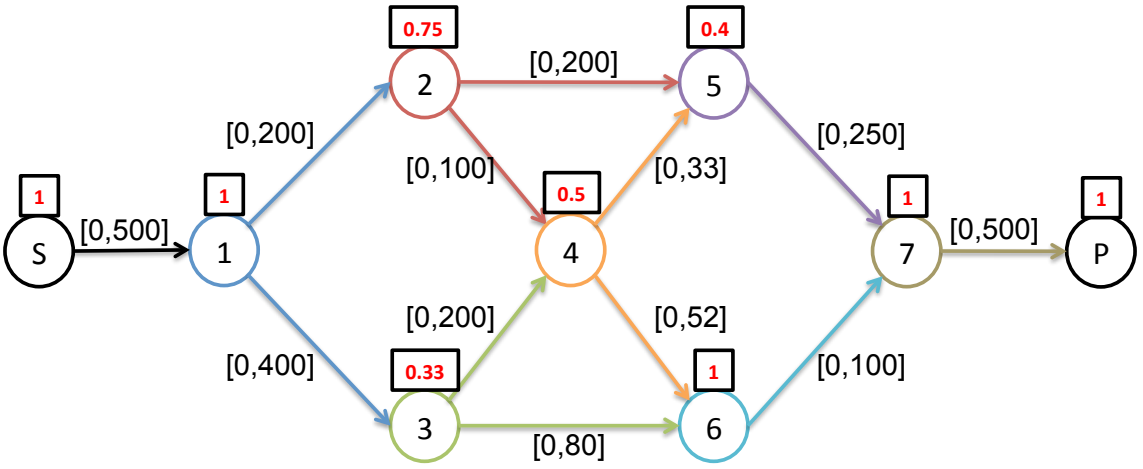


FIGURE 4.2 – Instance de graphe pour le problème FGED

Existe-t-il une fonction de flot $F : E \to \mathbb{N}$ telle que :
— $B_{min} = 400$: NON car la coupe minimale du graphe G est de 333 {(2,5),(4,5),(6,7)}.
— $B_{min} = 100$: OUI car il existe un certificat $CERT_{FGED}$ valide avec 151 unités atteignant le sommet P.

Arcs	(S,1)	(1,2)	(1,3)	(2,4)	(2,5)	(3,4)	(3,6)	(4,5)	(4,6)	(5,7)	(6,7)	(7,P)
Flot(F)	500	200	300	21	129	19	80	0	20	51	100	151

TABLE 4.1 – Exemple de certificat optimal pour le problème FGED

Propriétés

Théorème 1. *Le problème de flot généralisé entier avec diviseurs (FGED)* $\in NP$.

Démonstration. Soit $CERT_{FGED}$ un certificat associé à une instance de I_{FGED}.

1. $L(CERT_{FGED}) = M \leq N^2$ donc la longueur d'un certificat de $CERT_{FGED}$ est polynomiale.
2. L'algorithme de vérification prenant en entrée (I_{FGED},$CERT_{FGED}$) et vérifiant les 3 groupes de contraintes de FGED nécessite $O(M+N(N-1)+N) \leq O(N^2+N^2+N) = O(N^2+N)$ opérations pour répondre par OUI ou NON à la question posée. Cet l'algorithme est donc polynomial.

Sachant que la longueur du certificat et de l'algorithme de vérification de FGED sont polynomiaux, alors FGED $\in NP$.

□

Théorème 2. *Le problème de flot généralisé entier avec diviseurs (FGED) est NP-Complet.*

Démonstration. Réduction Partition $\{_p$ FGED :
Une instance de Partition est définie par un ensemble A de N entiers $S(a) \in \mathbb{N}^* \ \forall a \in A$.
La question que l'on se pose est de savoir s'il existe $A' \subset A$ tel que $\sum_{a \in A'} S(a) \geq \sum_{a \in A-A'} S(a)$.
[Karp, 1972][Karp, 1972] a montré que le problème Partition est NP-Complet. Afin de prouver la NP-Completude de FGED nous allons montrer par restriction que le problème Partition est un cas particulier de FGED et que l'algorithme de transformation de Partition vers IPDF est polynomial.
Soit le problème FGED défini par le graphe $G = (V, E)$ suivant :
$V = \{S, P, M_0, M_1, M_2, M_3, a_{11}, ..., a_{1N}, a_{21}, ..., a_{2N}, a_1, ..., a_N\}$ avec les sommets S et P des points de départ et d'arrivée, M_0 et M_3 des sommets fictifs de synchronisation, M_1, M_2 des sommets fictifs représentant les partitions A' et $A - A'$ du problème Partition , $a_{11}, ..., a_{1N}$ les sommets $\in \Gamma^+_{M_1}$, $a_{21}, ..., a_{2N}$ les sommets $\in \Gamma^+_{M_2}$, $a_1, ..., a_N$ les sommets qui représentent l'ensemble A.
$E = \{(S, M_0), (M_0, M_1), (M_0, M_2), (M_1, a_{11}), ..., (M_1, a_{1N}), (M_2, a_{21}), ..., (M_2, a_{2N}), (a_{11}, a_1), (a_{21}, a_1), ..., (a_{1N}, a_N), (a_{2N}, a_N), (a_1, M_3), ..., (a_N, M_3), (M_3, P)\}$.
On pose $B = \frac{1}{2} \sum_{a \in A} S(a)$.
Les capacités des arcs de E sont respectivement :
$C(E) = \{2B, B, B, S(a_1), ..., S(a_N), S(a_1), ..., S(a_N), 1, 1, \ ..., 1, 1, 1, ..., 1, N\}$
$Pr_i = 1 \ \forall i \in V - \{a_{11}, ..., a_{1N}, a_{21}, ..., a_{2N}\}$ et $Pr_i = \frac{1}{S(i)} \ \forall i \in \{a_{11}, ..., a_{1N}, a_{21}, ..., a_{2N}\}$
$B_S = \sum_{a \in A} S(a) = 2B$ et la cible $B_{min} = Card(A) = N$.
Au plus B_S unités passeront par les sommets M_1 ou M_2. Chaque sommet a_i avec $i \in [1..N]$ ne pourra être atteint que depuis les sommet $\in \Gamma^+_{M_1}$ ou bien par les sommets $\in \Gamma^+_{M_2}$. Au plus N sommets a_i avec $i \in [1..N]$ peuvent être traversés une seule fois.
Ainsi, trouver une fonction de flot F pour cette instance de FGED dont la valeur cible $B_{min} = N$ revient à résoudre le problème Partition qui est NP-Complet.
L'algorithme de transformation d'une instance de Partition vers une instance de FGED nécessite $O(2(3N + 6))$ opérations pour créer les sommets de V et leur probabilité de traversée, $O(2(5N + 4))$ opérations pour créer les arcs de E ainsi que leur capacités. Cet algorithme présenté ci-dessus a une complexité de $O(2(3N + 6) + 2(5N + 4)) = O(N)$ qui est polynomiale d'où Partition$\{_p$FGED.
Donc FGED est *NP-Complet*. □

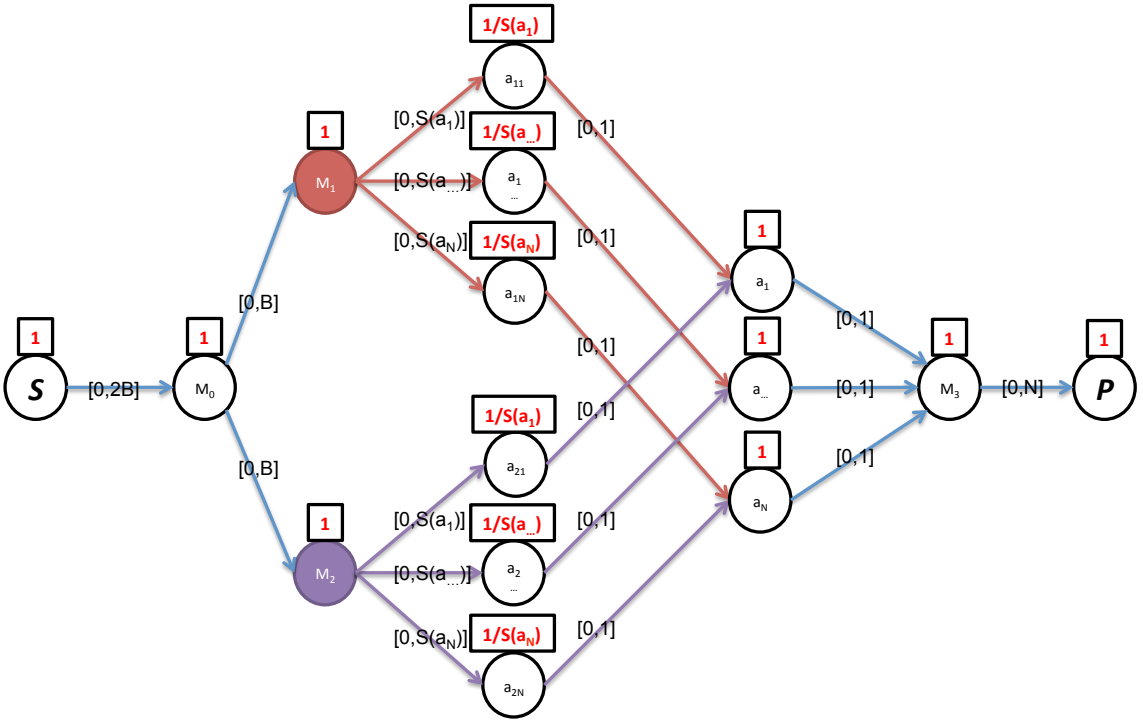


FIGURE 4.3 – Réduction polynomiale de Partition vers FGED

Théorème 3. *Le problème de flot généralisé entier avec diviseurs (FGED) est NP-Complet au sens fort.*

Démonstration. Réduction 3-Partition $\{_{ps}$ FGED
Une instance de 3-Partition est définie par une borne $B \in \mathbb{N}^*$, un ensemble A de 3N entiers $S(a)$ tels que $\forall a \in A$, $\frac{B}{4} < S(a) < \frac{B}{2}$.
La question que l'on se pose est de savoir s'il existe N sous-ensembles disjoints $A_i \subset A$ tels que $\forall i \in [1..N] \quad \sum_{a \in A_i} S(a) = B$.
[Garey et Johnson, 1975, Garey et Johnson, 1979] ont montré que le problème 3-Partition est NP-Complet au sens fort.
Afin de prouver que FGED est NP-Complet au sens fort nous allons montrer par restriction que le problème 3-Partition est un cas particulier de FGED et que l'algorithme de transformation de 3-Partition vers FGED est pseudo-polynomial.
Soit le problème FGED définit par le graphe $G = (V, E)$ suivant :
$V = \{S, P, M_0, M_1, ..., M_{N+1}, a_{1.1}, ..., a_{1.3N}, a_{2.1}, ..., a_{2.3N}, ..., a_{N.1}, ..., a_{N.3N}, a_1, ..., a_{3N}\}$ avec les sommets S et P des points de départ et d'arrivée, M_0 et M_{N+1} des sommets fictifs de synchronisation, M_i $i \in [1..N]$ des sommets fictifs représentant les partitions A_i du problème 3-Partition , $\{a_{i.1}, ..., a_{i.3N}\}$ les sommets $\in \Gamma^+_{M_i} \forall i \in [1..N]$, a_i $\forall i \in [1..N]$ les sommets qui représentent l'ensemble A.
$E = \{(S, M_0), (M_0, M_1), ..., (M_0, M_N), (M_1, a_{1.1}), ..., (M_1, a_{1.3N}), ..., (M_N, a_{N.1}), ..., (M_N, a_{N.3N}),$
$(a_{1.1}, a_1), ..., (a_{N.1}, a_1), (a_{1.2}, a_2), ..., (a_{N.2}, a_2), ..., (a_{1.3N}, a_{3N}), ..., (a_{N.3N}, a_{3N}),$

$(a_1, M_{N+1}), ..., (a_{3N}, M_{N+1}), (M_{N+1}, P)\}$.
Les capacités des arcs de E sont respectivement :
$C(E) = \{NB, B, ..., B, S(a_1), ..., S(a_{3N}), ..., S(a_1), ..., S(a_{3N}), 1, ..., 1, 1, ..., 1, ..., 1, ..., 1, 1, ..., 1, N\}$
$Pr_i = 1\ \forall i \in V - \{a_{j.1}, ..., a_{j.3N}\}$ et $Pr_i = \frac{1}{S(i)}\ \forall i \in \{a_{j.1}, ..., a_{j,3N}\}$ avec $j \in [1..N]$
$B_S = \sum_{a \in A} S(a) = NB$ et la cible $B_{min} = Card(A) = 3N$.
Au plus B_S unités passeront par chaqu'un des sommets M_i avec $i \in [1..N]$. Chaque sommet a_i avec $i \in [1..3N]$ ne pourra être atteint que depuis un seul sommet $M_j\ \ j \in [1..N]$. Au plus 3N sommets a_i avec $i \in [1..3N]$ peuvent être traversés une seule fois. Enfin au maximum 3 sommets a_i avec $i \in [1..3N]$ pourront être atteint depuis un sommet $M_j\ \ j \in [1..N]$.
Ainsi, trouver une fonction de flot F pour cette instance de FGED dont la valeur cible $B_{min} = 3N$ revient à résoudre le problème 3-Partition qui est NP-Complet au sens fort.
Soit f la fonction de transformation ci-dessous :

3-Partition	FGED
$I = (A, S)$	$G = (V, E)$
$Card(A) = 3N$	$3N^2 + N + 4\ sommets$
$S(A) = NB$	$B_S = NB$
$\forall a \in A, \frac{B}{4} < S(a) < \frac{B}{3}\ avec B \in \mathbb{N}^*$	$S(a_{i.j}) = S(a_j)\ \forall i \in [1..N], \forall j \in [1..3N]$
	$Pr_{i.j} = \frac{1}{S(a_{i.j})}\ \forall i \in [1..N], \forall j \in [1..3N]$
	$C(M_i, a_{i.j}) = a_j\ \forall i \in [1..N], \forall j \in [1..3N]$

TABLE 4.2 – Génération des données de FGED depuis 3-Partition

Soit I une instance de 3-Partition :
- $L(I) = N\lceil \log_2(B) \rceil$
- $M(I) = \max\{S(a_i)\}\ \forall i \in [1..3N]$

Soit J une instance de FGED :
- $L'(J) = Card(E) * \max\{C(i,j)\}\ \forall (i,j) \in E$
  $L'(J) \leq N^2 * \max\{C(i,j)\}\ \forall (i,j) \in E$
- $M'(J) = \max\{C(i,j)\}\ \forall (i,j) \in E$

Une fois la problème 3-partition transformé en FGED par la fonction f nous obtenons :
- $L'(f(I)) = (3N^2 + N + 4)^2 \lceil \log_2(NB) \rceil$
- $M'(f(I)) = NB$

Nous pouvons maintenant vérifier si les trois conditions particulières d'une réduction pseudo-polynomiale sont remplies :

1. $L'(f(I)) \leq 49N^4 \lceil \log_2(NB) \rceil \leq 49N^4 * N \lceil \log_2(B) \rceil \leq 49(N \lceil \log_2(B) \rceil)^5$
   $L'(f(I))$ est majorée par un polynôme en $L(I)$ et $M(I)$.
2. $L(I) \leq (3N^2 + N + 4)^2 \lceil \log_2(NB) \rceil$
   $L(I)$ est majorée par un polynôme en $L'(f(I))$.
3. $M'(f(I)) = NB \leq N * M(I) \leq L(I) * M(I)$
   $M'(f(I))$ est majorée par un polynome $L(I)$ et $M(I)$.

Les trois conditions étant vérifiées, nous pouvons dire que f est une réduction pseudo polynomiale.
Donc 3-Partition$\{_{ps}$FGED d'où le problème de flot généralisé entier avec diviseurs est NP-Complet au sens fort. □

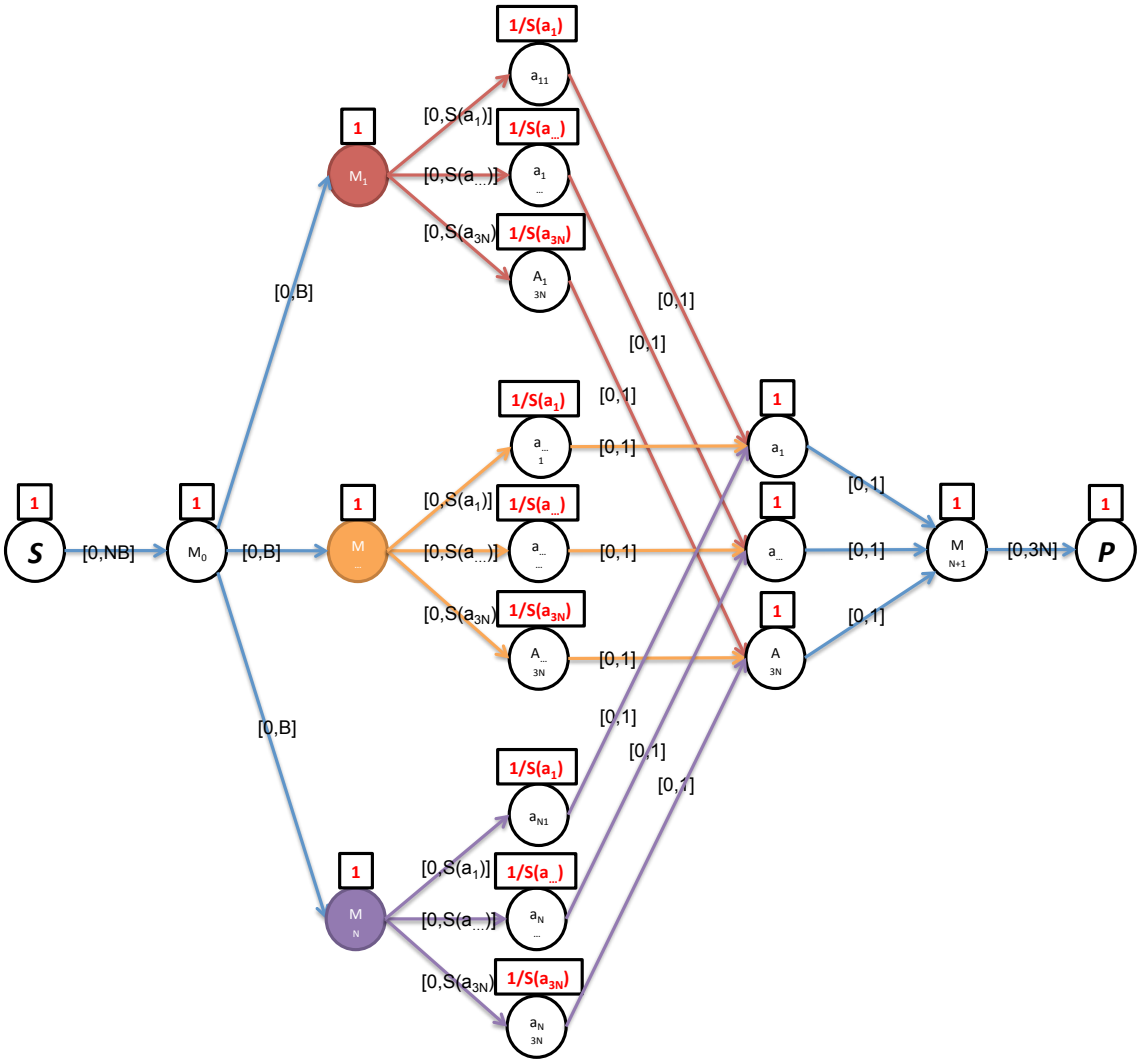


FIGURE 4.4 – Réduction pseudo-polynomiale de 3-Partition vers FGED

Du fait de la complexité du problème et de la taille des instances que nous devons résoudre dans le cadre de notre problème d'évacuation de masse, nous présentons une revue de la littérature des problèmes ayant quelques similitudes afin d'étudier les stratégies utilisées.

4.2.3 Problèmes connexes

Tout comme le problème de transbordement [Hoppe et Tardos, 2000], nous avons un problème d'évacuation où nous devons mettre en sécurité les personnes se situant sur un ensemble de points de rassemblement vers un ensemble de centres de secours ayant des capacités d'accueil. Nous avons la possibilité de prendre en compte la notion de réseau dynamique dans le temps en ajoutant des durées de trajets sur les arcs. Toutefois, contrai-

rement au problème de transbordement pour lequel toutes les personnes arriveront dans des centres de secours et où nous voulons minimiser la durée de l'évacuation, dans notre cas certains des évacués peuvent être blessés et ainsi attendre de l'aide afin de poursuivre leur périple. En outre nous ne souhaitons pas minimiser la durée de l'évacuation mais minimiser le nombre de personnes blessées durant l'évacuation (i.e. maximiser le nombre de personnes indemnes). Dans le but de prendre en compte la notion de sécurité durant l'évacuation, [Opasanon et Miller-Hooks, 2009] ont créé un problème dont le but est de trouver le plan d'évacuation le plus sûr pour l'évacué devant prendre le plus de risque. Cette approche originale permet l'évacuation de bâtiments ou de zones géographiques avec des niveaux de sécurité qui varient dans le temps. [Miller-Hooks et Stock Patterson, 2004] transforment tout problème de transbordement en un problème de "Quickest Flow" avec un super sommet source et un super sommet puits. A ce nouveau réseau, ils ajoutent des dates de départ impératives pour chaque groupe d'évacués ainsi que des niveaux de sécurité sur les arcs qui correspondent à la probabilité de traverser avec succès ces derniers. Tout comme notre problème d'évacuation, le modèle présenté n'a pas pour objectif de minimiser la durée de l'évacuation mais de maximiser la sécurité. Toutefois, contrairement au notre, leur but n'est pas de maximiser la sécurité de l'ensemble de la population mais de la personne devant prendre le plus de risques. En outre, l'algorithme proposé pour résoudre leur problème est pseudo-polynomial et peut ne pas donner de solution lorsqu'un groupe de personnes n'arrive pas à commencer son évacuation à la date de départ impérative.
Ainsi, nous pouvons noter trois différences majeures entre l'approche proposée par [Opasanon et Miller-Hooks, 2009] et notre problème d'évacuation avec pertes. En effet, dans leur cas, les niveaux de sécurité se trouvent sur les arcs, les niveaux de sécurité sont calculés en effectuant un produit des niveaux de sécurité des arcs traversés et ils ne minimisent que le risque encouru par la personne prenant le plus de risque. Les travaux de thèse effectués par [Wayne, 1999] sur les flots généralisés permettent de combler ce manque. Ce dernier a aussi développé plusieurs méthodes permettant de résoudre ce problème dans le cas général. Pour ce dernier, la formulation proposée par [Sahni, 1974] où les multiplicateurs sont sur les arcs a un certain nombre d'avantages. Cette approche est très pratique puisqu'elle permet d'établir un problème central permettant d'obtenir un plan d'évacuation macroscopique axée sur la notion de sécurité. Néanmoins, elle ne permet pas de prendre en compte le fait que le flot doit être entier comme les évacués sont indivisibles en parties voyageant indépendamment dans le réseau. Comme dans notre cas nous considérons un réseau étendu dans le temps sans arcs de durées de traversées nulles, cela veut aussi dire qu'il n'y a pas de cycles et qu'il est possible d'utiliser un graphe de niveau pour modéliser le réseau étendu dans le temps. Une telle modélisation est pratique car elle donne des réseaux sans cycles et permet ainsi d'améliorer les méthodes de résolution. Comme nous l'avons déjà montré dans la section 2.2.6.1, le fait de modéliser la sécurité sur les sommets au lieu des arcs ne change pas la nature du problème de flot généralisé. En effet, le passage d'un modèle à un autre s'effectue en ajoutant ou en retirant des sommets fictifs avec les niveaux de sécurité correspondants.
Du point de vue des méthodes de résolution possible, nous avons déjà montré que notre problème d'évacuation est $NP\text{-}Difficile$ au sens fort et n'est pas approximable. En outre, du fait de la taille des instances à prendre en compte pour effectuer une évacuation de masse, il n'est pas envisageable de considérer des méthodes de résolution exactes. Aussi,

dans la section suivante, nous présentons une métaheuristique permettant la résolution de ce dernier.

4.2.4 Algorithmes approchés

4.2.4.1 Codage d'une solution

Avant d'établir une méthode de résolution, nous devons dans un premier temps caractériser ce qu'est une solution ainsi que comment cette dernière est codée.
La figure 4.5 montre un exemple avec trois groupes de sommets. Pour chaque sommet $j \in V$ considéré, nous avons un ensemble de prédécesseurs $i_h \in V$ et un ensemble de successeurs $k_l \in V$. Aussi, du fait de la notion de pertes durant l'évacuation, la loi de conservation de flot de Kirchohoff n'est pas respectée. De ce fait, une des manières de modéliser une solution permettant de retrouver l'ensemble des individus est d'associer à chaque arc un pourcentage de répartition. Par voie de conséquence, considérant le sommet j, nous savons que $\delta\%$ des personnes indemnes sortant du sommet j iront vers le sommet k_1, $\epsilon\%$ vers le sommet k_2 et $\zeta\%$ vers le sommet k_3 avec $\delta\% + \epsilon\% + \zeta\% = 100\%$. De même iront vers le sommet j, $\alpha\%$ des personnes indemnes sortant du sommet i_1 , $\beta\%$ des personnes indemnes sortant de i_2 et $\gamma\%$ des personnes indemnes sortant de i_3. Cette architecture, permet de définir une répartition de la population sur les différents arcs durant l'évacuation mais aussi permet de retrouver ceux qui du à leurs blessures ne peuvent plus continuer à se mouvoir par leurs propres moyens. Les algorithmes que nous présenterons par la suite auront tous

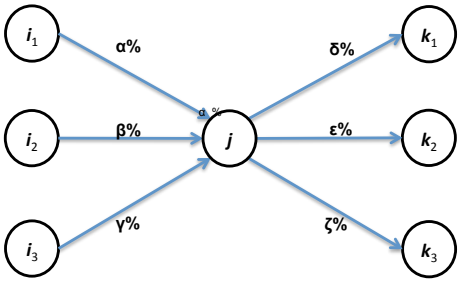


FIGURE 4.5 – Codage d'une solution du problème FGED

pour but de déterminer ces coefficients de répartition de la population. Aussi, comme il s'agira de mettre en place des approches heuristiques permettant d'établir des plans d'évacuation macroscopiques, nous utiliserons des approches basées sur la méthode sandwich (voir [Hamacher *et al.*, 2011b]) permettant d'allier les approches microscopiques et macroscopiques afin d'avoir des plans d'évacuation prenant en compte quelques caractéristiques humaines influant sur l'évacuation (voir section 2.3).

4.2.4.2 Similitudes entre les colonies de fourmis et les personnes à évacuer

L'optimisation par colonies de fourmis fait partie du domaine des métaheuristiques. En effet, contrairement aux heuristiques dont le but est de fournir une solution approchée à un problème difficile en se basant sur certaines caractéristiques du problème, les

métaheuristiques quant à elles font le parallèle entre des concepts réels ou abstraits et le problème à résoudre. Ainsi, il est possible de voir un problème d'optimisation comme un problème analogue à processus métallurgique (Recuit similé), de croisement d'individus (Algorithme génétique) ou encore trouver des veines ou niches d'or (Recherches locales). A l'image des exemples précédemment cités, il est possible de faire un parallèle entre le comportement des colonies de fourmis et le comportement de personnes en panique lors d'une évacuation de masse. Un état de l'art exhaustif des méthodes d'optimisation utilisant des colonies de fourmis peut être trouvé dans le chapitre "Fourmis Artificielles" du livre [Siarry, 2014]. D'autres contributions comme celles de [Forcael *et al.*, 2014] appliquent plus précisément l'optimisation par colonie de fourmis aux problèmes d'évacuation de masse en cas de catastrophes naturelles comme dans le cas de tsunamis. Plusieurs particularités lient ces dernières à notre problème d'évacuation avec pertes.
Tout d'abord, nous avons modélisé notre problème d'évacuation à l'aide d'un réseau étendu dans le temps ayant un seul super sommet source S et un seul super sommet puits P. Le nombre de personnes à évacuer est de B_S personnes et le nombre de personnes pouvant être accueilli est de B_P individus. Pour cela, les personnes doivent traverser des zones dangereuses et au fur et à mesure qu'ils se déplacent, ils rencontrent des personnes blessées (en attente de moyens de déplacement) leur indiquant si oui ou non le chemin qu'ils empruntent est sûr. Nous pouvons considérer la même configuration dans le cas d'une colonie de fourmis. En effet, à chaque colonie de fourmis est associée un nid (une fourmilière) où vivent les individus (i.e. B_S) fourmis. Ces dernières devant se nourrir, nous pouvons considérer qu'elles doivent quitter le nid afin de trouver une source de nourriture qui est disponible en au plus B_P unités au sommet P. Pour y parvenir, ils doivent explorer dans un premier temps leur environnement qui est rempli de dangers (prédateurs naturels, flaques d'eau, voies de circulations, ...). Cette exploration est limitée par la vue des évacués mais aussi des fourmis. En effet, aussi bien chez l'être humain que chez la fourmi, les décisions pour se déplacer se fondent sur les informations que l'on peut percevoir avec les différents sens comme la vue ou l'odorat. De ce fait, si des fourmis surmontent tous ces dangers et trouvent la source de nourriture alors elles déposent des phéromones sur les chemins empruntés afin que les autres puissent aussi les suivre. Ces échanges d'informations aussi bien entre les personnes qu'entre les fourmis permettent à ceux qui ont réussi à atteindre leur objectif de faire bénéficier aux autres de leurs expériences. Il est possible d'énumérer bon nombre de ressemblances entre l'homme et la fourmi comme les différences de corpulences ou leur rôles (reine, soldat, ouvrière,...) mais nous nous limiterons aux aspects permettant de résoudre le FGED. Dans la section 4.2.4.4 nous présentons une métaheuristique se basant sur l'optimisation par colonie de fourmis pour résoudre notre problème d'évacuation. Toutefois un plan d'évacuation basé sur une approche par colonie de fourmis mène à un plan d'évacuation microscopique. Dans la section suivante, nous montrons comment convertir un tel plan d'évacuation en un plan d'évacuation macroscopique.

4.2.4.3 Transformation d'une évacuation microscopique en macroscopique

Dans le cadre d'une optimisation par colonie de fourmis, chaque fourmi correspond à la représentation d'un homme devant voyager du super sommet source S au super sommet puits P. Ayant ses propres attributs, cette dernière doit emprunter une succession d'arcs et

traverser un certain nombre de sommets avec des niveaux de sécurité afin d'atteindre son objectif. Aussi, à chaque fourmi ayant atteint la source de nourriture (personne indemne) et à chaque fourmi ne pouvant plus se mouvoir (i.e. personne blessée) est associée un chemin allant du super sommet source S à leur position courante. L'agrégation de l'ensemble des chemins utilisés respectant les capacités maximales des arcs nous permet de déduire le schéma de répartition des fourmis dans le réseau considéré. Du fait que dans le cas d'une évacuation nous utilisons un réseau étendu dans le temps ne comprenant plus de cycles, ce dernier peut être transformé en graphe de niveaux. Dans chaque niveau se trouve un ensemble de sommets $j \in V$ ayant des prédécesseurs se trouvant dans les niveaux inférieurs et des successeurs se trouvant dans les niveaux supérieurs (voir figure 4.5). Par voie de conséquence, la validation des contraintes définies dans le modèle mathématique du problème FGED (voir 4.2.1.1) des niveaux inférieurs aux niveaux supérieurs du graphe de niveaux, garantie que la solution proposée est réalisable et qu'elle correspond à un modèle de flot macroscopique.
Dans la section suivante, nous présentons l'algorithme de colonie de fourmis permettant de déterminer la répartition des fourmis.

4.2.4.4 Algorithme de colonies de fourmis

Comme la plupart des métaheuristiques, les algorithmes basés sur les fourmis disposent d'un certain nombre de paramètres permettant de les configurer en fonction du problème que l'on veut traiter. Nous présentons ces derniers et soulignons leurs correspondances pour notre problème d'évacuation.

- Influence des phéromones : Lorsque les fourmis déposent des phéromones sur leur passage, ces derniers sont perçus par les fourmis suivantes et les incitent à suivre le même chemin. Cette incitation est assimilable à l'attractivité liée à un chemin plutôt qu'à un autre. De ce fait, plus un chemin est attractif, plus il aura de chance d'être emprunté.
- Évaporation : Le dépôt de phéromones est effectué par des fourmis ayant réussi à atteindre le sommet puits P ou ayant subi des dommages. Ainsi, plus un chemin sera utilisé, plus son niveau de phéromone augmentera et plus il sera attractif. Dans le cas contraire, plus il passera de temps sans que personnes ne l'utilise moins il restera des traces de passages et moins il sera attractif. L'évaporation étant un phéromone qui est fonction du temps, elle affecte l'ensemble des arcs du réseau.
- Influence de la visibilité : La visibilité permet de déterminer un sous graphe du réseau étendu dont toutes les informations sont connues par les fourmis. Cette visibilité restreint les choix de déplacement des fourmis à l'environnement qui les entourent. De ce fait, elles découvrent au fur et à mesure qu'elles se déplacent des parties du réseau.
- Valeur initiale des phéromones : Elle permet de guider la recherche initiale de solutions lors du processus d'optimisation. Si les valeurs de phéromones sont identiques, alors les fourmis ne seront pas influencées dans le cas contraire, plus le niveau de phéromones sur un arc sera élevé plus il sera préféré aux autres.
- Nombre de fourmis : Ce nombre permet de s'assurer de la pertinence d'une solution. Plus le nombre de fourmis est élevé plus il y aura de chance que des solutions

exotiques mais efficaces soient explorées.
— Borne inférieure des phéromones : Permet de s'assurer que le niveau de phéromone d'un arc ne diminue pas en dessous d'un certain seuil en supposant qu'il restera toujours des résidus. Cela permet aussi de s'assurer que des solutions ou arcs non explorés depuis un certain moment ne soient pas exclus de la recherche. Ainsi, la probabilité de son exploration est non nulle.
— Exploitation/exploration (intensification/diversification) : Permettent de diversifier l'espace de recherche après un certain nombre d'itérations sans amélioration mais aussi d'améliorer les solutions courantes.

Afin de résoudre ce problème d'évacuation, nous utilisons une métaheuristique utilisant les colonies de fourmis comme décrit dans la section 4.2.4.4. Par defaut, les phéromones se trouvant sur les arcs sont initialisés avec la même valeur (i.e. 1.0). Dans le but d'avoir de meilleures performances, nous avons aussi développé deux heuristiques $H3$ et $H4$ permettant d'avoir un niveau de phéromones sur les arcs plus pertinents pour l'algorithme d'optimisation par colonies de fourmis. Ces derniers sont décrites dans les sections 4.2.4.5 et 4.2.4.6.

Algorithme : Colonies de fourmis

Donnée(s) :
G : Instance du problème FGED.
B_S : Nombre de personne à évacuer.
$NbVieFourmi$: Nombre de vies pour chaque fourmi.
$NBColonies$: Nombre de colonies de fourmis.
$Pheromones$: Quantité de phéromone.
Début :
$ColoniesDeFourmies \leftarrow CreerColonies(NBColonies, B, NbVieFourmi)$
Pour chaque colonie $\in ColoniesDeFourmies$ **Faire**
  Pour chaque Fourmi $\in colonie$ **Faire**
    $Fourmi.AjouterPosition(S)$
      Tant que $Fourmi.EstEnVie()$ et $Fourmi.PositionCourante() \neq P$ **Faire**
        $PositionSuivante \leftarrow G.RecuperationAleatoireDunDesSucesseursVisibles(Fourmi.RecupererPositionCourante())$
        Si $G.Capacity(Fourmi.RecupererPositionCourante(),\ PositionSuivante) > 0$ **Alors**
          $Fourmi.AjouterPosition(PositionSuivante)$
          $Fourmi.DiminuerLeNombreDeVie(FinDeVieAleatoire(G.Safety(Fourmi.RecupererPositionCourante())))$
        Sinon
          $G.DiminuerLeNiveauDePheromone(Fourmi.RecupererChemin())$
          $Fourmi.DiminuerLeNombreDeVies(1)$

18) $Fourmi.EffacerLeChemin()$
19) $Fourmi.AjouterPosition(S)$
20) **Fin Si**
21) **Fin Tant que**
22) **Si** $Fourmi.EstEnVie()$ **Alors**
23) $G.DiminuerCapacite(Fourmi.RecupererChemin())$
24) $G.AugementerLesPhetomones(Fourmi.RecupererChemin())$
25) **Fin Si**
26) $G.MiseAJourDeLaVisibilite(Fourmi.RecupererChemin())$
27) **Fin Pour chaque**
28) $G.SauvegarderLaSolutionCouranteNormalisee()$
29) $G.ReinitialiserLesCapacitesDesArcs()$
30) **Fin Pour chaque**

Fin :
Sortie(s) :

31) $G.MeilleurPlanDevacuation$: Le plan d'évacuation permettant de maximiser le nombre de personnes indemnes et autonomes

L'algorithme de colonies de fourmis prend en données d'entrée un graphe G (respectivement un graphe étendu dans le temps) pour lequel chaque sommet a un niveau de sécurité et où à chaque arc est associé une capacité et un niveau de phéromone. Nous avons aussi le nombre total de personnes à évacuer B_S qui correspond au nombre total d'individus dans chaque colonie de fourmis d'une génération. A chaque fourmi, nous associons un nombre de vies qui correspond au nombre de chance de ressusciter que nous donnons à cette dernière lorsqu'elle a pris une mauvaise décision menant à sa mort ou à son immobilisation. Nous avons aussi comme donnée la variable $NBColonies$ qui correspond au nombre de génération sur laquelle portera notre étude en sachant que chaque génération de fourmis associée à une colonie transmettra à la génération suivante son expérience de la zone à évacuer à travers les phéromones laissés sur les arcs du graphe. Ce dépôt de phéromones positif ou négatif est effectué par chaque fourmi appartenant à une colonie (membre d'une génération). Cette dernière met à jour le niveau de pheromones sur les arcs qu'elle a eu à emprunter durant son périple du super sommet source S vers le super sommet puits P afin d'aider les générations suivantes à faire au moins aussi bien qu'elle.
Une des premières étapes de l'algorithme est de créer un ensemble de colonies de fourmis. Par la suite, pour chaque colonie et pour chaque fourmi membre d'une colonie nous définissons comme point de départ le super sommet source S. Ainsi tant que la fourmi courante considérée n'a pas épuisé toutes ses chances (vies) et qu'elle n'est pas sur le super sommet puits P, cette dernière doit se déplacer vers l'un des successeurs de sa position courante. Chaque successeur est plus ou moins visible que les autres en fonction de son niveau de sécurité, du niveau de phéromone sur l'arc y menant, et du chemin le plus sûr (voir [Opasanon et Miller-Hooks, 2009]) disponible allant du successeur vers le super sommet puits P. Dans cet algorithme les niveaux de visibilité des successeurs d'un sommet sont normalisés pour que la valeur soit uniformément répartie entre 0,0 et 100,0. Ainsi en tirant un nombre aléatoire entre 0,0 et 100,0, si ce dernier tombe dans la plage de définition d'un des successeurs, alors ce dernier est choisi et la fourmi se déplace vers ce dernier. Ainsi, si la

capacité résiduelle de l'arc menant de la position courante à la position suivante le permet, alors la fourmi s'y rend. Une fois sur le sommet suivant, on tire un pourcentage aléatoire entre 0.0% et 100.0% qui définit là où se situe les chances de survie de la fourmi confrontée aux danger du sommet sur lequel elle se trouve. Aussi, si le pourcentage tiré au hasard est entre 0.0 et le niveau de sécurité du sommet courant alors la fourmi reste en vie dans le cas contraire, on la considère comme morte et nous diminuons ses nombres de vies. Nous diminuons aussi le nombre de vies d'une fourmi si cette dernière n'arrive plus à se déplacer à cause de la capacité résiduelle des arcs qui s'offrent à elle. Dans les deux cas, cette dernière est replacée au sommet de départ S. Nous continuons ce processus jusqu'à ce que la fourmi ne dispose plus de vies ou qu'elle atteigne le super sommet puits P. Quand c'est le cas, si elle est toujours vivante, ce qui veut dire qu'elle est arrivée au sommet final P, nous augmentons le niveau de phéromones sur le chemin qu'elle a emprunté pour atteindre ce dernier en partant du super sommet S et nous décrémentons la capacité résiduelle d'une unité sur ce chemin. Dans le cas contraire, si elle n'a pas survécu jusqu'à la fin de son périple, nous décrémentons le niveau de phéromone sur le chemin ayant mené à sa disparition afin que ses congénères soient moins enclin à choisir le même chemin dangereux. Cette diminution du niveau de phéromone sur le chemin se fait de manière croissante avec une faible diminution sur les arcs proches du super sommet S et une forte diminution sur les arcs proches du super sommet P. Quand tous les membres d'une colonie ont été envoyés dans le graphe, nous utilisons un graphe de niveau afin de normaliser la solution obtenue et de s'assurer que l'ensemble des contraintes du modèle ont été satisfaites. La solution finale est alors sauvegardée et les capacités sur les arcs sont réinitialisées pour la prochaine génération de colonie de fourmis. A la fin de l'algorithme, nous normalisons la meilleure solution en terme de sécurité et nous conservons cette dernière comme plan d'évacuation. Dans les sections suivantes nous présentons les heuristiques permettant d'initialiser les phéromones sur les arcs avant l'utilisation de l'algorithme de colonie de fourmis.

4.2.4.5 Heuristique H3

L'heuristique H3 est basée sur la même approche que celle utilisé par le "Fat-Path" soit le chemin le plus large dévelopé par [Wayne, 1999]. Cette dernière a pour but d'envoyer le plus d'individus possible sur les arcs permettant aux plus de personnes de continuer leurs périples. Cette procédure est décrite ci-dessous :

1. Soit G un graphe sans cycle dont nous avons construit le graphe de niveau correspondant.
2. Nous débutons en nous positionnant dans le niveau 0 qui ne contient que le super sommet source S.
3. Pour chaque sommet se trouvant dans le niveau courant, nous trions ses successeurs suivant le nombre de personnes pouvant les traverser et poursuivre leur évacuation.
4. Nous envoyons autant de personnes que possible sur le meilleur successeur puis le second meilleur et ainsi de suite jusqu'à ce qu'il ne reste plus personnes à faire passer ou que les capacités résiduelles ne le permettent pas.
5. Lorsque tous les sommets appartenant à un niveau ont été propagés, nous passons alors vers le niveau suivant

6. Si le niveau courant contient le super sommet puits P alors nous arrêtons l'algorithme. dans le cas contraire, nous continuons le processus en continuant l'algorithme depuis l'étape 3.

Cependant, cette heuristique ne prend pas en compte le fait que les chemins construits au fur et à mesure peuvent devenir de moins en moins larges quand ils s'approche du super sommet puits P et de ce fait le "fat-path" [Wayne, 1999] peut devenir dangereux en générant des points critiques d'engorgement. La figure 4.6 définit une instance du problème FGED pour laquelle 4 personnes doivent être évacuées depuis le point de rassemblement S_1 vers le centre de secours P_1. Dans cet exemple 2 chemins sont utilisables , $\Phi_1 = (S, S_1, 1, 2, P_1, P)$ avec un niveau de sécurité de 0.25 (Respectivement $1\times1\times1\times0.25\times1\times1$) et $\Phi_2 = (S, S_1, 3, 4, P_1, P)$ avec un niveau de sécurité de 0.75 (Respectivement $1 \times 1 \times 0.75 \times 1 \times 1 \times 1$).

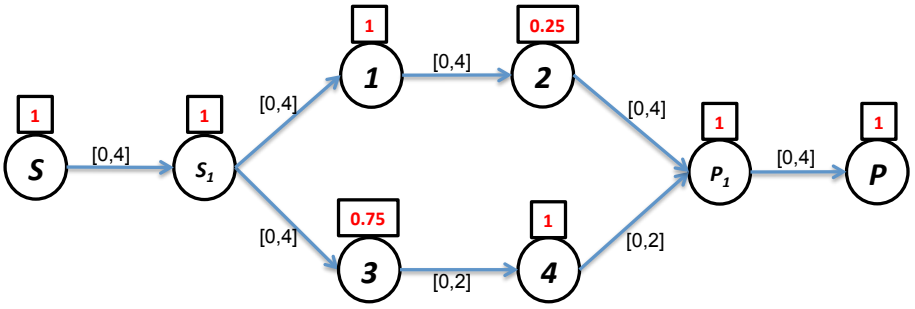


FIGURE 4.6 – Exemple d'instance pour laquelle H3 ne trouve pas de bonnes solutions

En utilisant l'heuristique H3, nous commençons au super sommet S. Comme le sommet S_1 est son seul successeur, toutes les personnes sortant du sommet S essaieront de s'y rendre. Comme le sommet S_1 dispose de 2 successeurs, le sommet 1 qui permet à 4 évacués de poursuivre leurs déplacements et le sommet 3 qui ne permet qu'à 2 de continuer. Ainsi les 4 personnes à évacuer sont envoyées au sommet 1 parmi lesquels seulement une seule personne pourra continuer et se rendre au sommet 2 dans le but t'atteindre P_1. Aussi, avec le chemin Φ_1, seulement une seule personne peut atteindre le super sommet puits P alors que 2 personnes auraient pu s'y rendre par leurs propres moyens en étant indemnes si nous avions envoyé toutes les personnes sur le chemin Φ_2. La principale insuffisance de cette heuristique est le fait de ne pas prendre en compte le niveau de sécurité associé au graphe. Nous présentons dans la partie suivante une heuristique qui comble ce manquement.

4.2.4.6 Heuristique H4

Heuristique H4 est basée sur l'approche proposée dans le cadre du problème d'évacuation "Safest Escape Problem" développé [Opasanon et Miller-Hooks, 2009] pour lequel les chemins les plus sûr sont obtenus en multipliant la sécurité de chaque arc traversé (respectivement chaque sommet dans notre cas). Les étapes qui décrivent l'heuristique sont détaillées ci-dessous :

1. Déterminer le chemin le plus sûr Φ allant du super sommet source S au super

sommet puits P.

2. Nous envoyons autant de personnes que possible sur le chemin ainsi déterminé en se basant sur la capacité résiduelle associée au chemin.
3. Si durant cette recherche de chemin, nous atteignons un sommet qui ne permet plus de continuer l'évacuation à cause d'un défaut de capacité résiduelle sur ses arcs sortants alors nous supprimons (désactivons) ce sommet du graphe ainsi que tous les arcs y menant.
4. Nous itérons à l'étape 1 jusqu'à ce qu'il ne reste plus personne à évacuer ou qu'il n'y ait plus de chemins permettant d'aller du sommet S au sommet P.

Toutefois, lorsqu'il s'agit de résoudre le problème d'évacuation FGED, cette approche ne permet pas de prendre en compte le fait que les chemins longs et réputés sûrs peuvent être plus risqués que ceux étant courts mais relativement dangereux. La figure 4.7 définit une instance du problème FGED pour laquelle 4 personnes doivent être évacuées depuis le point de rassemblement S_1 afin d'atteindre le centre de secours P_1. Dans cet exemple, 2 chemins sont disponibles à savoir $\Phi_1 = (S, S_1, 1, 2, 3, P_1, P)$ avec un niveau de sécurité de 0.970299 (respectivement $1 \times 1 \times 0.99 \times 0.99 \times 0.99 \times 1 \times 1$) et $\Phi_2 = (S, S_1, 4, P_1, P)$ avec un niveau de sécurité de 0.50 (respectivement $1 \times 1 \times 0.50 \times 1 \times 1$).
Aussi, seulement 1 personne pourra atteindre le super sommet puits en étant indemne

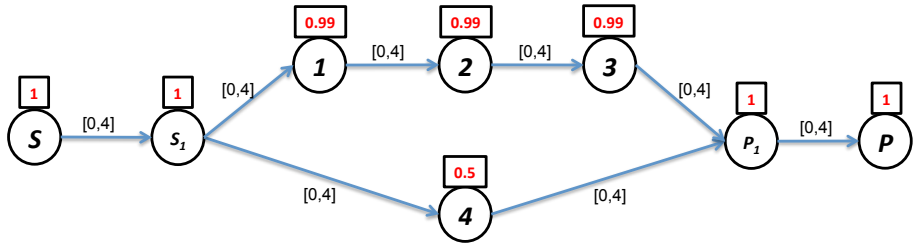


FIGURE 4.7 – Exemple d'instance pour laquelle H4 ne trouve pas de bonnes solutions

si nous prenons le chemin Φ_1 car $F(S, S_1) = 4$, $F(S_1, 1) = 4$, $F(1, 2) = 3$, $F(2, 3) = 2$, $F(3, P_1) = 1$ et $F(P_1, P) = 1$ alors que 2 personnes auraient pu être sauvées évacuées en empruntant le chemin Φ_2 avec $F(S, S_1) = 4$, $F(S_1, 4) = 4$, $F(4, P_1) = 2$ et $F(P_1, P) = 2$.

4.2.5 Résultats expérimentaux

Avant de pouvoir utiliser notre métaheuristique se basant sur l'optimisation par colonies de fourmis sur des instances de grande taille, nous avons effectué des campagnes de tests dont nous décrivons les paramètres ci-dessous :

4.2.5.1 Environment

Nous avons implémenté l'algorithme de résolution du problème FGED en utilisant le langage C++ avec comme compilateur gcc v. 4.8.2 avec l'option -O3. Toutes les expérimentations ont été effectuées sur une machine virtuelle "VirtualBox" utilisant un des

processeurs d'un Intel Core i7-3615QM cadencé à 2.30Ghz. Il avait 6Mo de mémoire cache et 4Go de mémoire RAM avec une fréquence de 1600Mhz. Le système d'exploitation utilisé est le Ubuntu 12.04 LTS x64.

4.2.5.2 Paramétrage

Avant d'appliquer nos heuristiques à des instances de taille réelle, nous évaluons leurs performances sur des instances de petite taille. Pour cela, nous avons implémenté le modèle mathématique de ce problème d'évacuation en utilisant le solveur CPLEX 12.5 afin d'obtenir la solution optimale correspondant au nombre maximum de personnes indemnes sauvées. Comme instances du problème d'évacuation, nous considérons des réseaux déjà étendus dans le temps. Une fois étendu, nous avons 4 classes de graphe de 30, 60, 90 et 120 sommets. Plus d'informations à propos de ces graphes peuvent être trouvées dans le tableau 4.3. La première colonne de ce tableau précise le nombre de sommets, la seconde le nombre d'instances générées pour chaque classe de graphe, la troisième colonne donne une idée du niveau de sécurité moyen d'une classe de graphe alors que la dernière et quatrième colonne permet de mesurer la dispersion à travers écart-type.

#NbSommets	#Instances	Sécurité moyenne des sommets	Ecart-type niveau de sécurité
30	480	0,641	0,135
60	480	0,731	0,123
90	480	0,757	0,160
120	600	0,708	0,137

TABLE 4.3 – Classes de graphe pour tester les performances des heuristiques

En outre, pour tous les algorithmes et les modèles mathématiques (résolus avec CPLEX), nous avons fixé une limite de temps de 20 minutes comme durée d'exécution ainsi qu'un espace mémoire utilisable de 1Go. Les temps d'exécution sont donnés en millisecondes pour toutes les instances de paramétrage alors que pour les instances de taille réelle ils sont donnés en seconde.

4.2.5.3 Résultats de tests

La difficulté à résoudre en pratique de manière exacte ce problème d'évacuation peut être constatée par les résultats du tableau 4.4 obtenus sur de très petites instances ayant les caractéristiques définies dans le tableau 4.3. Nous résolvons dans un premier temps le programme linéaire en nombres entiers du problème d'évacuation avec le solveur CPLEX. Les résultats obtenus montrent que le temps moyen de résolution augmente avec le nombre de sommets de l'instance. Il dépend aussi de la sécurité moyenne des sommets qui plus elle est faible moins il est facile pour le solveur de trouver une solution et de prouver qu'elle est optimale. Ceci peut être constaté en considérant les instances de la classe de graphe G_{60} qui ont un niveau de sécurité moyen de $0,731$ et un temps de résolution moyen de 39226

millisecondes alors que dans le même temps les graphes de la classe G_{90} ont un niveau de sécurité moyen de $0,757$ et un temps de résolution moyen de 34 secondes. La dernière colonne du tableau 4.4 montre que le pourcentage d'instances qui peuvent être résolues à l'optimalité décroit quand la taille du graphe augmente ce qui montre qu'il n'est pas viable de résoudre à l'optimalité des instances réelles de grande taille pour ce problème d'évacuation maximisant le nombre de personnes indemnes au sens strict.

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	% Résolues
G_{30}	3.308	12.061	913.611	66.225	98,5
G_{60}	3.214	39.226	1.032.699	143.537	87,9
G_{90}	3.322	34.300	1.146.259	117.783	72,7
G_{120}	2.785	62.207	1.106.551	182.341	64,8

TABLE 4.4 – Résultats de tests CPLEX avec le programme linéaire en nombres entiers (IP)

Nous avons aussi considéré le programme linéaire en nombres réels (LP) en relaxant la contrainte d'intégrité du flot (i.e. $F_{(i,j)} \in \mathbb{R}^+$). La relaxation du modèle mathématique initial permet d'avoir une borne supérieure correspondant au nombre fractionnaire de personnes indemnes à la fin de l'évacuation. La déviation entre le programme linéaire en nombres entiers (IP) et le programme linéaire en nombres réels (PL) est donné par la formule suivante $\frac{LP-IP}{\max(1,LP)}$. Les résultats de cette expérimentation sont présentés dans le tableau 4.5. Toutes les instances ont pu être résolues en moins de 3.736 millisecondes ce qui est relativement rapide comparé aux instantes résolues par le programme linéaire en nombres entiers. Cependant, lorsque la taille du graphe augmente, la déviation entre les solutions obtenues par le programme linéaire en nombres entiers et le programme linéaire en nombres réels tend à grandir jusqu'à atteindre une valeur moyenne de $18,1\%$.

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	%Résolues	% de déviation de la solution
G_{30}	2.015	2.246	3.736	109	100	9,2
G_{60}	2.068	2.259	2.656	63	100	10,7
G_{90}	2.138	2.357	2.671	72	100	9,4
G_{120}	2.213	2.590	3.293	251	100	18,1

TABLE 4.5 – Résultats de tests CPLEX avec le programme linéaire en nombres réels (LP)

Tout comme le montre les résultats expérimentaux il est impossible d'utiliser les approches exactes sur des données de grande taille étendues dans le temps. Ce problème relatif à la taille des instances est résolu grâce aux approches heuristiques. Ainsi, pour palier ce problème, nous présentons deux approches heuristiques qui permettent d'obtenir des bornes inférieures du nombre de personnes indemnes pour une date de fin d'évacuation fixée. La déviation de nos solutions heuristiques comparée au programme linéaire en

nombres entiers (IP) est donnée par la formule $\frac{IP-Heuristic_x}{\max(1,IP)}$. Les résultats de nos expérimentations sont détaillés dans le tableau 4.6 qui présente les performances obtenues en utilisant la version classique de l'algorithme d'optimisation par colonies de fourmis. Dans ce premier cas de figure, nous sommes partis d'un niveau de phéromones identique sur l'ensemble des arcs (i.e. 1.0 sur chaque arc). Cette approche à pour but de voir comment se comporte les fourmis lorsqu'elles n'ont pas d'information initiale leur permettant d'explorer efficacement le réseau en vue de trouver des centres de secours. Toutes les instances ont pu être résolues en moins de 1.438 millisecondes, ce qui est plus rapide que la résolution exacte par le solveur CPLEX du modèle relâché défini par le programme linéaire en nombres réels. Aussi tout comme pour les modèles de résolution exacte, nous constatons une augmentation de la durée moyenne de résolution en fonction de la taille des graphes traités. Toutefois cette augmentation est minime et dans le cas des graphes de la classe G_{120} la durée moyenne de résolution est inférieure à la seconde.
Toutefois, comme l'initialisation des phéromones sur les arcs n'a pas été orientée afin de ne pas influencer l'exploration de l'espace des solutions possibles, les solutions moyennes en terme de personnes indemnes atteignant les centres de secours sont 22% plus mauvaises que les solutions optimales obtenues en résolvant le programme linéaire en nombres entiers avec le solveur CPLEX. Nous pouvons aussi noter que la déviation moyenne par rapport aux solutions optimales demeure stable et qu'elle ne diverge pas lorsque l'on augmente la taille des instances des graphes de la classe G_{30} aux graphes de la classe G_{120}.
Dans le but d'améliorer la qualité des solutions obtenues, nous allons quelque peu influencer l'exploration de l'espace des solutions en initialisant le niveau de phéromones à l'aide des heuristiques H3 et H4. Ainsi les fourmis évacuées à l'aide de ces deux heuristiques peuvent être considérées comme des exploratrices laissant des indications pour la véritable population à évacuer. Comme nous avons déjà pu le montrer dans les sections 4.2.4.5 et 4.2.4.6, chacune de ces heuristiques a des avantages et des inconvénients. Nous effectuons donc des expérimentations sur les deux en utilisant les mêmes jeux de données précédemment définis afin d'évaluer leurs performances. Comme le montrent ces dernières, il nécessite en moyenne 0,239 milliseconde pour trouver une solution avec l'heuristique d'initialisation de phéromones H3 alors que cette durée est en moyenne de 1,112 millisecondes pour l'heuristique H4. Cependant, même si tous les deux résolvent l'ensemble des instances, l'heuristique H4 est en moyenne significativement meilleure que H3 car elle donne des solutions initiales qui sont à $11,37\%$ des solutions optimales obtenues grâce au solveur CPLEX alors que dans le même temps l'heuristique H3 donne des solutions qui dévient en moyenne de $28,82\%$ des meilleures solutions.

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	%Résolues	% de déviation de la solution
G_{30}	14	42	86	15	100	21,2
G_{60}	61	195	349	73	100	22,0
G_{90}	164	441	815	169	100	22,3
G_{120}	254	731	1.438	287	100	22,2

TABLE 4.6 – Résultat de tests de l'algorithme de colonies de fourmis

Taille du graphe	Durée moyenne	%Résolues	% de déviation de la solution
G_{30}	0,07	100	29,0
G_{60}	0,13	100	30,0
G_{90}	0,18	100	27,5
G_{120}	0,23	100	28,8

TABLE 4.7 – Résultats de tests de l'heuristique H3

Taille du graphe	Durée moyenne	%Résolues	% de déviation de la solution
G_{30}	0,24	100	11,1
G_{60}	0,51	100	11,6
G_{90}	0,76	100	9,6
G_{120}	1,12	100	13,0

TABLE 4.8 – Résultats de tests de l'heuristique H4

Comme nous avons pu le voir, l'heuristique d'initialisation H4 est meilleure que l'heuristique H3 sur nos instances de tests. Nous allons donc combiner l'heuristique H4 avec notre algorithme de résolution par colonies de fourmis. Les résultats de cette combinaison sont présentés dans le tableau 4.9. L'une des premières choses que nous pouvons noter est une petite augmentation de la durée moyenne de résolution des instances qui est due au temps nécessaire de détermination du niveau de phéromone initial. Toutefois, nous pouvons noter une amélioration significative en terme de personnes indemnes. En effet, la déviation de cette dernière par rapport à l'approche exacte n'est plus que de $4,46\%$. L'autre point important est que cette déviation par rapport aux solutions optimales reste stable lorsque l'on agrandit la taille du graphe et ainsi celle des instances à résoudre en passant des graphes de classe G_{30} aux graphes de classe G_{120}. Aussi, les expérimentations laissent penser que lors du passage à l'échelle avec des instances de taille réelle, notre méthode ne divergera pas.

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	%Résolues	% de déviation de la solution
G_{30}	0,24	46	155	38	100	4,4
G_{60}	0,61	341	837	189	100	4,7
G_{90}	1,76	890	2.032	457	100	4,0
G_{120}	1,12	1.451	13.058	726	100	4,7

TABLE 4.9 – Résultats de tests de l'heuristique H4 associée à l'algorithme de colonies de fourmis

4.2.5.4 Mise à l'échelle : Instances de taille réelle

Un des principale obstacle à la mise à l'échelle de la méthode de résolution par l'utilisation des algorithmes de fourmis est le temps passé à faire des opérations inhérentes à la méthode. En effet, si nous prenons par exemple la gestion des phéromones, ces derniers doivent être mis à jour (i.e. évaporation) après le routage fructueux ou malheureux de chaque génération de fourmis. Or cette mise à jour doit intervenir sur l'ensemble du graphe dynamique qui au fur et à mesure que le temps passe tend à être pseudo-polynomial en taille. Aussi, cette mise à jour intégrale du graphe dynamique permet de s'assurer de la consistance des données mais est toutefois très couteuse en temps de calcul sans pour autant être pertinent au regard de la fourmi que nous cherchons à router du nid (sommet source) à la nourriture (sommet puits). Aussi, pour optimiser le processus de mise à jour des phéromones, nous considérons que les données sont partiellement consistantes et ne le deviennent véritablement que lorsqu'elles doivent être utilisées lors du routage d'une fourmi. Pour y parvenir, nous ajoutons un attribut supplémentaire à chaque arc permettant de connaitre la dernière date à laquelle celui-ci avait était mis à jour. Ainsi, lorsqu'une fourmi cherchera à se déplacer de sa position courante vers un des sommets adjacents à une date donnée, nous procédons au rattrapage de mise à jour des phéromones associés aux arcs avant qu'elle ne puisse faire son choix de destination.
Dans le même esprit, d'autres optimisations sont effectuées afin d'éviter des mises à jour globales du réseau dynamique et de favoriser une actualisation des données lorsque nous en avons besoin.
Afin d'évaluer les performances de calcul sur des données de taille réelle, nous avons utilisé les mêmes 8 instances de la ville de Nice présentées dans la section 3.2. La différence principale, se situe au niveau des sommets qui permettent de modéliser la notion de sécurité. La note de sécurité provenant du plaquage des niveaux de dommages induis par le sinistre correspondant au scénario-catastrophe sur le réseau routier. A chaque instance est associée la durée minimale nécessaire pour évacuer l'ensemble de la population pour chaque type de scénario (voir tableau 3.4). Afin de percevoir le gain potentiel en nombre de personnes indemnes si nous nous donnons plus de temps, nous augmenterons cet horizon de planification par pas de 100 unités de temps jusqu'à atteindre une augmentation de 1000 unités par rapport à la durée minimale d'évacuation. Afin d'initialiser le niveau de phéromones dans le réseau en suivant la politique d'évacuation préconisée par l'heuristique H4, nous nous referons aux solutions d'affectations présentées dans la section 3.3.7. En effet l'approche heuristique permet de prendre en compte la notion de durée dans la recherche de chaînes augmentantes de sécurité maximale tout en évitant de construire l'intégralité du réseau étendu dans le temps. Sur la base de cette première répartition de la population que nous considérons comme une donnée d'entrée, nous utilisons notre algorithme de colonies de fourmis. Pour ce dernier, nous considérerons 100 générations de fourmis chacune se basant sur les choix de la génération précédente pour faire mieux.
Ces campagnes de tests sont présentées dans les tableaux 4.10 - 4.17 qui représentent les solutions non dominées (i.e. Personnes indemnes max) pour chaque jeux de paramètres et chaque instance.

Comme le montrent les résultats présentés dans les tableaux 4.10 - 4.17, nous pouvons voir que le nombre de personnes indemnes (i.e. colonne Personnes indemnes max) augmente

TABLE 4.10 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_1_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_1_1	1003	2467	2576	18461	41,07
1_1_1	987	2473	2584	18561	42,49
1_1_1	1582	3548	3696	18661	82,78
1_1_1	1929	4164	4332	18861	123,24
1_1_1	2238	4682	4902	19061	163,67
1_1_1	2453	5177	5339	19261	185,28
1_1_1	2402	5165	5348	19361	185,26
1_1_1	2560	5613	5812	19461	185,26

TABLE 4.11 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_1_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_1_2	1394	3372	3560	49287	6,46
1_1_2	2068	4811	4998	49487	12,83
1_1_2	2116	4817	5004	49587	12,81
1_1_2	2684	5669	5879	49687	18,97
1_1_2	3018	6452	6640	49887	25,37
1_1_2	2958	6432	6645	49987	25,73
1_1_2	3250	7094	7273	50087	32,26
1_1_2	3270	7099	7302	50187	31,73
1_1_2	3577	7660	7873	50287	38,88

au fur et à mesure que nous laissons plus de temps aux évacués pour rejoindre les centres de secours. De ce fait, ils peuvent trouver des chemins plus longs mais plus sûrs afin de se mettre à l'abri. Il est aussi à noter que le nombre moyen de personnes indemnes (i.e. colonne Personnes indemnes moy) est relativement proche de la meilleure solution prouvée (i.e. Personnes indemnes max) ce qui montre que la méthode converge rapidement vers la meilleure solution heuristique. Dans la majorité des instances, la colonne associée au nombre minimum de personnes indemnes (i.e. colone Personnes indemnes min) correspond à la première itération de l'algorithme. En effet, cette dernière se base sur le schéma d'affectation pour les résultats présentés à la section 3.3.7. Une fois normalisée avec la prise en compte des pertes sur les sommets le nombre d'individus indemnes baisse significativement. Enfin, la dernière colonne des tableaux présente le temps de résolution nécessaire. Ce dernier augmente mécaniquement avec l'augmentation de l'horizon de planification. En effet, lorsqu'une fourmi qui représente un évacué explore le réseau, chaque déplacement qu'elle effectue a un certain coût en temps (respectivement distance pour aller d'un point à un autre). Aussi, si la borne supérieure correspondant à l'horizon de planification est atteinte avant que la fourmi n'ait trouvé la nourriture (i.e. un des centres de secours) alors elle

TABLE 4.12 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_2_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_2_1	877	2167	2279	5633	22,92
1_2_1	899	2176	2296	5733	23,13
1_2_1	1251	2846	2958	5833	39,63
1_2_1	1292	2849	2979	5933	40,26
1_2_1	1603	3237	3413	6033	54,13
1_2_1	1641	3213	3363	6133	53,71
1_2_1	1808	3521	3683	6233	68,16
1_2_1	1763	3522	3697	6333	67,89
1_2_1	1980	3751	3886	6433	80,14
1_2_1	1931	3755	3894	6533	79,95
1_2_1	2023	3930	4067	6633	90,06

TABLE 4.13 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_2_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_2_2	1194	2946	3076	5721	38,00
1_2_2	1730	3782	3917	5921	67,72
1_2_2	1763	3790	3947	6021	67,95
1_2_2	2064	4363	4531	6121	97,06
1_2_2	2317	4793	5006	6321	124,92
1_2_2	2347	4821	5012	6421	123,53
1_2_2	2441	5160	5414	6521	151,31
1_2_2	2618	5483	5669	6721	180,18

perd une vie et doit recommencer son périple depuis le début. Il est aussi à noter qu'après une initialisation des niveaux de phéromones qui prend du temps, l'optimisation à l'aide de notre algorithme de fourmi se déroule rapidement et fournit des résultats en moins de $913,86$ secondes pour l'instance la plus difficile à résoudre (voir instance 2_1_2 avec durée d'évacuation de 15148 unités de temps).

TABLE 4.14 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_1_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_1_1	1613	3800	3951	7791	115,57
2_1_1	1598	3795	3973	7891	113,38
2_1_1	2367	5149	5336	7991	213,43
2_1_1	2944	6005	6212	8191	300,69
2_1_1	2973	6030	6241	8291	302,86
2_1_1	3425	6553	6746	8391	372,96
2_1_1	3437	6559	6751	8491	383,69
2_1_1	3675	6894	7113	8591	452,22
2_1_1	3920	7204	7447	8791	546,95

TABLE 4.15 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_1_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_1_2	2354	5428	5628	14148	194,58
2_1_2	2439	5428	5649	14248	196,30
2_1_2	3428	7287	7546	14348	364,50
2_1_2	4012	8374	8595	14548	503,47
2_1_2	4119	8328	8616	14648	493,81
2_1_2	4675	9014	9387	14748	647,65
2_1_2	4941	9518	9779	14948	771,22
2_1_2	5236	9911	10195	15148	913,86

4.2.6 Conclusion

Nous avons présenté un nouveau modèle permettant de prendre en compte la notion de sécurité et de durée durant une évacuation. En effet les personnes doivent traverser des zones dangereuses avant d'être en sécurité peuvent subir des dommages et blessures les empêchant de poursuivre leurs déplacements. Dans notre modélisation nous avons utilisé un réseau étendu dans le temps borné afin de pouvoir capturer la notion de durée et ainsi nous concentrer sur l'optimisation du critère de sécurité.

Les résultats expérimentaux sur de petites instances montrent combien il est important d'avoir une bonne solution initiale servant à initialiser le niveau de phéromones sur les arcs. L'heuristique H4 présentée ci-dessus correspond bien à cette description et permet d'obtenir en moyenne des solutions avec une déviation de $11,37\%$ par rapport aux solutions optimales obtenues avec le solveur mathématique CPLEX. Partant de cette solution initiale, l'algorithme d'optimisation par colonies de fourmis essaie de trouver de meilleures solutions en explorant de nouveaux chemins et en modifiant le niveau de phéromones sur le réseau. Cette approche à deux phases nous permet de résoudre l'ensemble des instances en quelques

TABLE 4.16 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_2_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_2_1	1310	2463	2575	6277	31,71
2_2_1	1349	2456	2566	6377	31,86
2_2_1	1860	3142	3275	6477	46,57
2_2_1	1889	3142	3294	6577	46,73
2_2_1	2203	3544	3730	6677	59,48
2_2_1	2486	3797	3969	6877	75,44
2_2_1	2640	3991	4127	7077	87,22
2_2_1	2622	4008	4141	7177	89,77
2_2_1	2769	4186	4362	7277	93,46

TABLE 4.17 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_2_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_2_2	1820	3196	3342	6277	42,77
2_2_2	1681	3187	3364	6377	44,11
2_2_2	2351	4001	4205	6477	68,88
2_2_2	2829	4488	4630	6677	91,25
2_2_2	2751	4473	4700	6877	94,27
2_2_2	3072	4807	5042	6877	112,12
2_2_2	3369	5066	5219	7077	134,25
2_2_2	3385	5093	5283	7177	139,67
2_2_2	3276	5276	5478	7277	140,18

milisecondes tout en ayant une déviation moyenne par rapport aux solutions optimales de 4.46%. En outre, la résolution d'instances de taille réelle permet d'estimer combien de personnes peuvent être sauvées avant une date limite T. Les algorithmes présentés ayant comme particularité de ne pas revenir sur les décisions déjà prises, nous avons pu mettre à profit les structures de données présentées dans le Chapitre 3 afin de pouvoir prendre en compte toutes les instances de taille réelle. Les expérimentations sur des données de taille réelle montrent aussi que plus nous donnons du temps aux évacués, plus nous avons de personnes indemnes capables de se déplacer par leur propre moyen des points de rassemblement vers les centres de secours en prenant les chemins les plus sûrs.

En faisant varier la date impérative de fin d'évacuation T, nous pouvons ainsi répondre à deux questions majeures à savoir :

— De combien de temps pouvons-nous retarder une évacuation en garantissant que tout le monde sera indemne et sans blessures ?
— Ayant une date impérative de fin d'évacuation T, combien de personnes pourront se mettre en sécurité en ne subissant aucun dommage ?

Aussi, pour savoir qu'elle est la durée minimale T (borne inférieure) nécessaire pour que tous les évacués arrivent indemnes par leurs propres moyens aux centres de secours, tous les sommets ayant un niveau de sécurité inférieur à $1,0$ doivent être supprimés. Dans ce cas de figure, le problème d'évacuation devient un problème de transbordement connu sous le nom de "Quickest Transshipment problem" qui peut être résolu en temps polynomial si les données concernant le réseau ne varient pas dans le temps (voir [Hoppe et Tardos, 2000]). Lorsque les données varient dans le temps nous avons proposé dans le Chapitre 3 un algorithme pseudo-polynomial permettant de connaitre cette date qui correspond à la date de fin d'évacuation.
De ce fait, si les circonstances font que la date limite de fin d'évacuation fixée T est inférieure à la borne inférieure déterminée par les deux précédentes méthodes, alors le problème d'évacuation doit être considéré comme un problème de flot généralisé entier avec diviseurs (FGED) afin de prendre en compte les blessures handicapantes subies par les évacués.
Toutefois, dans la pratique, lorsque l'environnement dévasté nuit au bon déroulement des opérations d'évacuation, il n'est pas rare de voir les forces de secours ou des personnes qualifiées sécuriser les points critiques ou aider les individus à les traverser. Nous traitons cet aspect dans la section suivante et étudions comment cela peut améliorer la qualité des solutions proposées.

4.3 Flot généralisé entier avec diviseurs et améliorations

4.3.1 Description du problème

Dans le cas des problèmes d'évacuation, au-delà de la minimisation de la durée d'évacuation (voir [Hamacher et Tjandra, 2001, Altay et Green III, 2006]) la prise en compte d'autres critères tels que la sécurité sont des domaines de plus en plus étudiés et dont l'intérêt est grandissant. Le but de ces différentes approches est de prendre en compte le plus d'aspects possible dans les modélisations afin que les modèles proposés soient réalistes et permettent aux preneurs de décisions de mieux s'organiser pour réagir aux situations critiques dans le but de sauver des vies (voir [Opasanon et Miller-Hooks, 2009, Lämmel *et al.*, 2011]).

L'utilisation de la notion de flots classique pour modéliser les problèmes d'évacuation est très connue dans la littérature (voir [Chalmet *et al.*, 1982, Yamada, 1996, Choi *et al.*, 1988b]). Dans cette même catégorie, les problèmes de flots avec multiplicateurs sont bien connus pour permettre la modélisation et la résolution de problèmes pratiques lorsque des objets sont créés ou détruits (voir [Oldham, 2001, Radzik, 1998, Wayne, 1999]). Comme nous l'avons montré dans la section précédente avec la modélisation du problème d'évacuation FGED, les problèmes de flots avec multiplicateurs peuvent aussi avoir de multiples applications dans le cadre des télécommunications avec l'amplification ou la perturbation de signaux, la modélisation de réseaux électriques ainsi que les pertes dues à la dissipation thermique de certains composants, le marché des changes et de devises, de l'équilibrage de charge des machines, de gestions de stocks ou encore modéliser des problèmes de satisfiabilité comme SAT ou 3-SAT.

Nous avons présenté dans le chapitre 3 comment réaliser une évacuation où l'on suppose que les individus pourront se déplacer par leurs propres moyens des zones dangereuses jusqu'aux centres de secours et dans la section 4.2 nous avons utilisé la notion de flot avec multiplicateurs (diviseurs) pour présenter comment organiser l'évacuation des personnes afin que ces dernières ne soient pas blessées et ne requièrent aucune aide extérieure pour continuer à se mouvoir. Toutefois, pour autant que nous sachions, il est à noter qu'aucun des travaux de la littérature ne permet de prendre en compte le fait qu'en cas d'évacuation de masse les personnes ne sont pas livrées à elles-mêmes.
En effet, lors d'évacuations de masse, les autorités déploient des forces de secours afin que ces dernières organisent le flux de personnes mais aussi s'assurent que les personnes puissent atteindre les centres de secours sans encombre. En outre, lorsqu'il existe des points critiques du réseau endommagé par un sinistre comme la chute d'arbres barrant des routes, un risque d'éboulement, un pont submergé par les eaux ou encore une route partiellement détruite par un tremblement de terre, il s'agit pour les autorités de déployer des forces de secours qui sont en nombre limité (policiers, pompiers, infirmiers,...) afin qu'ils sécurisent les zones traversées et donnent des directives permettant ainsi aux évacués d'avoir un meilleur taux de succès dans leur tentative de rallier les centres de secours.

Le choix du déploiement des forces de secours sur le réseau revient d'une certaine manière à le reconfigurer afin d'optimiser le processus d'évacuation.
De manière classique, le fait d'inverser le sens des arcs d'un graphe (i.e. inverser le sens d'une route) permet de s'assurer que les personnes puissent rapidement quitter une zone dangereuse et rendre difficile voir impossible toute tentative de s'en approcher. Aussi, si une route a deux voies opposées, le fait d'inverser un des arcs permet de doubler la capacité dans un des sens. Il devient donc évident qu'une reconfiguration peut être utile dans le but de minimiser par exemple la durée de l'évacuation. Cette reconfiguration nécessite que des individus qualifiés soient déployés sur le réseau à savoir des forces de secours sur ces arcs pour s'assurer que les personnes suivent les indications données pour optimiser l'utilisation du réseau. Des exemples d'études portant sur la reconfiguration de réseaux routiers peuvent être trouvés dans plusieurs travaux de recherche comme [Xie *et al.*, 2010, Xie et Turnquist, 2011]. De manière similaire au problème de déploiement des forces de secours limitées en nombre que nous traitons, dans le cas d'une reconfiguration de réseau routier, le nombre d'arcs pouvant être inversé est aussi limité en nombre. Toutefois contrairement à ces derniers, dans notre cas nous ne nous intéressons pas à la reconfiguration des arcs mais celle des sommets.
Plusieurs des travaux en rapport avec la reconfiguration de réseaux existent (voir [Schwarz et Krumke, 1998, Krumke *et al.*, 1998, Demgensky *et al.*, 2004, Ordóñez et Zhao, 2007, Dilkina *et al.*, 2011, Lin et Mouratidis, 2013, Campbell *et al.*, 2006]) mais dans notre cas, la reconfiguration ne porte pas sur les arcs et leur capacité mais sur les sommets et leur sécurité. Pour autant que nous sachions, cet aspect de la reconfiguration de réseau n'a pas encore été étudié.

En outre, le choix des zones à sécuriser est influencé par deux aspects les points critiques du réseau utilisés par une grande majorité de la population mais aussi le fait que l'on ne connaisse pas forcément à l'avance les niveaux d'endommagement de ces derniers afin de décider où affecter les individus, nous permettant de nous s'assurer que l'évacuation

se passera dans les meilleures conditions possibles. Dans ce dernier cas de figure, comme certaines données sont incertaines, il s'agit de réaliser un plan d'évacuation qui soit robuste et qui reste réalisable malgré les aléas inhérents aux sinistres (i.e. répliques de tremblement de terre, montée subite des eaux due à de fortes précipitations,...) et faisant que le niveau de dommages puisse varier.
Aussi, nous considérons deux cas de figure dans cette section où nous supposerons dans un premier temps que les niveaux de dommages sont connus et fixes, nous nommons ce problème d'évacuation FGEDA [1]. Dans un second temps nous étudions le cas où les autorités ont une idée approximative des dommages et que c'est seulement qu'une fois que les forces de secours sont déployées sur zone qu'il est possible de réellement évaluer le niveau de sécurité des points critiques associés au scénario-catastrophe considéré, nous nommons ce problème d'évacuation FRGEDA [2].

4.3.1.1 Modèle mathématique du problème FGEDA

Ci-dessous nous présentons la formulation mathématique du problème de Flot Généralisé Entier avec Diviseurs et Améliorations (FGEDA).
Données :

- $G = (V, E)$ Un réseau étendu dans le temps avec des données qui varient en fonction du temps. Ainsi les attributs de durée sont directement intégrés dans le réseau.
- V : Un ensemble de N sommets.
- S et P : Le super sommet source et le super sommet puits.
- E : Un ensemble de M arcs $\subseteq V \times V$.
- $C_{(i,j)}$: Le nombre maximum de personnes pouvant entrer dans l'arc $(i, j) \in E$.
- Pr_i : La probabilité de traverser sans dommages le sommet i avec $Pr_i \in \{x \in \mathbb{R} | 0 \leq x \leq 1\}$.
- Prg_i : Le gain de probabilité de traverser sans dommages le sommet i avec $Prg_i \in \{x \in \mathbb{R} | 0 \leq x \leq 1 - Pr_i\}$.
- Co_i : Le nombre de ressources nécessaires à la sécurisation su sommet i.
- B_S : Le nombre total de personnes à évacuer depuis le super sommet source S.
- B_P : Le nombre maximum de personnes pouvant atteindre le super sommet puits P.
- B_{min} : Le nombre minimum de personnes devant atteindre le super sommet puits P. Cette donnée est liée à la version décisionnelle du problème d'évacuation que nous expliciterons par la suite.
- FS : Le nombre total de groupes de forces de secours qui peuvent être déployés sur l'ensemble du réseau.
- RS : Le nombre total de ressources disponibles pour les forces de secours.

Variable :

- $F_{(i,j)}$: le nombre de personnes empruntant l'arc (i, j) avec $F_{(i,j)} \in \mathbb{N}$.
- Z_i : Variable indiquant si un groupe de forces de secours est déployé sur le sommet i avec Z_i une variable booléenne (i.e. $\in \{0, 1\}$).

1. Flot généralisé entier avec diviseur et améliorations
2. Flot robuste généralisé entier avec diviseurs et améliorations

— H_i : le nombre supplémentaire maximum de personnes pouvant sortir du sommet i suite à sa sécurisation.

Contraintes :

S.C

$$0 \leq F_{(i,j)} \leq C_{(i,j)} \qquad \forall (i,j) \in E \tag{4.5}$$

$$H_i \leq HV \times Z_i \qquad \forall i \in V \tag{4.6}$$

$$H_j \leq \sum_{i \in \Gamma_j^-} F_{(i,j)} \qquad \forall j \in V \tag{4.7}$$

$$\sum_{i \in V} Z_i \leq FS \tag{4.8}$$

$$\sum_{i \in V} Co_i \times Z_i \leq RS \tag{4.9}$$

$$Pr_j \times \sum_{i \in \Gamma_j^-} F_{(i,j)} + Prg_j \times H_j \geq \sum_{k \in \Gamma_j^+} F_{(j,k)} \qquad \forall j \in V \setminus P \tag{4.10}$$

$$\sum_{i \in \Gamma_S^+} F_{(S,i)} \leq B_S \tag{4.11}$$

$$\tag{4.12}$$

Ci-dessous, l'ensemble des contraintes permettant de décrire le problème d'évacuation avec pertes et améliorations.

— Le nombre maximum de personnes entrant dans l'arc $(i,j) \in E$ est inférieur ou égale à la capacité de l'arc considéré (4.5).
— La contrainte (4.6) permet de déterminer le nombre maximum de personnes en plus pouvant sortir du sommet i suite son amélioration ou non. La donnée HV est une donnée symbolisant un grand nombre. Cette dernière peut être fixée en utilisant le nombre de personnes devant être évacuées (i.e. $HV = B_S$).
— La contrainte (4.7) permet de corréler l'amélioration potentiellement attendu avec le nombre de personne arrivant à un sommet i.
— La contrainte (4.8) permet de s'assurer que les forces de secours ne puissent pas utiliser plus de ressources que disponible.
— La contrainte (4.9) permet de s'assurer que l'on ne déploie pas plus de forces de secours que disponible.
— Le nombre total de personnes indemnes sortant d'un sommet $i \in V$ est inférieur ou égal au nombre de personnes entrées multiplié par le niveau de sécurité amélioré du sommet considéré (4.3).
— Au plus B_S individus pourrons tenter de se mettre à l'abri en quittant le super sommet source S (4.4).

Fonctions objectif : Tout comme pour le problème présenté à la section 4.2 nous pouvons considérer les deux mêmes fonctions objectifs suivant le problème d'évacuation que nous cherchons à résoudre. En effet pour le problème décisionnel il s'agira de déterminer où placer les forces de secours afin qu'au moins B_{min} individus soient indemnes à la fin de l'évacuation alors que pour le problème d'optimisation nous chercherons à trouver le dé-

ploiement des forces de secours permettant de maximiser le nombre de personnes indemnes (voir l'équation (4.1)).
Dans la section suivante, nous étudions la complexité du problème FGEDA afin de connaitre ses caractéristiques.

4.3.2 Complexité du problème FGEDA

Nous effectuons une extension de la preuve de complexité du problème FGED afin de prendre en compte la notion d'amélioration du niveau de sécurité des sommets par les forces de secours. Nous utilisons aussi les mêmes notations que celles présentées dans la section 4.2.2.

Instance

Une instance du problème FGEDA est composée d'un graphe orienté $G = (V, E)$ avec N sommets et M arcs, d'un sommet de départ S et d'arrivée P, d'une probabilité $Pr_i \in [0..1] \ \ \forall i \in V - \{S, P\}$, d'une amélioration $Prg_i \in [0..1] \ \ \forall i \in V - \{S, P\}$ lorsque des forces de secours sont déployées sur un sommet, d'une capacité $C_{(i,j)} \in \mathbb{N}^* \ \ \forall (i, j) \in E$, d'un nombre d'unités B_S partant du sommet S et d'une valeur cible minimale $B_{min} \in [1..B_S]$ devant arriver au sommet P, du nombre maximum de forces de secours pouvant être déployé sur le réseau FS ainsi que du nombre de ressources utilisables RS.

Exemple

Soit une instance du problème FGEDA définie par la figure 4.8 et pour laquelle on essaie de faire voyager 500 personnes du sommet S vers le sommet P et pour laquelle nous disposons de deux forces de secours déployables sur le réseau (i.e. $FS = 2$) et de deux ressources (i.e. $RS = 2$). A chaque sommet est associé deux valeurs de sécurité (i.e. $Pr_i|Prg_i$) dont la première est le niveau de sécurité initial et la seconde l'amélioration auquelle nous pouvons prétendre si des forces de secours y sont déployées. Nous supposons que la quantité de ressources nécessaires à l'amélioration de chaque sommet est d'une unité (i.e. $Co_i = 1 \forall i \in V$).
Existe-t-il une fonction de flot $F : E \to \mathbb{N}$ telle que :

- $B_{min} = 400$: NON car la coupe minimale du graphe G est de 333 {(2,5),(4,5),(6,7)}.
- $B_{min} = 200$: OUI car il existe un certificat $CERT_{FGEDA}$ valide avec 246 unités atteignant le sommet P.

Arcs	(S,1)	(1,2)	(1,3)	(2,4)	(2,5)	(3,4)	(3,6)	(4,5)	(4,6)	(5,7)	(6,7)	(7,P)
Flot(F)	500	200	300	0	150	106	80	33	20	146	100	246

Sommets	S	1	2	3	4	5	6	7	P
Forces de secours(Z)	0	0	0	1	0	1	0	0	0

TABLE 4.18 – Exemple de certificat optimal pour le problème FGEDA

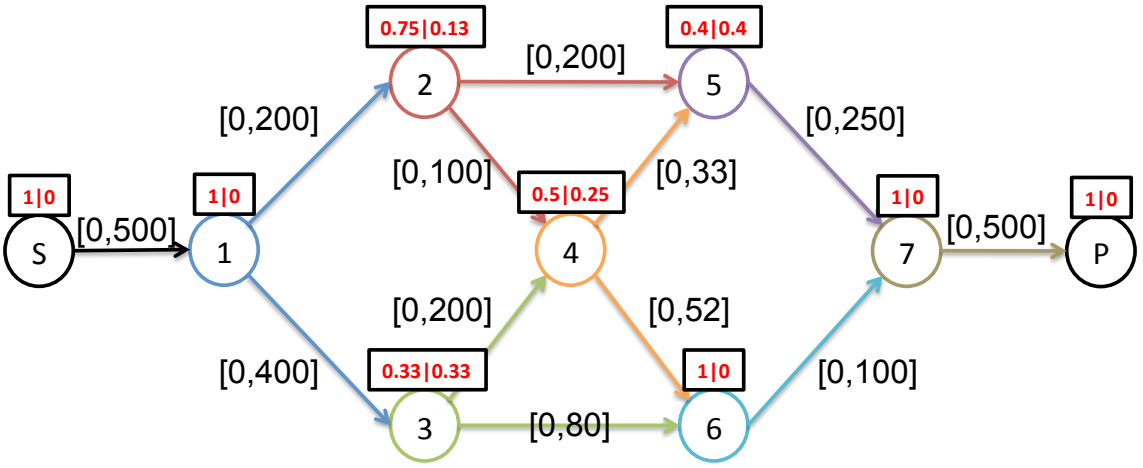


FIGURE 4.8 – Instance de graphe pour le problème FGEDA

Propriétés

Le problème FGEDA est une généralisation du problème d'évacuation FGED. En adaptant les démonstrations des théorèmes du problème FGED, il est trivial de montrer que le problème de décision de FGEDA est aussi $NP\text{-}Complet$ et $NP\text{-}Difficile$ pour le problème d'optimisation. Dans cette section nous ne nous intéresserons donc qu'à l'étude de la difficulté de résolution du problème d'optimisation.

Theorem 1. *Le problème d'évacuation FGEDA est $NP\text{-}Difficile$ au sens fort même si l'on relaxe la contrainte d'intégrité du flot et que les coûts de réparation des sommets sont d'une unité (i.e. $Co_i = 1$).*

Démonstration. Afin de prouver la difficulté de résolution du problème d'optimisation, nous considérons le problème 3-PARTITION. Ce problème est connu comme $NP\text{-}Difficile$ au sens fort [Garey et Johnson, 1975, Garey et Johnson, 1979] :
Pour rappel, une instance de 3-Partition est définie par une borne $B \in \mathbb{N}^*$, un ensemble A de $3N$ entiers $S(a)$ tels que $\forall a \in A$, $\frac{B}{4} < S(a) < \frac{B}{2}$.
La question que l'on se pose est de savoir s'il existe N sous-ensembles disjoints $A_i \subset A$ tels que $\forall i \in [1..N] \quad \sum_{a \in A_i} S(a) = B$.

Le graphe associé à la figure Fig4.9 décrit une instance du problème FGEDA construite à partir d'une instance générique du problème 3-PARTITION. A chaque sommet est associé un niveau de sécurité initial ainsi qu'une amélioration possible (i.e. $Pr_i|Prg_i$) et chaque arc dispose d'une capacité maximale. Les sommets S et P sont définis comme le super sommet source et le super sommet puits. Les sommets M_0 et M_{N+1} représentent des sommets fictifs de synchronisation alors que les sommets M_i avec $i \in [1, .., N]$ représentent les N sous-ensembles N disjoints de poids B. Chaque sous-ensemble M_i peut comprendre au plus 3 éléments parmi les $3N$ les dont la somme des poids est de B. Initialement, aucun élément ne peut être associé aux sous-ensembles M_i du fait de leurs niveaux de sécurité qui sont à 0. Nous posons $Co_i = 1 \forall i \in V$, et $RS = FS = 3N$. Ceci veut dire qu'afin qu'un

élément puisse être associé à un sous-ensemble de M_i, il faut que son niveau de sécurité soit amélioré en le faisant passer de 0 à 1. En matière d'évacuation, cela revient à créer un nouveau chemin sans danger utilisable par les évacués.

Maintenant considérons deux cas de figure :

— Flot entier : Si un élément j est entièrement associé au sous-ensemble M_i, alors cela génère un flux d'une unité sur l'arc $(a\prime_{ij}, a_j)$. Ainsi, il n'est donc pas possible qu'un même élément a_j soit associé à deux sous-ensembles distincts car la capacité de arcs (a_j, M_{N+1}) est d'une unité. Aussi, le fait de choisir 3 éléments (sommets) dont la sécurité doit être améliorée dans le sous-ensemble revient à les y affecter au sens du problème de 3-Partition. De ce fait, l'existence ou non de 3 tâches sur les $3N$ à améliorer pour chaque sous-ensemble permettant d'obtenir $3N$ unité de flot au super sommet puits P permet de répondre par OUI ou par NON à une instance du problème 3-Partition.

— Flot non entier : Supposons maintenant que le flot n'est pas entier et intéressons nous en particulier aux arcs $(a\prime_{ij}, a_j)$. Dans ce cas de figure, cela veut dire qu'un élément peut être partiellement associé à plusieurs sous-ensembles. Toutefois, comme nous avons un nombre limité de forces de secours et de ressources, à chaque fois que nous affectons deux ou plus forces de secours sur des copies d'un élément se trouvant sur plusieurs sous-ensemble, cela revient de fait à dire qu'il existera des éléments de A qui n'appartiendront à aucun sous-ensemble. Autrement dit, le fait d'allouer plus de forces de secours que nécessaires sur le même élément revient à pénaliser d'autres éléments et ainsi avec un nombre de personnes indemnes atteignant le super sommet puits P qui soit inférieur à $3N$ personnes à la fin de l'évacuation. Aussi, s'il existe une solution pour le problème 3-Partition, les solutions qui lui sont associées auront forcément un flot entier sur les arcs $(a\prime_{ij}, a_j)$ dans le but de maximiser le nombre d'éléments pouvant arriver au super sommet puits P à savoir $3N$. Cette configuration revient aussi à dire qu'à chaque sous-ensemble M_i avec $i \in [1, .., N]$ sont associés de manière unique 3 éléments parmi les $3N$ et dont la somme est égale à B.

□

Nous avons précédemment prouvé que le problème d'évacuation FGEDA est $NP\text{-}Difficile$ au sens fort. Afin de pouvoir résoudre des problèmes de taille réel à l'échelle d'une ville, nous allons présenter dans la section suivante comment prendre en compte l'amélioration du niveau de sécurité du réseau avant l'évacuation de la population.

4.3.3 Algorithmes approchés

Le problème FGEDA est une extension du problème FGED pour lequel nous avons déjà proposé un algorithme se basant sur l'utilisation des algorithmes d'optimisation par colonies de fourmis.
En effet, une fois les forces de sécurité déployées sur les sommets critiques du réseau afin de les améliorer, notre problème d'évacuation avec amélioration devient un problème classique de flot généralisé entiers avec diviseurs (i.e. FGED). Aussi, il convient de résoudre le

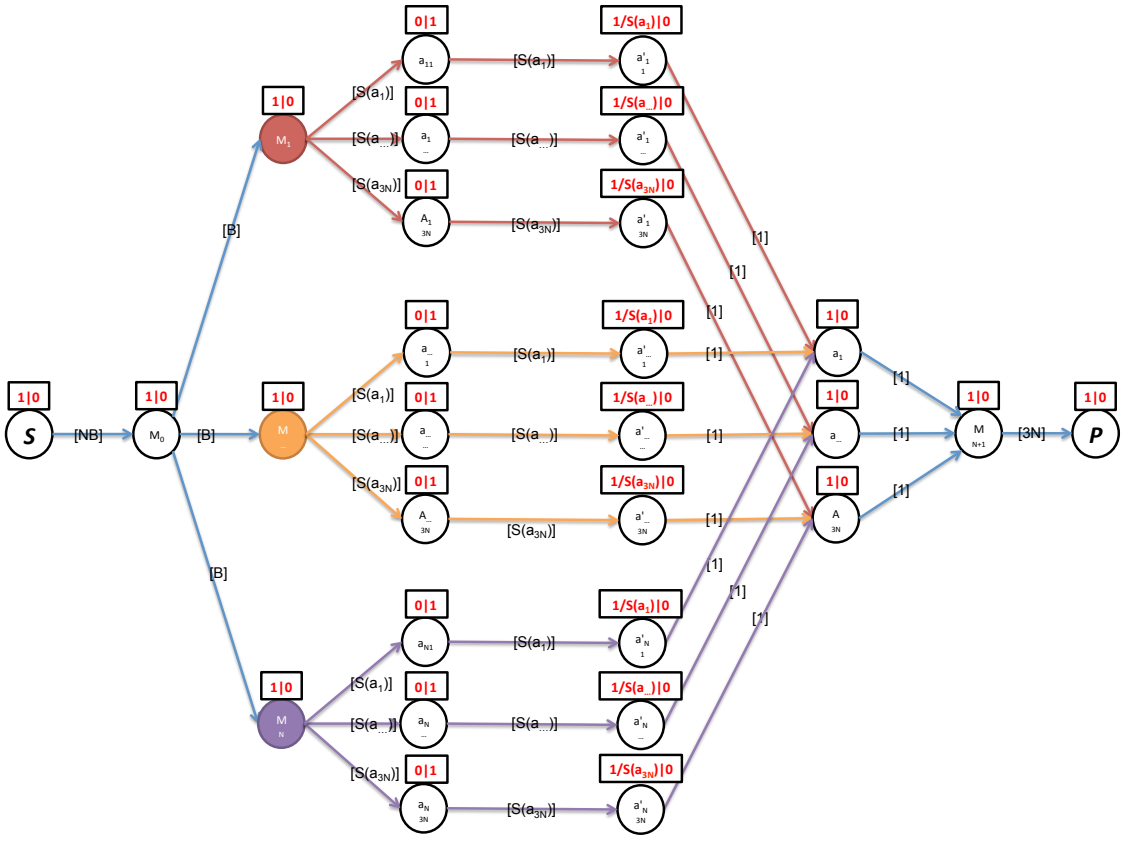


FIGURE 4.9 – Réduction pseudo-polynomiale de 3-Partition vers FGEDA

problème d'affectation des forces de sécurité.
Afin d'y parvenir, nous avons développé une métaheuristique basée sur une optimisation par recherche tabou (voir [Siarry, 2014]). Nous présentons l'approche développée dans les sections ci-dessous. Nous étudierons le cas où toutes les valeurs $Co_i = 1,\ \forall i \in V$ avec $FS = RS$. Pour le cas général, il est possible d'adapter les méthodes présentées afin de les prendre en compte.

Codage d'une solution

Afin de coder une solution pour le problème d'affectation de forces de secours, nous utilisons tout d'abord un vecteur de booléens de taille $Card(V) = N$ correspondant au nombre de sommets dans le graphe. Comme il est inutile de déployer des forces de secours sur des sommets dont le niveau de sécurité est déjà de 100% nous supprimons ces derniers pour ne plus garder que ceux dont le niveau de sécurité est perfectible. Si nous prenons l'instance définie par la figure Fig.4.8, notre vecteur de solution pour l'affectation de 2 forces de secours pourra être :

Sommets perfectibles	2	3	4	5
Affectation de forces de secours	0	1	0	1

Table 4.19 – Exemple de codage d'une solution d'affectation

Solution initiale

La solution initiale de notre métaheuristique est une solution de base permettant d'avoir une première répartition des forces de secours.
Afin de l'obtenir, nous résolvons dans un premier temps le problème d'optimisation FGED sans pour autant améliorer aucun des sommets à l'aide de l'algorithme par colonies de fourmis présenté à la section 4.2.4. Une fois cette étape effectuée, nous obtenons une solution réalisable pour laquelle nous pouvons déduire le nombre de personnes blessées et non indemnes sur chaque sommet. De ce fait, il est possible de mettre en place une liste de priorité qui se base sur le nombre de personnes blessées à cause d'un des sommets du graphe. Par exemple, si nous prenons l'instance 4.8 la solution correspondant au problème d'optimisation FGED est la suivante :

Arcs	(S,1)	(1,2)	(1,3)	(2,4)	(2,5)	(3,4)	(3,6)	(4,5)	(4,6)	(5,7)	(6,7)	(7,P)
Flot(F)	500	200	300	21	129	19	80	0	20	51	100	151

Sommets	S	1	2	3	4	5	6	7	P
Personnes non indemnes	0	0	50	201	20	78	0	0	0
Liste de priorité	5	5	3	1	4	2	5	5	5

Table 4.20 – Exemple de liste de priorité pour le déploiement des forces de sécurité

Ainsi, si $FS = 1$ seule le sommet 3 sera amélioré alors que si $FS = 2$ les sommets 3 et 5 le seront tous les deux. Lorsque deux sommets ont le même niveau de priorité nous en

choisissons arbitrairement un.

Évaluation d'une solution

L'évaluation d'une solution revient à savoir si le fait d'avoir amélioré les sommets a permis d'obtenir plus de personnes indemnes pouvant atteindre les centres de secours. Il est facile de constater qu'une fois les sommets à améliorer fixés, il nous reste à résoudre le problème d'évacuation FGED. Toutefois nous avons montré que ce problème d'optimisation est $NP\text{-}Difficile$ au sens fort et n'est pas approximable. Toutefois, nous avons présenté une métaheuristique basée sur l'optimisation par colonies de fourmis dont la déviation en moyenne par rapport aux solutions optimales était de de l'ordre de 4%. Aussi, à chaque solution du problème d'affectation des forces de secours considérée, nous utilisons l'algorithme d'optimisation par colonies de fourmis présenté à la section 4.2.4.

Voisinages

Une fois la meilleure solution courante en notre possession, nous déterminons les solutions qui lui sont voisines afin d'avoir un échantillonnage de l'espace de recherche. Pour cela, nous utilisons plusieurs opérateurs de voisinage comme nous présentons dans le tableau **??**.

Les opérateurs de voisinage décrits dans le tableau **??** permettent de s'assurer qu'à partir de toute solution d'affectation initiale des forces de secours il existe un nombre fini de permutations permettant d'atteindre tout autre état de l'ensemble de recherche. En outre, nous avons deux classes de voisinage. L'une permet d'effectuer une diversification de la solution courante tout en guidant la diversification grâce à une liste de priorité. La seconde classe permet une modification minime de la solution tout en garantissant qu'il est possible de créer de proche en proche des chemins sûrs initialement inexistants.

Gestion de la liste tabou

Même si du fait de l'aspect pseudo-aléatoire inhérent aux opérateurs de voisinage choisis, il est rare de retomber sur les mêmes états d'affectation des forces de secours, nous mettons tout de même en place un liste tabou qui correspond à une file de vecteurs d'affectation gérée en FIFO. Aussi, à chaque itération de notre métaheuristique basée sur la recherche tabou, nous enfilons la meilleure solution courante à la file (i.e. liste tabou). Nous avons arbitrairement fixé la taille de cette file à 10. Ce faisant, un même schéma d'affectation ne peut redevenir la meilleure solution avant au moins 10 itérations de notre méthode.

4.3.4 Résultats expérimentaux

Afin de pouvoir évaluer la performance de notre métaheuristique, nous avons effectué des campagnes de tests afin de comparer notre méthode avec les solutions optimales obtenues à l'aide du solveur CPLEX. Pour cela, nous utilisons les mêmes jeux de données et les mêmes paramétrages que ceux établis lors des tests effectués dans la section 4.2.5.

Opérateur de voisinage	Description de la permutation
Priorité-Permute-2	Pondérer chaque sommet en utilisant la liste de priorité obtenue avec la solution courante. Choisir aléatoirement parmi les sommets de priorités croissantes non améliorés (i.e. FAUX). Choisir aléatoirement parmi les sommets de priorités décroissantes améliorés (i.e. VRAI). Permuter l'état des deux sommets sélectionnés. Cet opérateur de voisinage permet de réaliser une diversification de la recherche
Aléatoire-Permute-Successeurs	Nous choisissons aléatoirement un sommet parmi les sommets déjà améliorés et nous déplaçons les forces de secours sur ses successeurs si ces derniers ne sont pas déjà améliorés. Cet opérateur de voisinage permet de réaliser une intensification de la recherche
Aléatoire-Permute-Prédécesseurs	Nous choisissons aléatoirement un sommet parmi les sommets déjà améliorés et nous déplaçons les forces de secours sur ses prédécesseurs si ces derniers ne sont pas déjà améliorés. Cet opérateur de voisinage permet de réaliser une intensification de la recherche

TABLE 4.21 – Opérateurs de voisinage

Dans le cadre de nos tests, nous considérerons que nous ne disposerons pas d'assez de forces de secours pour sécuriser plus de 20% des sommets dangereux du réseau routier. En outre du fait qu'en période de crise il n'est pas possible de s'assurer qu'une structure endommagée puisse être restaurée à 100% de son état initial, nous supposerons que lorsque les forces de secours sont déployées sur un sommet dangereux, seulement un gain de 20% pourra être espéré (i.e. $Prg_i = (1.0 - Pr_i) \times 0.2$).

4.3.4.1 Résultats de tests

Nous avons dans un premier temps résolu à l'optimalité le modèle mathématique en utilisant le solveur CPLEX sur nos jeux de données modifiées 4.2.5 (voir ci-dessous). Nous supposons que le flot d'individus indemnes est entier et notre but est de maximiser le nombre de personnes arrivant aux centres de secours sans aides. Les résultats obtenus sont présentés dans le tableau Tab.4.22. Nous pouvons constater que l'augmentation de la taille des instances rend le problème d'évacuation de plus en plus difficile à résoudre et que le pourcentage d'instances résolues chute de $95,8\%$ pour 30 sommets à $45,5\%$ pour 120 sommets. Il est aussi à constater qu'autant la résolution d'une instance peut nécessiter

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	% Résolues
G_{30}	2.750	29.956	1.118.410	118.874	95,8
G_{60}	3.289	61.963	1.193.430	178.432	69,5
G_{90}	3.360	36.549	709.448	109.275	58,5
G_{120}	3.214	66.534	1.177.734	183.278	45,5

TABLE 4.22 – Résultats de tests CPLEX avec le programme linéaire en nombres entiers (IP)

3 secondes autant elle peut nécessiter près de 20 minutes pour les mêmes tailles de données. Ceci reste problématique pour la résolution d'instances de taille réelle à l'échelle de la ville de Nice. En moyenne, hormis les graphes de la classe G_{60}, le temps de résolution augmentent avec la taille des instances.
Afin d'estimer le gain potentiel auquel nous pourrions nous attendre avec notre metaheuristique, nous avons relâché les contraintes d'intégrité du flot sur les arcs dans le modèle mathématique. Les résultats de ce dernier sont présentés dans le tableau Tab.4.23. L'une des premières constatation est que toutes les instances ont pu être résolues entre 3 et 5 secondes. L'autre aspect important est le gap entre les instances résolues à l'optimalité du modèle mathématique initial et ceux du modèle relâché.

En effet, la déviation entre les deux approches varie de $4,28\%$ à $6,60\%$, ce qui n'est pas loin de la solution optimale. Le modèle mathématique relâché étant une borne supérieure de bonne qualité pour ce problème d'évacuation, cela suggère que le gain potentiel obtenu par notre métaheuristique sera moindre. Afin d'évaluer notre métaheuristique basée sur l'approche tabou pour fixer l'affectation des forces de secours, nous avons effectué les tests sur les mêmes jeux de données. Les résultats présentés dans le tableau 4.24 montrent que l'ensemble des instances a pu être résolues en un peu plus de 4 secondes et qu'en moyenne

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	%Résolues	% de déviation de la solution
G_{30}	3.050	3.250	3.784	122	100	4,28
G_{60}	3.366	3.557	4.061	109	100	4,44
G_{90}	3.507	3.735	4.439	147	100	3,17
G_{120}	3.293	3.938	4.987	243	100	6,60

TABLE 4.23 – Résultats de tests CPLEX avec le programme linéaire en nombres réels (LP)

la durée de résolution augmente avec la taille du graphe considéré. En outre, nous pouvons noter une augmentation de la déviation par rapport à la meilleure solution connue pour le problème FGEDA comparé au problème FGED. En effet, celle-ci est en moyenne de $7,47\%$ pour le problème FGEDA et de $4,46\%$ pour le problème FGED. Ce décalage étant du à la métaheuristique de recherche tabou utilisée pour affecter les forces de secours sur les sommets critiques du réseau. Dans la section suivante, nous appliquons cette méthode à la

Taille du graphe	Durée minimale	Durée moyenne	Durée maximale	Ecart-type de la durée	%Résolues	% de déviation de la solution
G_{30}	0,21	103,60	201	23	100	7,52
G_{60}	0,87	452,34	776	196	100	7,87
G_{90}	1,98	976,11	2.221	478	100	6,91
G_{120}	1,11	1.662,84	4.055	765	100	7,57

TABLE 4.24 – Résultats de tests de l'approche tabou et de la meilleure configuration pour l'algorithme de colonies de fourmis

ville de Nice pour mesurer l'impact du déploiement des forces de secours sur la sureté des plans d'évacuation proposés lors de différents scénarios de catastrophes naturelles.

4.3.4.2 Mise à l'échelle : Instances de taille réelle

Afin de pouvoir mettre à l'échelle notre méthode de résolution nous avons appliqué les mêmes améliorations que celles présentées dans la section 4.2.5.4 concernant l'évaluation du flot d'individus. Afin de pouvoir explorer le plus de solutions possible et traiter rapidement le voisinage de la meilleure solution courante de la méthode tabou, nous avons parallélisé l'évaluation des solutions. Aussi pour y parvenir, nous avons changé d'architecture de tests. En effet, les expérimentations ont été effectuées sur une machine virtuelle "VirtualBox" utilisant 8 processeurs d'un Intel Core i7-4910QM cadencé à 2.90Ghz. Il avait 8Mo de mémoire cache et 16Go de mémoire RAM avec une fréquence de 1600Mhz. Le système d'exploitation utilisé est le Ubuntu 14.04 LTS x64. Les résultats obtenus sont présentés dans les tableaux 4.25 - 4.32

TABLE 4.25 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_1_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_1_1	1080	2752	2899	18461	76,65
1_1_1	1038	2737	2890	18561	76,56
1_1_1	1768	3896	4049	18661	141,98
1_1_1	2136	4594	4797	18861	192,02
1_1_1	2335	5207	5379	19061	254,34
1_1_1	2594	5692	5866	19261	320,21
1_1_1	2670	5704	5902	19361	319,16
1_1_1	2799	6107	6286	19461	377,92

TABLE 4.26 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_1_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_1_2	1524	3740	3862	49287	107,73
1_1_2	1465	3729	3892	49387	107,37
1_1_2	2297	5281	5519	49487	215,92
1_1_2	2839	6264	6439	49687	317,62
1_1_2	2848	6252	6440	49787	320,19
1_1_2	3233	7067	7240	49887	433,17
1_1_2	3288	7054	7261	49987	431,67
1_1_2	3476	7705	7950	50087	538,83
1_1_2	3519	7714	7974	50187	540,51
1_1_2	3813	8232	8544	50287	655,04

L'une des premières remarques que nous pouvons faire est que le fait de paralléliser les tâches d'évaluation de solutions permet de limiter le temps de résolution. En effet, l'ensemble des instances a pu être résolues en moins de 1450 secondes (voir instance 2_1_2 avec l'horizon de planification fixée 15148 unités de temps). Il est aussi à noter que le nombre de personnes indemnes augmente avec l'amélioration de la sécurité des sommets. Sur l'ensemble des instances traitées, le déploiement des forces de secours permet d'avoir en moyenne $252,29$ personnes indemnes de plus représentant une amélioration de $4,3\%$ de la qualité des solutions comparées à celles du problème FGED. Rappelons que seulement 20% des sommets pouvaient être améliorés et que les améliorations (i.e. sécurisation) ne pouvaient porter que sur 20% des dommages de chaque sommet.

4.3.5 Conclusion

Dans cette section, nous avons proposé un modèle mathématique permettant de prendre en compte le déploiement des forces en cas de scénarios catastrophes (FGEDA). Ce pro-

TABLE 4.27 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_2_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_2_1	819	2191	2328	5633	35,88
1_2_1	918	2207	2339	5733	35,73
1_2_1	1301	2880	3011	5933	60,55
1_2_1	1598	3237	3420	6033	81,75
1_2_1	1549	3259	3436	6133	81,28
1_2_1	1784	3557	3729	6233	102,17
1_2_1	1905	3796	3938	6433	123,78
1_2_1	1938	3815	3948	6533	121,05
1_2_1	2082	3991	4122	6633	140,53

TABLE 4.28 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 1_2_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
1_2_2	1237	2979	3105	5721	58,57
1_2_2	1229	2998	3123	5821	58,64
1_2_2	1755	3835	3990	5921	105,28
1_2_2	1724	3849	4028	6021	104,08
1_2_2	2092	4407	4596	6121	146,08
1_2_2	2162	4410	4627	6221	146,38
1_2_2	2348	4849	5034	6321	186,17
1_2_2	2287	4873	5043	6421	187,83
1_2_2	2549	5254	5455	6521	227,05
1_2_2	2649	5572	5804	6721	262,57

blème d'optimisation étant $NP\text{-}Difficile$, nous avons étendu l'approche proposée dans le cadre du problème (FGED) pour prendre en compte cet aspect supplémentaire. Pouvant à premiers abords être perçu comme un problème d'affectation des forces de secours puis de routage, nous avons opté pour une approche inverse se basant sur le routage des individus suivi du déploiement des forces de secours pour les épauler durant leur périple. A travers les expérimentations sur de petites instances, nous avons pu montrer que notre méthode était efficace avec une déviation de $7,47\%$ par rapport aux meilleures solutions connues. Nous avons aussi pu le mettre à l'échelle afin de l'appliquer à 8 instances de scénarios catastrophes de taille réelle touchant la ville de Nice. Ces dernières montrent une amélioration de la qualité des solutions qui est en moyenne de $4,3\%$ par rapport à l'approche sans déploiement des forces de secours. Aussi, le problème d'évacuation FGEDA permet aux décideurs de savoir pour un scénario de catastrophe donné, s'ils disposent d'assez de forces de sécurité afin que les opérations d'évacuation se passent bien. Il permet aussi à ces derniers d'anticiper le déploiement des forces de secours et de les positionner à des endroits

TABLE 4.29 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_1_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_1_1	1725	4037	4182	7791	174,12
2_1_1	1668	4036	4200	7891	182,31
2_1_1	2534	5455	5644	7991	327,43
2_1_1	2479	5472	5694	8091	319,71
2_1_1	3140	6328	6546	8291	450,09
2_1_1	3599	6822	7031	8391	577,38
2_1_1	3607	6832	7124	8491	582,62
2_1_1	3891	7251	7484	8591	699,52
2_1_1	4124	7601	7822	8791	835,89

TABLE 4.30 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_1_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_1_2	534	5740	5936	14148	300,22
2_1_2	3505	7637	7843	14348	547,46
2_1_2	3540	7640	7850	14448	552,69
2_1_2	4258	8718	9001	14548	782,58
2_1_2	4172	8721	9070	14648	792,99
2_1_2	4796	9374	9620	14748	1036,70
2_1_2	5260	9881	10164	14948	1257,03
2_1_2	5597	10430	10740	15148	1450,17

stratégiques pour que le plus de personnes indemnes puissent être mises à l'abri.
Cependant il demeure une certaine incertitude quant aux niveaux de dommage des différentes infrastructures du réseau du fait de l'état de crise. En effet, il est possible d'avoir une certaine idée des dommages des infrastructures mais ce n'est qu'après un certain temps que la véritable gravité de la situation est évaluée. Dans la section suivante, nous prenons en compte cette variabilité des données afin de mieux déployer les forces de secours.

4.4 Modèle de flot robuste pour l'évacuation de personnes

4.4.1 Description du problème

Ce travail est effectué en collaboration avec Marc Goerigk qui effectue son post-doctorat au sein de la faculté de mathématiques de l'université technique de Kaiserslautern (Optimization Research Group). Nous avons montré dans la section 4.3 comment il est possible d'améliorer la sécurité associée au processus d'évacuation lors d'une catastrophe naturelle. Toutefois comme nous l'avons déjà dit, les niveaux de dommages peuvent être très variables

TABLE 4.31 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_2_1

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_2_1	1368	2497	2625	6277	47,04
2_2_1	1908	3180	3329	6477	70,13
2_2_1	2228	3587	3779	6677	90,02
2_2_1	2400	3832	3977	6877	102,36
2_2_1	2556	4034	4219	7077	117,52
2_2_1	2772	4213	4362	7277	130,69

TABLE 4.32 – Résultats de tests de l'algorithme 4.2.4.4 appliqué à l'instance 2_2_2

Instance	Personnes indemnes min	Personnes indemnes moy	Personnes indemnes max	Durée de l'évacuation	Temps d'exécution
2_2_2	1747	3223	3370	6277	66,85
2_2_2	2356	4042	4214	6477	96,18
2_2_2	2752	4519	4704	6677	128,96
2_2_2	2969	4843	5080	6877	156,78
2_2_2	3190	5094	5262	7077	181,11
2_2_2	3193	5126	5319	7177	181,58
2_2_2	3455	5341	5516	7277	209,66

ou mal évalués dans les premiers instants suivant la catastrophe. Aussi, il existe une certaine incertitude quant aux données relatives à la sécurité pouvant mener à une mauvaise utilisation des forces de secours dont le rôle est de rendre l'évacuation plus sûre. Comme les niveaux de dommages peuvent être connus dans l'absolu, nous supposerons qu'ils sont définis dans un intervalle donné. Ainsi, dans le meilleur des cas de figure le niveau de sécurité d'un sommet correspondra à la borne supérieure de cet intervalle, alors que dans le pire des cas, le niveau de sécurité correspondra à la borne inférieure. Cette modélisation de l'incertitude de la sécurité est réaliste car même si la résistance intrinsèque d'une structure peut être garantie face à un type de sinistre, la chute d'autres éléments proches comme des bâtiments ou d'arbres consécutifs au sinistre peuvent l'endommager encore plus et ainsi réduire son niveau de sécurité.
Ne sachant pas où ces événements de sur-risques peuvent intervenir dans le réseau, notre but sera de déployer les forces de secours de manière à ce que quels que soient les sur-risques qui puissent intervenir (i.e. répliques de tremblement de terre) en garantissant que le maximum d'individus indemnes puisse atteindre les différents centres de secours.

4.4.2 Problèmes connexes

Pour y parvenir, nous étendons le concept de flot généralisé entier avec diviseurs et amélioration que nous avons déjà présenté à la section 4.3 afin de prendre en compte le fait

que les niveaux de dommages des sommets ne sont pas connus dans l'absolu. Par exemple, nous pouvons considérer le cas où des personnes doivent quitter une zone dangereuse à pied après un sinistre pouvant être un tremblement de terre ou une inondation. Suivant les chemins empruntés ces individus peuvent traverser des ponts, des routes partiellement immergées ou une zone boisée encourant comme risque que le pont cède, que le débit de l'eau sur la route immergée augmente ou que des branches cassées ou arbres fragilisés ne chutent sur eux. Nous avons déjà montré comment modéliser ces phénomènes en utilisant un modèle de flot généralisé entier avec diviseurs (voir section 4.2, 4.3 et [Ndiaye *et al.*, 2014e]). Dans le domaine de l'optimisation robuste avec des données incertaines, nous nous referons aux travaux de [Kouvelis et Yu, 1997, Aissi *et al.*, 2009, Bertsimas et Sim, 2004, Bertsimas et Sim, 2003, Ben-Tal *et al.*, 2009, Goerigk et Schöbel, 2013]. La prise en compte de l'aspect robuste dans le déploiement des forces de secours peut être effectué en considérant une approche en deux étapes ajustable (voir [Ben-Tal *et al.*, 2004]). Toutefois, le problème d'optimisation robuste résultant combine plusieurs problèmes d'optimisation $NP - Difficile$. En outre, la taille des instances à résoudre fait qu'il devient impossible d'utiliser une approche directe afin de résoudre l'ensemble du problème.

Afin de simplifier le problème traité, nous relaxons la contrainte d'intégrité du flot afin d'être entre $4,28\%$ et $6,60\%$ du flot optimal entier (voir 4.23) et nous utilisons une méthode itérative de génération des scénarios (voir [Agra *et al.*, 2013, Billionnet *et al.*, 2014, Gabrel *et al.*, 2014, Zeng et Zhao, 2013]).

4.4.3 Flot Robuste Généralisé Entier avec Diviseurs et Amélioration

Afin de prendre en compte la notion de robustesse du déploiement des forces de secours, nous étendons le modèle mathématique défini pour le problème FGEDA 4.3.1.1. Ceci nous permet de modéliser le fait que la sécurité des sommets ne peut qu'être estimée et que nous ne pouvons connaitre leur niveau réel de dangerosité. A la place, nous supposons comme connu un sous-ensemble de scénarios incertains $\mathcal{U} \subseteq \mathbb{R}^{|V|}$ pouvant affecter les différents points critiques du réseau routier. A chacun de ces scénarios sont associées des valeurs différentes de niveau de sécurité initial Pr. Ainsi, nous devons décider où envoyer les forces de secours avant de connaitre l'état complet de dégradation des lieux. Une fois cette affectation effectuée, et que les forces de secours sont déployées, le niveau de dommage des sommets du scénario considéré est révélé et les personnes à évacuer prennent les chemins plus sûrs disponibles après les améliorations effectuées par les forces de secours. En considérant une approche robuste pour répondre au pire cas de figure, nous pouvons nous poser la question suivante : "Quels sommets du réseau doivent être sécurisés afin de maximiser le nombre fractionnaire d'individus indemnes mis en sécurité dans le pire scénario qui puisse se produire ?"
Cette approche suit le principe de robustesse ajustable (voir [Ben-Tal *et al.*, 2004]) en supposant que les variables Z_i doivent être fixées au tout début sans attendre (i.e. ici et maintenant) et que les variables $F(i,j)$ et H_i peuvent être fixées par la suite après avoir eu une meilleure visibilité sur le problème à résoudre (i.e. attendre et voir). Une approche similaire en deux étapes peut être trouvé dans [Liebchen *et al.*, 2009] sachant que les données de la première phase peuvent changer une fois que le scénario définitif est connu.

4.4.3.1 Ensemble fini d'incertitudes

Nous considérons dans un premier temps un ensemble fini de scénarios $\mathcal{U}^f = \{Pr^1, \ldots, Pr^N\}$ associés à une catastrophe donnée et posons $\mathcal{N} := \{1, \ldots, N\}$.

Nous supposons aussi que $Pr_i^k + Prg_i \leq 1 \ \forall i \in V,\ k \in \mathcal{N}$. Le problème d'optimisation robuste à deux étapes résultant peut être modélisé comme suit :

$$\max \min_{k \in \mathcal{N}} \sum_{i \in \Gamma_P^-} F_{(i,P)}^k \tag{4.13}$$

S.C

$$0 \leq F_{(i,j)}^k \leq C_{(i,j)} \qquad \forall (i,j) \in E, k \in \mathcal{N} \tag{4.14}$$

$$Z_i \in \{0,1\} \qquad \forall i \in V \tag{4.15}$$

$$H_i^k \geq 0 \qquad \forall i \in V, k \in \mathcal{N} \tag{4.16}$$

$$\sum_{i \in \Gamma_S^+} F_{(S,i)}^k \leq B_S \qquad \forall k \in \mathcal{N} \tag{4.17}$$

$$\sum_{i \in V} Co_i \times Z_i \leq RS \tag{4.18}$$

$$H_i^k \leq HV \times Z_i \qquad \forall i \in V, k \in \mathcal{N} \tag{4.19}$$

$$H_j^k \leq \sum_{i \in \Gamma_j^-} F_{(i,j)}^k \qquad \forall j \in V, k \in \mathcal{N} \tag{4.20}$$

$$Pr_j^k \sum_{i \in \Gamma_j^-} F_{(i,j)}^k + Prg_j H_j^k = \sum_{q \in \Gamma_j^+} F_{(j,q)}^k \qquad \forall j \in V, k \in \mathcal{N} \tag{4.21}$$

Nous noterons aussi FRGEDA($\mathcal{U}^f$) lorsque nous faisons référence à un ensemble de scénarios incertains associés à un sinistre nécessitant une évacuation. Pour chaque scénario $k \in \mathcal{N}$, nous avons introduit les variables F^k et H^k qui dépendent du niveau de sécurité des sommets associés à un scénario. Il est est possible de réécrire la fonction objectif afin d'éliminer la minimisation comme suit :

$$\max \alpha$$

$$\alpha \leq \sum_{i \in \Gamma_P^-} F_{(i,P)}^k \qquad \forall k \in \mathcal{N}$$

(4.14–4.21)

Notons que tout comme le problème FGEDA, seules les variables de décision Z_i correspondant au déploiement des forces de secours. Une fois fixées, nous n'avons plus qu'à résoudre un problème FGED en relaxant la contrainte d'intégrité du flot pour tous les scénarios afin de trouver le pire cas de figure pouvant se présenter.

4.4.3.2 Ensemble borné d'incertitudes

Afin de rendre le modèle précédent plus réaliste, nous supposons que le niveau de dommage de chaque sommet PR_i ne varie pas de 0.0 à 1.0 suivant le sur-risque mais est

défini sur l'intervalle $[\underline{Pr_i}, \overline{Pr_i}]$ avec $\underline{Pr_i}$ le pire niveau de sécurité et $\overline{Pr_i}$ le meilleur niveau de sécurité pour un scénario fixé. Il serait possible de fixer le niveau de sécurité de tous les sommets à $[\underline{Pr_i}$ mais cela reviendrait à être pessimiste et à considérer que tous les sommets seront dégradés par des événements extérieurs. Une hypothèse réaliste est de supposer que le nombre de sommets soumis à un sur-risque et qui dévieront de leur meilleurs $\overline{Pr_i}$ est borné (voir [Bertsimas et Sim, 2004]). Nous modélisons cette borne comme suit :

$$\mathcal{U} = \mathcal{U}(K) = \Big([\underline{Pr}_1, \overline{Pr}_1] \times \ldots \times \left[\underline{Pr}_{|V|}, \overline{Pr}_{|V|}\right] \Big) \cap \Big\{ Pr \in \mathbb{R}^{|V|} \;\Big|\; \#\left\{ i \in V : Pr_i < \overline{Pr_i} \right\} \leq K \Big\}$$

Nous supposons toujours que $\overline{Pr_i} + Prg_i \leq 1$. La borne $K \in \mathbb{N}$ est un paramètre qui permet de contrôler le nombre sommets qui pour un sinistre donné subiront une dégradation supplémentaire due à un sur-risque (i.e. réplique de tremblement de terre, fragilisation de pont,...). Si $K = N$, l'ensemble $\mathcal{U}$ sera constitué du produit cartésien de l'ensemble des intervalles générant ainsi un ensemble exponentiel de scénarios pour un sinistre et avec un niveau de robustesse élevé. A l'opposé, si $K = 0$, cela revient à résoudre le problème d'évacuation initial avec amélioration du réseau à savoir FGEDA. Aussi, ce paramètre K permet de déterminer la qualité des solutions obtenues.

Notons que la modélisation du problème FRGEDA($\mathcal{U}$) en programme linéaire mixte résultant comporte un nombre infini de variables et de contraintes et ne peut donc pas être résolu en pratique. Toutefois lorsque nous avons des niveaux de sécurité $Pr_i < \overline{Pr_i}$, nous pouvons supposer que $Pr_i = \underline{Pr_i}$ comme ces types de scénarios seront dominants dans le pire cas. De ce fait, il devient alors possible d'énumérer l'ensemble des scénarios pertinents liés à un sinistre.

Lemme 6. *Le problème d'optimisation FRGEDA($\mathcal{U}$) peut être résolu à travers un problème de FRGEDA($\mathcal{U}^f$), avec* $|\mathcal{U}^f| = \binom{|V|}{K}$

Nous pouvons ainsi réduire le problème d'optimisation robuste avec un nombre infini de scénarios incertains à un nombre fini de scénarios incertains qui contiennent tous les scénarios possibles où K sommets subiront des dégradations supplémentaires et le reste des sommets des dégradations normales. Le nombre de scénarios restant tout de même exponentiel, il faut considérer une meilleure approche pour résoudre le problème d'évacuation robuste FRGEDA.

4.4.3.3 Évaluation d'une solution

Considérons le sous-problème suivant : "Ayant un nombre limité de sommets améliorables Z, quel scénario de dommages $Pr \in \mathcal{U}$ est responsable du Pire Plan d'Evacuation Optimal en terme de personnes indemnes ?" . Il est possible d'y répondre en modélisant le problème d'optimisation comme suit :

$$
\begin{aligned}
WC(Z) := \min\ & F^*(Pr) \\
& Pr_i = \overline{Pr}_i + (\underline{Pr}_i - \overline{Pr}_i)w_i + Prg_i Z_i && \forall i \in V \\
& \sum_{i \in V} w_i \leq K \\
& w_i \in \{0,1\} && \forall i \in V \\
& Pr_i \geq 0 && \forall i \in V
\end{aligned}
$$

Nous devons ainsi décider si le sommet i subit une dégradation supplémentaire suite au sinistre à l'aide de la variable binaire w_i. Nous voulons ainsi fixer cette variable de manière à ce que le flot de personnes indemnes soit le plus petit possible. Notons que $F^*(Pr)$ représente le flot optimal de personnes indemnes atteignant le sommet puits P dans le cadre du scénario défini par les niveaux de dommages Pr du réseau. Une fois cette transformation effectuée, nous avons de nouveau à résoudre un problème d'évacuation FGED avec une relaxation de la contrainte d'intégrité du flot comme suit :

$$
\begin{aligned}
F^*(Pr) := \max\ & \sum_{i \in \Gamma_P^-} F_{(i,P)} && && (4.22) \\
& \sum_{i \in \Gamma_S^+} f_{(S,i)} \leq B_S && && (4.23) \\
& Pr_j \sum_{i \in \Gamma_j^-} F_{(i,j)} \geq \sum_{q \in \Gamma_j^+} F_{(j,q)} && \forall j \in V && (4.24) \\
& 0 \leq F_{(i,j)} \leq C_{(i,j)} && \forall (i,j) \in E && (4.25)
\end{aligned}
$$

Pour intégrer le calcul de $F^*(Pr)$ au problème du pire plan d'évacuation optimal $WC(Z)$, nous considérons son problème dual qui se présente comme suit :

$$
\begin{aligned}
\min\ & \sum_{(i,j) \in E} C_{(i,j)} \alpha_{(i,j)} + B_S \beta \\
& \alpha_{(i,j)} + \gamma_i - p_j \gamma_j + \chi_S(i)\beta \geq \chi_P(j) && \forall (i,j) \in A \\
& \alpha_{(i,j)} \geq 0 && \forall (i,j) \in A \\
& \beta \geq 0 \\
& \gamma_i \geq 0 && \forall i \in V \\
& \gamma_S = \gamma_P = 0
\end{aligned}
$$

avec

$$
\chi_i(j) = \begin{cases} 1 & \text{si } j = i \\ 0 & \text{sinon} \end{cases}
$$

Nous utilisons cette formulation duale du problème FGED relaxé pour réécrire le problème $WC(Z)$:

$$\min \sum_{(i,j)\in E} C_{(i,j)}\alpha_{(i,j)} + B_S\beta$$
$$\sum_{i\in V} w_i \leq K$$
$$\alpha_{(i,j)} + \gamma_i - (\overline{Pr}_j + (\underline{Pr}_j - \overline{Pr}_j)w_j + Prg_jZ_j)\gamma_j + \chi_S(i)\beta \geq \chi_P(j) \quad \forall (i,j) \in E$$
$$\alpha_{(i,j)} \geq 0 \quad \forall (i,j) \in E$$
$$\beta \geq 0$$
$$\gamma_i \geq 0 \quad \forall i \in V$$
$$\gamma_S = \gamma_P = 0$$
$$w_i \in \{0,1\} \quad \forall i \in V$$

A cause du produit $w_j\gamma_j$, ce modèle mathématique n'est pas linéaire. Nous rappelons qu'à ce stade les variables Z_i sont fixées et que nous cherchons à trouver à affecter un niveau de dommage supplémentaire aux sommets afin de diminuer le nombre de personnes indemnes à la fin de l'évacuation. Aussi, le produit $z_j\gamma_j$ est linéaire dans ce modèle. Pour résoudre le problème de non-linéarité, nous posons $\gamma'_i = w_i\gamma_i$. Nous avons ainsi le modèle suivant :

$$\min \sum_{(i,j)\in E} C_{(i,j)}\alpha_{(i,j)} + B_S\beta \tag{4.26}$$
$$\sum_{i\in V} w_i \leq K \tag{4.27}$$
$$\alpha_{(i,j)} + \gamma_i - (\overline{Pr}_j + Prg_jZ_j)\gamma_j + (\overline{Pr}_j - \underline{Pr}_j)\gamma'_j + \chi_S(i)\beta \geq \chi_P(j) \quad \forall (i,j) \in E \tag{4.28}$$
$$\gamma'_i \leq \gamma_i \quad \forall i \in V' \tag{4.29}$$
$$\gamma'_i \leq HVw_i \quad \forall i \in V' \tag{4.30}$$
$$\alpha_{(i,j)} \geq 0 \quad \forall (i,j) \in E \tag{4.31}$$
$$\beta \geq 0 \tag{4.32}$$
$$\gamma_i, \gamma'_i \geq 0 \quad \forall i \in V \tag{4.33}$$
$$\gamma_s = \gamma_t = 0 \tag{4.34}$$
$$w_i \in \{0,1\} \quad \forall i \in V \tag{4.35}$$

Avec HV une constante assez grande. Soit i^* le sommet ayant le plus de successeurs, Alors $HV = \max(C_{(i,j)}) \times \Gamma^+_{i^*}$ est une valeur suffisante pour la constante.

4.4.4 Algorithme de résolution

4.4.5 Génération des scénarios

En utilisant le modèle mathématique défini par les contraintes (4.26–4.35) pour le problème $WC(Z)$ avec Z fixé, nous pouvons trouver la valeur de la fonction objectif correspondant à cette affectation des forces de secours mais aussi trouver le scénario de sur-risque w pour laquelle cette valeur est atteinte. Dans cette section nous présentons comment

cette dernière est utilisée dans l'algorithme de résolution. Nous commençons avec un sous-ensemble fini de scénarios $\mathcal{U}^0 \subseteq \mathcal{U}$. Ce sous-ensemble peut être vide, $\mathcal{U}^0 = \emptyset$ ou bien nous pouvons considérer qu'aucun des sommets n'a subi de sur-risque. $\mathcal{U}^0 = \{\overline{Pr}\}$. Ainsi, résoudre le problème FRGEDA($\mathcal{U}^0$) nous permet d'obtenir une solution au problème d'évacuation associée à l'affectation des forces de secours Z^0 ainsi que le nombre fractionnaire de personnes indemnes OBJ^0. Afin de trouver la meilleure contre mesure à cette affectation des forces de secours, nous résolvons le problème $WC(Z^0)$ afin de déterminer un scénario de dégradation supplémentaire des sommets w^0 ainsi que le nouveau nombre fractionnaire de personnes indemnes WC^0 atteignant le sommet puits P. De ce fait, poser $\mathcal{U}^1 := \mathcal{U}^0 \cup \{w^0\}$ permet de considérer une nouvelle ensemble de solutions qui peut être utilisé pour déterminer un autre déploiement des forces de secours. Ce processus de génération des scénarios est ainsi poursuivi de manière itérative. Nous arrêtons l'algorithme lors du déploiement des forces de secours Z^k, si la fonction objectif du pire scénario $WC(Z^k)$ est égale à la fonction objectif du problème FRGEDA($\mathcal{U}^k$). Comme tout problème d'évacuation FRGEDA($\mathcal{U}^k$) pour l'ensemble de sénario $\mathcal{U}^k \subseteq \mathcal{U}$ est une relaxation du problème global FRGEDA($\mathcal{U}$), la valeur optimale de la fonction objectif associée au déploiement de forces de secours Z^k est une borne supérieure pour la solution optimale du problème FRGEDA($\mathcal{U}$). Aussi, la valeur optimale de la fonction objectif est atteinte pour le pire scénario lorsque cette dernière est égale à la borne supérieure. Nous présentons la méthode de résolution dans l'Algorithme 1.

Algorithm 1 (Algorithme exact pour résoudre FRGEDA($\mathcal{U}$))

Require: An instance of FRGEDA($\mathcal{U}$).

1: $k \leftarrow 0$
2: $\mathcal{U}^0 \leftarrow \{\overline{p}\}$
3: Solve FRGEDA($\mathcal{U}^k$). Let z^k be the resulting solution for the improvements, et OBJ^k the resulting objective value.
4: Solve $WC(z^k)$. Let p^k be the resulting scenario, et WC^k the resulting objective value.
5: **if** $OBJ^k = WC^k$ **then**
6: **return** Optimal improvements z^k et objective value OBJ^k.
7: **else**
8: $\mathcal{U}^{k+1} \leftarrow \mathcal{U}^k \cup \{Pr^k\}$
9: $k \leftarrow k + 1$
10: Goto 3
11: **end if**

Notons que la valeur de la fonction objectif OBJ^k est décroissante du fait qu'au fur et à mesure, nous ajoutons des variables et des contraintes au modèle mathématique. Toutefois, la valeur de la fonction objectif associée à WC^k n'est pas toujours croissante. La figure Figure 4.10 donne un exemple illustrant l'exécution de l'Algorithm 1. Les cercles montrent les meilleures solutions trouvées par le modèle WC et qui sont de meilleures qualités que leurs prédécesseurs.

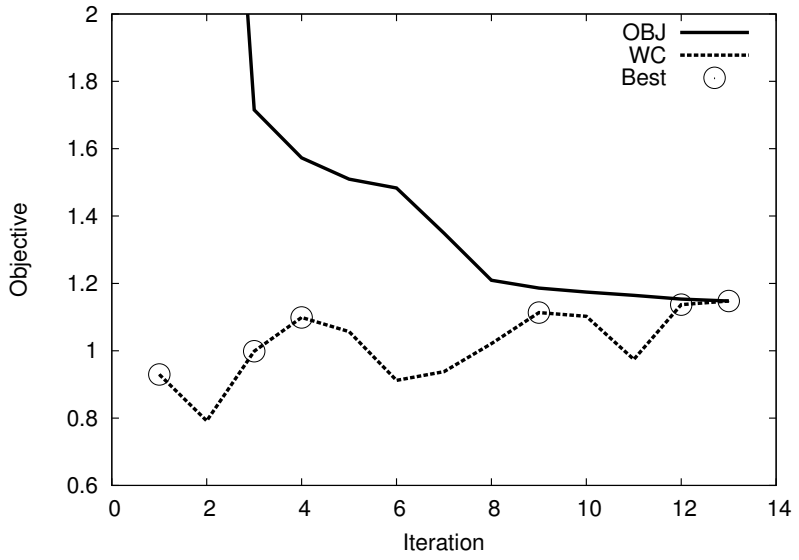


FIGURE 4.10 – Exemple d'execution de l'Algorithm 1. Valeur optimale atteinte lorsque $OBJ^k = WC^k$

4.4.6 Ensemble borné de scénarios

Résoudre le sous-problème défini par les contraintes (4.26–4.35) pour un ensemble d'amélioration de sommets fixé mène à un modèle mathématique ayant $O(|E|)$ variables et $O(|E|)$ contraintes. Toutefois, le problème défini par les contraintes (4.13–4.21) doit être résolu à chaque itération. A la $k^{\text{ième}}$ itération, ce dernier comprend $O(k|E|)$ variables et $O(k|E|)$ contraintes. Ainsi, autant le temps de résolution du sous-problème $WC(Z)$ reste relativement stable entre chaque itération, celui du problème FRGEDA($\mathcal{U}$) augmente à chaque itération de l'algorithme Algorithme 1.

Aussi, si nous avions une méthode permettant de supprimer les scénarios redondants de l'ensemble $\mathcal{U}^k$, alors le temps de calcul pourrait être amélioré. Dans cette optique, nous considérons une variante de l'Algorithm 1 :
Soit $L \in \mathbb{N}$ le paramètre d'un algorithme permettant de contrôler la taille de l'ensemble $\mathcal{U}^k$. Lorsqu'un scénario Pr^k est ajouté à l'ensemble $\mathcal{U}^k$ à la ligne 8 de l'algorithme, nous vérifions si $|\mathcal{U}^k| \leq L - 1$. Si c'est bien le cas, alors le scénario est ajouté comme à l'accoutumée. Si toutefois $|\mathcal{U}^k| = L$ (i.e. la taille maximale de l'ensemble des scénarios est atteinte) nous choisissons et supprimons un des scénarios courants de l'ensemble $\mathcal{U}^k$.
Si la valeur de L choisie est trop petite, cela peut mener à un algorithme qui ne converge pas et boucle dans l'exploration des scénarios. Afin de prévenir ce problème, nous sauvegardons l'ensemble des scénarios qui ont été générés depuis le début dans un ensemble complémentaire $\overline{\mathcal{U}}^k$. Lorsqu'un scénario généré avait déjà été utilisé par le passé (i.e. $Pr^k \in \overline{\mathcal{U}}^{k-1}$), nous augmentons le paramètre L d'une unité.
Pour le choix du scénario qui doit être enlevé de l'ensemble des scénarios courants, plusieurs approches sont possibles. Ces approches permettent d'arriver à la solution optimale du problème FRGEDA($\mathcal{U}$). Ce qui les dédifférencie est leur impact sur le temps de calcul ou le nombre d'itérations pour parvenir à la solution optimale.

Dans notre cas, nous utilisons la stratégie suivante :
Nous enlevons le scénario $Pr^j \in \mathcal{U}^k$ pour lequel $\sum_{i\in\Gamma_P^-} F^j_{(i,P)}$ est maximale dans les solutions les plus récentes pour la résolution du problème FRGEDA. En d'autres mots, le scénario que nous enlevons est celui qui permet au plus de personnes indemnes d'être mises en sécurité car n'oublions pas que nous cherchons le pire scénario qui puisse se produire. Cependant, à cause de la structure max-min de la fonction objectif du problème FRGEDA, une solution optimale ne nécessite pas forcément que le flot soit maximal dans chaque scénario. Afin de s'assurer que ce soit bien le cas, nous utilisons une fonction objectif lexicographique de la forme

$$\max \alpha + \varepsilon \sum_{k\in\mathcal{N}} \sum_{i\in\Gamma_P^-} F^k_{(i,P)}$$

avec $1 \gg \varepsilon > 0$.

4.4.7 Création de scénarios artificiels

Comme le problème d'évacuation FGEDA est déjà $NP\text{-}Difficile$ et que la version robuste FRGEDA ne fait qu'augmenter la difficulté de résolution, nous présentons maintenant une approche heuristique permettant de résoudre des instances de taille raisonnable. En effet, afin d'évaluer la pertinence d'un déploiement des forces de secours sur le réseau Z il faut résoudre le programme linéaire en variables mixtes du problème $WC(z)$. Cette nécessité rend les heuristiques inefficaces car elles se basent sur l'évaluation de plusieurs solutions (i.e. recherche locale, algorithme génétique). A la place, nous voulons une solution du problème FGEDA relaxé associée à un seul scénario $Pr \in \mathbb{R}^{|V|}$ et qui soit assez représentative de l'ensemble d'incertitude $\mathcal{U}$. Il n'est pas nécessaire que le scénario Pr soit réaliste (i.e. $Pr \notin \mathcal{U}$ est accepté). Toutefois, autant une fois que le scénario est fixé il est relativement facile de le résoudre, autant la qualité de ce dernier est déterminée par la manière dont le scénario Pr est créé. Pour cela, nous considérons les cas de figure suivants :

1. Nous posons $Pr = \lambda\underline{Pr} + (1-\lambda)\overline{Pr}$ avec $\lambda \in [0,1]$. Cette approche inclut aussi le meilleur et le pire scénario possibles lorsque $Pr = \underline{Pr}$ ou $Pr = \overline{Pr}$.
2. Il y a aussi le cas où on suppose que le sur-risque est uniformément distribué en fonction du nombre de sommets K soumis au sur-risque ainsi que du nombre total de sommet N. Aussi dans ce cas de figure nous posons $Pr = \lambda\underline{PR} + (1-\lambda)\overline{Pr}$ avec $\lambda = K/N$.
3. Enfin, si on suppose qu'il est irréaliste de considérer que les sur-risques soient uniformément sur les K sommets, il est possible de donner un certain avantage aux sommets par lesquels transitent beaucoup d'évacués. En estimant que sur un graphe avec N sommets, seulement $\sqrt{N}$ de ces derniers seront utilisés du fait que le graphe est dynamique et que l'on utilise souvent les mêmes chemins ou portions de chemins, nous pouvons considérer le cas plus pessimiste où $\lambda = \min\{1, K/\sqrt{N}\}$. Il est possible d'affiner cette estimation si le graphe utilisé a certaines caractéristiques remarquables.

Les trois approches précédentes pour répartir le sur-risque sur K sommets sont des heuristiques non exhaustives qui mènent donc à des solutions heuristiques en matière de

déploiement des forces de secours Z. Afin d'évaluer les solutions, nous avons toujours besoin de résoudre un problème de type $WC(Z)$. Aussi, pour des instances de grande taille, l'évaluation d'une solution pourrait donc nécessiter plus de temps de calcul afin de trouver le meilleur déploiement possible des forces de secours.

4.4.8 Résultats expérimentaux

Nous présentons deux ensembles d'expérimentation. Dans le premier, nous générons aléatoirement un graphe sous forme de grille (grid graph) afin d'évaluer les algorithmes de résolution heuristiques comparés à la méthode exacte. Dans un second temps, nous appliquons l'approche heuristique sur une instance de taille réelle de la ville de Nice qui est notre cas d'étude. Toutes les expérimentations concernant cette partie sont effectuées sur un ordinateur avec un processeur Intel Xeon E5-2670 cadencé à 2.60 GHz et disposant de 20MB de mémoire cache. Le système d'exploitation est un Ubuntu 12.04. et le programme linéaire en variables mixtes est résolu grâce au solveur Gurobi v. 5.5. Le langage de programmation utilisé est le C++ avec comme compilateur gcc v. 4.5.4. et l'option -O3. Pour la première partie des expérimentations sur les données générées aléatoirement, nous utilisons seulement 4 processeurs. Afin de limiter la variance liée à la génération des instances, nous générons pour chaque ensemble de paramétrage 4 instances afin d'obtenir des valeurs moyennes. Pour la seconde partie des expérimentations sur les données de taille réelle, nous utiliserons l'ensemble des 16 processeurs. Les temps de calcul seront exprimés en nombre de signaux d'horloge.

4.4.9 Partie 1 : Instances aléatoires

Instances Nous avons généré 7 types d'instance (i.e. grilles) $\mathcal{I}_\ell$, $\ell = 3, \ldots, 9$ afin de pouvoir comparer notre méthode de résolution heuristique avec l'approche exacte pour résoudre le problème d'évacuation FRGEDA. Chaque graphe est une grille de taille $\ell \times \ell$, où les évacués doivent partir du coin inférieur gauche représentant le sommet source S afin de se rendre au sommet supérieur droit qui représente le sommet puits P. Dans l'ensemble des graphes, nous supposerons que nous évacuons une centaine d'individus (i.e. $B_S = 100$). La quantité de ressources utilisables par les forces de secours RS peut aussi être assimilée au nombre de groupes de forces de secours FS que l'on peut déployer sur le réseau. La génération des valeurs qui leur sont associées sera effectuée de manière uniforme dans l'intervalle de valeurs $\{1, \ldots, \ell\}$. La capacité des arcs sera aussi générée dans l'intervalle de valeurs $\{1, \ldots, 100\}$ alors que les sur-risques $\underline{Pr}_i \in [0.001, 0.999]$, $\overline{Pr}_i \in [\underline{p}_i, 1]$. Les améliorations en cas de déploiement seront définies par la suite avec $Prg_i \in [0, 1 - \overline{Pr}_i]$. Le coût de sécurisation d'un sommet Co_i sera aléatoirement fixé dans l'intervalle $[0.5, 1.5]$. A ces 7 valeurs possibles de l (i.e. $\ell = 3, \ldots, 9$), nous générons 20 instances différentes d'où un total de 140 instances.
Pour résoudre les instances, nous utilisons Algorithm 1 et sa variante permettant de borner la taille de l'ensemble d'incertitude (voir section 4.4.6) avec comme borne $L = 5$ et $L = 10$.

Pour faire référence aux différentes solutions associées aux algorithmes, nous utilisons OPT-F lorsque l'ensemble des scénarios est considéré et OPT-5, OPT-10 lorsque nous utilisons la version bornée de l'algorithme de résolution. En outre, nous déterminons les

Instance	W(0.00)	W(0.25)	W(0.50)	W(0.75)	W(1.00)	W(K/n)	W($K/\sqrt{N}$)
$\mathcal{I}_3$	0.96	0.98	0.99	0.97	0.99	0.98	0.99
$\mathcal{I}_4$	0.89	0.91	0.92	0.93	0.95	0.90	0.93
$\mathcal{I}_5$	0.89	0.90	0.92	0.94	0.93	0.88	0.89
$\mathcal{I}_6$	0.76	0.76	0.83	0.89	0.88	0.75	0.85
$\mathcal{I}_7$	0.80	0.81	0.83	0.83	0.84	0.80	0.85
$\mathcal{I}_8$	0.78	0.78	0.79	0.82	0.81	0.78	0.82
$\mathcal{I}_9$	0.73	0.73	0.83	0.82	0.83	0.74	0.84

TABLE 4.33 – Valeur moyenne relative du nombre fractionnaire d'individus indemnes

solutions heuristiques en générant des scénarios artificiels comme décrit dans la section 4.4.7 avec $\lambda \in \{0,00; 0,25; 0,50; 1,00; K/n, K/\sqrt{N}\}$. Les solutions qui leur sont associées seront présentées comme W(λ). Pour chaque instance ainsi définie, nous donnons la valeur de la fonction objectif ainsi que le temps de calcul associé.

Résultats Les résultats sont présentés dans les tableaux 4.33–4.36. Le tableau 4.33 nous permet de comparer l'approche exacte et notre approche heuristique en comparant l'écart moyen sur le nombre fractionnaire d'individus mis en sécurité au terme de l'évacuation. Les valeurs sont normalisées de manière à ce que $1,0$ corresponde à la solution pour laquelle le déploiement des forces de secours est optimal pour le pire scénario considéré. Ainsi, la valeur de $0,96$ en haut et à gauche du tableau signifie qu'en moyenne sur 20 instances issues du graphe (grille) de taille 3×3 les meilleures solutions en affectation des forces de secours ainsi que de routage des individus dévient de 4% de la solution optimale (i.e. $100\% - 96\%$). Une des premières remarques que nous pouvons effectuer est que le nombre fractionnaire d'individus indemnes décline lorsque l'on augmente la taille du graphe. Ceci reste cohérent avec les observations effectuées dans le cadre des problèmes FGED et FGEDA. En effet, plus ces derniers doivent traverser de sommets dangereux, plus le nombre fractionnaire de personnes indemnes baisse. Nous pouvons aussi noter que plus les scénarios considérés sont pessimistes, plus notre approche est robuste et permet de trouver des solutions proches des solutions optimales (i.e. $\lambda \in 0,75; 1,0; K/\sqrt{N}$). Les algorithmes associés à ces paramétrages n'indiquent pas de différences majeures en terme de personnes indemnes sauvées.

Le tableau 4.34 présente le temps d'exécution de nos heuristiques. Les résultats sont similaires à ceux du tableau 4.33. Tous les algorithmes ont besoin d'une solution du problème FRGEDA ainsi que le pire scénario associé afin de pouvoir évaluer cette dernière. Toutefois, autant les temps de calcul liés aux précédents paramétrages sont fonction de la taille des graphes, autant ce n'est pas le cas des algorithmes itératifs exacts comme le montrent les résultats présentés dans le tableau 4.35.

Nous présentons séparément les temps d'exécution liés aux bornes inférieures pour le problème $WC(Z)$ ainsi que pour la borne supérieure à savoir le problème FRGEDA. Comme le montrent les résultats, la majeure partie du temps de résolution de l'approche exacte est due à la résolution du problème FRGEDA qui est de plus en plus difficile à résoudre à chaque itération. En comparant les trois approches exactes proposées, nous pouvons constater que le fait de borner le nombre de scénarios d'incertitude ne nous aide pas à réduire le temps de résolution et peut même dans les graphes de grande taille augmenter

Instance	W(0.00)	W(0.25)	W(0.50)	W(0.75)	W(1.00)	W(K/n)	W($K/\sqrt{n}$)
$\mathcal{I}_3$	0.02	0.02	0.02	0.02	0.02	0.02	0.02
$\mathcal{I}_4$	0.03	0.03	0.03	0.03	0.03	0.03	0.03
$\mathcal{I}_5$	0.05	0.05	0.05	0.05	0.05	0.05	0.05
$\mathcal{I}_6$	0.10	0.09	0.09	0.10	0.11	0.10	0.11
$\mathcal{I}_7$	0.18	0.16	0.17	0.16	0.27	0.17	0.23
$\mathcal{I}_8$	0.69	0.77	0.77	0.70	0.66	0.73	0.66
$\mathcal{I}_9$	1.96	1.89	1.22	1.67	1.90	2.00	1.67

TABLE 4.34 – Temps de calcul des heuristiques en secondes

	LB ($WC(Z)$)			UB ($FRGEDA$)			Total		
Instance	OPT-F	OPT-5	OPT-10	OPT-F	OPT-5	OPT-10	OPT-F	OPT-5	OPT-10
$\mathcal{I}_3$	0.02	0.01	0.01	0.02	0.02	0.02	0.04	0.03	0.03
$\mathcal{I}_4$	0.03	0.03	0.03	0.05	0.04	0.04	0.08	0.07	0.07
$\mathcal{I}_5$	0.08	0.08	0.08	0.18	0.17	0.17	0.26	0.25	0.25
$\mathcal{I}_6$	0.46	0.56	0.45	2.69	2.64	2.44	3.15	3.20	2.89
$\mathcal{I}_7$	2.03	2.78	2.04	14.41	18.29	13.93	16.44	21.07	15.97
$\mathcal{I}_8$	9.49	14.59	10.73	153.89	192.99	180.56	163.38	207.58	191.29
$\mathcal{I}_9$	47.56	98.39	66.94	370.96	884.88	521.70	418.53	983.27	588.64

TABLE 4.35 – Temps de calcul des méthodes exactes en secondes

ce dernier.

Comme prévu, l'utilisation de cette variante, augmente en contrepartie le nombre d'itérations nécessaires à la convergence de la borne inférieure et de la borne supérieure (voir Tableau 4.36) alors que dans le même temps la durée de résolution des itérations diminue dans le plupart des cas. Une comparaison directe des temps de résolution entre OPT-F et OPT-5 pour les instances nécessitant au moins 6 calculs de bornes supérieures est montrée par la figure Fig 4.11. Afin de pouvoir mieux visualiser le graphique, ce dernier a été normalisé en appliquant un double logarithme. Il est alors plus facile de constater que le fait de borner la taille de l'ensemble de scénarios d'incertitudes permet de réduire la durée de résolution de quelques instances alors que la plupart du temps cela l'augmente. Aussi des recherches supplémentaires doivent être effectuées afin de trouver une meilleure stratégie qui permettent de mieux choisir quel scénario doit être éliminé à chaque itération.

	#UB			Temps de calcul par UB		
Instance	OPT-F	OPT-5	OPT-10	OPT-F	OPT-5	OPT-10
$\mathcal{I}_3$	2.05	2.05	2.05	0.01	0.01	0.01
$\mathcal{I}_4$	2.30	2.30	2.30	0.02	0.02	0.02
$\mathcal{I}_5$	3.15	3.15	3.15	0.05	0.05	0.05
$\mathcal{I}_6$	5.25	6.20	5.30	0.30	0.22	0.27
$\mathcal{I}_7$	6.00	7.48	6.10	1.29	1.09	1.21
$\mathcal{I}_8$	8.85	12.90	10.00	12.90	10.24	12.70
$\mathcal{I}_9$	7.70	12.15	8.90	24.63	26.58	24.29

TABLE 4.36 – Nombre moyen de bornes supérieures et leur temps de calcul en secondes

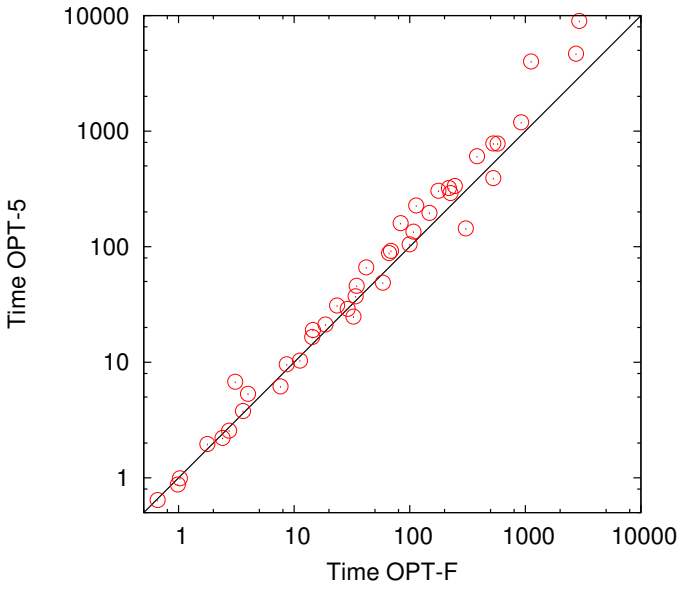


FIGURE 4.11 – Comparaison des temps de calcul de OPT-F et OPT-5 en secondes

En résumé, les résultats obtenus montre que l'approche exacte n'est pas utilisable lorsque la taille du graphe augmente car la durée de résolution croit de manière exponentielle. En effet, le nombre d'itération ainsi que la durée de résolution des itérations augmentent. A l'opposé, l'approche heuristique est beaucoup moins sensible à l'augmentation de la taille des instances et mène à des solutions qui dans les pires scénarios considérés se trouvent à 20% de la solution optimale (i.e. $100\% - 80\%$).

4.4.10 Partie 2 : Instances réalistes

Instances Afin de résoudre des instances réelles, nous avons testé notre approche heuristique sur notre cas d'étude à savoir la ville de Nice dont nous étudions en particulier l'état du centre ville. Afin de pouvoir l'appliquer, nous avons simplifié le graphe de la ville tout en gardant la topologie du graphe initial. Ce dernier comporte 500 sommets et 1124 arcs. Le sinistre considéré est celui du tremblement de terre de 1887 de Ligure d'une magnitude de 6,9 sur l'échelle de richter. L'épicentre du séisme a été relocalisé au large de la ville de Nice générant ainsi un petit tsunami. En se basant sur les simulations du sinistre, les niveaux de dommages sur le bâti et les infrastructures sont donnés par la figure 4.13. La figure 4.12 quant à elle montre l'instance du problème d'évacuation considéré avec les points de rassemblement des individus ainsi que les centres de secours. Afin de déterminer la population impactée, nous utilisons la même approche présentée à la section 4.2.5. Comme le montre la figure 4.13, les centres de secours se trouvant en dehors de la zone dangereuse (i.e. zone en rouge). Dans notre instance de taille réelle, nous avons 17 points de rassemblements et 7 centres de secours. Sachant que ni les données en rapport avec les forces de secours ni les ressources matérielles disponibles ne sont connues, nous supposons que chaque sommet endommagé nécessitera un cout aléatoire de réparation défini dans l'intervalle $[0.5, 1.5]$ avec un cout unitaire de réparation estimé à 10.000 Euros.

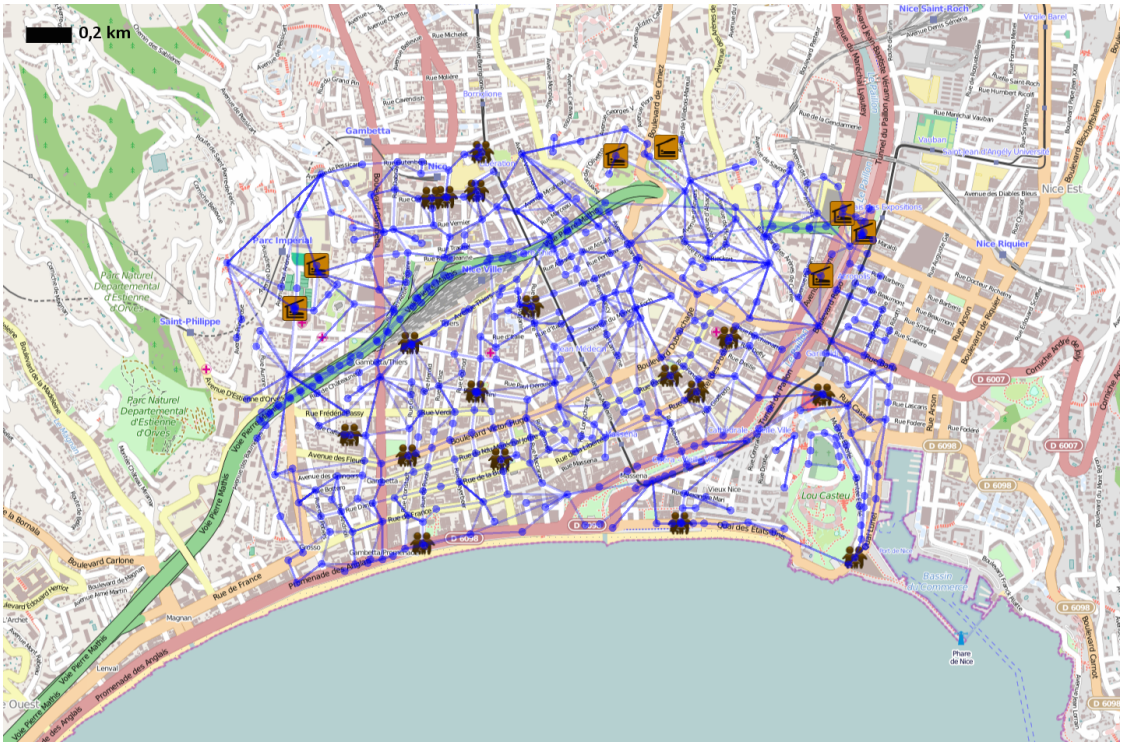


FIGURE 4.12 – Graphe simplifié de la ville de Nice

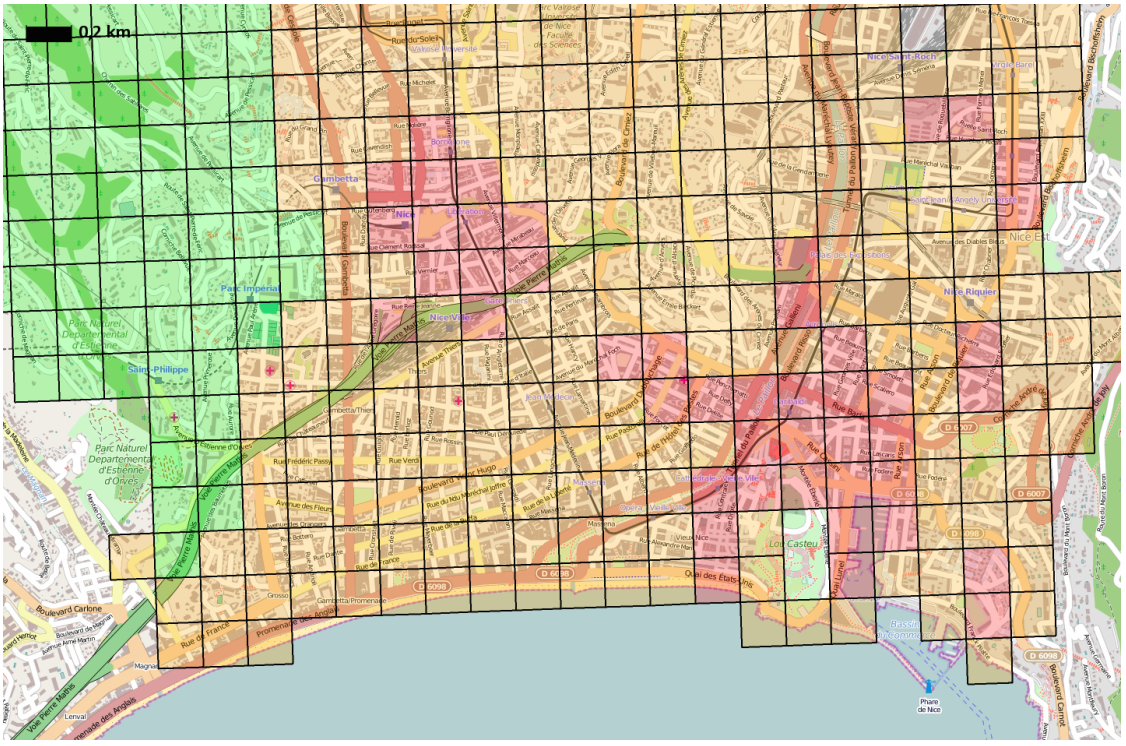
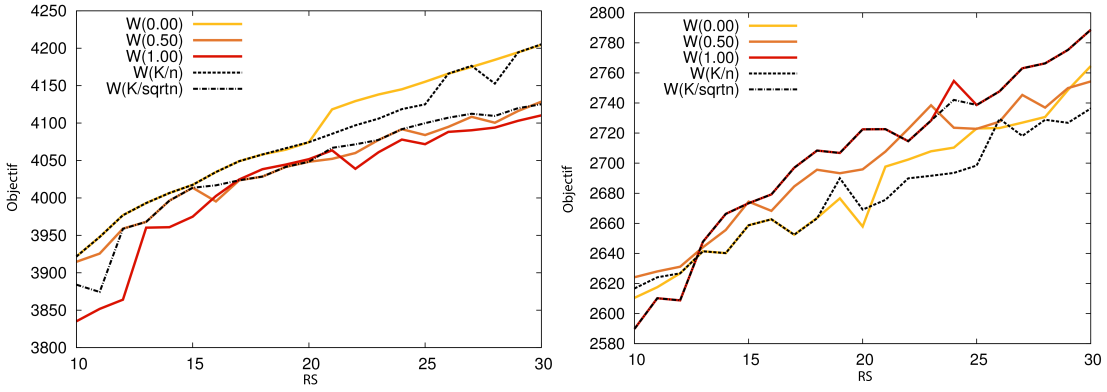


FIGURE 4.13 – Niveaux de dommage du scénario de tremblement de terre Ligure

Nous considérons deux ensembles de scénarios, $\mathcal{U}(10)$ (i.e. 2% des sommets seront dans le pire état sécurité possible) et le scénario plus réaliste $\mathcal{U}(50)$ (i.e. 10% des sommets seront dans le pire état). Nous résolvons ces instances de manière heuristique en utilisant $\lambda \in \{0.00, 0.50, 1.00, K/n, K/\sqrt{n}\}$ tout en faisant varier la quantité de ressources disponibles $RS \in [10, 30]$. Afin de réduire le temps de résolution, nous imposons une limite de 120 secondes pour la détermination de chaque borne inférieure.

Résultats Nous comparons la fonction objectif des heuristiques avec comme paramètres $K = 10$ et $K = 50$ dont les valeurs sont présentées par les figures 4.14(a) et 4.14(b). Alors que pour le cas de figure le plus optimiste (i.e. $K = 10$) $\lambda = 0.00$ et $\lambda = K/n$ donnent des résultats qui dominent les autres paramètres en permettant d'optimiser le nombre fractionnaire de personnes indemnes. Ce n'est toutefois plus le cas en considérant le scénario le plus pessimiste où 50 sommets subiront des dégradations supplémentaires. En outre, sur l'ensemble des tests effectués, le paramétrage utilisant $\lambda = K/\sqrt{n}$ permet d'obtenir un meilleur compromis au regard des deux scénarios de base considérés. En outre, dans l'ensemble des tests effectués, le calcul des bornes supérieures pour le problème ne nécessite que quelques secondes alors que le calcul des bornes inférieures utilise l'intégralité des 120 secondes disponibles.



(a) $K = 10$ (b) $K = 50$

FIGURE 4.14 – Valeurs des fonctions objectifs en faisant varier les ressources disponibles (i.e RS).

Les résultats présentés par la figure 4.14 permettent aux autorités compétentes de répondre à deux questions : "Quel est le gain espéré en terme de sécurité en investissant des ressources dans les infrastructures du réseau lors d'une catastrophe naturelle ?" et "Combien de personnes indemnes peuvent être mises en sécurité une fois les forces de secours déployées ?". Aussi sur une population sinistrée estimée entre 2500 et 4200 personnes, le déploiement des forces de secours permet en moyenne d'avoir pour chaque sommet hautement endommagé restauré 8 évacués supplémentaires pouvant le traverser en étant indemnes. Lorsqu'un sommet est faiblement endommagé, son amélioration augmentera en moyenne le nombre de personnes indemnes pouvant le traverser de 15 individus. Ces résul-

tats permettront aux décideurs d'avoir des indicateurs leur permettant de mieux préparer une situation de crise.

4.4.11 Conclusion

Dans cette section, nous avons proposé une nouvelle manière de modéliser les problèmes d'évacuation avec la prise en compte de la notion des ressources matérielles et humaines pouvant être mises en œuvre afin de rendre une potentielle évacuation la plus réaliste et la plus sûre possible.
Ce modèle prend aussi en compte le fait que certaines des données soient incertaines comme les niveaux de dommage subis par les infrastructures. Du fait que le nombre de scénarios possibles, mais aussi que le modèle mathématique soient de tailles exponentielles, il est impossible de résoudre le programme linéaire mixte associé pour des instances de taille réelle à l'aide d'un solveur mathématique. Afin de pallier ce problème, nous avons proposé une approche heuristique qui augmente itérativement la taille du problème traité afin de trouver le pire scénario possible pour une solution courante proposée.
L'approche heuristique proposée a été appliquée sur une instance de taille réelle de la ville de Nice sur un ensemble de deux scénarios de base avec pour chacun un certain nombre de sommets subissant des sur-risques. Nous avons aussi proposé plusieurs paramétrages permettant d'estimer l'impact d'un sinistre sur une ville ainsi que les incertitudes qui en découlent au regard de la "non-connaissance" complète de l'état de la ville.
D'après ces expérimentations, les forces de secours doivent prioritairement être déployées sur des sommets ayant subi de faibles dommages et il n'est pas pertinent de déployer ensemble des moyens disponibles pour restaurer des infrastructures lourdement endommagées.

4.5 Conclusion du problème d'évacuation avec pertes

Dans cette partie, nous avons traité trois types de problèmes d'évacuation avec pertes avec des niveaux de difficulté et de réalisme croissants. Dans un premier temps nous avons modélisé grâce au problème FGED le fait que les personnes puissent être blessées durant le processus d'évacuation au point de nécessiter d'être prises en charge (i.e. fractures, brulures intoxication au monoxyde de carbone,...). Ce problème d'optimisation étant $NP\text{-}Difficile$ nous avons avons proposé une métaheuristique à base de colonies de fourmis avec une déviation moyenne de $4,46\%$ par rapport aux meilleures solutions connues. Ce faisant, nous avons pu appliquer cette dernière à des instances de taille réelle de la ville de Nice et ainsi qu'avec des scénarios-catastrophes d'ampleur variable.
Afin de prendre en compte le fait que les individus ne seront pas livrés à eux-mêmes en cas de catastrophe d'origine naturelle ou industrielle, nous avons ajouté grâce au problème FGEDA le fait que des forces de secours (i.e. policiers, pompiers, infirmiers,...) puissent être déployés dans le réseau pour aider les évacués. Étant aussi un problème d'optimisation $NP\text{-}Difficile$, nous avons étendu notre méthode d'optimisation par algorithme de fourmis en y intégrant la recherche Tabou. Cette dernière permet d'explorer l'espace de recherche associé à l'affectation des forces de secours sur les points critiques du réseau. La déviation de cette approche par rapport aux meilleures solutions connues est en moyenne de $7,47\%$. Du fait de la nature même de la solution proposée, nous avons pu paralléliser certaines des tâches ce qui facilite grandement la mise à l'échelle de notre approche sur des instances de taille réelle de la ville de Nice (i.e. moins de 25 minutes pour l'instance la plus difficile).
Pour finir, nous avons pris en compte l'incertitude liée aux données de sécurité ainsi que son influence sur le déploiement des forces de secours avec le problème d'évacuation FRGEDA. Dans ce cadre, nous avons proposé des méthodes exactes et heuristiques permettant de trouver la meilleure affectation des forces de secours dans le pire des scénarios qui puissent se présenter. Les résultats obtenus dans ce dernier problème sont dans la même lignée que ceux obtenus par la méthode de résolution prônée par l'approche FGEDA. En effet sur les instances testées, la meilleure politique est de privilégier la sécurisation des points critiques faiblement endommagés et plébiscités par les évacués afin de les accompagner tout au long de leurs déplacements.

Chapitre 5

Applications et intégrations

5.1 Introduction de la valorisation des recherches

Comme nous l'avons déjà dit dans le chapitre 1, le projet DSS_Evac_Logistique est constitué d'un ensemble d'acteurs dont le but de réaliser un logiciel d'aide à la décision nommé $Visual\ Flow$ et prenant en compte l'ensemble de la chaîne des événements en cas d'évacuation due à une catastrophe d'origine industrielle ou naturelle. Le but étant de fournir un outil d'aide à la conception de plans d'évacuation. Pour y parvenir, plusieurs compétences venant de domaines variés sont nécessaires afin que la modélisation et la résolution des problèmes soient la plus réaliste possible. De plus, même si les problèmes traités sont pour la plupart $NP\text{-}Difficile$ et peuvent nécessiter en pratique un temps de résolution exponentiel, l'ensemble des algorithmes de résolution proposés par les acteurs sont des déclinaisons heuristiques de méthodes exactes permettant de résoudre des instances de taille réelle en au plus quelques minutes. L'autre aspect important de la solution développée est son architecture logicielle. En effet, cette dernière doit aussi bien pouvoir fonctionner en ligne qu'en local. Aussi, le logiciel développé doit, entre autres, permettre de réaliser plusieurs tâches comme :

- l'importation de données géographiques de la zone d'étude.
- l'application de niveaux de dommages suivant les sinistres considérés
- le choix des zones à évacuer et des zones d'accueil
- les différents modes d'évacuation (piétons, voitures personnelles, transports en communs)
- la visualisation et l'évaluation d'un plan d'évacuation
- la comparaison de plusieurs plans d'évacuation.

Dans ce chapitre nous allons présenter l'ensemble de l'architecture du logiciel utilisé dans le cadre du projet ainsi que certains des algorithmes d'optimisation intégrés à ce dernier. Enfin nous présenterons la chaîne complète de traitement d'une catastrophe naturelle grâce aux cas d'utilisation et fonctionnalités du logiciel en allant de la définition des scénarios-catastrophes et la collecte des données à la prise en charge de la population dans les centres de secours par les autorités compétentes.
Nos travaux et notre contribution à cet outil logiciel s'inscrivent dans les développements du WP5.

5.2 Architecture logicielle

L'architecture du logiciel d'aide à la décision $Visual_Flow$ peut être vue comme suivant une architecture MVC [1] comme décrit par la figure 5.1. Cette dernière à la particularité

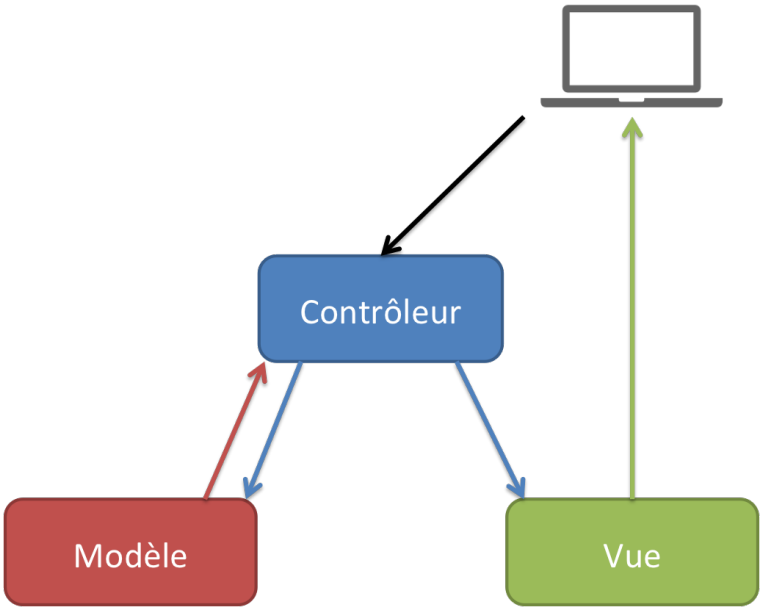


FIGURE 5.1 – Architecture MVC

de diviser l'application en trois entités indépendantes. En effet, la partie Modèle contient l'ensemble des données de manière structurée, la partie contrôleur traite les requêtes des utilisateurs en récupérant les données définies dans la partie Modèle. A la suite des traitements sur les données, la partie Contrôleur génère une Vue qui est retournée à l'utilisateur en fonction de son type d'interface $Visual\_Flow$.
Ci-dessous, nous présentons de manière synthétique les éléments constitutifs de cette architecture.

Partie contrôleur

La partie contrôleur est un élément central du logiciel. C'est elle qui reçoit les différentes requêtes de l'utilisateur afin de les traiter. Ces requêtes peuvent être l'affichage d'une zone de sinistre, la résolution d'un problème d'évacuation ou encore la comparaison de plusieurs solutions en fonction de plusieurs critères. Afin de l'aider dans sa tâche, ce dernier peut faire appel à l'ensemble des solveurs développés dans le cadre du projet afin de résoudre plusieurs problèmes d'optimisation comme la localisation des centres de secours, le routage des piétons, des voitures personnelles ou encore des transports en commun comme les bus. A ces solveurs, s'ajoutent d'autres exécutables lui permettant de simuler un plan

1. Modèle-Vue-Contrôleur

d'évacuation, de détecter les conflits potentiels entre les personnes à évacuer ou encore de reconfigurer le réseau sinistré.
Afin de pouvoir réaliser l'ensemble de ces tâches, la partie contrôleur a besoin de données qui lui sont fournies par la partie modèle.

Partie modèle

La partie modèle est composée de l'ensemble des données dont nous disposons. Afin que le logiciel soit robuste en cas de défaillances des infrastructures de télécommunication, nous avons une redondance des données. En effet, ces dernières existent sous deux formes à savoir une base de données principale mais aussi sous forme de fichiers XML qui sont des extractions des éléments se trouvant dans la base de données.
Les sources de données alimentant notre base de données principale sont variées, nous les présentons ci-dessous :

- OpenStreetMap : Cette plateforme collaborative est la principale source de données concernant le réseau routier d'une ville. Sa force est le fait qu'elle soit mise à jour assez régulièrement par les contributeurs en faisant ainsi la source la plus à jour possible. Un autre avantage de cette plateforme est de permettre d'extraire des graphes de villes particulières comme Nice et Kaiserslautern afin de les exploiter en mode hors-ligne à l'aide de la plateforme annexe $geofabrik$. Toutefois ces données n'apportent aucune information sur le sinistre à étudier ainsi que les niveaux de dommages causés.
- Citeres et BRGM : Afin de connaitre la population qui sera impactée par un sinistre et qui devra être évacuée, il est important d'étudier la robustesse des infrastructures et les caractéristiques des différents sols mais aussi la propagation des effets du sinistre sur ce dernier. La combinaison de ces deux études permet de déterminer les niveaux de dommage des bâtiments sous forme de $Shapefiles$. Aussi, suivant la gravité des dommages, les personnes y résidant doivent être évacuées et mises en sécurité. Ces dernières constituent la population à évacuer.
- AKGA [2] : Du coté allemand, le groupe de travail de sécurité de Kaiserslautern qui représente les forces de secours (médecins, pompiers, policiers) dispose d'un outil web interfacé à la base de données principale afin de mettre à jour les informations de celle-ci en fonction des sinistres considérés. Ainsi les utilisateurs autorisés disposant d'un accès à la plateforme peuvent mettre à jour en temps réel les données terrain en vue d'une évacuation due à un sinistre de type industriel (i.e. découverte d'une bombe de la seconde guerre mondiale). Cet outil permet par exemple de délimiter la zone à évacuer, les différents centres de secours à ouvrir ainsi que leurs attributs (capacité d'accueil, les personnes en charge de les aménager,...), les différents points de collecte des personnes en vue d'une évacuation par transport en commun (i.e. bus)

2. Arbeitskreis Gefahrenabwehr Kaiserslautern

Partie vue

La partie vue du modèle est celle qui est générée après les traitements de la partie contrôleur et constitue ce que voit l'utilisateur à travers son interface d'utilisation. Dans le cadre du projet, il existe deux types de vues inhérentes aux applications utilisées. La première est une application desktop $JAVA$ permettant une utilisation de l'application $Visual\_Flow$ en mode hors-ligne et en mode en-ligne. La seconde est une interface web utilisable depuis un navigateur web et qui dispose des mêmes fonctionnalités en-ligne que l'application $JAVA$.

5.3 Algorithmes implémentés pour le logiciel VisualFlow

Dans le cadre de cette thèse, nous avons intégré certains de nos algorithmes à l'outil d'aide à la décision du projet DSS_Evac_Logistique. Suivant les hypothèses effectuées et les données disponibles, nous avons implémenté dans le logiciel VisualFlow les algorithmes d'évacuation sans pertes présentés dans le chapitre 3.

- Affectation des évacués : Cet algorithme permet d'obtenir l'affectation des individus aux différents centres de secours ainsi que leur date d'arrivée à ces derniers. Pour y parvenir, nous utilisons l'algorithme de Klinz (voir [Hoppe et Tardos, 2000]) sur les données statiques du réseau routier sans prise en compte de la notion de sécurité. Cette dernière permet d'obtenir une borne inférieure en rapport avec la date de fin d'évacuation au plus tôt et le schéma d'affectation des évacués sur les centres de secours. L'avantage principal de cet algorithme est qu'il est polynomial. Toutefois il ne permet pas d'avoir le plan d'évacuation (le flot) de personnes. Le fait d'avoir l'affectation des personnes aux différents centres de secours dont les arrêts de bus permet au WP5 d'avoir les données nécessaires d'arrivées des personnes à évacuer en bus ainsi que leur nombre.
- Approche lexicographique : Cet algorithme est l'implémentation de la méthode présentée dans la section 3.2.5.1 permettant une première prise en compte de la notion de durée, de sécurité mais aussi la variabilité des données du réseau durant l'évacuation. Il est centré sur l'optimisation de la durée nécessaire pour évacuer l'ensemble de la population. Une première prise en compte de la notion de sécurité est effectuée. En effet, lors de la création du plan d'évacuation, si deux chemins ont la même longueur alors celui ayant le niveau de sécurité le plus élevé est favorisé. Grâce à une nouvelle structure de données, nous pouvons utiliser des graphes de taille réelle ne nécessitant pas de simplification. En outre l'approche proposée est insensible à la taille des instances relatives au nombre de points de rassemblement et de centre de secours. Il peut ainsi facilement résoudre les instances les plus difficiles de la ville de Nice avec le scénario de tremblement de terre Ligure (voir 3.2.7).
- Enumération du front de Pareto : Cet algorithme est l'implémentation de la version approchée de l'algorithme 3.3.3.1. Il prend en compte l'ensemble des éléments de notre problème d'évacuation. Il permet de déterminer les chemins empruntés par les piétons et les voitures personnelles durant le processus d'évacuation tout en essayant de contourner les zones dangereuses. Contrairement aux précédents algorithmes, ce dernier est centré sur la maximisation de la sécurité pour un horizon de planification

déterminée. Une évaluation de la durée minimale nécessaire pour évacuer toute la population est obtenue en utilisant les deux précédents algorithmes. Une fois cette date fixée, nous maximisons le nombre fractionnaire de personnes indemnes à la fin de l'évacuation. En se donnant plus de temps et en permettant aux personnes de faire des détours pour contourner les zones dangereuses, nous arrivons à optimiser la sécurité globale du plan d'évacuation.

Sachant que la différence principale entre les piétons et les utilisateurs de voitures personnelles résident dans la vitesse de déplacement et donc la durée relative de parcours des arcs, les algorithmes précédents s'appliquent sans perte de généralités aux deux cas de figure.

5.4 Contributions de chaque working package au logiciel VisualFlow

Dans cette partie nous présentons quelques contributions réalisées dans chaque working package à l'outil d'aide à la décision. Ces dernières sont illustrées par des captures d'écrans issues du logiciel "VisualFlow".

WP1 : Evac-Scenario

Dans le cadre du projet, deux types de scénario-catastrophe ont été étudiés pour la ville de Nice à savoir ceux associés aux tremblements de terre de Ligure et Vésubie dont l'épicentre a été relocalisé au large de la ville. L'étude de la vulnérabilité des bâtiments et l'impact de ces sinistres sur ces derniers sont présentés sur les figures Fig.3.9 et Fig. grâce à des fichiers shapefiles. Les couleurs associées aux niveaux de dommages peuvent être interprétées comme suit :

— Vert : Niveau de dommage faible, les piétons, voitures et bus peuvent circuler.
— Orange : Niveau de dommage moyen, les piétons et les bus peuvent circuler.
— Rouge : Niveau de dommage élevé, seuls les piétons peuvent circuler.

Afin de rester en adéquation avec les besoins de l'ensemble des acteurs comme pour la partie allemande où il s'agit principalement de catastrophes industrielles comme en cas de découverte d'une bombe de la seconde guerre mondiale, il est aussi possible de déterminer le niveau de danger d'une zone de manière arbitraire comme l'indique la figure Fig.5.2. Il s'agit de délimiter une zone à l'aide d'un polygone et de lui appliquer un niveau de dommage.

WP2 : Evac-Location

Une fois les niveaux de dommage déterminés par le WP1 sur le réseau, ce dernier est passé en paramètre au WP2 dont le rôle est de déterminer les positions des points de rassemblement ainsi que des centres de secours. Pour la ville de Nice, un travail en amont a été effectué par le laboratoire Citeres dans le cadre de stages étudiants. Ce travail a consisté à répertorier dans chaque zone d'habitation les potentiels points de rassemblement où les personnes peuvent se réunir afin de recevoir les premières consignes d'évacuation. Durant ces stages, ils ont aussi répertorié l'ensemble des emplacements du réseau pouvant être consi-

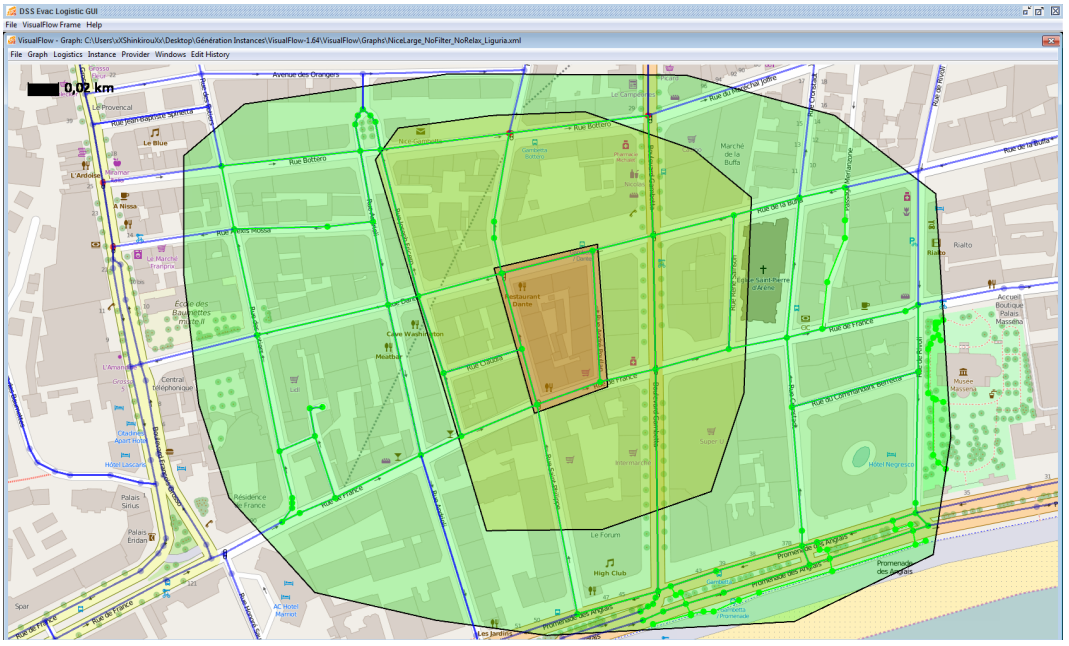


FIGURE 5.2 – Illustration des niveaux de dommages

dérés comme des centres de secours. En outre, dans la pratique le terme centre de secours fait référence à trois entités qui sont les arrêts de bus, les centres de secours temporaires et les centres de secours définitifs. Les arrêts de bus sont des centres de secours à capacités limitées et restaurables dans le temps lorsqu'un ou plusieurs bus y passent pour récupérer des personnes à évacuer. L'objectif principal de ces arrêts de bus est d'éviter que les piétons sans voitures ne parcourent plusieurs kilomètres à pied pour se mettre en sécurité. Les centres de secours temporaires sont des zones où les évacués se rendent pour bénéficier de consignes et des premiers soins s'ils sont blessés. Aussi étant des centres temporaires, leur capacité d'accueil est limitée et les personnes s'y rendant doivent soit reprendre leur route soit attendre un bus pour les mettre définitivement en sécurité dans un centre de secours définitif. Les centres de secours définitifs quand à eux sont des emplacements où les individus peuvent résider pour une durée allant jusqu'à plusieurs jours. Ils ont une grande capacité d'accueil et disposent d'infrastructures plus abouties. Un exemple de ces différents points de rassemblements et des centres de secours est décrit dans la figure Fig.5.3. Dans

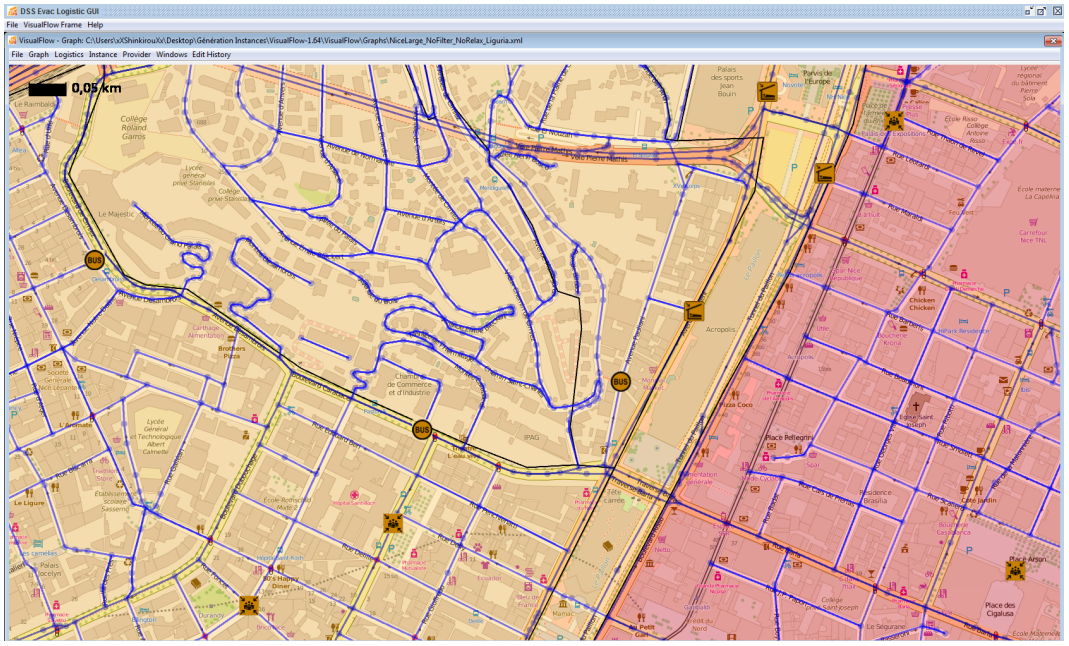


FIGURE 5.3 – Points de rassemblement et centres de secours

cet exemple associé à la ville de Nice, il est possible de voir des points de rassemblement de personnes se situant à des emplacements à découvert comme des jardins afin d'éviter la chute de débris sur les personnes, des arrêts de bus connus par les riverains et où les personnes peuvent se rendre afin d'être évacués par les autorités à l'aide de bus ainsi que des centres de secours définitifs se trouvant au Palais des sports Jean Bouin, ou le centre des congrès (Acropolis) étant des lieux vastes, robustes et récemment construits pouvant

accueillir les évacués pendant une certaine durée après quelques aménagements.
Le choix de l'ouverture et la fermeture des différents points de rassemblement et des centres de secours en fonction des coûts induits est réalisé par nos collègues universitaires allemands et en particulier Philipp Heβler dans le cadre de sa thèse au sein de la faculté de mathématiques de l'université technique de Kaiserslautern (Optimization Research Group).

WP3 : Evac-EvalTraffic

Ce working package vient en complément des WP1 et WP2. Il permet de reconfigurer le réseau considéré afin de le rendre plus performant durant le processus d'évacuation mais aussi de le simplifier afin que les méthodes proposées puissent être appliquées.
L'une des fonctionnalités en rapport avec la reconfiguration du réseau est la relaxation du graphe. En effet, la taille des graphes peut être très importante et nécessiter une simplification afin de pouvoir être traités en temps et en espace mémoire acceptable.
Il est ainsi possible de fixer à l'avance le nombre de sommets et d'arcs que contiendra le graphe simplifié. Pour y parvenir, il s'agit de fusionner ou de supprimer des sommets et des arcs du graphe. Toutefois en cas de simplification extrême, il y a une perte d'informations ainsi que de la topologie du réseau. La figure Fig 5.4 montre le réseau réel associé à la zone résidentielle se trouvant en face du centre des congrès (Acropolis) de Nice.

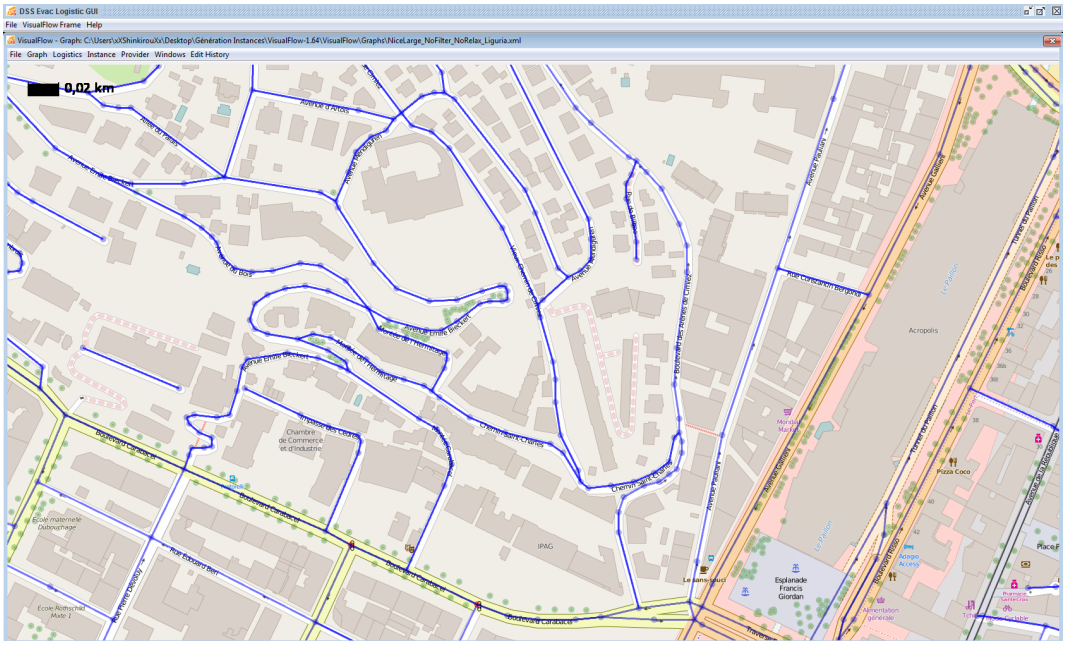


FIGURE 5.4 – Avant la relaxation du graphe

Nous pouvons constater qu'il existe de longues routes tortueuses composées de plusieurs

sommets et arcs. A l'échelle de la ville entière de Nice, cela représente un graphe composé de 46990 sommets et 91412 arcs. Avec la prise en compte de l'aspect temporel lors des déplacements, la prise en compte de données de telles tailles devient impossible pour des méthodes de résolution utilisant notamment des solveurs. En appliquant la relaxation du graphe et en conservant la topologie initiale, on obtient le graphe présenté par la figure Fig.5.5 .

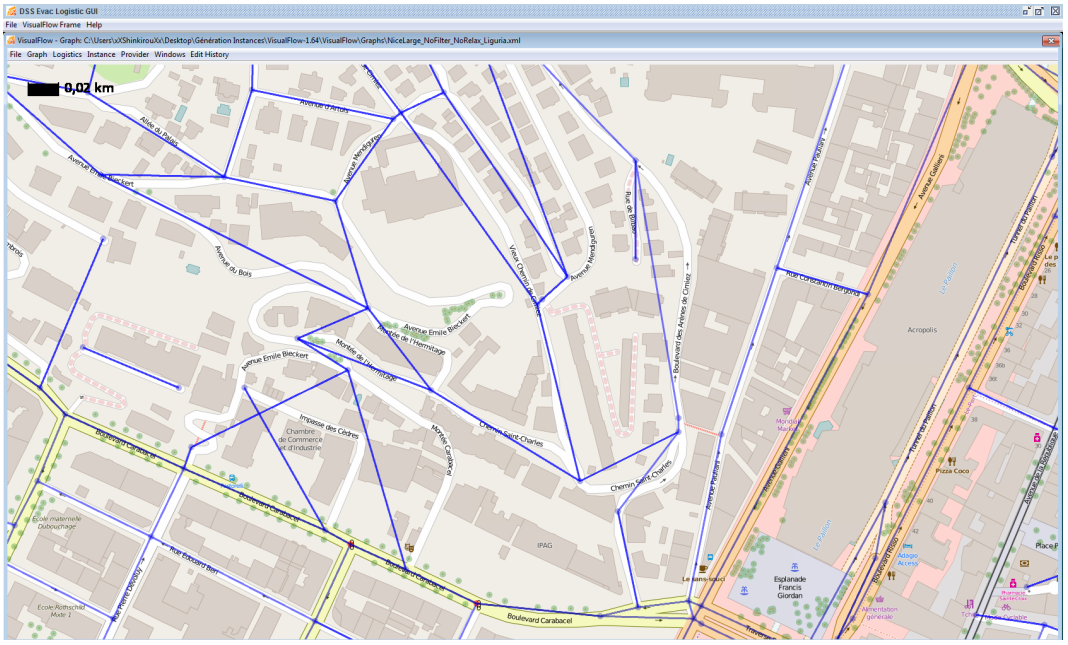


FIGURE 5.5 – Après la relaxation du graphe

Cela permet de réduire la taille du graphe en faisant passer cette dernière à 9396 sommets et 22052 arcs. Le principal contributeur de cette partie est Bob Grün qui effectue une thèse au sein de la faculté de mathématiques de l'université technique de Kaiserslautern (Optimization Research Group). Toutefois, de par la nature des problèmes traités durant la présente thèse (critères additif et multiplicatif), nous n'utilisons pas cette simplification de graphe du fait de l'agrégation des données et de la perte d'informations en rapport avec la sécurité sur les arcs ou sommets.

WP4 : Evac-Scheduling

Ce working package a pour principal rôle d'évacuer les piétons attendant aux arrêts de bus vers des centres de secours pouvant les accueillir afin de leur éviter de parcourir plusieurs kilomètres pour se mettre définitivement en sécurité. Ce problème d'optimisation peut être

vu comme VRPPD [3]. En effet, un certain nombre de bus à capacité limité partent tous d'un ou plusieurs dépôts, visitent un ou plusieurs arrêts de bus pour récupérer les clients (i.e. les piétons) puis rejoignent un des centres de secours temporaires ou définitifs afin de les y déposer avant de reprendre leur processus jusqu'à la fin de l'évacuation.
Afin de résoudre ce problème, ce dernier a été modélisé comme un problème d'atelier où les machines représentent les bus et les personnes à évacuer les tâches. La figure 5.6 montre une évacuation par bus sur un réseau simplifié de la ville de Nice. Il est aussi possible de voir le diagramme de Gantt associé à la planification des tournées de bus avec en bleu les moments où les bus sont vides et en vert les moments où ils ont des passagers. Ce travail est réalisé

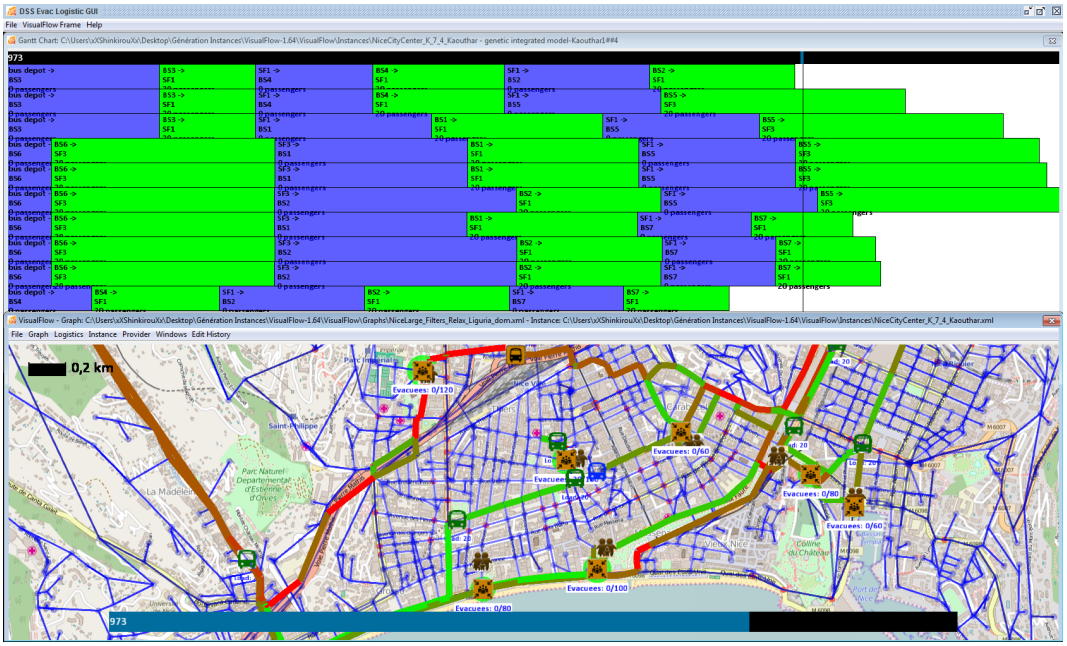


FIGURE 5.6 – Évacuation par bus

par Kaouthar Deghdak dans le cadre d'une thèse encadrée au Laboratoire d'Informatique de l'Université de Tours.

WP5 : Evac-Solve

Ce working package est celui associé à ce travail de thèse et vient à la suite du WP3. Comme données d'entrée à cette partie, nous disposons d'un réseau routier dont les niveaux de dommage sont définis, d'un ensemble de points de départ (points de rassemblement) et d'un ensemble de points d'arrivée potentiels (arrêts de bus, centres de secours temporaires, centres de secours définitifs). A chaque point de départ sont associées des personnes devant

3. Vehicle Routing Problem with Pick-Up et Delivery

quitter leur domicile à cause des dommages subits et les rendant dangereux. Ces personnes ont le choix entre évacuer à pied ou en voiture personnelle. Les points d'arrivée disposent d'une certaine capacité d'accueil pour chaque type de mode de transport (i.e. les arrêts de bus ne peuvent accueillir que des piétons alors que les centres de secours temporaires et définitifs peuvent accueillir les deux). La figure Fig.5.7 montre la solution associée à une instance du tremblement de terre Vésubie sur le centre ville de Nice ($NICE\_SMALL$). Le graphe de cette instance est composé de 10279 sommets et 18805 arcs. Le nombre de piétons à évacuer est de 8013 alors que le nombre de voitures personnelles est de 3038. Les individus étant répartis sur 512 départs (i.e. points de rassemblement) et pouvant potentiellement atteindre 119 points d'arrivée (i.e. 31 arrêts de bus, 33 centres de secours temporaires et 55 centres de secours définitifs).

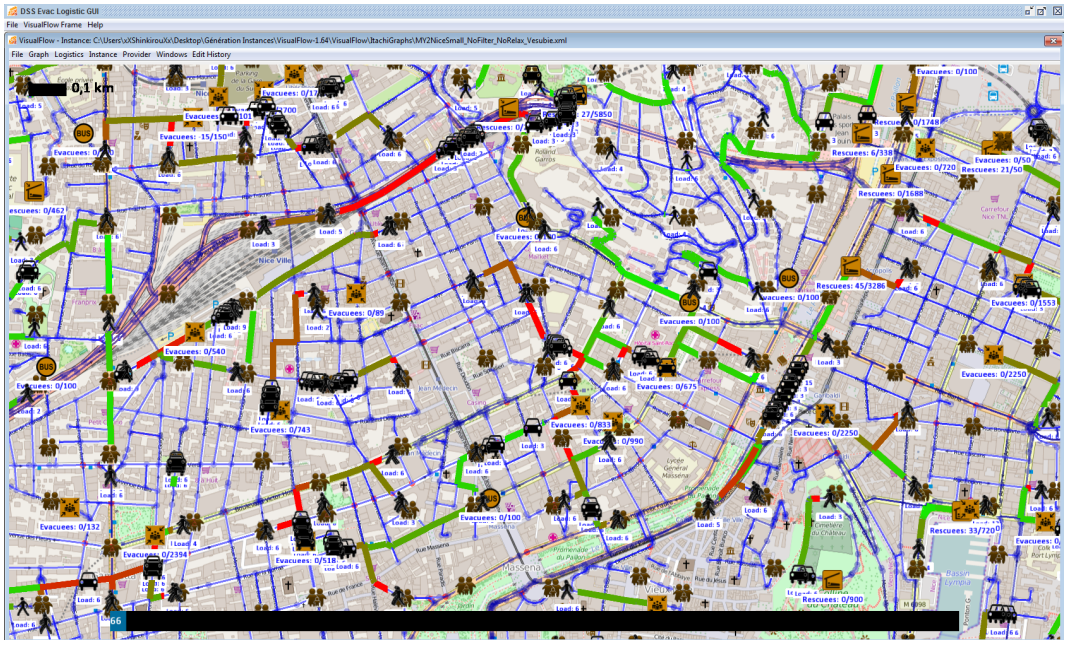


FIGURE 5.7 – Évacuation de piétons et de voitures à la date 66

Sur la figure 5.8 nous pouvons voir qu'à la date $t = 266$ l'ensemble des voitures personnelles se trouvant sur la zone l'ont déjà quittée. En effet, ces dernières ayant une vitesse de déplacement qui est fonction de la limite de vitesse sur les routes utilisées, elles se déplacent bien plus rapidement que les piétons.

Si l'on ne considère que les arrêts de bus, ces derniers reçoivent tout au long du processus d'évacuation des personnes à évacuer en bus. Dans le cadre du WP4, il s'agit de la date de disponibilité des différents travaux à ordonnancer sur les machines (i.e. les bus). Lorsqu'un bus passe par un arrêt saturé et récupère des personnes à transporter, la capacité d'accueil de l'arrêt augmente, ce qui fait qu'il devient de nouveau possible d'y envoyer des piétons.

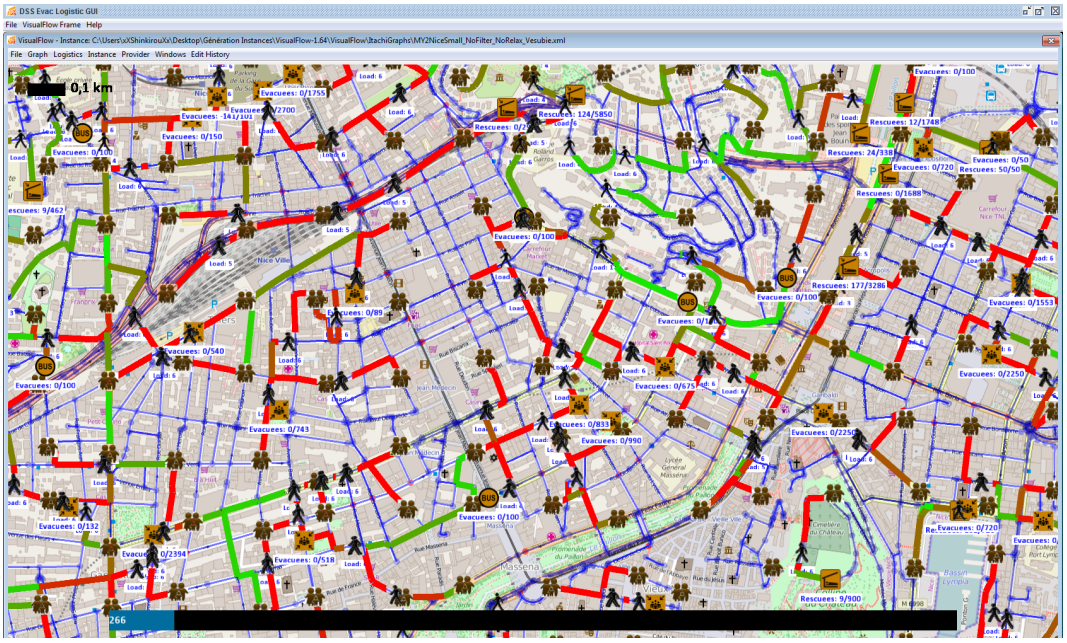


FIGURE 5.8 – Évacuation de piétons et de voitures à la date 266

WP6 : Evac-MCDM

Les deux précédents "Working Packages" fournissent un ensemble de solutions de compromis (i.e. durée, sécurité) qui peuvent être en nombre important. L'échantillonnage et la visualisation de ces solutions sont donc des aspects importants afin que les décideurs puissent avoir un ensemble représentatif de solutions. Cela devient d'autant plus important lorsque le nombre de critères augmente et dépasse 3 comme dans la partie Allemande avec des critères comme le coût matériel, la satisfaction des évacués ou encore leur temps d'attente moyen avant d'être pris en charge.
L'outil proposé est sous forme de "spiderchart" avec pour chaque critère considéré la possibilité de déterminer sa borne supérieure et sa borne inférieure. Ainsi seules les solutions respectant ces contraintes seront proposées (voir figure 5.9). Une fois une solution sélec-

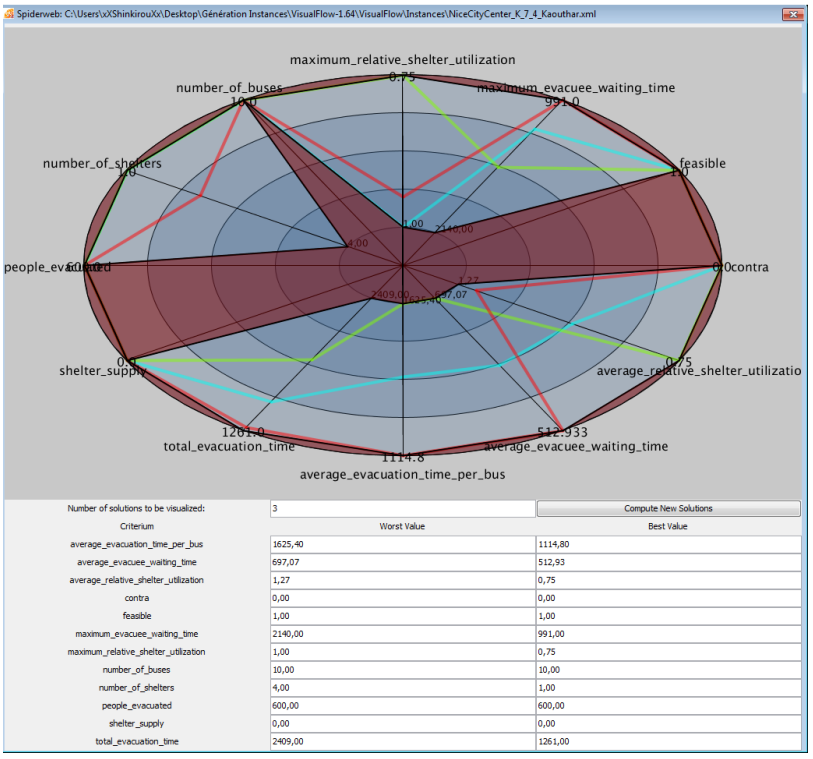


FIGURE 5.9 – Échantillonnage de front de paréto et affichage des solutions

tionnée, il est alors possible de la visualiser (voir les figures 5.7, 5.8 et 5.6).
Celui qui est en charge de cette partie est Tobias Huhn qui effectue sa thèse au sein de la faculté de mathématiques de l'université technique de Kaiserslautern (Optimization Research Group).

WP7 : Evac-Simulate

Une fois une solution sélectionnée, il est possible de faire une simulation microscopique afin d'évaluer le plan d'évacuation et de vérifier qu'il ne mène pas à des situations aber-

rantes. Le partenaire industriel en charge de cette partie est l'entreprise française CERV-VAL dont l'un des domaines d'expertise est la simulation d'environnement urbain. Une fois cette simulation effectuée, et les données mises à jour, il est de nouveau possible de faire appel aux WP4 et WP5, afin de déterminer de nouveaux plans d'évacuation prenant en compte les données actualisées.

WP8 : Evac-Tool

Ce working package est celui qui s'occupe d'interfacer les différentes productions scientifiques des différents acteurs associés au projet DSS_Evac_Logistique. Il s'agit du logiciel "VisualFlow" qui est développé par la société allemande "INFORM GmbH".

5.5 Conclusion de la valorisation des recherches

Dans ce chapitre, nous avons présenté l'architecture du logiciel d'aide à la décision pour l'évacuation de masse en cas de catastrophes naturelles dans le cadre du projet DSS_Evac_Logistique. Comme nous l'avons montré avec des exemples de contribution de chaque working package, le logiciel ainsi développé permet de prendre en compte l'ensemble de la chaîne des événements en cas de catastrophes naturelles ou industrielles.
Dans le WP1 nous avons vu comment étaient définis les scénarios-catastrophes. Dans le WP2 nous avons vu comment le choix des lieux de départ et d'arrivée des évacués était effectué. Un exemple de reconfiguration du réseau afin d'optimiser le processus d'évacuation est présenté dans le WP3. Une fois tous les paramètres du problème d'évacuation fixés, les WP4 et WP5 permettent de déterminer les chemins empruntés par les évacués ainsi que leurs modes de déplacements (i.e. à pied, en voiture, en bus). Dans le package WP6 permet de comparer et de sélectionner des solutions en fonction de plusieurs critères. Le package WP7 permet d'effectuer une simulation de l'évacuation afin de détecter les éventuels problèmes dans les solutions comme la formation d'embouteillages ou de mouvements de foule. Enfin le WP8 permet de fédérer l'ensemble des travaux des acteurs afin d'avoir un logiciel d'aide à la décision fonctionnel comme nous l'avons vu dans ce chapitre.
Nous avons aussi montré comment nous avons contribué à l'élaboration de ce logiciel à travers les travaux du working package WP5 (i.e. Evac-Solve).Pour y parvenir, nous avons conçu des algorithmes permettant de résoudre des problèmes d'optimisation de flots multicritère sans pour autant que ces derniers ne souffrent des problèmes dus à la taille des données ou à la complexité théorique des problèmes traités.

Conclusion générale et perspectives

Les travaux réalisés durant cette thèse portent sur le routage de masse avec une application aux problèmes d'évacuation en cas de catastrophes d'origine naturelle ou industrielle. S'intégrant au projet Franco-Allemand DSS_Evac_Logistique, nous avions comme cas d'étude la ville de Nice pour la partie française et la ville de Kaiserslautern pour la partie allemande avec pour objectif de proposer un outil d'aide à la décision permettant d'organiser les opérations d'évacuation en cas de sinistres. Dans le cadre de cette thèse (i.e. Working Package 5) nous avions en charge la résolution de problèmes multicritère (durée/sécurité) pour la conception de plans d'évaluation de personnes qui autrement dit consiste à produire des plans d'évacuation multicritère minimisant la durée de l'évacuation tout en maximisant la sécurité des individus.
Une des hypothèses de base sur ces individus étant le fait qu'ils puissent se déplacer par leurs propres moyens (i.e. à pied ou en groupe par voitures personnelles). Pour réaliser cette tâche, nous avons divisé notre travail en 5 parties.

Dans le chapitre 1, nous avons présenté le contexte dans lequel le projet DSS_Evac_Logistique avait été réalisé afin de situer notre thèse dans ce dernier. Ainsi, nous avons présenté l'ensemble des données d'entrée et de sortie de chaque acteur ainsi que les contraintes auxquelles les modèles qu'ils ont développés sont soumis. Aussi, nous avons montré que le projet traitait l'ensemble de la chaîne des événements en partant de la génération des scénarios-catastrophes à la sélection et visualisation des différentes solutions de compromis (i.e. plans d'évacuation).

Dans le chapitre 2, nous avons situé notre travail de thèse par rapport à l'existant dans la littérature des problèmes d'évacuation. Pour ce faire, nous avons scindé l'état de l'art en deux blocs englobant les modèles d'évacuation macroscopiques et microscopiques. Dans le premier cas, nous avons montré comment modéliser le processus d'évacuation comme des problèmes de flots dynamique. Cette approche part du postulat que les individus ont des caractéristiques identiques (i.e. âge, vitesse de déplacement, corpulence, résistance au stress,...). Ces caractéristiques sont vues comme représentant un individu moyen représentatif de la population. En n'oubliant pas qu'une population est composée d'individus hétérogènes, nous avons dans un second temps étudié l'existant en matière d'approche microscopique pour pallier à cette situation. Pour ce faire, des méthodes à base d'automates cellulaires et de systèmes multi-agents ont été présentées. Ces dernières permettent aussi

bien de prendre en compte les interactions entre les individus (i.e. bousculades, transmission d'informations ou de stress,...) mais aussi entre les individus et leur environnement. Enfin nous avons montré les avantages et les inconvénients des approches macroscopiques et microscopiques afin de savoir laquelle serait utilisable dans le cadre de la modélisation d'une évacuation de masse d'une ville.

Dans le chapitre 3, nous partons du postulat que l'ensemble de la population pourra être évacuer par ses propres moyens. Pour y parvenir, nous avons utilisé la meilleure politique d'évacuation possible (i.e. Earliest Arrival Flow) en rapport avec le critère de la durée car incluant les propriétés des autres problèmes d'évacuation. Toutefois ce problème d'optimisation est $NP\text{-}Difficile$ et est d'autant plus problématique à utiliser sur des données de taille réelle du fait de la complexité spatiale qui lui était jusqu'alors associée. Pour résoudre ce problème, nous avons mis en place une structure de données permettant de ne construire que les données nécessaires et d'accéder aux données en temps quasi-constant (i.e. $O(LogT) \approx O(Constante) \approx O(1)$). Sans perte de généralité, nous avons étendu cette approche avec une méthode lexicographique permettant une première prise en compte de la sécurité lors de l'évacuation. Cette extension permet d'obtenir un optimum de Pareto faible ayant la meilleure durée d'évacuation possible. Pour énumérer l'ensemble du front de Pareto, nous avons proposé une méthode de résolution exacte qui a une complexité temporelle et spatiale pseudo-polynomiale. Comme cette dernière est limitée du point de vue applicatif sur des données de taille réelle, nous avons aussi proposé une méthode d'énumération approchée qui se base sur l'usage de la mémoire de décision. Ces travaux ont fait l'objet de communications en conférences nationales et internationales (voir [Ndiaye et Neron, 2013a, Ndiaye et Neron, 2013b]) ainsi que d'une publication en seconde lecture (voir [Ndiaye *et al.*, 2014c]).

Dans le chapitre 4, nous partons du postulat que l'ensemble de la population ne pourra pas évacuer par ses propres moyens et qu'une partie aura besoin d'une aide extérieure. Pour cela nous prenons comme problème central la maximisation du nombre personnes indemnes pouvant se déplacer durant l'évacuation sans aides extérieures. Nous avons dans un premier temps modélisé le problème d'évacuation comme un flot généralisé prenant en compte l'intégrité des individus (i.e. problème FGED). Ce problème optimisation étant $NP\text{-}Difficile$, nous avons proposé une métaheuristique se basant sur l'optimisation par colonies de fourmis. En outre la méthode proposée permet d'unifier les approches macroscopiques et microscopiques dans une certaine mesure afin de pouvoir l'appliquer à des données de taille réelle. Pour rendre ce problème d'évacuation plus réaliste, nous avons aussi ajouté le fait de pouvoir déployer des forces de secours afin d'assister les personnes durant leur périple (i.e. problème FGEDA). Nous avons avons pour cela étendu notre algorithme de colonies de fourmis afin de prendre en compte l'affectation des différentes forces de secours sur les points critiques du réseau afin qu'ils les sécurisent. Pour y parvenir, nous avons utilisé une recherche tabou avec des opérateurs de voisinages issus de l'étude des personnes blessées sur les différents types de sommets. Enfin, en collaboration avec Marc Goerigk, nous avons étudié le cas particulier où les données concernant les niveaux de dommages sont incertaines. Pour cette dernière partie de notre travail, nous avons proposé une approche exacte ainsi que des approches heuristiques permettant de trouver le pire scéna-

rio pouvant se produire et comment déployer les forces de secours afin de répondre à cette situation exceptionnelle (i.e. problème FRGEDA). Ces travaux ont fait l'objet de communications en conférences nationales et internationales (voir [Ndiaye *et al.*, 2014a, Ndiaye *et al.*, 2014b, Ndiaye *et al.*, 2014d]), ainsi que deux publications dont une acceptée (voir [Ndiaye *et al.*, 2014d]) et une autre en seconde lecture (voir [Goerigk et Ndiaye, 2014]).

Enfin, dans le chapitre 5, nous avons présenté le fruit du travail des différents acteurs du projet DSS_Evac_Logistique à travers l'architecture du logiciel développé mais aussi une énumération de quelques fonctionnalités disponibles. Pour ce logiciel "VisualFlow", nous avons implémenté trois méthodes de résolution issues du chapitre 3. Ces dernières permettent de prendre en compte les notions de durée et de sécurité dans un contexte de crise avec des données variant dans le temps durant l'évacuation tout en permettant de router les piétons et les voitures personnelles.

En terme de perspectives de recherches, le domaine de l'évacuation de masse est connexe à celui du routage de masse ouvrant ainsi beaucoup de possibilités.
Aussi, à court et moyen termes, nous souhaitons investiguer les points suivants :

- **Incertitude des durées de trajet** : Dans notre thèse nous, avons traité la variabilité des données dans le temps. Toutefois, plusieurs facteurs comme les embouteillages, feux de tricolore ou mouvements de foule peuvent influer sur les durées de déplacements. Afin de prendre en compte ces différents éléments, il convient de proposer des modèles prenant en compte ces perturbations en temps réel et non juste proposer des plans de routage robustes. Cette voie de recherches a la particularité de pouvoir bénéficier l'essor de dispositifs de traçage GPS aussi bien dans les voitures personnelles que dans les "smartphones" des piétons.
- **Approche microscopique a grande échelle et différentiation des individus** : Au travers de nos métaheuristiques d'optimisation par colonies de fourmis, nous avons montré comment prendre en compte l'aspect microscopique en appliquant à chaque fourmi un ensemble de caractéristiques qui peuvent être unique (vitesse de déplacement, perception accrue ou non des dangers, corpulence, ...). Toutefois, ces dernières n'ont qu'un seul objectif à savoir partir d'un point de rassemblement et atteindre un centre de secours quelconque. Il s'agira donc d'étudier le cas ou la destination est fixée (i.e. partir de sa maison et aller à son travail et vice-versa).
- **Routage multimodal de masse** : Cette partie à pour but d'unifier les travaux des Working Package 4 et 5 mais aussi de les étendre. Il s'agira de permettre à chaque individu associé à un point de départ et d'arrivée et un ensemble de modes de transports possible (i.e. à pied, en bus, en métro, en voiture, vélo,...) de bénéficier d'un plan de routage. En outre, ce plan de routage ne concernant pas une seule personne mais un ensemble d'individus, il convient donc de faire en sorte qu'il n'y ait pas de goulet d'étranglement en proposant le même routage à plusieurs personnes (i.e. éviter par exemple que 200 personnes attendent un bus ayant 50 places et proposer à certain d'utiliser d'autres moyens de transport les faisant arriver tous aussi vite à destination).

Enfin, à long terme nous souhaitons étudier les points suivants :

— **Critères supplémentaires** : Il est possible de prendre en compte des critères supplémentaires comme la minimisation du temps d'attente moyen des individus entre deux moyens de transport, la maximisation de la satisfaction des personnes, la prise en compte de la déclinaison des routes quant à l'effort relatif nécessaire pour les traverser, minimiser la formation de foule ou d'embouteillage.

— **Prédictions des états futures du réseau** : Il s'agit d'étudier le comportement des personnes se déplaçant dans un réseau afin de déterminer leurs habitudes en fonction du temps et des jours et pouvoir faire prédire à l'avance ce qui pourrait se passer. Cette approche a pour but de pouvoir encadrer les incertitudes liées au réseau considéré.

Annexe A

Annexes

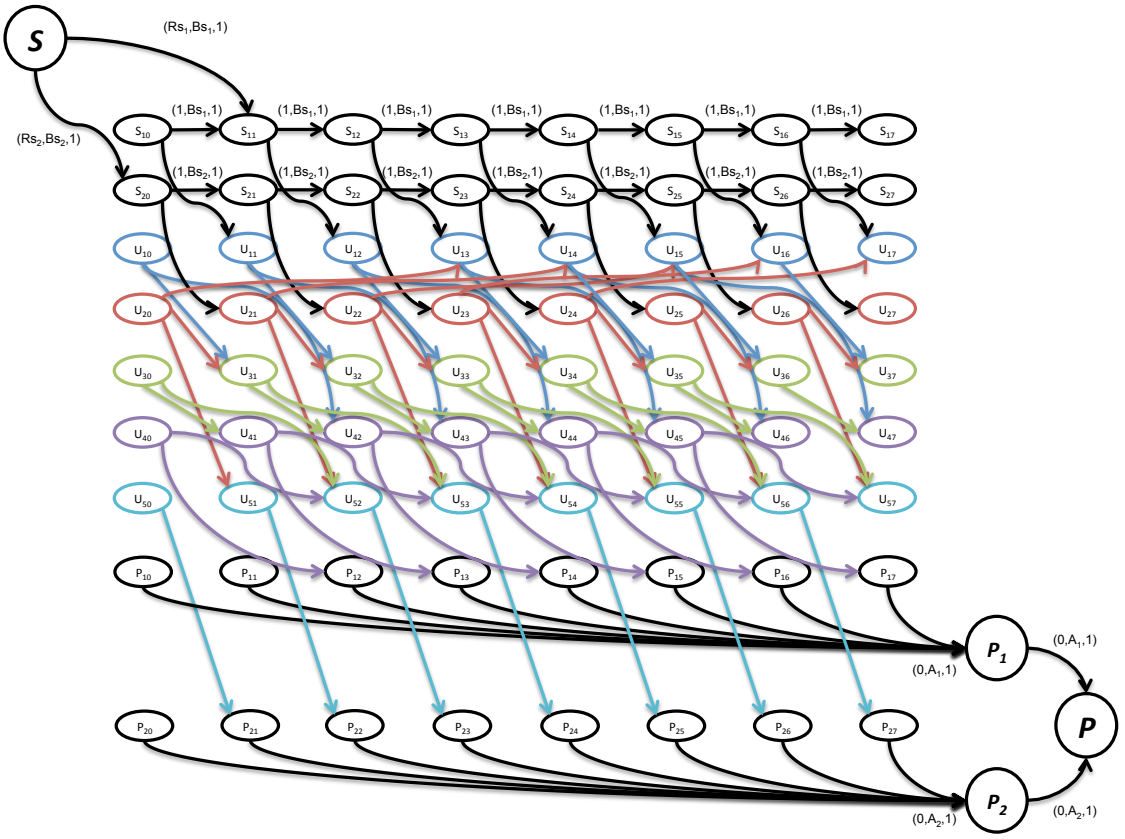


FIGURE A.1 – Exemple de graphe étendu dans le temps

TABLE A.1 – Niveau d'utilisation des arcs à chaque instant t

Arcs	Time t											
	0	1	2	3	4	5	6	7	8	9	10	11
(S, S_1)	6	0	0	0	0	0	0	0	0	0	0	0
(S, S_2)	12	0	0	0	0	0	0	0	0	0	0	0
(S_1, S_1)	0	3	0	0	0	0	0	0	0	0	0	0
(S_1, U_1)	0	3	3	0	0	0	0	0	0	0	0	0
(S_2, S_2)	10	8	6	4	2	1	0	0	0	0	0	0
(S_2, U_2)	2	2	2	2	2	1	1	0	0	0	0	0
(U_1, U_3)	0	0	0	0	0	0	0	0	0	0	0	0
(U_1, U_4)	0	0	3	3	0	0	0	0	0	0	0	0
(U_2, U_1)	0	0	0	0	0	0	0	0	0	0	0	0
(U_2, U_3)	0	1	1	1	1	1	1	1	0	0	0	0
(U_2, U_5)	0	1	1	1	1	1	0	0	0	0	0	0
(U_3, U_4)	0	0	1	1	1	1	1	1	1	0	0	0
(U_3, U_5)	0	0	0	0	0	0	0	0	0	0	0	0
(U_4, U_5)	0	0	0	0	0	0	0	0	0	0	0	0
(U_4, P_1)	0	0	0	1	4	4	1	1	1	1	0	0
(U_5, P_2)	0	0	1	1	1	1	1	0	0	0	0	0
(P_1, P)	0	0	0	0	0	1	4	4	1	1	1	1
(P_2, P)	0	0	0	1	1	1	1	1	0	0	0	0

TABLE A.2 – Niveau de sécurité de chaque chemin utilisé et le nombre de personnes l'utilisant à chaque instant t

Paths	Time t						
	0	1	2	3	4	5	6
(S_1, U_1, U_4, P_1)	$1, 0/3$	$1, 0/3$	•	•	•	•	•
$(S_2, U_2, U_3, U_4, P_1)$	$1, 0/1$	$1, 0/1$	$1, 0/1$	$1, 0/1$	$1, 0/1$	$1, 0/1$	$1, 0/1$
(S_2, U_2, U_5, P_2)	$0, 80/1$	$0, 80/1$	$0, 80/1$	$0, 80/1$	$0, 80/1$	•	•

TABLE A.3 – État de la mémoire de décision à l'instant $t = 3$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	
(S, S_2)	$(\bullet, 1.0, 1)$;
(S_1, U_1)	
(S_2, U_2)	$(0, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	
(U_2, U_1)	
(U_2, U_3)	
(U_2, U_5)	$(1, 0.8, 1)$;
(U_3, U_4)	
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	
(U_5, P_2)	$(2, 1.0, 1)$;
(P_1, P)	
(P_2, P)	$(\bullet, 1.0, 1)$;

TABLE A.4 – État de la mémoire de décision à l'instant $t = 4$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	
(S, S_2)	$(\bullet, 1.0, 2)$;
(S_1, U_1)	
(S_2, U_2)	$(0, 1.0, 1)$;$(1, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	
(U_2, U_1)	
(U_2, U_3)	
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;
(U_3, U_4)	
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;
(P_1, P)	
(P_2, P)	$(\bullet, 1.0, 2)$;

TABLE A.5 – État de la mémoire de décision à l'instant $t = 5$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	
(S, S_2)	$(\bullet, 1.0, 4)$;
(S_1, U_1)	
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 1)$;$(2, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 1)$;
(P_2, P)	$(\bullet, 1.0, 3)$;

TABLE A.6 – État de la mémoire de décision à l'instant $t = 6$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	$(\bullet, 1.0, 3)$;
(S, S_2)	$(\bullet, 1.0, 6)$;
(S_1, U_1)	$(1, 1.0, 3)$;
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 2)$;$(2, 1.0, 1)$;$(3, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	$(2, 1.0, 3)$;
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;$(2, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;$(4, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;$(4, 1.0, 4)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 5)$;
(P_2, P)	$(\bullet, 1.0, 4)$;

TABLE A.7 – État de la mémoire de décision à l'instant $t = 7$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	$(\bullet, 1.0, 6)$;
(S, S_2)	$(\bullet, 1.0, 8)$;
(S_1, U_1)	$(1, 1.0, 3)$;$(2, 1.0, 3)$;
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 2)$;$(2, 1.0, 2)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	$(2, 1.0, 3)$;$(3, 1.0, 3)$;
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;$(2, 1.0, 1)$;$(3, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;$(4, 0.8, 1)$;$(5, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;$(4, 1.0, 4)$;$(5, 1.0, 4)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 9)$;
(P_2, P)	$(\bullet, 1.0, 5)$;

TABLE A.8 – État de la mémoire de décision à l'instant $t = 8$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	$(\bullet, 1.0, 6)$;
(S, S_2)	$(\bullet, 1.0, 9)$;
(S_1, U_1)	$(1, 1.0, 3)$;$(2, 1.0, 3)$;
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 2)$;$(2, 1.0, 2)$;$(3, 1.0, 2)$;$(4, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	$(2, 1.0, 3)$;$(3, 1.0, 3)$;
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;$(4, 0.8, 1)$;$(5, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;$(4, 1.0, 4)$;$(5, 1.0, 4)$;$(6, 1.0, 1)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 10)$;
(P_2, P)	$(\bullet, 1.0, 5)$;

TABLE A.9 – État de la mémoire de décision à l'instant $t = 9$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	$(\bullet, 1.0, 6)$;
(S, S_2)	$(\bullet, 1.0, 10)$;
(S_1, U_1)	$(1, 1.0, 3)$;$(2, 1.0, 3)$;
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 2)$;$(2, 1.0, 2)$;$(3, 1.0, 2)$;$(4, 1.0, 2)$;
(U_1, U_3)	
(U_1, U_4)	$(2, 1.0, 3)$;$(3, 1.0, 3)$;
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;$(4, 0.8, 1)$;$(5, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;$(4, 1.0, 4)$;$(5, 1.0, 4)$;$(6, 1.0, 1)$;$(7, 1.0, 1)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 11)$;
(P_2, P)	$(\bullet, 1.0, 5)$;

TABLE A.10 – État de la mémoire de décision à l'instant $t = 10$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	$(\bullet, 1.0, 6)$;
(S, S_2)	$(\bullet, 1.0, 11)$;
(S_1, U_1)	$(1, 1.0, 3)$;$(2, 1.0, 3)$;
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 2)$;$(2, 1.0, 2)$;$(3, 1.0, 2)$;$(4, 1.0, 2)$;$(5, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	$(2, 1.0, 3)$;$(3, 1.0, 3)$;
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;$(4, 0.8, 1)$;$(5, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;$(7, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;$(4, 1.0, 4)$;$(5, 1.0, 4)$;$(6, 1.0, 1)$;$(7, 1.0, 1)$;$(8, 1.0, 1)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 12)$;
(P_2, P)	$(\bullet, 1.0, 5)$;

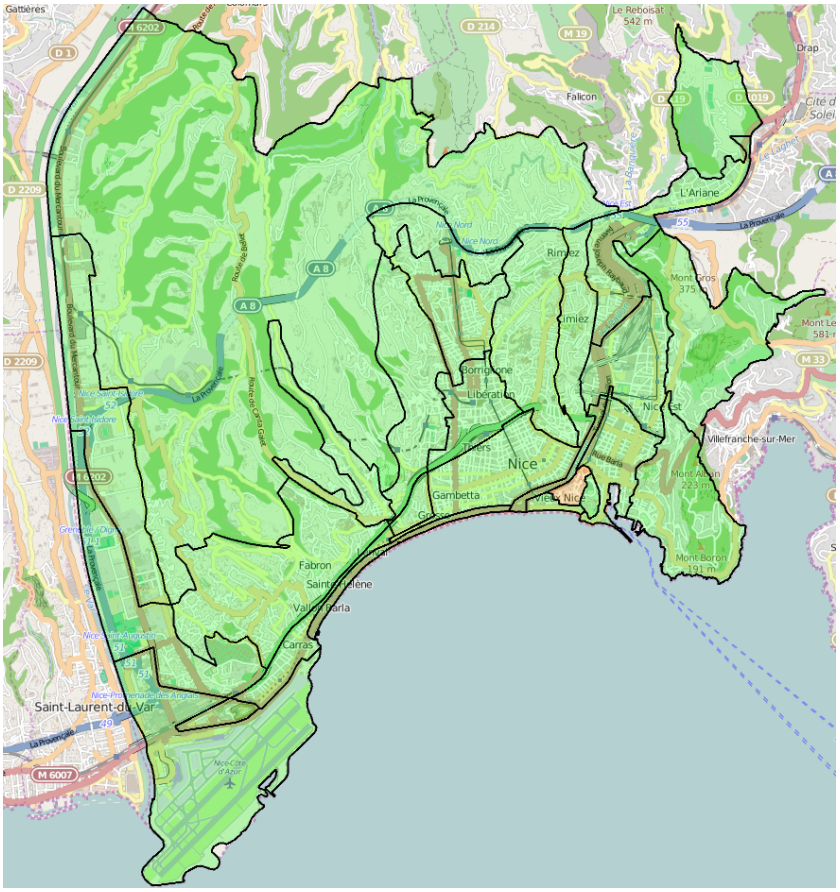


FIGURE A.2 – Zone impactée par le scénario de Vésubie sur l'ensemble de la ville de Nice

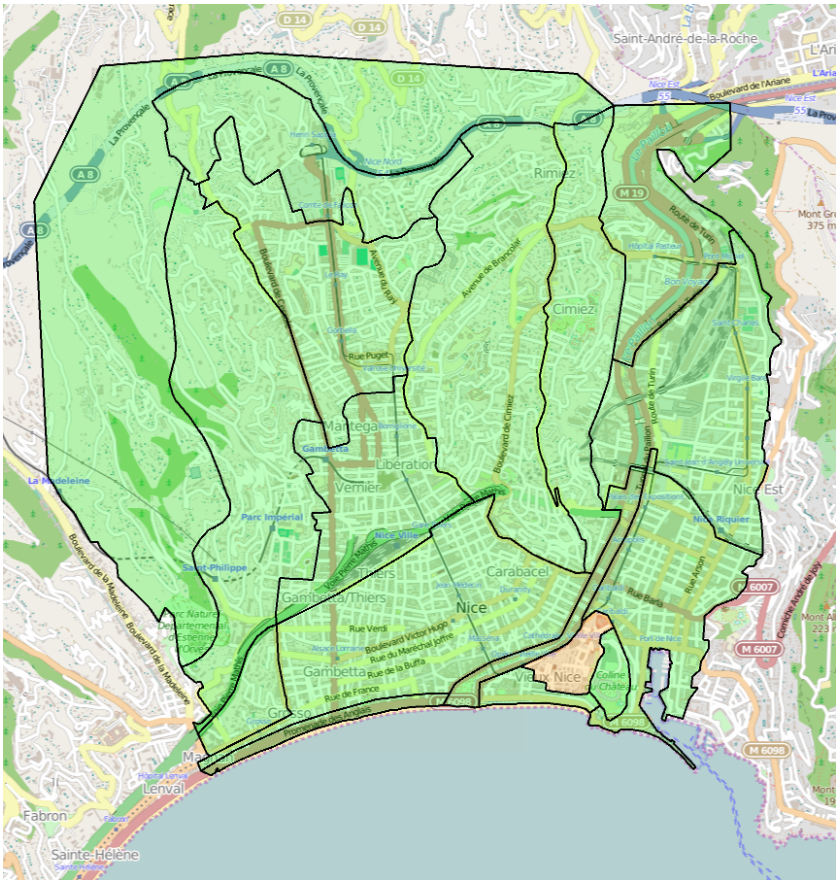


FIGURE A.3 – Zone impactée par le scénario de Vésubie sur le centre ville de Nice

TABLE A.11 – État de la mémoire de décision à l'instant $t = 11$

Arcs	Moment, Sécurité et Utilisation de chaque arc (M,S,U)
(S, S_1)	$(\bullet, 1.0, 6)$;
(S, S_2)	$(\bullet, 1.0, 12)$;
(S_1, U_1)	$(1, 1.0, 3)$;$(2, 1.0, 3)$;
(S_2, U_2)	$(0, 1.0, 2)$;$(1, 1.0, 2)$;$(2, 1.0, 2)$;$(3, 1.0, 2)$;$(4, 1.0, 2)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(U_1, U_3)	
(U_1, U_4)	$(2, 1.0, 3)$;$(3, 1.0, 3)$;
(U_2, U_1)	
(U_2, U_3)	$(1, 1.0, 1)$;$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;$(7, 1.0, 1)$;
(U_2, U_5)	$(1, 0.8, 1)$;$(2, 0.8, 1)$;$(3, 0.8, 1)$;$(4, 0.8, 1)$;$(5, 0.8, 1)$;
(U_3, U_4)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;$(7, 1.0, 1)$;$(8, 1.0, 1)$;
(U_3, U_5)	
(U_4, U_5)	
(U_4, P_1)	$(3, 1.0, 1)$;$(4, 1.0, 4)$;$(5, 1.0, 4)$;$(6, 1.0, 1)$;$(7, 1.0, 1)$;$(8, 1.0, 1)$;$(9, 1.0, 1)$;
(U_5, P_2)	$(2, 1.0, 1)$;$(3, 1.0, 1)$;$(4, 1.0, 1)$;$(5, 1.0, 1)$;$(6, 1.0, 1)$;
(P_1, P)	$(\bullet, 1.0, 13)$;
(P_2, P)	$(\bullet, 1.0, 5)$;

Bibliographie

[Adams et Galea, 2011] Adams, A. et Galea, E. (2011). Evacuation dynamics of children – walking speeds, flows through doors in daycare centers. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 128–138. Springer US.

[Administration, 1990] Administration, F. A. (1990). 14 cfr 25.803 - emergency evacuation. `http://lessonslearned.faa.gov/ChinaAirlines120/25.803.pdf`.

[Agra *et al.*, 2013] Agra, A., Christiansen, M., Figueiredo, R., Hvattum, L. M., Poss, M. et Requejo, C. (2013). The robust vehicle routing problem with time windows. *Computers & Operations Research*, 40(3):856 – 866. Transport Scheduling.

[Aissi *et al.*, 2009] Aissi, H., Bazgan, C. et Vanderpooten, D. (2009). Min–max and min–max regret versions of combinatorial optimization problems : A survey. *European Journal of Operational Research*, 197(2):427 – 438.

[Alizadeh, 2011] Alizadeh, R. (2011). A dynamic cellular automaton model for evacuation process with obstacles. *Safety Science*, 49(2):315–323.

[Altay et Green III, 2006] Altay, N. et Green III, W. G. (2006). OR/MS research in disaster operations management. *European Journal of Operational Research*, 175(1):475 – 493.

[Appert-Rolland *et al.*, 2014] Appert-Rolland, C., Cividini, J., Hilhorst, H. et Degond, P. (2014). Pedestrian flows : From individuals to crowds. *Transportation Research Procedia*, 2(0):468 – 476. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Aronson, 1989] Aronson, J. E. (1989). A survey of dynamic network flows. *Annals of Operations Research*, 20(1):1–66.

[Baumann, 2006] Baumann, N. (2006). Solving evacuation problems efficiently — earliest arrival flows with multiple sources. *In FOCS 2006 Proceedings*, pages 399–408.

[Baumann, 2007] Baumann, N. (2007). *Evacuation by Earliest Arrival Flows*. Thèse de doctorat.

[Baumann et Köhler, 2007] Baumann, N. et Köhler, E. (2007). Approximating earliest arrival flows with flow-dependent transit times. *Discrete Applied Mathematics*, 155(2): 161 – 171. 29th Symposium on Mathematical Foundations of Computer Science {MFCS} 2004.

[Baumann et Skutella, 2009] Baumann, N. et Skutella, M. (2009). Earliest arrival flows with multiple sources. *Mathematics of Operations Research*, 34(2):499–512.

[Ben-Tal *et al.*, 2009] Ben-Tal, A., Ghaoui, L. E. et Nemirovski, A. (2009). *Robust Optimization*. Princeton University Press, Princeton and Oxford.

[Ben-Tal *et al.*, 2004] Ben-Tal, A., Goryashko, A., Guslitzer, E. et Nemirovski, A. (2004). Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376.

[Bengtson *et al.*, 2011] Bengtson, S., Kecklund, L., Siré, E., Andrée, K. et Willander, S. (2011). How do people with disabilities consider fire safety and evacuation possibilities in historical buildings ? *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 275–284. Springer US.

[Bertsimas et Sim, 2003] Bertsimas, D. et Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming Series B*, 98:2003.

[Bertsimas et Sim, 2004] Bertsimas, D. et Sim, M. (2004). The price of robustness. *Operations Research*, 52(1):35–53.

[Best *et al.*, 2014] Best, A., Curtis, S., Kasik, D., Senesac, C., Sikora, T. et Manocha, D. (2014). Ped-air : A simulator for loading, unloading, and evacuating aircraft. *Transportation Research Procedia*, 2(0):273 – 281. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Billionnet *et al.*, 2014] Billionnet, A., Costa, M.-C. et Poirion, P. (2014). 2-Stage Robust MILP with continuous recourse variables . *Discrete applied mathematics*, 170(19): 21–32.

[Bish, 2011] Bish, D. R. (2011). Planning for a bus-based evacuation. *OR Spectrum*, 33(3):629–654.

[Borrmann *et al.*, 2012] Borrmann, A., Kneidl, A., Köster, G., Ruzika, S. et Thiemann, M. (2012). Bidirectional coupling of macroscopic and microscopic pedestrian evacuation models. *Safety Science*, 50(8):1695 – 1703. Evacuation and Pedestrian Dynamics.

[Bretschneider, 2012] Bretschneider, S. (2012). *Mathematical models for evacuation planning in urban areas*, volume 659. Springer Science & Business Media.

[Bretschneider et Kimms, 2011] Bretschneider, S. et Kimms, A. (2011). A basic mathematical model for evacuation problems in urban areas. *Transportation research part A : policy and practice*, 45(6):523–539.

[Burkard *et al.*, 1993] Burkard, R. E., Dlaska, K. et Klinz, B. (1993). The quickest flow problem. *Zeitschrift für Operations Research*, 37(1):31–58.

[Campbell *et al.*, 2006] Campbell, A. M., Lowe, T. J. et Zhang, L. (2006). Upgrading arcs to minimize the maximum travel time in a network. *Networks*, 47(2):72–80.

[Chalmet *et al.*, 1982] Chalmet, L., Francis, R. et Saunders, P. (1982). Network models for building evacuation. *Fire Technology*, 18(1):90–113.

[Choi *et al.*, 1988a] Choi, W., Hamacher, H. et Tufekci, S. (1988a). Modeling of building evacuation problems by network flows with side constraints. *European Journal of Operational Research*, 35(1):98 – 110.

[Choi *et al.*, 1988b] Choi, W., Hamacher, H. et Tufekci, S. (1988b). Modeling of building evacuation problems by network flows with side constraints. *European Journal of Operational Research*, 35(1):98 – 110.

[Coutinho-Rodrigues *et al.*, 2012] Coutinho-Rodrigues, J., Tralhão, L. et Alçada-Almeida, L. (2012). Solving a location-routing problem with a multiobjective approach : the design of urban evacuation plans. *Journal of Transport Geography*, 22(0):206 – 218. Special Section on Rail Transit Systems and High Speed Rail.

[Daamen et Hoogendoorn, 2012] Daamen, W. et Hoogendoorn, S. (2012). Emergency door capacity : Influence of door width, population composition and stress level. *Fire Technology*, 48(1):55–71.

[Dantzig, 1951] Dantzig, G. (1951). Application of the simplex method to a transportation problem. *In* Koopmans, T., éditeur : *Activity analysis of production and allocation*, pages 359–373. J. Wiley, New York.

[Demgensky *et al.*, 2004] Demgensky, I., Noltemeier, H. et Wirth, H.-C. (2004). Optimizing cost flows by edge cost and capacity upgrade. *Journal of Discrete Algorithms*, 2(4):407 – 423. The 26th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2000).

[Dilkina *et al.*, 2011] Dilkina, B., Lai, K. J. et Gomes, C. P. (2011). Upgrading shortest paths in networks. *In* Achterberg, T. et Beck, J., éditeurs : *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 6697 de *Lecture Notes in Computer Science*, pages 76–91. Springer Berlin Heidelberg.

[Disser et Skutella, 2015] Disser, Y. et Skutella, M. (2015). The simplex algorithm is np-mighty. *In Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 858–872. SIAM.

[Edmonds et Karp, 1972] Edmonds, J. et Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2):248–264.

[Fleischer et Skutella, 2007] Fleischer, L. et Skutella, M. (2007). Quickest flows over time. *SIAM Journal on Computing*, 36(6):1600–1630.

[Fleischer, 2001] Fleischer, L. K. (2001). Universally maximum flow with piecewise-constant capacities. *Networks*, 38(3):115–125.

[Forcael *et al.*, 2014] Forcael, E., González, V., Orozco, F., Vargas, S., Pantoja, A. et Moscoso, P. (2014). Ant colony optimization model for tsunamis evacuation routes. *Computer-Aided Civil and Infrastructure Engineering*, 29(10):723–737.

[Ford et Fulkerson, 1962] Ford, L. et Fulkerson, D. R. (1962). *Flows in networks*, volume 1962. Princeton University Press.

[Ford et Fulkerson, 1958] Ford, L. R. et Fulkerson, D. R. (1958). *Constructing Maximal Dynamic flows from Static flows*, volume 6. Operations Research.

[Fujiyama et Tyler, 2011] Fujiyama, T. et Tyler, N. (2011). Free walking speeds on stairs : Effects of stair gradients and obesity of pedestrians. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 95–106. Springer US.

[Gabrel *et al.*, 2014] Gabrel, V., Lacroix, M., Murat, C. et Remli, N. (2014). Robust location transportation problems under uncertain demands. *Discrete Applied Mathematics*, 164, Part 1:100 – 111. Combinatorial Optimization.

[Gale, 1959] Gale, D. (1959). Transient flows in networks. *Michigan Math. J.*, 6(1):59–63.

[Garey et Johnson, 1975] Garey, M. R. et Johnson, D. S. (1975). Complexity results for multiprocessor scheduling under resource constraints. *SIAM Journal on Computing*, 4(4):397–411.

[Garey et Johnson, 1979] Garey, M. R. et Johnson, D. S. (1979). Computers and intractability : A guide to the theory of np-completeness. *Series of Books in the Mathematical Sciences*.

[Goerigk *et al.*, 2013] Goerigk, M., Grün et Hessler, P. (2013). Branch and bound algorithms for the bus evacuation problem. *Computers and Operations Research*, 40(12): 3010 – 3020.

[Goerigk et Ndiaye, 2014] Goerigk, M. et Ndiaye, I. A. (2014). Robust flows with losses and improvability in evacuation planning.

[Goerigk et Schöbel, 2013] Goerigk, M. et Schöbel, A. (2013). Algorithm engineering in robust optimization. Rapport technique, Preprint-Reihe, Institut für Numerische und Angewandte Mathematik, Universität Göttingen. Submitted.

[Goldberg *et al.*, 1989] Goldberg, A. V., Plotkin, S. A. et Éva Tardos (1989). Combinatorial algorithms for the generalized circulation problem. *MATHEMATICS OF OPERATIONS RESEARCH*, 16:351–379.

[Gondran et Minoux, 2009] Gondran, M. et Minoux, M. (2009). *Graphes et algorithmes*. Collection EDF R&D. Edition TEC&DOC Lavoisier.

[Göttlich *et al.*, 2011] Göttlich, S., Kühn, S., Ohst, J. P., Ruzika, S. et Thiemann, M. (2011). Evacuation dynamics influenced by spreading hazardous material. *NHM*, 6(3):443–464.

[Greenwood *et al.*, 2014] Greenwood, D., Sharma, S. et Johansson, A. (2014). Gap analysis of current industrial challenges and the state-of-the-art in pedestrian modelling. *Transportation Research Procedia*, 2(0):219 – 227. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Groß et Skutella, 2012] Gross, M. et Skutella, M. (2012). Generalized maximum flows over time. *In Proceedings of the 9th International Conference on Approximation and Online Algorithms*, WAOA'11, pages 247–260. Springer-Verlag.

[Groß et Skutella, 2015] Gross, M. et Skutella, M. (2015). A tight bound on the speed-up through storage for quickest multi-commodity flows. *Operations Research Letters*, 43(1):93–95.

[Hamacher *et al.*, 2011a] Hamacher, H., Heller, S., Klein, W., Köster, G. et Ruzika, S. (2011a). A sandwich approach for evacuation time bounds. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 503–513. Springer US.

[Hamacher *et al.*, 2011b] Hamacher, H., Heller, S., Klein, W., Köster, G. et Ruzika, S. (2011b). A sandwich approach for evacuation time bounds. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 503–513. Springer US.

[Hamacher et Tjandra, 2001] Hamacher, H. W. et Tjandra, S. A. (2001). Mathematical modeling of evacuation problems : A state of the art. *In Pedestrian and Evacuation Dynamics (Schreckenberg, M. and Sharma, S. D. eds)*, 1964:227–266.

[Hamacher et Tjandra, 2002] Hamacher, H. W. et Tjandra, S. A. (2002). Mathematical modeling of evacuation problems : A state of the art. *In* Schreckenberg, M. et Sharma, S. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 227–266. Springer Verlag.

[Hamacher et Tjandra, 2003] Hamacher, H. W. et Tjandra, S. A. (2003). Earliest arrival flows with time-dependent data. Rapport technique 88, Fachbereich Mathematik - TU Kaiserslautern. `https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/1449`.

[Hiyoshi *et al.*, 2014] Hiyoshi, H., Tanioka, Y., Hamamoto, T., Matsumoto, K. et Chiba, K. (2014). Pedestrian movement model based on voronoi cellular automata. *Transportation Research Procedia*, 2(0):336 – 343. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Hoppe et Tardos, 1994] Hoppe, B. et Tardos, E. (1994). Polynomial time algorithms for some evacuation problems. *In Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '94, pages 433–441, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

[Hoppe et Tardos, 2000] Hoppe, B. et Tardos, E. (2000). The quickest transshipment problem. *Math. Oper. Res.*, 25(1):36–62.

[INSEE, 2015] INSEE (2015). Data. `http://www.insee.fr`.

[Jarvis et Ratliff, 1982] Jarvis, J. J. et Ratliff, H. D. (1982). Note—some equivalent objectives for dynamic network flow problems. *Management Science*, 28(1):106–109.

[Jewell, 1962] Jewell, W. S. (1962). New methods in mathematical programming—optimal flow through networks with gains. *Operations Research*, 10(4):476–499.

[Karp, 1972] Karp, R. M. (1972). *Reducibility among combinatorial problems*. Springer.

[Kinsey *et al.*, 2011] Kinsey, M. J., Galea, E. R. et Lawrence, P. J. (2011). Stairs or lifts ? - a study of human factors associated with lift/elevator usage during evacuations using an online survey. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 627–636. Springer US.

[Klinz et Woeginger, 2004] Klinz, B. et Woeginger, G. J. (2004). Minimum-cost dynamic flows : The series-parallel case. *Networks*, 43(3):153–162.

[Klüpfel *et al.*, 2001] Klüpfel, H., Meyer-König, T., Wahle, J. et Schreckenberg, M. (2001). Microscopic simulation of evacuation processes on passenger ships. *In* Bandini, S. et Worsch, T., éditeurs : *Theory and Practical Issues on Cellular Automata*, pages 63–71. Springer London.

[Klüpfel, 2014] Klüpfel, H. (2014). Large scale multi-modal simulation of pedestrian traffic. *Transportation Research Procedia*, 2(0):446 – 451. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Köhler *et al.*, 2008] Köhler, E., Möhring, R. H. et Spenke, I. (2008). Quickest flows : A practical model. *Technische Universität Berlin*.

[Kouvelis et Yu, 1997] Kouvelis, P. et Yu, G. (1997). *Robust Discrete Optimization and Its Applications*. Kluwer Academic Publishers.

[Krumke *et al.*, 1998] Krumke, S. O., Marathe, M. V., Noltemeier, H., Ravi, R. et Ravi, S. (1998). Network improvement problems. *Network Design : Connectivity and Facilities Location, AMSDIMACS Volume Series in Discrete Mathematics and Theoretical Computer Science*, 40:247–268.

[Lämmel *et al.*, 2011] Lämmel, G., Klüpfel, H. et Nagel, K. (2011). Risk minimizing evacuation strategies under uncertainty. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 287–296. Springer US.

[Larusdottir et Dederichs, 2011] Larusdottir, A. R. et Dederichs, A. S. (2011). Evacuation dynamics of children – walking speeds, flows through doors in daycare centers. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 139–147. Springer US.

[Lemoine *et al.*, 2014] Lemoine, A., Bernardie, S., Brivois, O., De Martin, F., Desramaut, N., Le Roy, S., Monfort Climent, D., Negulescu, C., Pedreros, R., Sedan, O., Chan Vong, Q., Vagner, A. et Foerster, E. (2014). Ligurian earthquake : Seismic and tsunami scenario modeling, from hazard to risk assessment towards evacuation planning. *In 2nd European conference on earthquake engineering and seismology : 2ECEES 2014*, Istanbul, Turkey.

[L.G.Chalmet *et al.*, 1982] L.G.Chalmet, Francis, R. et Saunders, P. (1982). Network models for building evacuation. *Fire Technology*, 18(1):90–113.

[Li *et al.*, 2011] Li, H., Maohua, Z., Congling, S., Jiehong, S., Haicheng, C. et Qiaoxiang, X. (2011). Experimental research on investigation of metro passenger evacuation behaviors in case of emergency. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 173–184. Springer US.

[Liebchen *et al.*, 2009] Liebchen, C., Lübbecke, M., Möhring, R. et Stiller, S. (2009). The concept of recoverable robustness, linear programming recovery, and railway applications. *In* Ahuja, R., Möhring, R. et Zaroliagis, C., éditeurs : *Robust and Online Large-Scale Optimization*, volume 5868 de *Lecture Notes in Computer Science*, pages 1–27. Springer Berlin Heidelberg.

[Lin et Mouratidis, 2013] Lin, Y. et Mouratidis, K. (2013). Best upgrade plans for large road networks. *In* Nascimento, M., Sellis, T., Cheng, R., Sander, J., Zheng, Y., Kriegel, H.-P., Renz, M. et Sengstock, C., éditeurs : *Advances in Spatial and Temporal Databases*, volume 8098 de *Lecture Notes in Computer Science*, pages 223–240. Springer Berlin Heidelberg.

[Lu, 2011] Lu, C. (2011). The evacuation training problems of an earthquake in china. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 121–127. Springer US.

[Miller-Hooks et Stock Patterson, 2004] Miller-Hooks, E. et Stock Patterson, S. (2004). On solving quickest time problems in time-dependent, dynamic networks. *Journal of Mathematical Modelling and Algorithms*, 3(1):39–71.

[Minieka, 1973] Minieka, E. (1973). Maximal, lexicographic and dynamic network flows. 21(2):517–527.

[Minieka, 1974] Minieka, E. (1974). Dynamic network flows with arc changes. *Networks*, 4(3):255–265.

[Ndiaye et Neron, 2013a] Ndiaye, I. A. et Neron, E. (2013a). Calcul de chemins multicritères durée/sécurité pour la construction de plans d'évacuation de personnes. *In 14ème congrès de la Société de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, pages –, Troyes, France.

[Ndiaye et Neron, 2013b] Ndiaye, I. A. et Neron, E. (2013b). Multicriteria evacuation plan for natural disasters. *In 22nd International Conference on Multiple Criteria Decision Making*, pages –, Malaga, Spain.

[Ndiaye *et al.*, 2014a] Ndiaye, I. A., Neron, E. et Goerigk, M. (2014a). Evacuation de piétons lors de catastrophes naturelles avec prise en compte de la sécurité. *In ROADEF - 15ème congrès annuel de la Société française de recherche opérationnelle et d'aide à la décision*, Bordeaux, France. Société française de recherche opérationnelle et d'aide à la décision.

[Ndiaye *et al.*, 2014b] Ndiaye, I. A., NERON, E. et Goerigk, M. (2014b). Multicriteria pedestrian evacuation plan for natural disasters with safety and duration. *In Proc. 3rd International Symposium on Combinatorial Optimization(Lisboa, Portugal, March 2014)*.

[Ndiaye *et al.*, 2014c] Ndiaye, I. A., Neron, E. et Jouglet, A. (2014c). Macroscopic evacuation plans for natural disasters : A lexicographical approach for duration and safety criteria : Lex((q|s) flow).

[Ndiaye *et al.*, 2014d] Ndiaye, I. A., Neron, E., Linot, A., Monmarche, N. et Goerigk, M. (2014d). A new model for macroscopic pedestrian evacuation planning with safety and duration criteria. *Transportation Research Procedia*, 2:486 – 494. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Ndiaye *et al.*, 2014e] Ndiaye, I. A., Néron, E. et Goerigk, M. (2014e). Multicriteria pedestrian evacuation plan for natural disasters with safety and duration. *In Proc. 3rd International Symposium on Combinatorial Optimization(Lisboa, Portugal, March 2014)*.

[Ogier, 1988] Ogier, R. G. (1988). Minimum-delay routing in continuous-time dynamic networks with piecewise-constant capacities. *Networks*, 18(4):303–318.

[Okamoto *et al.*, 2011] Okamoto, E., Hasemi, Y., Moriyama, S. et Okada, N. (2011). Experiments for the feasibility study of the evacuation by moving escalator in public space. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 649–659. Springer US.

[Oldham, 2001] Oldham, J. D. (2001). Combinatorial approximation algorithms for generalized flow problems. *Journal of Algorithms*, 38(1):135 – 169.

[Olfati-Saber, 2006] Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems : algorithms and theory. *Automatic Control, IEEE Transactions on*, 51(3):401–420.

[Onaga, 2006] Onaga, K. (2006). A dynamic programming of optimum flows in lossy communication nets. *IEEE Transactions on Circuit Theory*, 13:282–287.

[Opasanon et Miller-Hooks, 2009] Opasanon, S. et Miller-Hooks, E. (2009). The safest escape problem. *Journal of the Operational Research Society*, 60(12):1749–1758.

[OpenStreetMap, 2015a] OpenStreetMap (2015a). Openstreetmap data for provence alpes-cote-d'azur (including nice city). `http://download.geofabrik.de/europe/france/provence-alpes-cote-d-azur.html`.

[OpenStreetMap, 2015b] OpenStreetMap (2015b). Openstreetmap website. `http://openstreetmap.fr`.

[Ordóñez et Zhao, 2007] Ordóñez, F. et Zhao, J. (2007). Robust capacity expansion of network flows. *Networks*, 50(2):136–145.

[Peacock *et al.*, 2010] Peacock, R. D., Hoskins, B. L. et Kuligowski, E. D. (2010). Overall and local movement speeds during fire drill evacuations in buildings up to 31 stories. *National Institute of Standards and Technology*, 1675(2):1 – 32.

[Powell *et al.*, 1995] Powell, W. B., Jaillet, P. et Odoni, A. (1995). Stochastic and dynamic networks and routing. *Handbooks in operations research and management science*, 8:141–295.

[Radzik, 1998] Radzik, T. (1998). Faster algorithms for the generalized network flow problem. *Mathematics of Operations Research*, 23(1):69–100.

[Rosen *et al.*, 1991] Rosen, J. B., Sun, S.-Z. et Xue, G.-L. (1991). Algorithms for the quickest path problem and the enumeration of quickest paths. *Computers & Operations Research*, 18(6):579–584.

[Rupprecht *et al.*, 2011] Rupprecht, T., Klingsch, W. et Seyfried, A. (2011). Influence of geometry parameters on pedestrian flow through bottleneck. *In* Peacock, R. D., Kuligowski, E. D. et Averill, J. D., éditeurs : *Pedestrian and Evacuation Dynamics*, pages 71–80. Springer US.

[Sahni, 1974] Sahni, S. (1974). Computationally related problems. *SIAM Journal on Computing*, 3(4):262–279.

[Schmidt et Skutella, 2014] Schmidt, M. et Skutella, M. (2014). Earliest arrival flows in networks with multiple sinks. *Discrete Applied Mathematics*, 164:320–327.

[Schwarz et Krumke, 1998] Schwarz, S. et Krumke, S. (1998). On budget-constrained flow improvement. *Information Processing Letters*, 66(6):291 – 297.

[Siarry, 2014] Siarry, P. (2014). *Métaheuristiques recuit simulé, recherche avec tabous, recherche à voisinages variables, méthode GRASP, algorithmes évolutionnaires, fourmis artificielles, essaims particulaires et autres méthodes d'optimisation.* Eyrolles, Paris.

[Takizawa *et al.*, 2012] Takizawa, A., Inoue, M. et Katoh, N. (2012). An emergency evacuation planning model using the universally quickest flow. *The Review of Socionetwork Strategies*, 6(1):15–28.

[Tjandra, 2003] Tjandra, S. A. (2003). *Dynamic network optimisation with application to the evacuation problem.* Thèse de doctorat.

[Twarogowska *et al.*, 2014] Twarogowska, M., Goatin, P. et Duvigneau, R. (2014). Comparative study of macroscopic pedestrian models. *Transportation Research Procedia*, 2(0):477 – 485. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Wang *et al.*, 2008] Wang, P., Luh, P. B., Chang, S. et Sun, J. (2008). Modeling and optimization of crowd guidance for building emergency evacuation. *In* Xiong, C., Liu, H., Huang, Y. et Xiong, Y., éditeurs : *Intelligent Robotics and Applications*, volume 5315 de *Lecture Notes in Computer Science*, pages 1–6. Springer Berlin Heidelberg.

[Wayne, 1999] Wayne, K. D. (1999). *Generalized Maximum Flow Algorithms*. PhD Thesis. Cornell University, New York, United States.

[Wayne, 2002] Wayne, K. D. (2002). A polynomial combinatorial algorithm for generalized minimum cost flow. *Math. Oper. Res.*, 27(3):445–459.

[Wilkinson, 1971] Wilkinson, W. L. (1971). An algorithm for universal maximal dynamic flows in a network. *Operations Research*, 19(7):1602–1612.

[Xie *et al.*, 2010] Xie, C., Lin, D.-Y. et Waller, S. T. (2010). A dynamic evacuation network optimization problem with lane reversal and crossing elimination strategies. *Transportation Research Part E : Logistics and Transportation Review*, 46(3):295 – 316.

[Xie et Turnquist, 2011] Xie, C. et Turnquist, M. A. (2011). Lane-based evacuation network optimization : An integrated lagrangian relaxation and tabu search approach. *Transportation Research Part C : Emerging Technologies*, 19(1):40 – 63.

[Yamada, 1996] Yamada, T. (1996). A network flow approach to a city emergency evacuation planning. *International Journal of Systems Science*, 27(10):931–936.

[Yuan et Han, 2010] Yuan, F. et Han, L. D. (2010). A multi-objective optimization approach for evacuation planning. *Procedia Engineering*, 3(0):217 – 227. 1st Conference on Evacuation Modeling and Management.

[Zeng et Zhao, 2013] Zeng, B. et Zhao, L. (2013). Solving two-stage robust optimization problems using a column-and-constraint generation method. *Operations Research Letters*, 41(5):457 – 461.

[Zheng *et al.*, 2010] Zheng, X., Li, W. et Guan, C. (2010). Simulation of evacuation processes in a square with a partition wall using a cellular automaton model for pedestrian dynamics. *Physica A : Statistical Mechanics and its Applications*, 389(11):2177 – 2188.

[Zhi-Ming *et al.*, 2014] Zhi-Ming, F., Wei, L., Xiao-Dong, L. et Wei-Guo, S. (2014). Study of boeing 777 evacuation using a finer-grid civil aircraft evacuation model. *Transportation Research Procedia*, 2(0):246 – 254. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

Résumé :

Les travaux présentés dans cette thèse visent à proposer des méthodes de routage d'une population de masse à travers un réseau perturbé dont les données varient dans le temps pour l'aide à la conception de plan d'évacuation. Ce problème s'illustre parfaitement en cas de catastrophe d'origine humaine ou naturelle où les populations (potentiellement) impactées par ces sinistres doivent quitter leur lieux de vie pour une période pouvant aller d'un à plusieurs jours.

Dans la littérature, ces routages de masse sont souvent modélisés comme des problèmes de flots dynamiques dont l'objectif est de minimiser la durée globale du transfert des individus depuis un certain nombre de points de départs dangereux vers des points d'arrivée sûrs. Toutefois, peu de travaux prennent en compte la notion de sécurité durant ce routage et encore moins le déploiement d'agents (policiers, sapeur-pompiers, ambulanciers,...) pouvant sécuriser et/ou faciliter le déplacement des personnes. Dans ce cadre, la notion de sécurité peut être vue comme un élément ayant une influence sur la qualité de vie des personnes dont nous organisons le déplacement. En effet, elle peut être liée à un nuage radioactif, un feu de forêt, un tsunami, un tremblement de terre ou une inondation pouvant rendre les chemins empruntés dangereux et moins praticables pour les individus. Ainsi nous avons un problème où deux critères potentiellement conflictuels sont à optimiser afin de garantir que les individus aient le moins de dommages possibles dus à l'évacuation. Pour minimiser la durée de l'évacuation, il faudra prendre un ensemble de plus courts chemins alors que l'optimisation de la sécurité consistera à faire des détours pour prendre des chemins éventuellement plus longs mais aussi plus sûrs.

Pour résoudre ce problème d'optimisation, une approche itérative permettant de résoudre différents niveaux de complexité est proposée. Notre première contribution porte sur l'amélioration de la complexité des algorithmes concernant le problème d'optimisation de la durée moyenne d'évacuation puis par extension, nous prenons en compte la notion de sécurité durant ce routage grâce à une approche lexicographique. Notre seconde contribution porte sur la modélisation de la notion de sécurité comme un problème de flot généralisé avec pertes. Enfin, notre dernière contribution porte sur le déploiement des forces de secours dans un contexte de flot généralisé avec perte où ces dernières seront positionnées sur les zones critiques.

Mots clés :

réseau dynamique, évacuation, transbordement, modèle microscopique, modèle macroscopique, flot généralisé, multicritère, durée, sécurité, routage.

Abstract :

The work presented in this thesis aims to propose methods for routing a mass population through a disturbed network whose data vary over time. This problem can be raised by disasters due to humans or natural events where people (potentially) affected have to leave their living places for a period of one to several days.

In the literature, mass routing are often modeled as dynamic flow problems whose objective is to minimize the overall duration of the evacuation process from a set of gathering points towards another set of shelter locations. However few papers take into account the concept of safety during this routing nor deploying task forces that can secure or facilitate this process. In this context, the safety can be seen as a danger affecting the quality of life of people we organize the trip. So, the safety can be seen as a danger that influence the health of the people we are trying to evacuate. Indeed, hazardous event can be related to a radioactive cloud, a fire, a tsunami, an earthquake or a flooding which make some of paths becoming dangerous or less usable by evacuees. So we have an optimization problem where two conflicting criteria are to be optimized to guarantee that the individuals will have least damage possible during the evacuation. Then, during the evacuation, it will be necessary to have a set of shorter paths(ways) while the optimization of the safety(security) will consist in making detours to take longer but safer paths.

To solve this optimization problem, an iterative approach allowing to solve various levels of complexity is proposed. Our first contribution concerns the improvement of the complexity of the algorithms regarding the optimization the average duration of the evacuation, then by extension, we take into account the notion of safety by applying a lexicographical approach. Our second contribution concerns the modelling of the notion of safety as a generalized flow with losses problem. Finally, our last contribution concerns the deployment of the task forces over critical points of the network to improve their safety.

Keywords :

dynamic network, evacuation, transshipment, macroscopic model, microscopic model, generelized flow, multicriteria, duration, safety, routing.