Automatisation et optimisation d'une plateforme logistique

par

Rémy CARPENTIER

RAPPORT DE PROJET PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE COMME EXIGENCE PARTIELLE À L'OBTENTION DE LA MAÎTRISE EN GÉNIE DE LA PRODUCTION AUTOMATISÉE

MONTRÉAL, LE 12 AVRIL 2018

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

PRÉSENTATION DU JURY

CE RAPPORT DE PROJET A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Amin Chabaane, directeur de projet
Génie de la production automatisée à l'École de technologie supérieure

M. Mustapha Ouhimmou, président du jury
Génie de la production automatisée à l'École de technologie supérieure

REMERCIEMENTS

Tout d’abord, je tiens à remercier l’entreprise Transport Bourassa, tout particulièrement Marie-Ève Turcot, directrice générale des opérations et Stéphane Benoit, directeur du quai de transbordement, d’avoir rendu ce projet possible, de m’avoir chaleureusement accueilli dans leurs locaux et d'avoir répondu à toutes mes questions.

Je voudrais également remercier mon directeur de projet, le professeur Amin Chaabane, de m’avoir guidé, de m'avoir fait confiance et de m'avoir encouragé durant ces huit mois de projet.

Un grand merci à Rim Larbi d’avoir partagé ses connaissances et pour son aide lors du développement du modèle mathématique. Ses recherches et sa thèse portent exactement sur le sujet de ce projet, ce qui m’a permis de progresser plus rapidement.

Enfin, merci à tous mes proches de m’avoir soutenu durant ces deux ans de maîtrise.

AUTOMATISATION ET OPTIMISATION D'UNE PLATEFORME LOGISTIQUE

Rémy CARPENTIER

RÉSUMÉ

Le transport est une activité vitale pour notre mode de vie moderne. Le cross-docking est une technique qui permet d'importantes économies financières et un service de transport plus rapide. L'entreprise avec qui nous avons réalisé ce projet utilise cette technique, mais arrive aujourd'hui à saturation concernant les produits transitant par son entrepôt. Pour faire face à la concurrence qui continue de s'étendre, il nous a été demandé de trouver une solution pour automatiser et optimiser l'affectation des camions aux quais en prenant en compte leurs contenus. Pour répondre à ce besoin, nous nous sommes appuyés sur les recherches actuelles dans le domaine pour réaliser un modèle mathématique propre au fonctionnement de l'entreprise. Nous avons mis en place des méthodes capables de résoudre des problèmes de petite taille et développé un algorithme génétique capable de résoudre des problèmes de taille industrielle.

Mots-clés : cross-docking, logistique, "*truck-to-door scheduling*", algorithme génétique, étude de cas

AUTOMATISATION ET OPTIMISATION D'UNE PLATEFORME LOGISTIQUE

Rémy CARPENTIER

ABSTRACT

Transport is an essential activity for our modern way of life. The cross-docking approach improves speed delivery service and allows money savings on storage cost. This project has been made with a partner company which uses cross-docking. Recently, the company has reached the maximum capacity of its warehouse. To face the competition, we have to find a way to optimize and to automate assignation of trucks to docks while taking into account cargo of trucks. First of all, we studied literature and actual research on cross-docking. We then made a mathematical model who works for the real warehouse. Afterwards, we developed a mathematical program to solve small problems and then a genetic algorithm to solve industrial-sized problems.

Keywords: cross-docking, logistics, truck-to-door scheduling, genetic algorithm, case study

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

LTL	*Less than truck load*
FTL	*Full truck load*
GA	*Genetic algorithm*
TS	*Tabu search*
PSO	*Particle swarm optimization*
ACO	*Ant colony optimization*
DE	*Differential evolution*
VNS	*Variable neighborhood search*
ICA	*Imperialist competitive algorithm*
SA	*Simulated annealing*
EMA	*Electro-magnetism algorithm*

INTRODUCTION

0.1 Contexte de recherche

Depuis le début du XXe siècle, la mondialisation est devenue un terme important dans la croissance économique. Après deux guerres mondiales, la délocalisation vers l'Asie, la consommation en forte hausse, l'essor des nouvelles technologies et le commerce sur internet, le siècle passé a rendu vital une bonne gestion mondiale des flux de marchandises, nouvellement appelée logistique. Utilisée initialement pour définir les opérations de déplacements d'équipements militaires, la logistique englobe maintenant tous les moyens mis en œuvre pour organiser le bon fonctionnement d'une entreprise, comme la manutention, le transport et le conditionnement.

De nombreux axes de recherches ont émergé pour répondre aux problèmes complexes auxquels les entreprises doivent faire face pour suivre la mondialisation des marchés. Un de ces axes est la recherche opérationnelle. Ce domaine de recherche a vu le jour après la Deuxième Guerre mondiale pour répondre de façon optimale à la reconstruction de l'Angleterre. C'est cette notion d'optimisation qui est reprise par les industries dans les années 60 pour gérer au mieux leur production, leur stock, et leur distribution. L'aide informatique et la programmation linéaire permettent de résoudre des problèmes complexes de recherche opérationnelle. L'heuristique et la métaheuristique sont des domaines fortement utilisés en logistique. Ces algorithmes ont vu le jour pour résoudre, au mieux, des problèmes encore plus complexes ayant un nombre de solutions déraisonnable pour tous les énumérer dans des temps de calcul acceptables.

Le cross-docking, transbordement en français, est une technique en logistique pour répondre efficacement et rapidement à un besoin matériel. Dans un entrepôt classique, on réceptionne les marchandises et on les stocke. Lorsque le client a besoin d'un produit, on va le chercher et on l'expédie dans le premier camion sortant. Dans une plateforme de cross-docking, ou cross dock, les produits sont déchargés, triés selon leurs destinations et réexpédiés immédiatement.

Idéalement, les produits ne sont pas stockés ou alors pendant de courtes périodes, inférieurs à 24 heures. Cette technique permet d'expédier rapidement les marchandises aux clients tout en réduisant les coûts de stockage. Toutefois, son fonctionnement est complexe et nécessite souvent une aide informatique et algorithmique pour produire de réels bénéfices.

L'entreprise Transport Bourassa, notre partenaire industriel pour ce projet, utilise actuellement ce genre de technique pour le fonctionnement de son entrepôt. Ce rapport va présenter le travail réalisé pour optimiser l'utilisation de cet entrepôt de cross-docking à l'aide de la recherche opérationnelle et l'utilisation de métaheuristique.

0.2 Motivation de l'étude

Dans notre étude, nous avons travaillé pour améliorer un cas industriel réel. L'entreprise en question est l'entreprise Transport Bourassa basée à Saint-Jean-sur-Richelieu, au Canada [1]. Cette entreprise est spécialisée dans le transport de marchandises depuis 1956 à l'est du Canada et des États-Unis. Actuellement leader pour les transports LTL au Québec, elle compte environ 500 employés et 200 camions.

Le fonctionnement d'une livraison est le suivant : la journée, les camions font les cueillettes des marchandises chez les clients. Lorsque la remorque est pleine, les camions reviennent à l'entrepôt en soirée. Durant toute la nuit, l'activité du cross-dock est à son maximum. Les camions sont déchargés et rechargés avec les marchandises correspondantes, selon les nouvelles destinations. À l'aube, l'activité du cross-dock est terminée et les chauffeurs peuvent aller livrer les marchandises puis recommencer leur cueillette une fois les livraisons terminées (Figure 0.1). Actuellement, avec les ressources et les infrastructures disponibles, l'entreprise considère que le cross dock est arrivé à sa capacité maximale de marchandises traitées. Sa croissance est donc nulle.

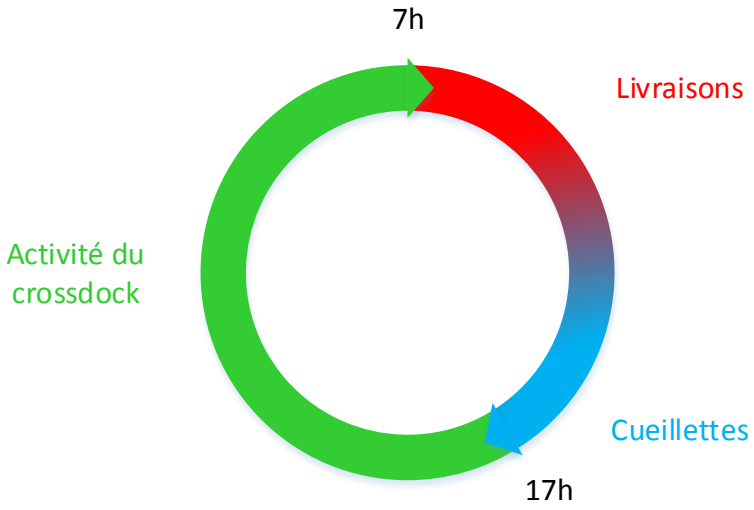


Figure 0.1 Fonctionnement opérationnel de l'entreprise

0.3 Problématique et objectifs de recherche

La problématique qui nous a été posée est de trouver une méthode pour optimiser et améliorer l'affectation des camions aux portes. En ce moment, cette tâche est gérée de façon manuelle à l'aide d'un code couleur sur la disponibilité des marchandises par chargement. La capacité interne du cross-dock étant limitée, il arrive qu'il y ait des engorgements de marchandises ou à l'inverse, que certaines marchandises prioritaires soient en attente pour faute de quais libres. Avec un algorithme d'optimisation, l'entreprise espère avoir un gain de productivité en minimisant les mouvements inutiles, et les doubles manutentions ainsi que de réduire les erreurs humaines.

Au niveau de la recherche, l'entreprise utilise un type de cross-docking peu étudié, ce que nous verrons ultérieurement durant la revue de littérature. La raison principale est la suivante : l'ordre de chargement et de déchargement est à prendre en compte. D'autres paramètres rendent le fonctionnement peu commun comme la préemption autorisée et le mode de service mixte. La seconde raison, comme nous l'avons dit, nous avons réalisé cette étude avec une entreprise. Cela a apporté un réel intérêt à notre étude puisque notre état de l'art démontre que peu d'articles s'appuient directement sur des problèmes et des données réels. Nous avons donc

dû collecter et utiliser directement les données de l'entreprise et eu très peu de flexibilité sur la modélisation et sur la qualité des résultats obtenus.

Notre objectif principal est de produire un algorithme d'affectation des camions aux portes. Ce type de problème est appelé « *Truck-to-Door scheduling* » dans la littérature [2]. Nous reviendrons sur ces termes un peu plus loin. Pour être utile à l'entreprise, l'algorithme doit être le plus fidèle à son fonctionnement actuel, intégrant les limites physiques de la plateforme et prenant en compte les mouvements des marchandises à l'intérieur de ce dernier. Il doit aussi être rapide puisque nous avons un problème de niveau opérationnel, avec des données qui se renouvellent quotidiennement. Cela signifie que notre algorithme devra être capable de trouver une solution dans un temps relativement faible, de l'ordre de quelques heures.

L'apport de ce projet pour l'entreprise est double. À court terme, les bénéfices d'une meilleure gestion des camions aux portes vont engendrer une meilleure efficacité globale du cross-docking et optimiser le temps de travail des manutentionnaires. À long terme, l'entreprise sera plus apte à faire ses choix stratégiques pour augmenter son nombre de clients et son volume de commandes maximum.

0.4 Livrables et contributions de l'étude

Pour répondre aux objectifs précédents, nous allons vous présenter les deux aspects majeurs de notre travail. Premièrement, l'élaboration d'un nouveau modèle mathématique correspondant au plus proche du fonctionnement de l'entreprise. Deuxièmement, l'élaboration d'un algorithme génétique pour résoudre le modèle précédent avec les données de l'entreprise. Le Tableau 0.1ci-dessous présente en détail les livrables du projet ainsi que leurs durées.

Notre contribution pour la recherche est double. Premièrement, nous avons développé un nouveau modèle mathématique s'appuyant sur un cas industriel concret de « *truck-to-door scheduling* ». Deuxièmement, l'utilisation d'un algorithme génétique n'a pas encore été

utilisée pour résoudre ce type de modèle mathématique. Dans ce rapport, nous allons vous présenter les résultats que nous avons obtenus avec cet algorithme.

0.5 Structure et organisation du rapport

Le rapport est articulé autour de quatre grands axes. Le premier est consacré à la définition de la technique de cross-docking et à une revue de littérature sur ce dernier. Le deuxième chapitre traite de la méthodologie mise en place pour répondre à nos objectifs. Le chapitre 3 est spécifique à l'étude du cas industriel. Pour finir, le chapitre 4 clôturera ce rapport, il est dédié à la présentation des résultats obtenus et à une discussion sur les futures recherches.

	Year 2017			Year 2018		
	1 September	2 October	3 November	4 December	5 January	6 February
Theme 1 *(6 months)*	**Delivrable 1 : Model development for truck scheduling in cross docking at Transport Bourassa**					
	Data collection *: Observation and interview with the industrial partner*					
	Constraints *: Identify the specific constraints for Transport Bourassa cross-dock*					
	Model development *: Mathematical model formulation using operational research theory*					
	Exact solutions *for small size problems and model validation*					
Theme 2 *(4 months)*			**Delivrable 2 : Heuristic solution for Transport Bourassa cross-docking terminal optimization**			
			Propose heuristics *for solving large size problems*			
			Data collection *: Select test cases from available data at Transport Bourassa for heuristics validation*			
			Experimental evaluation *and performance validation of the heuristics*			
Theme 3 *(3 months)*				**Delivrable 3 : Optimization platform for Truck-to-Door sequencing problem**		
				Excel Spreadsheet tool programming		
				Connect the optimization tool *with the existing information system*		
				Validate the new tool *with daily operations at Transport Bourassa cross-dock*		

Tableau 0.1 Calendrier des livrables du projet

CHAPITRE 1

REVUE DE LITTÉRATURE

Cette première partie de rapport est consacrée à l'état de l'art du cross-docking au travers d'une revue d'articles scientifiques. Cet exercice va nous permettre de mieux comprendre l'état de la recherche actuelle, d'assimiler tous les différents aspects liés à la technique du cross-docking et de voir les différentes données disponibles pour ce genre de problème. La synthèse de cette revue mettra en avant les lacunes de la recherche actuelle pour résoudre des cas industriels ainsi que les opportunités de recherche disponibles.

1.1 Présentation du cross-docking

Avant de commencer la revue de littérature, il nous a semblé approprié de définir le fonctionnement d'un entrepôt en cross-docking et d'expliquer les différentes caractéristiques qu'un cross-docking peut avoir. Aussi pour alléger la lecture, les termes suivants désigneront la même chose : camion et remorque comme le moyen de transport des marchandises à l'extérieur de l'entrepôt ; entrepôt, cross dock, plateforme de transbordement, terminal comme le lieu physique de transbordement des marchandises ; marchandises, produits, palettes comme étant l'entité à transporter. Pour la même raison, les camions entrants correspondent à la réception des marchandises, arrivant de l'expéditeur vers l'entrepôt et les camions sortants correspondent à l'expédition des marchandises, de l'entrepôt vers le destinataire.

Le fonctionnement d'une plateforme de transbordement se découpe en deux parties, l'extérieur et l'intérieur de l'entrepôt. Tout d'abord, d'un point de vue extérieur, un entrepôt de cross-docking peut être considéré comme un entrepôt classique, un nœud situé dans la chaîne logistique entre un fournisseur et un client. Les marchandises ne font que transiter par cet entrepôt. Pour ce qui est des avantages, un entrepôt de cross-docking apporte relativement les mêmes avantages qu'un entrepôt classique : premièrement, faire des économies d'échelle sur le transport en remplissant les remorques entièrement. On parle de FTL, « *full truck load* ». Deuxièmement, le service client est amélioré puisque le stock situé dans l'entrepôt est

généralement plus proche du client que la distance séparant le fournisseur du client, réduisant ainsi les délais de livraison.

On peut avoir différents types d'utilisation d'un entrepôt. La gestion la plus classique est d'approvisionner l'entrepôt par des FTL venant des différents fournisseurs et d'expédier les marchandises de l'entrepôt vers les clients lorsque la remorque est pleine. L'idée derrière cette gestion est de réduire au maximum les coûts de transport en remplissant au maximum les cargaisons. Toutefois, la capacité de stockage de l'entrepôt doit être très importante. D'autres modes de gestion ont vu le jour pour essayer de réduire la quantité de stock tout en minimisant les coûts de transports. On parle de consolidation, de séparation ou « *break-bulk* » ou encore d'entrepôt mixte [3]. Le premier, la consolidation, correspond à la réception de marchandises provenant de plusieurs fournisseurs dans des transports LTL. Une fois déchargées dans l'entrepôt, ces marchandises sont chargées de façon à ne remplir plus qu'une seule remorque à destination du client. Autrement dit, on a consolidé les marchandises pour n'avoir plus qu'un seul voyage FTL vers le client (Figure 1.1).

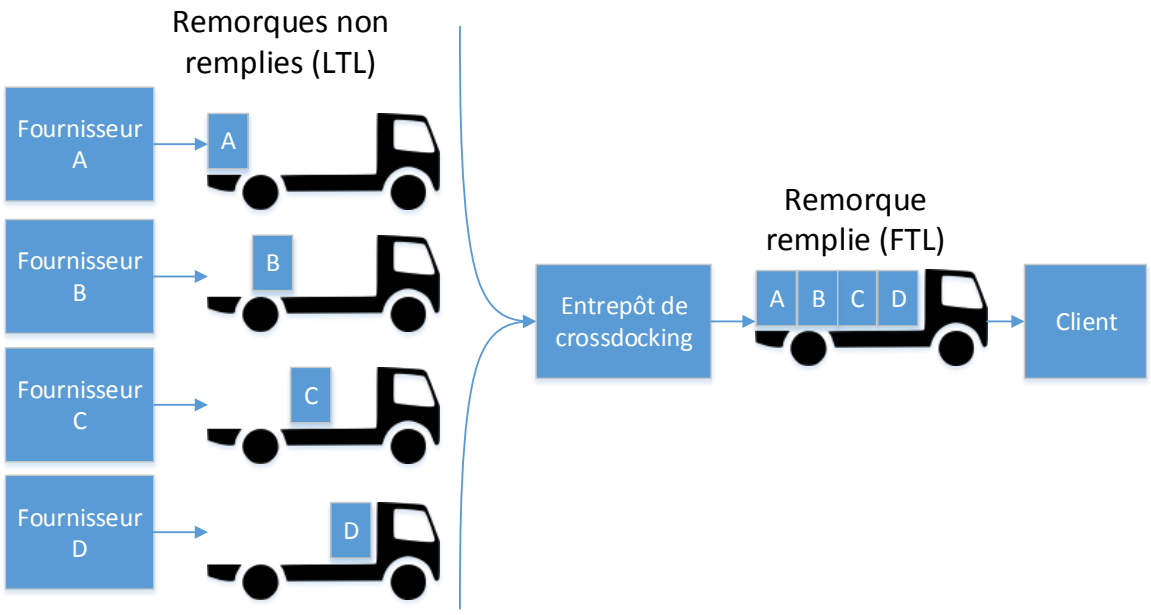


Figure 1.1 Utilisation d'un cross-docking en consolidation [3]

Pour que cette pratique soit économiquement intéressante, il faut que l'entrepôt soit placé proche des fournisseurs pour réduire le plus possible les coûts de transport LTL très onéreux. La séparation est l'inverse de la consolidation [3]. La chaîne logistique possède un seul fournisseur, mais de nombreux clients. On a donc des transports FTL entre le fournisseur et l'entrepôt. Les marchandises sont ensuite réparties dans différents transporteurs LTL jusqu'à leurs clients respectifs. À l'inverse de la consolidation [3], l'entrepôt doit être placé le plus proche des clients, toujours pour minimiser les transports LTL (Figure 1.2). Le dernier mode d'utilisation d'un entrepôt serait de mélanger les deux modes précédents. Ainsi, dans le cas plus réaliste où il y a plusieurs fournisseurs et plusieurs clients, l'entrepôt reçoit des remorques pleines de marchandises en provenance des fournisseurs. Il y a un tri et une nouvelle allocation des marchandises dans les remorques de façon à pouvoir réexpédier les marchandises vers les clients dans des transports FTL (Figure 1.3).

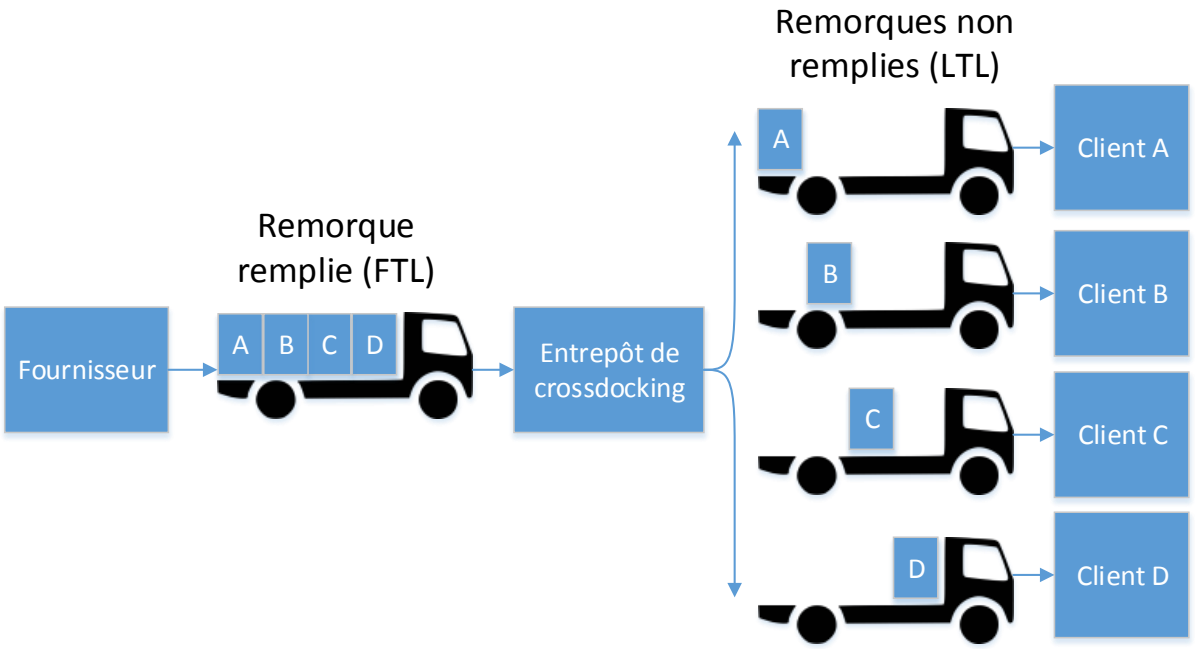


Figure 1.2 Utilisation d'un cross-docking en séparation [3]

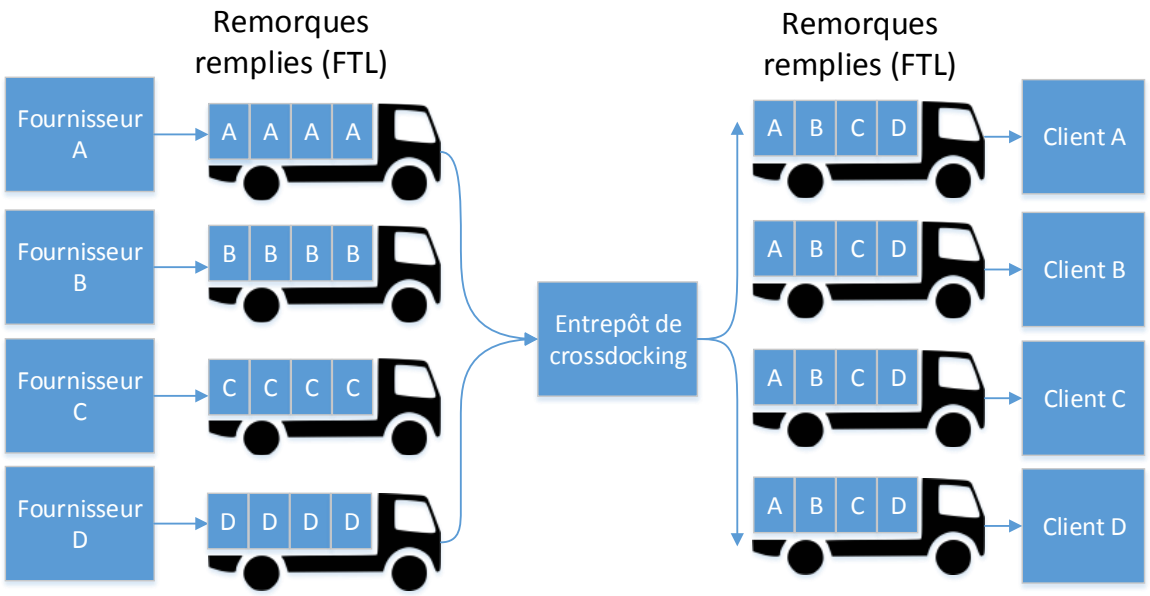


Figure 1.3 Utilisation mixte d'un cross-docking [3]

Un entrepôt géré en cross-docking reprend les concepts d'entrepôt mixte. On rajoute à cela que la phase de tri doit se faire de façon très rapide. Les marchandises doivent repartir du cross dock en moins de 24 heures. En réduisant au maximum le stockage des marchandises, on réduit drastiquement les coûts de manutention et de stockage. Des économies ont lieu aussi sur les infrastructures, moins d'espaces sont nécessaires pour le stockage par rapport à un entrepôt classique. Le Tableau 1.1 ci-dessous représente la répartition des coûts selon le mode de gestion de son entrepôt [4] : la distribution directe entre fournisseurs et clients, la gestion classique d'un entrepôt vu précédemment et le cross-docking.

Tableau 1.1 Répartition des coûts selon le mode de livraison [4]

Type de distribution	Distribution des coûts		
	Stockage	Manutention	Transport
Entrepôt classique	Fort	Fort	Faible
Cross-docking	Faible	Moyen	Moyen
Livraison directe	Faible	Moyen	Fort

L'entreprise Transport Bourassa offre des services de livraison, c'est-à-dire que ces camions doivent réaliser la cueillette et la livraison des marchandises. Dès 7 h du matin, les remorques pleines de marchandises partent pour livrer les clients. Une fois la remorque vide, le chauffeur peut recommencer les cueillettes de marchandises avant de revenir au cross dock en soirée (Figure 0.1 et Figure 1.4). Cette partie du cross dock ne sera pas étudié dans ce travail. Une équipe de 40 personnes, appelées planificateurs, se relaient jour et nuit pour s'occuper d'optimiser l'itinéraire de la flotte de camions en fonction des lieux de cueillettes et de livraison.

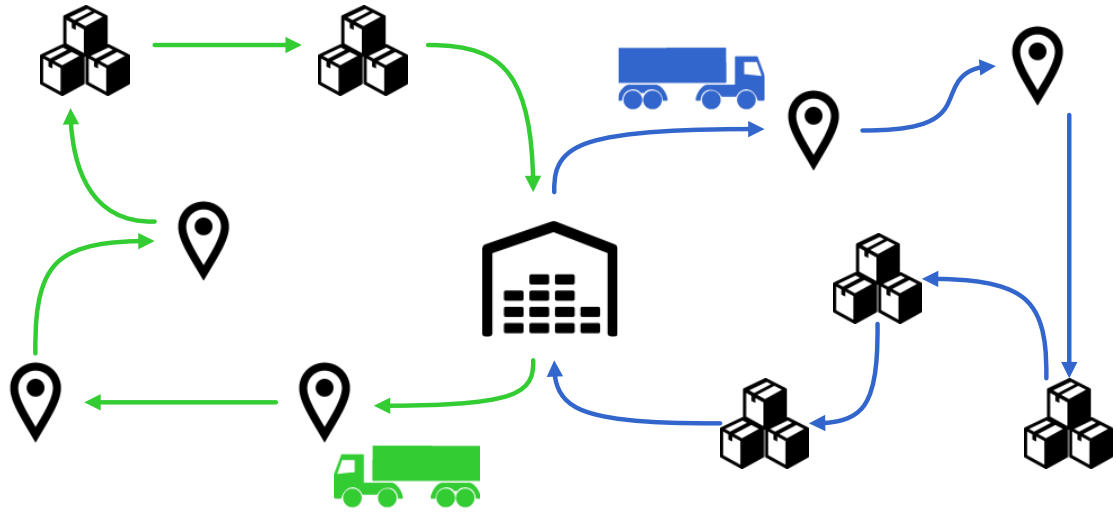


Figure 1.4 Fonctionnement extérieur du cross dock de l'entreprise

Du point de vue de son fonctionnement interne, un entrepôt de cross-docking doit faire face à de nombreux problèmes : comment gérer les quais (destination fixe, réception et expédition séparée), comment affecter les camions aux quais, comment décharge-t-on les marchandises des camions, quelles sont les ressources humaines et matérielles nécessaires, etc. Chaque réponse à ses questions aura un impact direct sur la performance de l'entrepôt et, par conséquent, sur la qualité du service offert au client [5]. Le travail que nous avons réalisé se concentre principalement sur l'affectation des camions aux quais de chargements et de déchargements.

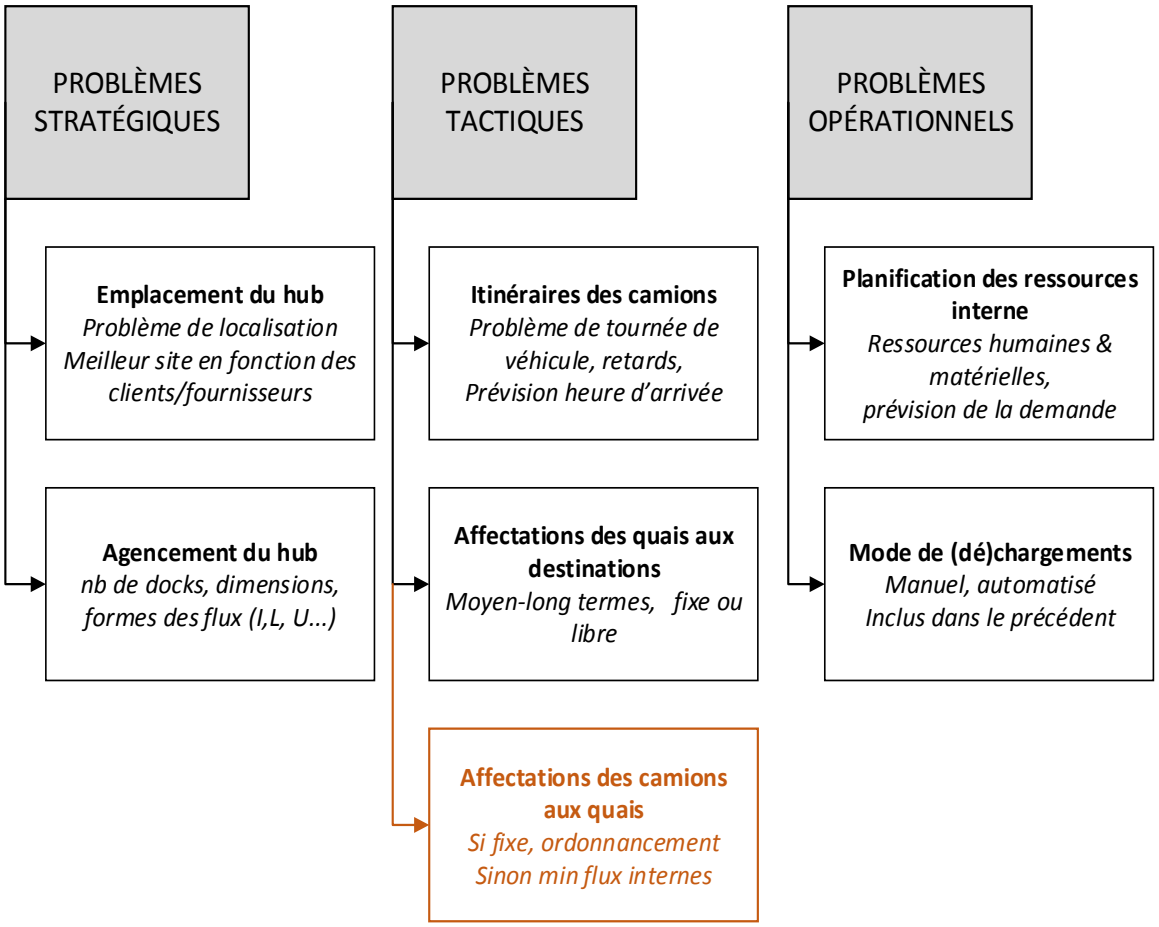


Figure 1.5 Axes de recherche liés au cross-docking

1.2 Caractéristiques d'un entrepôt de cross-docking

Même si l'idée de cross-docking est assez simple, il n'existe pas d'entrepôt ni de fonctionnement optimal et universel capable d'exploiter au maximum les bénéfices de cette gestion. Chaque entreprise a ses propres infrastructures, des marchandises et des clients bien spécifiques, des moyens financiers différents et surtout des fonctionnements différents de la concurrence. Ces spécificités se retrouvent très bien dans la diversité des articles scientifiques traitant du cross-docking.

Pour faciliter la compréhension de la revue de littérature de ce rapport, nous avons décidé de reprendre les termes utilisés dans les papiers de Boysen et Fliedner [6], de Van Belle et coll.

[7] et de Ladier et Alpan [2]. Ces trois articles ont le point commun de rassembler un grand nombre d'articles traitant du cross-docking et d'essayer d'établir une classification selon différentes caractéristiques du cas étudié. Nous allons vous présenter en détail ceux communs aux trois publications et ceux qui nous ont été utiles pour la suite de ce travail.

Tableau 1.2 Caractéristique d'un cross-docking

Niveau stratégique	**Niveau tactique**	**Niveau opérationnel**
• Forme • Nombre de portes • Transport interne	• Mode de service • Préemption • Capacité du stockage temporaire • Capacité des ressources internes	• Heure d'arrivée • Heure de départ • Interchangeabilité des produits

Niveau stratégique :

- Forme : On parle ici de la forme physique de l'entrepôt. Il est d'usage d'utiliser une lettre pour la décrire (I, L, U, T, X, H…). L'emplacement des portes est également important, qu'elles soient sur tout le périmètre, seulement sur deux faces ou uniquement sur un bord, etc.
- Nombre de portes (ou quai) : Une porte correspond à l'emplacement pour le chargement ou le déchargement d'un camion dans l'entrepôt. C'est un critère déterminant sur la taille de la plateforme de cross-docking. On peut préciser le nombre de portes d'entrée ou de sortie selon le mode de service (voir mode de service).
- Transport interne : Définis la façon dont les produits sont déplacés à l'intérieur de l'entrepôt. Cela peut se faire être manuellement (transpalettes, chariots élévateurs…), automatiquement (tous les types de convoyeurs, les AGV, *« Automated Guided Vehicle »* …) ou une combinaison des deux.

Niveau tactique :

- Mode de service : Le mode de service est la façon dont est utilisé un quai lors de l'affectation avec un camion. Il existe 4 modes de service : exclusif, mixte, combinaison et géographique. Une porte en service exclusif sera dédiée uniquement à la réception (camions entrants) ou uniquement aux expéditions (camions sortants). À l'opposé, un mode de service mixte signifie qu'une porte peut être utilisée pour la réception et pour l'expédition, selon le besoin présent. Avec ce mode de service, on peut décharger et charger une remorque sans la changer de place. Il arrive que certains entrepôts aient un mode de service exclusif pour quelques quais et mixte pour les autres. On parle alors du mode de service combinaison. Enfin, le dernier mode de service, géographique, correspond à l'attribution d'une zone géographique fixe à une porte. Cela peut être une ville, une région, un état. On parle de destination exclusive si c'est pour les expéditions et d'origine exclusive si c'est pour les réceptions. Seuls les camions allant ou venant de cette zone peuvent être affectés à cette porte.
- Préemption : La préemption est l'interruption du chargement d'un camion au profit d'un autre. En d'autres termes, si une remorque en attente est plus importante qu'une autre déjà à quai, on peut changer de remorque même si le chargement n'est pas terminé. La remorque mise de côté sera réaffectée à une porte pour terminer son chargement ultérieurement.
- Capacité du stockage temporaire : Lorsqu'un produit est déchargé et que son camion d'expédition n'est pas encore affecté à une porte, il est mis en attente dans la zone de stockage. Cette zone de stockage correspond à une surface définie physiquement dans l'entrepôt. Si la limite est contraignante, on parle de capacité limitée. Il sera intéressant de chiffrer cette limite. Si la limite est très supérieure aux quantités de produits stockés, l'entrepôt n'est jamais plein, on la considère alors comme illimitée. Enfin, certaines marchandises ne doivent pas être stockées, comme les produits congelés, on parle alors de capacité de stockage nulle.
- Capacité des ressources internes : Selon le type de transport à l'intérieur du cross dock, cela peut correspondre à la cadence maximale d'un tapis roulant ou bien au nombre maximal d'employés présents dans l'entrepôt. Tout comme le stockage, elle peut être limitée ou illimitée en fonction de la criticité de ce point pour l'entrepôt.

Niveau opérationnel :

- Heure d'arrivée : Comme son nom l'indique, cela concerne l'heure d'arrivée des camions au cross dock. On a deux cas de figure : soit les camions sont tous disponibles à l'instant zéro et sont prêts à être affectés aux portes à n'importe quel moment ; soit ils arrivent à des horaires différents (selon leurs tournées de livraison, selon le trafic routier…) et ne peuvent être physiquement affectés à un quai qu'après cet horaire. À noter que cela s'applique pour les camions entrants, pour les sortants ou pour les deux.
- Heure de départ : On parle ici d'horaire à laquelle un camion doit partir, quitter le cross dock. La raison principale est que la marchandise est attendue par le destinataire à un horaire prévu. Selon le cross dock étudié, on a plusieurs politiques sur les départs. On peut autoriser un camion à ne partir que si sa cargaison est complète même si cela engendre du retard ou, à l'inverse, le camion doit partir à l'heure prévue même si son chargement n'est pas complet. Encore une fois, ce paramètre s'applique aux camions entrants et sortants indépendamment.
- Interchangeabilité des produits : On parle d'interchangeabilité lorsque le lien entre le produit et le camion peut être modifié. C'est-à-dire que tant que les produits arrivent à destination, le camion qui les a transportés n'a pas d'importance. On a trois types d'interchangeabilité : post-distribution, destination ou prédistribution. Pour le premier, on n'a aucune information sur la cargaison des camions entrants. Les seules choses que l'on connaît sont les différents produits à charger pour chacun des camions sortants. Le deuxième mode est assez similaire, à la différence que l'on connaît la quantité de produits à expédier par destination. On ne parle plus de camions en eux-mêmes, mais de destination. Enfin dans le dernier mode, celui de prédistribution, on connaît à l'avance la cargaison des camions entrants et le besoin des camions sortants. À l'inverse des deux premiers modes, on ne peut pas interchanger des marchandises puisque tout est déjà connu et déjà optimisé.

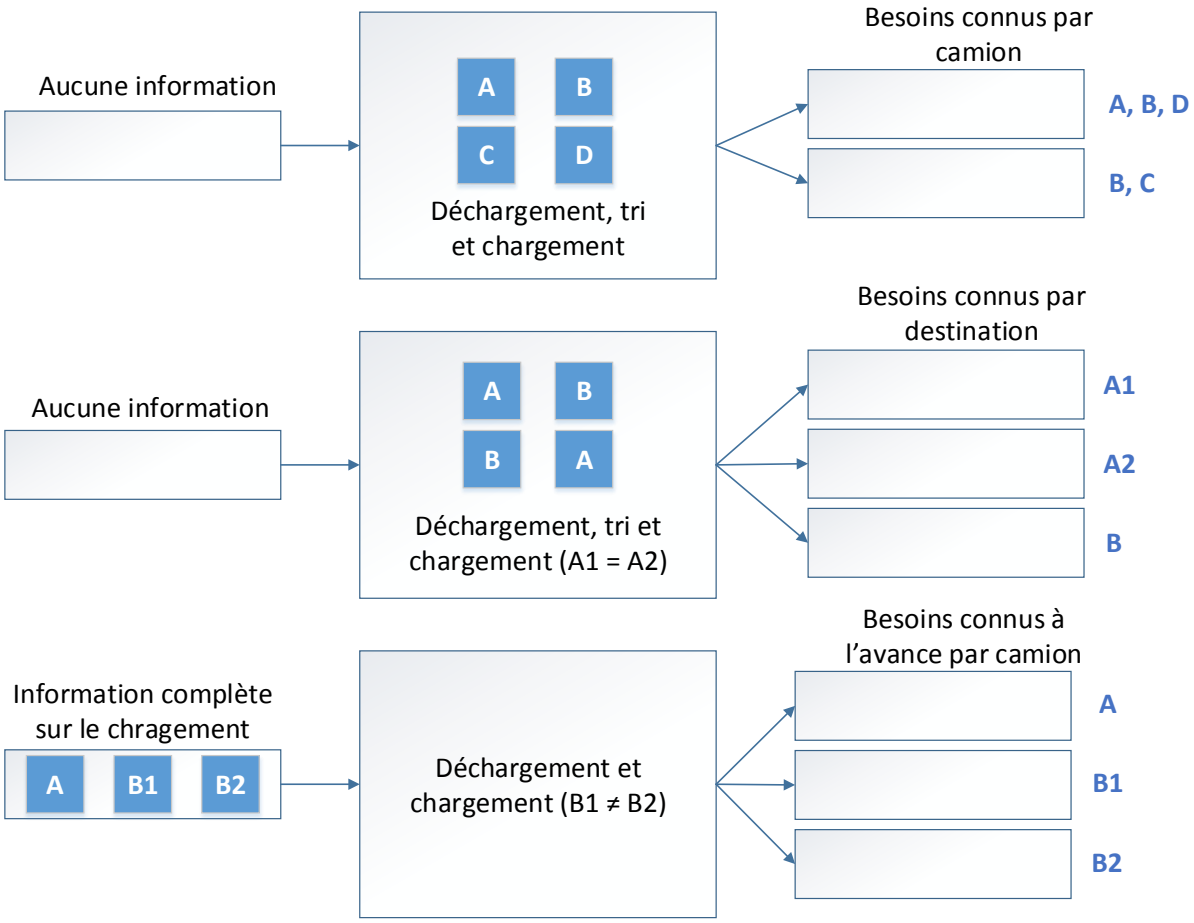


Figure 1.6 Interchangeabilité des produits : post-distribution, destination et prédestination

Les deux derniers points présentés ci-dessous sont aussi présents dans la classification de Ladier et Alpan [2]. Il s'agit de la mesure de la performance du modèle ou autrement dit, la fonction du modèle à optimiser, et la méthode, l'algorithme mis en place pour résoudre le modèle. La mesure de performance est très souvent le « *makespan* » ou bien le temps mis pour réaliser toutes les opérations du cross dock. Cela correspond à la plage horaire entre l'arrivée du premier camion entrant jusqu'au départ du dernier camion sortant. Cette mesure de performance est largement utilisée par la communauté scientifique, mais très peu par les industriels qui préfèrent unanimement calculer les nombres d'heures de travail, le premier coût d'un entrepôt de cross-docking. Il peut aussi y avoir des combinaisons de plusieurs indicateurs plus ou moins contradictoires. Dans le cas de modèles multiobjectifs, les résultats obtenus ne permettent pas de trouver la meilleure solution, mais plutôt un ensemble de solutions

optimales, appelé front de Pareto. Une personne devra privilégier un critère plutôt qu'un autre selon ses besoins. Nous avons listé ci-dessous d'autres indicateurs de performances présents dans différents articles :

- Quantité maximum de produits stockés dans l'entrepôt
- Total des heures de travail des opérateurs dans le cross dock
- Distance parcourue par les produits à l'intérieur de l'entrepôt
- Temps/Distance de congestion
- Temps total des palettes à l'intérieur du cross dock
- Temps total de chargement et de déchargement
- Temps total d'occupation des portes
- Total des écarts entre la prévision et le temps mis pour réaliser le transbordement
- Nombre de produits non chargés à temps, mal triés, perdus
- Coût de préemption
- Nombre de manutention pour le même produit

Enfin, le dernier critère est l'algorithme mis en place pour résoudre le modèle. Il permet d'avoir une vision plus globale sur l'efficacité d'un algorithme ainsi que sa popularité au sein de la communauté scientifique. Pour ce qui est des problèmes d'affectation aux portes dans un cross-docking, on peut noter trois familles d'algorithmes de résolution : les méthodes exactes, les heuristiques et les métaheuristiques. La première famille, la plus ancienne, propose d'énumérer l'ensemble des solutions du modèle et de trouver la meilleure d'entre elles. Les avantages de ces méthodes sont : la solution optimale globale et la facilité de l'algorithme. Le gros désavantage de ces méthodes est que parcourir toutes les solutions peut devenir très long voir infaisable pour des problèmes de très grandes tailles. Pour répondre à ce problème, les heuristiques et les métaheuristiques proposent de ne calculer qu'un certain nombre de solutions pour ensuite proposer une solution au mieux optimale ou bien qui s'en rapproche fortement, mais en un temps de calcul bien plus faible. La deuxième méthode, les heuristiques sont des algorithmes développés spécialement pour un type de problème. Ils permettent d'aller plus vite que l'énumération totale puisqu'ils reposent sur un raisonnement humain qui évite de calculer toutes les solutions. Toutefois, ce type d'algorithme ne garantit pas de trouver la solution

optimale globale du problème. Enfin la dernière méthode, les métaheuristiques, propose des algorithmes génériques permettant de converger vers une solution performante. Ils peuvent être appliqués à n'importe quel type de problème puisqu'ils ne sont pas spécifiques développés pour un fonctionnement précis.

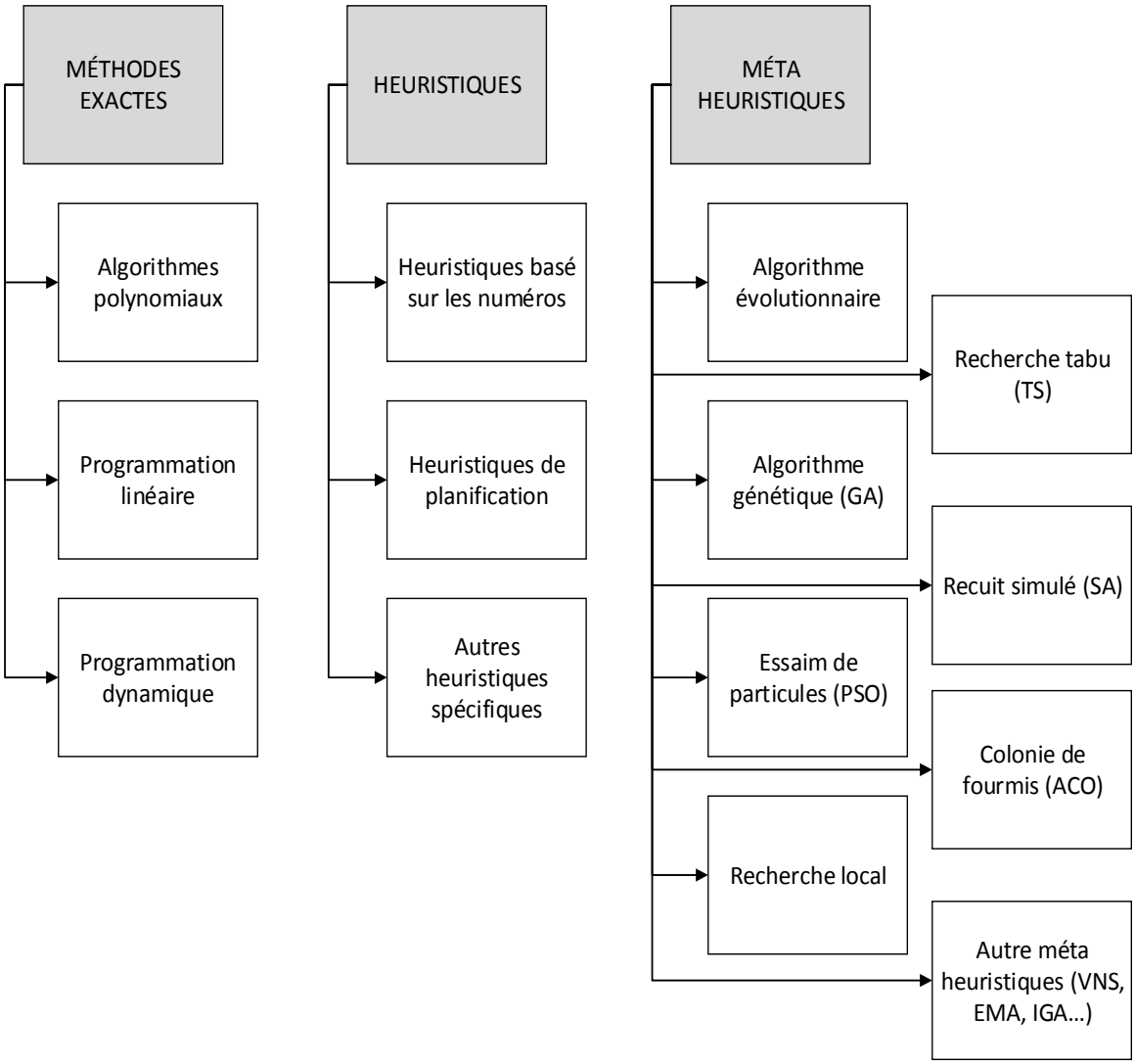


Figure 1.7 Classification des algorithmes pour des problèmes de cross-docking

1.3 Historique des recherches

La gestion de plateforme en cross-docking n'est pas une nouveauté en logistique. La preuve avec l'article de Tsui et Chang [8] datant de 1990. Les auteurs ont mis en place un programme bilinéaire capable de faire l'assignation des portes aux camions entrants et sortants de façon à minimiser la distance parcourue par les palettes à l'intérieur de la plateforme. Le modèle mathématique est simple, mais permet de prendre en compte la distance entre les quais et la quantité de palettes transférées d'un camion à un autre. Les auteurs ont utilisé une heuristique en trois étapes pour résoudre leur modèle. La solution approchée obtenue est une aide à la décision et permet au décideur d'avoir un point de vue supplémentaire.

De façon plus formelle, Rohrer en 1995 [9] est parmi les premiers auteurs à avoir publié un article traitant de cross-docking explicitement. Il présente différentes méthodes de modélisation avec leurs avantages et leurs inconvénients. Il explique aussi que la modélisation et les simulations permettent de bâtir des entrepôts optimaux dès le début de leur exploitation et d'anticiper les problèmes. Enfin, il termine avec le point suivant : pour qu'un entrepôt de cross-docking fonctionne de façon optimale, il faut que les parties logicielles et matérielles soient parfaitement interconnectées.

Plusieurs entreprises ont appliqué et amélioré ce type de gestion dans les années 1990 comme Walmart (Stalk et coll., 1992 [10]), UPS (Forger, 1995 [11]), American Home Foods (Schwind, 1996 [12]), Office dépôt (Ross, 1997 [13]), Toyota (Witt, 1998 [14]), Goodyear GB Ltd (Kinnear, 1997 [15]) et Kodak Co. (Katz, 2006 [16]). En comparaison avec les méthodes classiques d'approvisionnement, le cross-docking permet de faire de gros bénéfices et d'être plus compétitif : Walmart arrivait à proposer des produits 2 à 3 % moins dispendieux que la concurrence en 1992 [10]. Dans les années 2000, les articles scientifiques liés au cross-docking se diversifient. De nouveaux axes de recherches émergent pour répondre le plus précisément aux problématiques des entreprises. De nouveaux paramètres apparaissent comme l'augmentation du nombre de camions et de quais, le stockage temporaire, les heures d'arrivées et de départs des camions, etc. En plus de la diversification des types de cross-docking, on a

également remarqué l'apparition de benchmark sur des modèles classiques pour pouvoir comparer la vitesse et la robustesse des différents algorithmes. Nous développerons ce point un peu plus loin.

Un autre travail souvent cité pour les problèmes de cross-docking est l'article de Apte & Viswanathan [3]. Comme nous l'avons développé plus haut, l'article permet de mieux comprendre le fonctionnement des notions de consolidation et de séparation. Ils discutent aussi des techniques à mettre en place pour obtenir la meilleure efficacité globale d'une chaîne de distribution ayant un cross-dock, ainsi que le type de produits avantageux à traiter en cross-docking. Pour finir, la forme de l'entrepôt et le partage d'information au sein de la chaîne de distribution sont aussi discutés.

Gue en 2007 [4], lui aussi souvent cité pour définir le cross-docking, compare dans son article les deux stratégies de gestion d'un entrepôt, classique ou en cross-docking. Il liste aussi sept points importants à considérer avant de changer de stratégie. Pour qu'un entrepôt de cross-docking soit bénéfique, il faut que la demande des produits soit forte et stable ; les clients doivent être capables d'attendre quelques jours leurs produits ; la stratégie de distribution doit être en mode « *push* », pour éviter d'avoir du stock ; avoir de la flexibilité dans les dates de réceptions et même, si possible, confier aux fournisseurs le tri et la répartition des marchandises vers les clients ; une gestion rigoureuse de la manutention (effectif, coûts, pertes, organisation) ; tous les coûts doivent être connus et analysés attentivement ; une collaboration étroite avec les vendeurs pour une réussite partagée. Il termine son article en rappelant que la meilleure solution est toujours d'avoir les deux types de gestions pour plus de flexibilité.

1.4 Articles de revues

Comme nous l'avons dit dans l'introduction, il n'existe pas un seul type de cross-docking universel et optimal. Dans l'article de Boysen & Fliedner [6] de 2010, les auteurs ont regroupé une quarantaine d'articles traitants de cross-docking et ont mis en place une notation pour classer les différents cross docks. Ils font la distinction entre les problèmes liés au cross-

docking au niveau opérationnel et aux autres niveaux. Autrement dit, ils ne prennent pas en compte les problèmes stratégiques ou tactiques comme l'emplacement géographique du cross dock ou les problèmes de tournées de véhicules. La classification se fait selon trois grands critères et 18 sous critères. Les trois catégories sont les suivantes (avec quelques exemples de sous-catégories) : les docks (nombre et mode de services), les caractéristiques internes (heures de départ et d'arrivées des camions, stockage intermédiaire, temps de chargement…) et les paramètres à optimiser (temps total, réduction du retard global ou par camions, stock maximal…). En plus de cette classification, les auteurs proposent un nouveau modèle mathématique avec des horaires fixes pour les camions sortants du cross dock.

Van Belle et coll. [7] ont aussi mis en place une méthodologie pour pouvoir classer et mieux comparer les différents cross docks. Leur approche repose aussi sur trois niveaux, caractéristiques physiques, opérationnelles et du flux interne. Les sous-critères sont assez semblables à ceux proposer par Boysen & Fliedner [6] avec quelques ajouts et une meilleure répartition dans les catégories. Ils ont appliqué leur classement aux différents articles cités par Boysen & Fliedner [6] ainsi qu'à une vingtaine d'articles et de conférences supplémentaires et plus récents. Ils introduisent aussi le fait que la gestion d'un cross dock doit se faire à une échelle plus haute qui prendrait en compte l'ensemble de la chaîne logistique en temps réel, ce qu'ils appellent le cross-docking « *on-line* ».

La revue de littérature la plus récente et la plus intéressante est celle publiée par Ladier et Alpan en 2016 [2]. En plus de reprendre les critères de classification des deux articles précédents, les auteurs proposent une nouvelle distinction entre les cross-docks selon leurs fonctionnements. Si la plateforme considère tous les quais comme étant identiques, on parle de « *truck problem* », s'ils sont différents (prise en compte de la distance entre les quais, spécificités des quais…), on parle alors de « *truck-to-door problem* ». Le deuxième point est l'aspect temporel. Si le cross-dock ne prend pas en compte cet aspect, on parle d'affectation. Si l'on considère seulement l'ordre, on parle de séquencement. Finalement, si l'on intègre totalement une variable temporaire, on a alors un problème d'ordonnancement (Tableau 1.3). Ils classifient ainsi 112 articles sur le cross-docking. Enfin, ils font la comparaison des travaux

scientifiques avec huit plateformes réelles qu'ils ont observés. Cela leur permet de terminer leur article sur les écarts qu'il y a entre les problèmes résolus par la communauté scientifique et les problèmes concrets des entreprises.

Tableau 1.3 Classification des problèmes d'affectation

	À quelle porte ?	À quel moment ?	Dans quel ordre ?
Truck-to-door assignment	✓	N/A	N/A
Truck-to-door sequencing	✓	N/A	✓
Truck-to-door scheduling	✓	✓	N/A
Truck sequencing	N/A	N/A	✓
Truck scheduling	N/A	✓	N/A

1.5 Analyse comparative

Pour revenir sur les différentes familles d'algorithmes énoncées au début de cette revue de littérature, nous avons dit que généralement les méthodes exactes sont plus lentes que les heuristiques et les métaheuristiques. Pour affirmer cela, on peut citer plusieurs articles qui compare les performances de l'heuristique ou de la métaheuristique qu'ils aient développé avec une méthode exacte : comme ceux de Álvarez-Pérez & coll. [17] et de Lim & coll. [18] qui comparent à leurs programmes linéaires, une recherche tabou (TS), ou bien Chen & Song [19] et Song & Chen [20] qui comparent une heuristique de planification avec leur programmation linéaire, ou encore Yu & coll. [21] et Fazel Zarandi & coll. [22] qui comparent les programmes linéaires avec un algorithme génétique (GA). Certains articles deviennent même des références pour d'autres auteurs afin de comparer la vitesse du nouvel algorithme développé.

C'est le cas de l'article de Yu & Egbelu [23] qui fait partie des articles les plus cités avec celui de Apte & Viswanathan [3] et celui de Van Bell & coll. [7]. C'est le premier modèle à intégrer une variable de temps et donc le premier problème de « *truck scheduling* ». La modélisation mise au point par les auteurs correspond à l'entrepôt suivant : un quai de réception et un quai

d'expédition ; tous les camions sont disponibles au temps zéro ; pas de stockage à long terme, le total des palettes entrantes est égal au total de palettes sortantes ; les temps de chargement, déchargement, déplacement et de changement de camions sont pris en compte ; le stockage intermédiaire est autorisé et est de capacité illimitée. Le modèle développé par Yu & Egbelu a souvent été repris à l'identique ou alors avec de faibles modifications par plusieurs auteurs parmi lesquels Shiguemoto [24] qui comparent un GA, une recherche tabou (TS) et un algorithme hybride avec les deux précédents ; Vahdani & Zandieh [25] qui comparent cinq métaheuristiques différentes (GA, TS, recuit simulé (SA), algorithme basé sur la théorie d'électromagnétisme (EMA), et une recherche à voisinage variable (VNS)) ; Boloori Arabani & coll. [26] comparent les métaheuristiques suivantes : GA, TS, optimisation par essaim particulaire (PSO), algorithme de colonies de fourmis (ACO) et un algorithme à évolution différentielle (DE) ; et Forouharfard & Zandieh [27] qui comparent un GA avec un algorithme de compétition impérialiste (ICA). Yu & Egbelu proposent aussi de réutiliser les 32 scénarios qu'ils ont aussi publiés dans leur article [23] afin de pouvoir tester les algorithmes sur les mêmes instances avec différentes complexités.

1.6 Classification d'articles récents

Dans l'article de 2016 de Cota & al [28], l'approche du cross-docking est très différente. Ils ont modélisé les portes d'entrées et de sorties comme un problème de planification à deux niveaux. Leur modèle est donc très différent de ceux en rapport avec les cross docks. Les caractéristiques du cross dock étudié sont toutefois ordinaires, mode de service exclusif, temps de déchargement fixe. Le modèle prend en compte plusieurs portes, mais pas les distances entre elles. Ils ont aussi développé trois heuristiques différentes et 30 scénarios différents pour réaliser leurs tests.

Kuo en 2013 [29] propose une nouvelle modélisation intégrant tous les camions en attente d'affectation à une porte du cross dock. Son modèle assigne les camions sur le principe du premier arrivé, premier servit. Ce qui revient à se focaliser sur l'affectation des portes et non

l'ordre des camions. Il a réalisé un algorithme reposant sur la métaheuristique VNS et l'a comparé à 4 autres métaheuristiques basées sur le fonctionnement du SA.

Keshtzari & coll, ont publié en 2016 [30] un article présentant leur nouveau modèle ainsi que leur algorithme. Leur modèle est une amélioration du modèle présenté par Yu & Egbelu 2008 [23]. Le nouveau modèle permet de mieux gérer les heures d'arrivée et de départs des camions, mais n'a qu'une seule porte d'entrée et une de sortie. L'algorithme proposé est un mélange entre deux métaheuristiques qui sont le PSO et le SA. Ils comparent leur nouvel algorithme avec un algorithme génétique et un EMA.

Nous avons rassemblé dans le Tableau 1.4 ci-dessous, 10 articles récents (ultérieurs à 2016) que nous avons aussi étudiés pour le projet. Ils n'apportent pas directement d'informations utiles pour notre projet. Les auteurs sont les suivants : Assadi & coll. [31], Cota & coll [28], Golshahi-Roudbaneh & coll. [32], Keshtzari & coll. [30], Ladier et Alpan [33], Maknoon & coll. [34], [35], Mohtashami [36], Rahmanzadeh Tootkaleh & coll. [37] et Wisittipanich et Hengmeechai [38]. Nous les avons classifiées dans un tableau comme ont fait Ladier et Alpan [2] dans leur la revue. Nous avons aussi rajouté les différents algorithmes utilisés dans ces articles.

Tableau 1.4 Classification d'articles de cross-docking

			Strategic				Tactical		Operational			Performances measures	Algorithms		
	Problem type	On witch doors	Shape	Nb inb. doors	Nb outb. Doors	Internal transp.	Service mode	Pre-emption	Arrival time	Departure time	Inter-chang.		Heur.	Meta-heur.	Hybrid
Assadi et al. 2016	TtD sequ.	Both	*	*	*	ns	excl.	No	/truck	No	Dest	Truck process. time dev. + Makespan		SA, EA	
Cota & al. 2016	T sched.	Both	*	*	*	ns	excl.	No	Zero	No	Pré-D	Makespan	1		
Golshashi-Roudbaneh & al. 2017	T sequ.	Both	n/a	1	1	Man.	excl.	No	Zero	No	Post-D	Makespan	2	CR, GA, SA, KA, SFS	PSO+SA
Keshtzari & al. 2016	T sequ.	Both	n/a	1	1	ns	excl.	No	Zero	No	Pré-D	Makespan			PSO+SA
Ladier & Alpan. 2016b	T sched.	Both	*	*	*	ns	excl.	No	Window	No	Pré-D	Inventory level + Truck process. time dev. + Door utilisation + Makespan	9		
Maknoon et al. 2016	TtD Sched.	Both	*	*	*	Man.	mix. ?	No	Zero	No	Pré-D	Nb of touch		VNS	
Maknoon et al. 2017	TtD Sched.	Both	n/a	1	1	Man.	excl.	No	Zero	No	Pré-D	Nb of touch		B&B	
Mohtashami. 2015	T sequ.	Both	n/a	1	1	ns	excl.	OutB.	Zero	No ?	Dest	Makespan		GA	
Tootkaleh & al. 2016	TtD sched.	InB.	*	*	*	ns	excl.	No	Zero	No ?	Post-D	Inventory level	1		
Wisittipanich & al. 2017	TtD Sched.	Both	I	*	*	ns	excl.	No	Zero	No	Dest	Makespan		PSO, mPSO	

1.7 Modèles en lien direct avec l'entreprise

L'article de Shakeri & coll. de 2012 [39] nous présente un modèle mathématique très semblable au fonctionnement de l'entreprise. Le modèle comprend les caractéristiques suivantes : un mode de service mixte, un nombre de portes quelconque, la prise en compte des distances séparant les portes, du temps pour charger, décharger, changer de camions, les palettes sont transportées une par une par un manutentionnaire, un seul manutentionnaire par chargement ou déchargement, pas de préemption, le chargement peut débuter dès que le déchargement est terminé, les palettes ne sont pas sécables, le stockage temporaire est illimité et surtout, la prise en compte de l'ordre de chargement et de déchargement des palettes dans les camions. En deuxième partie, ils présentent une heuristique pour résoudre le problème sur de grands problèmes. Cette heuristique est en deux partie, la première permet de trouver une séquence pour l'ordre d'affectation des camions afin d'éviter les combinaisons infaisables du fait que le nombre de camions est supérieur au nombre de portes et que la préemption n'est pas autorisée. La deuxième partie de cette heuristique réutilise la séquence trouvée par le premier algorithme et se concentre à trouver la meilleure combinaison entre les camions et les portes afin de minimiser les distances parcourues par les palettes dans l'entrepôt.

Assadi & coll. [31] proposent un modèle mathématique qui minimise les écarts entre l'horaire réel de départ d'un camion avec celle fixée à l'avance. Pour cela, ils prennent en compte l'heure de départ des camions sortants effective (le camion sort dès que sa cargaison est chargée) et l'heure de départ prévue en plus des critères suivants : un nombre de portes quelconque, un temps de transfert diffèrent selon la distance entre les portes, différents produits et un mode de service exclusif. Pour la résolution de leur modèle, ils utilisent Cplex pour des problèmes de petite taille et ont développé deux métaheuristiques, un DE et un SA basé sur une population, pour les problèmes de plus grandes tailles.

Le modèle de Wisittipanich & Hengmeechai 2017 [38] est assez ordinaire : minimisation du temps total, pas de préemption, temps de chargement fixe, capacité du stockage intermédiaire illimitée. Ils utilisent ensuite une métaheuristique PSO. Ce qui est intéressant ici, c'est la façon

dont ils ont construit leur solution. Étant donné qu'ils ont un problème de « *truck-to-door scheduling* », leur solution doit contenir l'ordre de traitement des camions ainsi que l'affectation des camions aux portes. Nous nous resservirons de cet article pour la construction de notre chromosome. Ils comparent ensuite les algorithmes selon les résultats obtenus avec le logiciel LINGO pour des petits problèmes.

1.8 Conclusion

Pour conclure cette partie, nous avons mis en avant les origines et le fonctionnement d'un cross-docking. Nous avons aussi montré que la recherche est poussée par les forts besoins des entreprises pour toujours avoir des entrepôts toujours plus performants. Ensuite, nous vous avons présenté les différents types de cross-docking, leurs spécificités et l'ensemble des méthodes misent en place pour les résoudre.

Pour clôturer cette revue de littérature, nous pouvons reprendre quelques chiffres de l'article de Ladier et Alpan [2] sur leur comparaison entre les articles publiés et les entrevues faites avec des entreprises en activités : 63 % des entrepôts qu'ils ont visités utilisent un mode de service mixte contre 14 % dans la littérature, le stockage temporaire est limitant dans 88 % des entrepôts alors que seulement 9 % des articles prennent en compte cette caractéristique. La préemption n'est pas une priorité dans la recherche puisque seulement 9 % des articles l'utilisent. Enfin, l'ordre de chargement et de déchargement n'est même pas pris en compte dans la classification. Vous noterez que ces derniers points sont des caractéristiques observées dans l'entreprise partenaire que nous allons développer dans la partie suivante.

CHAPITRE 2

MÉTHODOLOGIE

Grâce à la revue de littérature, nous avons une idée précise de ce qu'est un entrepôt géré en cross-docking. Nous avons pris connaissance des différentes caractéristiques qu'il peut y avoir d'un modèle mathématique à un autre. Malgré les nombreux modèles que nous avons lus, aucun ne ressemble exactement aux caractéristiques de l'entreprise partenaire. Ce chapitre est consacré à l'explication du cheminement que nous avons suivi pour aboutir à un nouveau modèle. Nous parlerons aussi des différents aspects que nous avons retenus de nos observations du fonctionnement de l'entreprise, des hypothèses et des modifications que nous avons dû faire pour rendre stable et solvable le modèle. Pour conclure cette section, nous vous présenterons et expliquerons le modèle mis au point ainsi que la validation de celui-ci avec la programmation linéaire.

2.1 L'entreprise partenaire, Transport Bourassa

2.1.1 Présentation de l'entreprise

L'entreprise avec laquelle nous avons fait cette étude se nomme Transport Bourassa. Nous allons vous présenter brièvement l'entreprise pour mieux comprendre l'étendue de ce projet. En chiffre, l'entreprise représente 450 employés, dont 150 chauffeurs et environ 190 camions et 550 remorques. Le terminal se situe au 800 rue de Dijon à Saint-Jean-sur-Richelieu, au Québec, Canada, à une quarantaine de kilomètres au sud-est de Montréal. L'entreprise dispose d'une surface d'environ douze hectares, incluant un entrepôt de 23 000 pieds carrés et un garage de 13 000 pieds carrés. La compagnie est classée 306^{e} parmi les 500 sociétés les plus importantes du Québec en 2017 [40] et est le leader québécois en transport *« less than truckload* ».

Les services offerts par l'entreprise sont le transport de marchandises au Québec, en Ontario et au nord-est des États-Unis, ainsi que l'entreposage [1]. Transport Bourassa offre ses services

pour tous types de marchandises, que ce soit une remorque entière de marchandise, un simple colis ou même des matières dangereuses. Pour répondre le plus efficacement aux besoins du client, Transport Bourassa dispose de nombreuses et diverses remorques, à 2 ou 3 essieux, de différentes longueurs, chauffées ou réfrigérées, fermées, à rideau, ouvertes ou encore porte conteneur. Pour les destinations lointaines, l'entreprise fait appel à d'autres compagnies de transporteurs pour répondre au besoin du client. Aussi, pour tous les besoins douaniers, l'entreprise dispose d'un entrepôt routier de douane dans son terminal à Saint-Jean-sur-Richelieu pour accélérer les démarches administratives.

Figure 2.1 Logo de l'entreprise partenaire

Concernant sa filiale d'entreposage, Entreposage Plaspack Inc. Elle dispose d'une surface totale de stockage de 274 000 pieds carrés, dont 42 000 directement relié au terminal [1]. Elle propose les services suivants en plus de l'entreposage : réception et distribution des marchandises, inventaire informatisé, étiquetage, emballage, préparation des commandes et entreposage de matières dangereuses.

Voici quelques moments clés dans l'histoire de l'entreprise Transport Bourassa [1] :

- 1956 : Premier service de livraison par André et Guy Bourassa autour de Granby.
- 1970 : Achat d'un permis de transport pour les villes de Saint-Jean-sur-Richelieu, Montréal, Sherbrooke, Drummondville et Saint-Hyacinthe.
- 1981 : Rachat des parts de l'entreprise par Jean et Daniel Bourassa – Création de l'actuel Bourassa.
- 1982 : Agrandissement du service de livraison aux villes de Trois-Rivières et Québec.
- 1983 : Acquisition d'un terminal de 5000 pieds carrés à Saint-Jean-sur-Richelieu.
- 1985 : Lancement d'un service de transport aux États-Unis et d'un service douanier.
- 1990 : Déménagement de l'entreprise dans le terminus actuel (23 000 pieds carrés)

- 1996 : Achat d'un garage pour la maintenance des camions
- 2008 : Augmentation du nombre de quais à 40 et de 2 accès au train

Pour revenir sur notre problématique, l'entreprise a atteint ses limites en termes d'infrastructure et cherche une solution pour continuer de se développer. Nous allons voir dans les prochaines parties, comment fonctionne l'entreprise et notre cheminement pour aboutir à notre modèle mathématique.

2.1.2 Observations de l'entrepôt

Après avoir fait la revue de littérature, nous avons passé deux nuits en observation chez l'entreprise. Nous avons été avec Stéphane Benoit, au poste de superviseur. Son rôle est d'affecter les remorques aux portes et de distribuer aux manutentionnaires les chargements et les déchargements de marchandise. Le but était de comprendre le fonctionnement du cross-dock actuel et de le catégoriser selon les classes vues dans l'article de Ladier et Alpan [2] afin de trouver un modèle mathématique adéquat. Nous allons vous présenter les différents points notables ci-dessous. À savoir que l'entreprise utilise le terme commande pour désigner une ou un ensemble de palettes provenant d'un même client et allant vers un destinataire unique et le terme montage pour désigner le chargement, l'ensemble des commandes d'un camion sortant du cross-docking.

Questionnaire

Concernant les caractéristiques pour le modèle, nous avons premièrement repris le questionnaire utilisé par Ladier [41]. Ce questionnaire permet de rapidement déterminer à quel type de cross-docking nous avons à faire au travers de quelque points clé dont nous avons discuté avec l'entreprise. L'intégralité du questionnaire est en ANNEXE I de ce rapport. Les points majeurs extraits de ce questionnaire sont les suivants :

- Mode de service mixte,
- Ressources internes limitées (de 10 à 15 manutentionnaires),
- Stockage temporaire limité (environ 750 palettes),

- Heures de départs pour les camions sortants,
- Préemption autorisée,
- Ordre de déchargement et de chargement à respecter,
- Temps de transfert en fonction de la distance et du type de produit,
- Fonctionnement différent selon les jours de la semaine.

Caractéristiques pour le modèle

- L'entrepôt possède 40 portes et des nombreuses rangées utilisées pour le stockage temporaire des marchandises. La Figure 2.2 permet de voir la forme de l'entrepôt ainsi que l'emplacement des portes et des rangées.
- Tous les montages ont une heure de départ prévue, c'est-à-dire que chaque remorque doit être chargée avant cet horaire.
- En cas de retards prévisibles sur cet horaire (camions entrants retardés), les superviseurs du cross-docking peuvent modifier cette heure de départ. Nous n'avons observé qu'une seule fois ce cas de figure durant notre observation.
- Le nombre de manutentionnaires est compris entre 10 et 15. Ils ont tous des chariots élévateurs électriques identiques.
- Les marchandises en partance pour les États-Unis et pour l'Ontario sont prioritaires sur celles à destination du Québec. Cela se généralise par des heures de départ situées entre 18 h et 22 h pour l'Ontario et les États-Unis, alors que pour le Québec, les départs sont entre 3 h et 7 h du matin.
- Les informations concernant le contenu des camions entrants sont connues en temps réel selon les cueillettes, de même pour les informations concernant les montages qui s'actualisent selon les produits cueillis. En général, les cueillettes sont terminées et les montages sont disponibles vers 17 h, 18 h pour l'Ontario et les États-Unis et vers 21 h pour le Québec. Toutefois, il y a toujours un peu de données manquantes.
- Le fonctionnement de l'entrepôt est différent selon les jours de la semaine. Du lundi au jeudi, l'entrepôt fonctionne normalement : réception des camions à partir de 17 h, expédition vers les États-Unis et l'Ontario avant 22 h et vers le Québec avant 7 h du matin. Le vendredi, les expéditions pour le Québec n'ont pas lieu et celle pour l'Ontario et les

États-Unis se terminent plus tôt. Le samedi, le cross-docking est fermé. Il n'y a que la réception des marchandises qui est ouverte. Le dimanche, il n'y a aucune réception, toutes les marchandises sont déjà présentes dans la cour. Seulement l'entrepôt est en fonctionnement et réalise le maximum de montages possibles.

- La mesure des performances se fait actuellement selon trois critères : les retards sur les départs des camions, le nombre de montages réalisés à l'heure par manutentionnaire et le temps mis pour réaliser tous les montages, ou « *makespan* ».
- La préemption est autorisée. Étant donné que le nombre de portes est limité, il est arrivé plusieurs fois que le superviseur soit obligé de reculer une remorque à moitié déchargée pour libérer une porte pour des marchandises prioritaires.
- Il y a toujours des remorques vides disponibles dans la cour. Cela permet au superviseur de ne pas avoir à attendre de vider totalement une remorque pour la réutiliser pour un chargement.
- L'entreprise dispose d'un « *shunter* », un petit camion dont le rôle est de déplacer et de changer les remorques aux portes. C'est une ressource non limitante pour l'entrepôt. Le cas échéant, on peut ajouter un deuxième « *shunter* ».
- Nous avons remarqué qu'il y a environ une dizaine de commandes par montage.

<u>Cas particuliers</u>

- Les camions sortants partent rarement en avance par rapport à l'heure prévue. La raison est que cela consomme des ressources inutilement.
- Le nombre de manutentionnaires excède rarement quinze puisqu'au-delà de ce nombre, leur efficacité baisse en raison du trop grand trafic dans l'entrepôt.
- Il est arrivé que les manutentionnaires déplacent les palettes deux par deux.
- Certaines remorques sont expédiées vers d'autres compagnies de transport. Pour ce genre de cargaisons, nous n'avons aucune information sur le contenu de la remorque et les marchandises sont chargées en vrac, sans prendre en compte un ordre particulier.
- Il existe différents types de conditionnement transitant par le cross-docking. Majoritairement, les marchandises sont sur des palettes, mais il arrive que certaines marchandises soient hors gabarit comme des tiges en acier de plusieurs mètres de long, des

pneus de voitures et des boîtes à chaussures sans conditionnement, ou encore des colis de tailles diverses. Les temps de chargement varient énormément pour ces genres de produits.

- Certaines marchandises sont stockées jusqu'à une semaine avant de partir. Cela dépend de la popularité de la destination puisque l'entreprise essaie de n'avoir que des voyages FTL.

Informations complémentaires

- L'équipe des planificateurs se charge de créer des montages réalisables. Leurs rôles sont de s'assurer que l'ensemble des commandes mises dans un montage n'excède pas les limites de poids légales, de s'assurer de la disponibilité de la remorque affectée au montage selon plusieurs critères (longueur de 48 ou 53 pieds, remorque fermée, réfrigérée, porte-conteneur…) tout en prenant en compte la tournée de livraison. Il arrive aussi qu'ils fassent des modifications de dernières minutes tout au long de la nuit.
- L'entreprise propose des services d'étiquetage et de pesage des marchandises. Cela impacte la durée pour transférer les marchandises dans l'entrepôt.
- Étant donné que c'est le rôle du superviseur d'affecter les camions aux portes, il fait des zones géographiques dans l'entrepôt. Cela lui permet de mettre les camions ayant les mêmes destinations et de mêmes provenances dans la même zone pour éviter aux manutentionnaires de longs déplacements dans l'entrepôt.

Heuristique actuelle

Le superviseur a à sa disposition l'ensemble des montages à effectuer, informatiquement et en temps réel, avec le détail des commandes qui les constitue. Chacune des commandes est représentée avec un code couleur :

- En vert : la commande est présente dans l'entrepôt, dans une rangée de stockage temporaire.
- En bleu : la commande est dans une remorque présentement à quai.
- En jaune : la commande est dans un camion en attente dans la cour.
- En rouge : la commande n'est pas sur le site du terminal, elle n'est soit en route vers l'entrepôt, soit pas encore cueillie.

Ensuite, un algorithme va placer les montages ayant le plus de commandes vertes en tête, puis les montages ayant du bleu, puis du jaune et en dernier, les montages ayant des commandes en rouge. La logique sous-jacente à cette heuristique est de charger les commandes présentes dans l'entrepôt en priorité pour éviter d'engorger ce dernier.

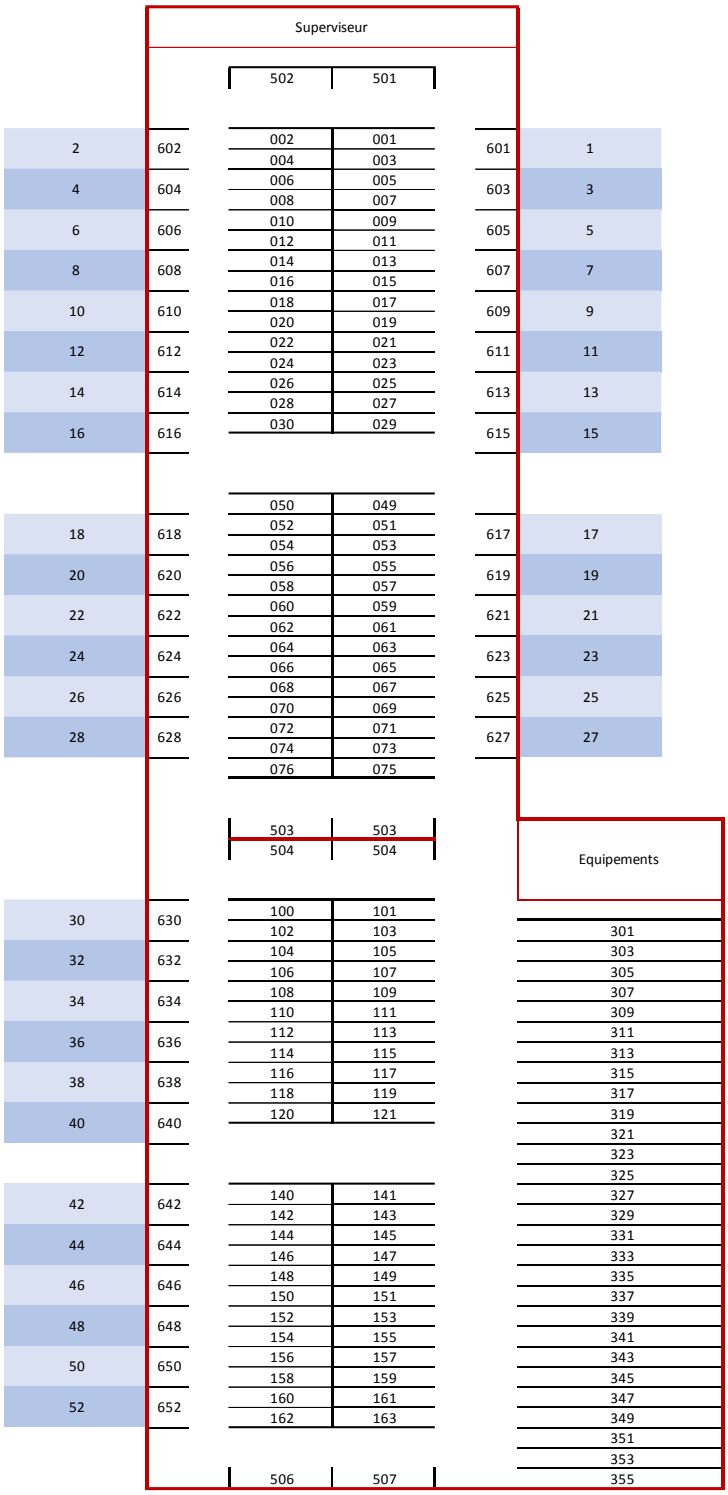


Figure 2.2 Vue d’ensemble de l’entrepôt avec les rangées et les portes

2.2 Modélisation

2.2.1 Analyse et classification de l'entreprise

Avec les observations précédentes, nous pouvons classer le fonctionnement de l'entreprise avec les caractéristiques proposées par Ladier et Alpan [2]. Nous avons un problème de type « *truck-to-door scheduling* » puisque la distance entre les portes est à prendre en compte et parce que nous avons des horaires de départs pour les camions sortants, la variable horaire est aussi importante.

Ensuite, nous avons un mode de service mixte puisque les portes servent pour la réception et l'expédition. La forme de l'entrepôt est assimilable à un I comme le montre la Figure 2.2. Nous avons 42 portes. La préemption est autorisée. Étant donné que nous sommes avec un entrepôt physique, les ressources internes et la capacité de stockage sont limitées. L'heure d'arrivée des camions dépend de la journée étudiée : si l'on est en semaine, les camions ont des horaires d'arrivée différents les uns des autres et à l'inverse, le dimanche, les camions sont tous présents dans la cour et disponibles dès le début de l'activité. Enfin, comme nous l'avons dit, les camions sortants ont un horaire à respecter et les informations concernant l'interchangeabilité des produits sont connues à l'avance et il faut y rajouter l'ordre.

Connaissant les caractéristiques du problème, nous avons cherché un article scientifique avec les mêmes caractéristiques pour réutiliser leurs modèles. Nous avons donc comparé l'entreprise avec les 112 modèles présents dans l'article de Ladier et Alpan [2] ainsi qu'avec la dizaine d'articles plus récents que nous avons étudiés. Nous n'avons trouvé aucune correspondance exacte. Nous avons répertorié seulement sept articles prenant en charge la préemption, 12 avec un mode de services mixtes et 23 ayant des heures de départs.

Nous avons donc combiné plusieurs modèles pour aboutir à un modèle très proche du fonctionnement de l'entreprise. Le Tableau 2.1 représente le fonctionnement de l'entreprise ainsi que quelques articles se rapprochant du fonctionnement de l'entreprise et étant

complémentaires. Nous avons supprimé les caractéristiques identiques pour simplifier le tableau.

Tableau 2.1 Comparaison de l'entreprise avec la littérature

		Strategic	Tactical				Operational			
	Problem type	Nb doors	Service mode	Pre-emption	Temp. Storage capacity	Ressource capacity	Arrival time	Departure time	Interchang.	Performances measures
Tr. Bourassa semaine	TtD Sched.	42	Mixed	Yes	limited	limited	/truck	OutB.	Pré-D + ordre	N/A
Tr. Bourassa dimanche	TtD Sched.	42	Mixed	Yes	limited	limited	Zero	OutB.	Pré-D + ordre	N/A
Assadi et al. 2016	TtD sequ.	*	Excl.	No	∞	∞	/truck	No	Dest	Truck process. time dev. + Makespan
Shakeri & al	TtD Sched.	*	Mixed	No	0	limited	Zero	No	Dest + ordre	Makespan
Wisittipanich & al. 2017	TtD Sched.	*	Excl.	No	∞	∞	Zero	No	Dest	Makespan

2.2.2 Hypothèses et modifications

Nous avons utilisé le modèle de Shakeri & coll. [39] comme notre point de départ. La raison est que dans cet article, les auteurs proposent un modèle qui prend en compte l'ordre de chargement et de déchargement des marchandises tout en conservant une forme de modèle plutôt classique pour un cross-docking. Grâce à cela, nous avons pu modifier des fonctions pour remplacer les temps de chargement et de déchargement pour une palette par des valeurs réalistes obtenues avec les données de l'entreprise. Nous avons aussi rajouté une variable de quantité. En effet, puisqu'une commande peut être composée d'une ou de plusieurs palettes, les temps de chargement et de déchargement dépendent donc du nombre de palettes par commandes. Cela impacte aussi les temps de déplacement, puisque les manutentionnaires ne peuvent déplacer qu'une palette à la fois.

L'autre ajout majeur au modèle est les heures d'arrivées et de départs des camions. Pour cela, nous nous sommes inspirés de l'article de Van Balle & coll. [42]. Dans leur modèle, comme dans le nôtre, ils ont une variable correspondant au moment où un camion est affecté à une porte et une autre lorsque le camion quitte sa porte. Pour contraindre l'heure d'arrivée des camions, les auteurs utilisent une simple inéquation de telle sorte que l'heure d'affectation à

une porte doit être supérieure ou égale à l'heure d'arrivée du camion au terminal. Il faut rajouter dans les données initiales au problème, l'heure d'arrivée de tous les camions entrants. Pour les heures de départ des camions sortant, ils utilisent une contrainte souple, « *soft constraints* ». C'est-à-dire que si un camion part après l'heure initialement prévue, cela engendra du retard. Sachant que dans la fonction à minimiser il y a la variable de retard, on comprend que le modèle va essayer de réduire au maximum les retards. Cependant, s'il ne peut pas faire autrement, le modèle peut proposer une solution comportant un peu de retard, mais qui minimise mieux les autres objectifs, comme ici, la somme des temps de transfert des marchandises. Nous avons donc repris la même équation pour contraindre les heures d'arrivée. Mais pour les heures de départ, nous avons utilisé une contrainte inflexible, « *hard constraints* ». C'est-à-dire que nous ne tolérons aucun retard. L'heure de départ de tous les camions sortants doit être inférieure à l'heure de départ prévu. C'est le cas chez Transport Bourassa, tous les camions partent avant ou à l'heure prévue. D'ailleurs, leurs heures de départ ne sont pas fixées en fonction du temps que mettent les manutentionnaires à vider puis remplir le camion, mais plutôt en fonction du dernier délai pour ne pas prendre du retard sur toute la tournée de livraison.

Autre point important pour correspondre au fonctionnement de l'entreprise, la préemption. Si l'on reprend le modèle de Shakeri & coll. [39], la fonction (2.1), voir partie suivante, oblige chaque camion à être affecté à une seule porte. On ne peut donc plus modifier l'affectation d'une remorque une fois que celle-ci est à une porte. Pour empêcher totalement la préemption dans leur modèle, les auteurs ont deux fonctions (2.2) et (2.10) qui modélise les phrases suivantes : un camion ne peut être affecté à une porte que si son prédécesseur a fini son chargement, plus le temps de déplacer les remorques à la porte ; un camion a fini son chargement lorsque toutes ces marchandises attitrées ont été chargées. En d'autres termes, si un camion n'a pas terminé son chargement, il ne peut pas quitter son quai. On n'a donc pas de préemption.

Pour pallier ce manque, nous avons essayé de relaxer la fonction (2.1) en changeant le signe égal pour un signe supérieur ou égal. Cela revient à dire qu'une remorque peut être affectée à au moins une porte. Les résultats obtenus avec cette modification n'avaient pas de sens. La

raison est que le modèle interdit de quitter une porte si le chargement n'est pas complet. Il aurait donc fallu modifier les fonctions (2.2) et (2.10). Toutefois, ces deux fonctions sont directement reliées avec 50 % des fonctions et impactent la majorité des variables du modèle. En comparaison, notre relaxation ne touche qu'une seule fonction. Pour éviter de modifier la quasi-totalité du modèle, nous avons pris la décision de continuer en ne prenant pas en compte la préemption.

Enfin, selon la classification Ladier et Alpan [2], le modèle de Shakeri & coll. [39] n'a pas de stockage temporaire. Notre compréhension du modèle qu'ils proposent est la suivante : un manutentionnaire qui décharge une palette la déplace vers ce qu'appellent les auteurs, une zone de transit, devant le camion sortant correspondant. La palette attend dans cette zone jusqu'à ce qu'un autre manutentionnaire vienne la charger une fois le camion arrivé. Les auteurs précisent que les zones de transit n'affectent pas les performances des manutentionnaires. Nous considérons donc ces zones de transit comme du stockage temporaire de capacité infinie.

2.2.3 Modèle mathématique

Cette partie est entièrement consacrée à la présentation et à l'explication du modèle mathématique. Après avoir expliqué la signification de chacun des indices, paramètres et variables, nous vous présenterons l'ensemble des équations du modèle.

Indices

m portes, $m \in M = \{1, \cdots, |M|\}$.

j remorques, $j \in J = \{1, \cdots, |J|\}$.

p commandes, $p \in P = \{1, \cdots, |P|\}$.

Avec M, J *et* P respectivement le nombre total de portes, de remorques et de commandes. Les trois paramètres précédents définissent la taille du problème. La complexité du problème dépend du rapport entre le nombre de remorques et le nombre de portes : plus il y a de remorques et plus on peut faire des combinaisons différentes. À l'inverse, si le nombre de

portes est supérieur ou égal au nombre de camions, le problème devient trivial puisqu'on peut affecter tous les camions en même temps à l'entrepôt. Enfin, le nombre de commandes est aussi déterminant dans la complexité du problème puisque c'est directement selon la répartition des commandes dans les remorques que l'on va les affecter aux différentes portes.

Paramètres d'entrée

$t_{mm'}$	Temps de transport entre les portes m et m'.
U_j	Lots de palettes à décharger de la remorque j.
L_j	Lots de palettes à charger de la remorque j.
A_j	Heure d'arrivée des camions entrants j.
D_j	Heure de départs des camions sortants j.
β_p	Position de déchargement de la commande p.
k_p	Quantité de palettes pour la commande p.
ULT	Temps pour décharger une palette.
LT	Temps pour charger une palette.
T^c	Temps de changement des remorques à une porte.
Q	Grand nombre.

Nous avons ci-dessus les différents paramètres nécessaires pour le modèle mathématique. Premièrement, le temps mis par un manutentionnaire pour déplacer une palette d'un quai à un autre. Nous parlons ici de temps et non de distance comme dans l'article de Shakeri [39]. Cela nous a paru plus adéquat puisque toutes les autres variables sont temporelles. Les deux paramètres suivants, U_j et L_j, listent les palettes à charger ou à décharger pour chacune des remorques. Ci-dessous, nous vous avons représenté les matrices U_j et $L_j$ainsi que la première remorque de chacune des deux matrices pour une meilleure compréhension.

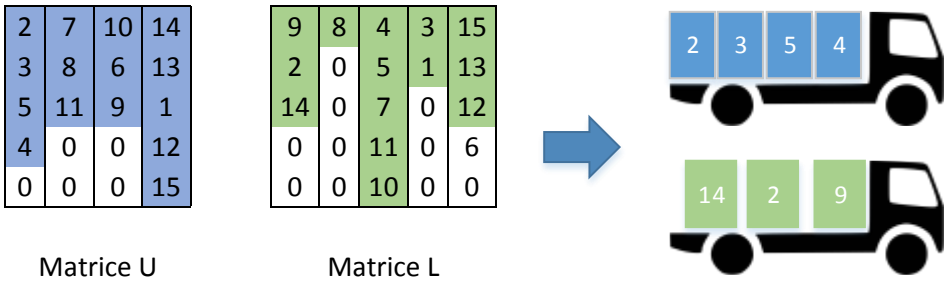


Figure 2.3 Illustration des données U et L

On remarque que l'ordre des commandes dans les remorques est inversé entre U_j et L_j. La manière dont nous avons construit ces matrices est la suivante : pour U_j, la première commande est la première à être déchargée, donc à gauche sur le schéma ; pour L_j, la première commande est la première à être chargée dans le camion donc à droite sur le schéma. Chaque colonne correspond à une remorque.

Les paramètres A_j et D_j correspondent respectivement aux heures d'arrivée des camions entrants et aux heures de départ pour les camions sortants. L'explication de ces paramètres a déjà été faite dans la partie précédente. Le paramètre suivant, β, représente l'ordre de déchargement des palettes. En d'autres mots, la palette déchargée en première aura une valeur de bêta égal à 1, la deuxième sera égale à 2, etc. Le paramètre bêta ne prend pas en compte la remorque dans laquelle la palette doit être déchargée. Le paramètre k représente le nombre de palettes par commande.

Avec les deux paramètres suivants, ULT et LT, les temps de déchargement et de chargement pour une palette, et la quantité de palettes par commande, on peut donc prend en compte le temps de chargement et de déchargement de manière réaliste et précise. Le paramètre T^c correspond au temps mis pour changer de camion à une porte. C'est-à-dire le temps de réaliser les actions suivantes : fermer la remorque, l'amener dans la cour, aller chercher la nouvelle remorque et l'amener à la même porte. Le dernier paramètre, Q, est directement lié au modèle. Il permet de simplifier les expressions avec des variables binaires.

Variables de décision

C_{max}	Temps total ou « *makespan* ».
a_j	Temps d'affectation, d'entrée de la remorque j à une porte de l'entrepôt.
c_j	Temps de désaffectation, de sortie de la remorque j à une porte de l'entrepôt.
λ_p	Temps de début du mouvement de la palette p.
μ_p	Temps de fin du mouvement de la palette p.
σ_p	Temps de fin de chargement de la palette p
$\delta_{pp'}$	1, si les palettes p et p' sont stockées au même endroit (camion ou stockage) et si p est avant p', 0 sinon.
$\gamma_{pp'}$	1, si les palettes p et p' sont chargées dans la même remorque et si p est chargé avant p', 0 sinon.
x_{jm}	1, si la remorque j est affectée à la porte m, 0 sinon.
$y_{jj'}$	1, si les remorques j et j' sont affectées à la même porte et si j est avant j', 0 sinon.
$v_{jmj'm'}$	1, si $x_{jm} = 1$ et $x_{j'm'} = 1$, 0 sinon.

La partie ci-dessus regroupe les variables du modèle. Le premier, le temps total, correspond au temps écoulé entre l'affectation du premier camion au quai et le départ du dernier camion. C'est cette variable qui va nous intéresser lors des tests avec le modèle pour savoir si oui ou non l'algorithme a bien résolu le problème. Les deuxièmes et troisièmes paramètres, a_j et c_j, représentent respectivement l'heure à laquelle une remorque rentre et sort de l'entrepôt. Les trois paramètres suivants, lambda, mu et sigma, sont essentiels pour que le mouvement des commandes ait lieu. Ils représentent respectivement le temps auquel une commande a fini d'être déchargée et est prête à être déplacée, le temps où elle a fini d'être déplacée et le temps où son chargement est terminé (voir Figure 2.4). Les dernières variables $\delta_{pp'}$, $\gamma_{pp'}$, x_{jm}, $y_{jj'}$ et $v_{jmj'm'}$ sont des variables binaires. Elles permettent d'assurer l'affectation des remorques aux portes et le respect de l'ordre des camions et des palettes.

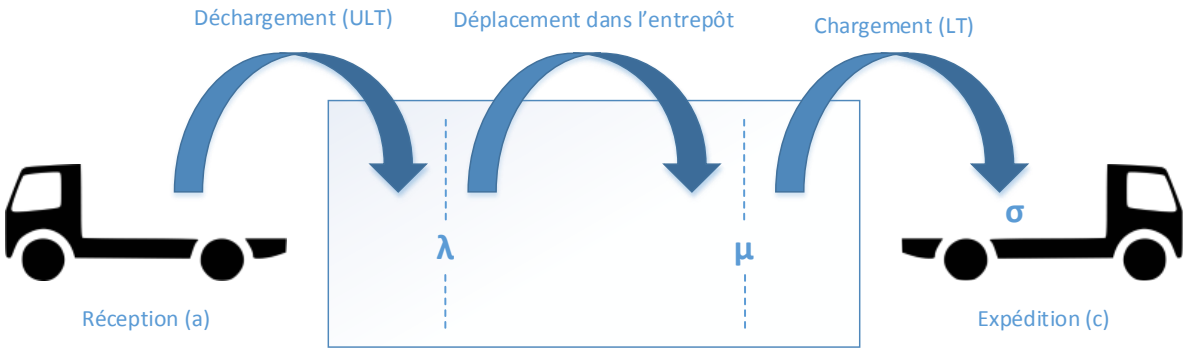


Figure 2.4 Représentation des variables temporaires

Formulation du modèle mathématique

Minimiser C_{max} sous les contraintes suivantes :

$$\sum_{m \in M} x_{jm} = 1\,, \quad j \in J \tag{2.1}$$

$$a_{j'} \geq c_j + T^c - Q.\left(1 - y_{jj'}\right), \quad j, j' \in J, \ j \neq j' \tag{2.2}$$

$$\lambda_p \geq a_j + \sum_{p=1}^{\beta_p} k_p.ULT\ , \quad p \in U_j\ ,\ j \in J \tag{2.3}$$

$$\lambda_p \geq a_j\,, \quad p \in L_j\,,\ j \in J \tag{2.4}$$

$$\lambda_{p'} \geq \lambda_p + 2.t_{mm'}.v_{jmj'm'}.k_p - Q.\left(1 - \delta_{pp'}\right), \quad p \in U_j \cap L_{j'}\,,\ p' \in P,\ p \neq p',\ j, j' \in J,\ m, m' \in M \tag{2.5}$$

$$\mu_p \geq \lambda_p + \left(2.k_p - 1\right) t_{mm'}.v_{jmj'm'}\,, \quad p \in U_j \cap L_{j'}\,,\ j, j' \in J,\ m, m' \in M \tag{2.6}$$

$$\sigma_p \geq \mu_p + LT.k_p\ , \quad p \in P \tag{2.7}$$

$$\sigma_p \geq a_j + \sum_{p'=1}^{\max_{p' \in U_j} \beta_{p'}} k_{p'}.ULT + k_p.L\ —\ , \quad p \in L_j\ ,\ j \in J \tag{2.8}$$

$$\sigma_{p'} \geq \sigma_p + k_p.LT - Q.\left(1 - \gamma_{pp'}\right), \quad p, p' \in L_{j'}\ ,\ p \neq p',\ j \in J \tag{2.9}$$

$$c_j \geq \sigma_p\ , \quad p \in L_j\ ,\ j \in J \tag{2.10}$$

$$C_{max} \geq c_j\ , \quad j \in J \tag{2.11}$$

$$a_j \geq A_j\ , \quad j\ \in U_j \tag{2.12}$$

$$c_j \geq D_j\ , \quad j\ \in L_j \tag{2.13}$$

La fonction (2.1) oblige tous les camions à être affectés à une porte. La fonction (2.2) permet de respecter la chronologie pour deux camions successifs à une même porte. La fonction (2.3) indique qu'une commande est prête à être déplacé dans l'entrepôt une fois que toutes les palettes des commandes précédentes ont été déchargées ainsi que celle en question. La fonction (2.4) indique que le début du déplacement de la commande dans l'entrepôt n'a lieu que lors que la remorque à charger correspondante est à quai. Autrement dit, on ne déplace pas de commandes tant que son camion destinataire n'est pas affecté à une porte. La fonction (2.5) modélise le temps mis par un manutentionnaire pour déplacer une commande puis pour revenir pour en déplacer une autre. La fonction (2.6) correspond au moment où une commande a fini son déplacement. La fonction (2.7) représente le temps de chargement d'une commande dans la remorque. La fonction (2.8) oblige à attendre que le déchargement de la remorque soit complété, que la remorque soit vide, avant de commencer les chargements. La fonction (2.9) sert à ce que l'ordre de chargement soit respecté. La fonction (2.10) indique qu'une remorque

quitte le quai que lorsqu'elle a chargé toutes ses commandes. Enfin, la fonction (2.11) définit le « *makespan* » comme étant l'heure de départ du dernier camion. C'est cette variable que notre modèle vise à minimiser. Nous avons rajouté les fonctions (2.12) et (2.13). Ces deux fonctions permettent d'avoir une heure d'arrivée pour les camions entrants et une heure de sortie pour les camions sortants. Ce sont des contraintes inflexibles, « *hard constraints* », c'est-à-dire qu'on ne peut pas avoir de retard sur ces horaires.

$$y_{jj'} + y_{j'j} = \sum_{m \in M} v_{jmj'm}\,, \quad j, j' \in J,\ \ j \neq j' \tag{2.14}$$

$$\delta_{jj'} + \delta_{j'j} = \sum_{m \in M} v_{jmj'm}\,, \quad p \in U_j\,,\ \ p' \in U_{j'}\,,\ \ p \neq p',\ \ j, j' \in J' \tag{2.15}$$

$$\gamma_{jj'} \geq \left(\mu_{p'} - \mu_p\right)/Q\,, \quad p, p' \in L_j\,,\ \ p \neq p',\ \ j \in J \tag{2.16}$$

$$\gamma_{jj'} \leq 1 + \left(\mu_{p'} - \mu_p\right)/Q\,, \quad p, p' \in L_j\,,\ \ p \neq p',\ \ j \in J \tag{2.17}$$

$$v_{jmj'm'} \leq x_{jm}\,, \quad j, j' \in J,\ \ m, m' \in M \tag{2.18}$$

$$v_{jmj'm'} \leq x_{j'm'}\,, \quad j, j' \in J,\ \ m, m' \in M \tag{2.19}$$

$$v_{jmj'm'} \geq x_{jm} + x_{j'm'} - 1\,, \quad j, j' \in J,\ \ m, m' \in M \tag{2.20}$$

$$Q = 2.|P| + \sum_{m \in M} \sum_{m' \in M} t_{mm'} + |J|.T^c \tag{2.21}$$

Les fonctions ci-dessus assurent le bon fonctionnement des variables binaires. La fonction (2.14) évite que l'ordre des camions aux portes devienne chaotique. La fonction (2.15) s'assure aussi que l'ordre soit respecté dans la zone de stockage pour les palettes. Les fonctions (2.16) et (2.17) assurent l'ordre des palettes lors de leurs chargements. Les trois fonctions (2.18),

(2.19) et (2.20) permettent de définir la variable $v_{jmj'm'}$ et d'éviter de la rendre non linéaire, comme étant un produit de deux autres variables. Enfin, la fonction (2.21) permet de fixer une valeur pour Q qui est supérieure à n'importe quelle durée d'opération dans l'entrepôt.

2.3 Programmation linéaire

Après avoir mis en place le modèle mathématique expliqué juste avant, nous avons voulu le tester sur de petites instances. Pour cela, nous avons utilisé le programme GUSEK GLPK. Ce programme permet de résoudre des programmes linéaires. Nous avons déjà réalisé plusieurs projets avec ce programme. Nous avons donc réécrit notre modèle mathématique en langage GUSEK GLPK. Toutefois, nous avons dû arrêter ce logiciel pour deux raisons : la première est que le modèle que nous avons écrit dans GUSEK GLPK est incapable de trouver une solution faisable à notre problème. Deuxièmement, le manque de support pour déboguer et améliorer le script nous a poussé à changer de programme.

Nous avons utilisé un autre programme capable de résoudre des programmes linéaires, LINGO. Nous avons donc réécrit notre modèle mathématique en langage LINGO. Avec ce logiciel, nous avons pu confirmer que notre modèle est fonctionnel et correspond au fonctionnement de l'entreprise. Nous avons réalisé plusieurs scénarios pour tester les différents paramètres et l'influence des différentes données. Nous avons aussi réalisé un lien entre LINGO et Excel. D'un côté le modèle mathématique dans LINGO et dans une feuille Excel, nous avons regroupé les données et les résultats. Nous avons aussi réalisé plusieurs codes en VBA pour permettre à l'utilisateur de choisir le type de problème qu'il veut tester au travers d'une fenêtre. Nous avons aussi créé un code pour générer des données aléatoirement selon les variables rentrées par l'utilisateur, voir les Figure 2.5 et Figure 2.6 ci-dessous. Un troisième code nous permet de mettre en forme, de manière plus visuelle, les résultats obtenus avec le logiciel LINGO, voir Figure 2.7.

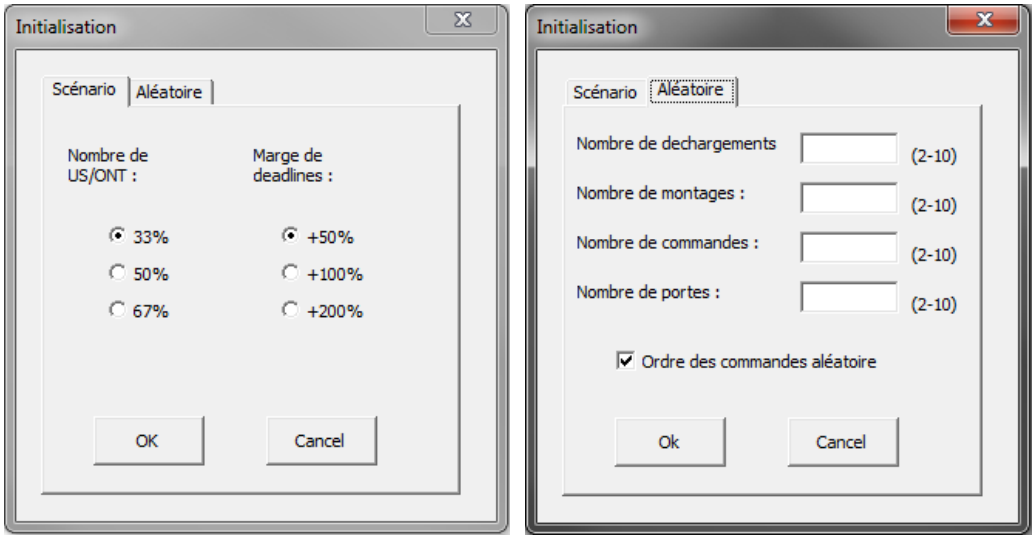


Figure 2.5 Fenêtre de choix des données pour LINGO

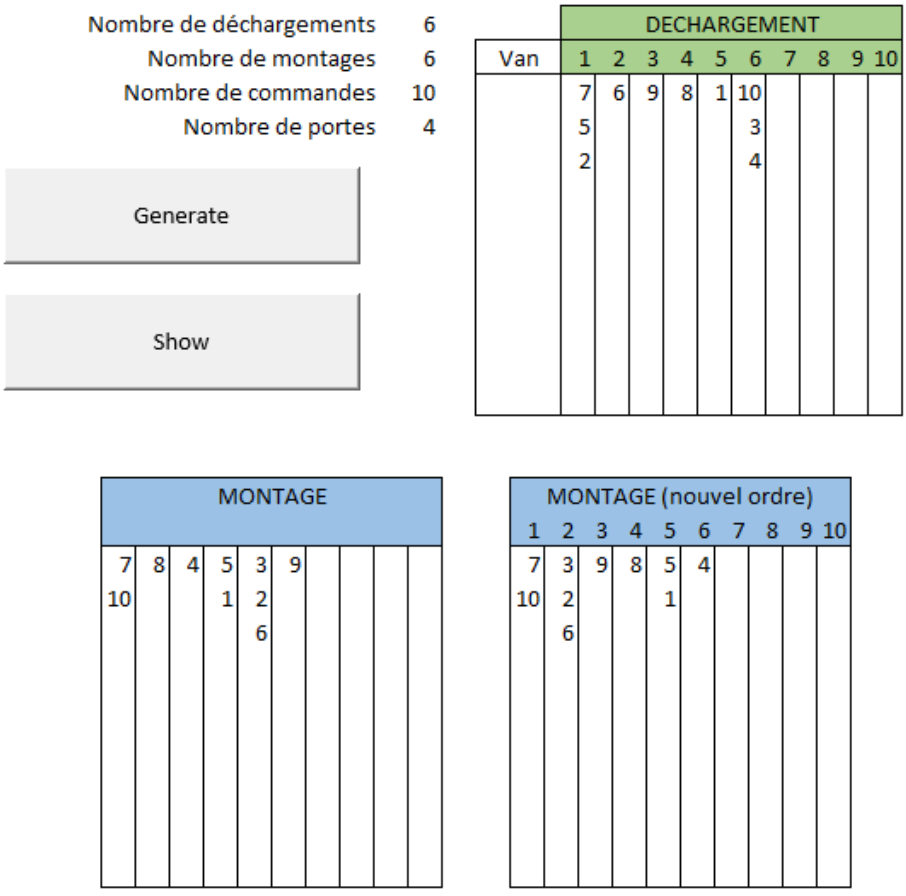


Figure 2.6 Exemple de données générées aléatoirement

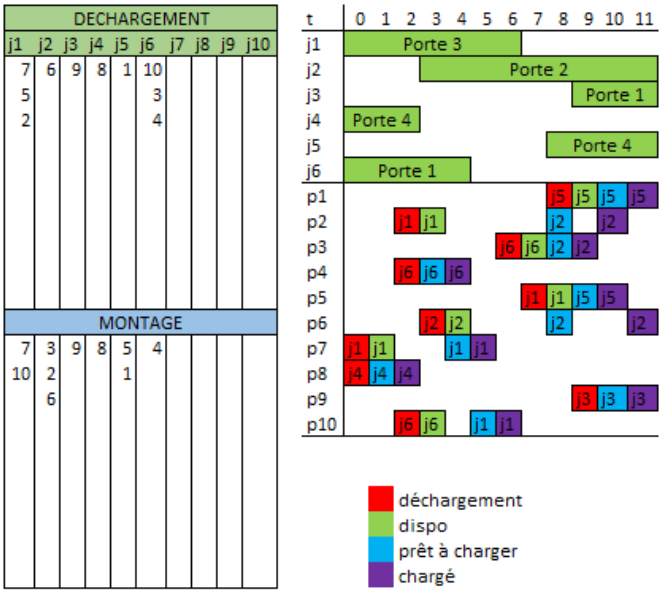


Figure 2.7 Exemple de l'affichage des résultats

L'ensemble des fichiers présentés ci-dessus sont disponibles en libre accès à l'adresse suivante : https://github.com/RemyC/Bourassa-Genetique. Vous pouvez y retrouver le script fait avec GUSEK GLPK ainsi que ceux réalisés avec LINGO. Le fichier Excel que nous avons créé avec toutes les macros est aussi disponible à la même adresse. Remarquez que nous avons limité le nombre de portes à 10, le nombre de remorques à 10 et le nombre de commandes à 30 simplement pour avoir des exemples solvables par LINGO.

À noter que lorsque l'on créer des données de tests, la code VBA donnait des valeurs aléatoires aux paramètres A_j et D_j, les heures d'arrivées et de départs des camions. Il est arrivé plusieurs fois que les valeurs de D_j rendaient le problème infaisable, étant donné que nous avons des limites inflexibles. Nous avons donc dû augmenter les valeurs possibles de D_j pour que le modèle soit solvable à chaque fois.

2.4 Conclusion

Dans cette partie, nous vous avons présenté l'entreprise avec laquelle nous avons réalisé ce projet. Leur fonctionnement et leurs besoins ont été mis en avant. Dans un second temps, nous vous avons présenté le modèle mathématique que nous avons mis en place pour répondre aux besoins de l'entreprise. Nous nous sommes appuyés sur plusieurs articles scientifiques pour pouvoir mettre au point ce modèle. Malgré les efforts que nous avons fournis, certains aspects comme la préemption sont trop complexes à intégrer à notre modèle sans le modifier au complet, tandis que d'autres paramètres comme D_j peuvent rendre le modèle très instable. Enfin, le travail que nous avons réalisé concernant la programmation linéaire a été discuté. Nous verrons dans le chapitre 4 les résultats obtenus avec le logiciel LINGO.

CHAPITRE 3

ÉTUDE DE CAS

Dans cette partie, nous allons vous présenter l'algorithme développé pour pouvoir résoudre notre modèle mathématique mis en place précédemment. Nous vous présenterons le détail du fonctionnement de notre algorithme pour bien comprendre chacune de ces étapes. La question de la collecte des données sera elle aussi soulevée. Nous vous présenterons comment nous avons dû adapter les données transmissent par l'entreprise pour pouvoir les utiliser.

3.1 Limites de la recherche opérationnelle

Nous avons réalisé plusieurs tests du modèle avec différents scénarios sous LINGO. Rapidement, en variant les paramètres, on s'est rendu compte que le nombre de variables et de contraintes est très important même pour des petits problèmes. Avec seulement 4 remorques, 4 portes et 10 commandes, on obtient déjà 427 variables, dont 422 en nombres entiers et un total de 2601 contraintes. En augmentant de deux le nombre de camions, on vient presque doubler ces nombres ; on a alors 779 variables et 3 589 contraintes.

Le temps de calcul pour ces deux premiers problèmes est égal à une seconde. Toutefois, si l'on continue d'augmenter le nombre de commande en passant de 10 à 25, là le temps de résolution passe d'une seule seconde à plus de 15 secondes. Enfin, si l'on augmente encore la taille du problème avec 25 commandes, 10 remorques au lieu de 4 et toujours 4 portes alors, le programme linéaire a besoin de presque 50 minutes. Notre but principal dans ce projet est de répondre aux problèmes de l'entreprise. Pour rappel, la taille du problème de l'entreprise est de l'ordre de 60 camions et 500 commandes par nuit. On comprend alors que la recherche opérationnelle est limitante dans notre cas pour des raisons de temps de calcul. C'est pourquoi, dans la suite de ce chapitre, nous allons voir la métaheuristique que nous avons développée pour contourner ce problème.

3.2 Algorithme génétique

3.2.1 Sélection de l'algorithme

Avec le petit exemple ci-dessus, on comprend que les méthodes dites itératives, qui calculent chacune des solutions, sont extrêmement lentes voir infaisables pour des problèmes de grande taille. Notre solution repose sur les heuristiques qui ont la particularité de ne pas comparer toutes les solutions du problème, mais seulement un petit nombre et de donner comme résultat le meilleur parmi cet échantillon. En d'autres termes, une heuristique permet de trouver rapidement une solution performante, mais pas forcément optimale. La différence entre une méthode heuristique et une métaheuristique est la suivante : une heuristique est un algorithme de résolution développé pour un problème bien spécifique alors qu'une métaheuristique a toujours le même fonctionnement et peut être appliquée pour résoudre différents problèmes.

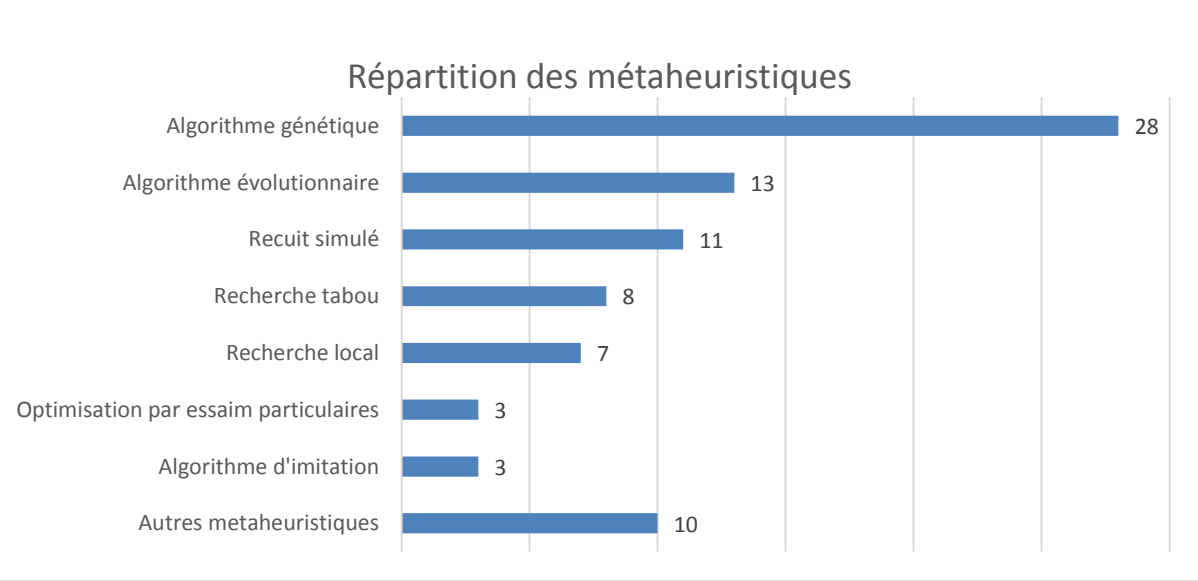


Figure 3.1 Répartition des métaheuristiques dans la revue de Ladier et Alpan [2]

La métaheuristique que nous avons utilisée dans la suite de ce travail est l'algorithme génétique. Il repose sur le même procédé que la sélection naturelle. On assimile notre échantillon à une population, à l'intérieur de laquelle chaque solution est appelée individu.

Grâce à des étapes de sélection, croisement et de mutation, on diversifie la population de sorte à avoir une solution finale très performante. Nous avons fait ce choix pour trois raisons. Premièrement, ce type d'algorithme n'a pas été utilisé sur des modèles similaires au nôtre, ce qui présente une nouveauté. Deuxièmement, il est l'algorithme le plus utilisé dans la littérature (voir Figure 3.1) et enfin, nous avons déjà travaillé avec des algorithmes génétiques par le passé.

3.2.2 Analyse combinatoire

Il faut comprendre qu'en augmentant la taille du problème, le nombre de solutions augmente lui aussi et souvent de façon exponentielle. Nous allons donc à présent, faire l'analyse de la combinatoire du problème, c'est-à-dire dénombrer l'ensemble des solutions faisables en fonction des paramètres du problème. Avec notre modèle, nous avons trois paramètres d'entrée : le nombre de remorques, le nombre de portes et le nombre de palettes. En prenant un nombre de remorques égal à 66, soit la quantité de remorques par nuit que nous avons observée, et une seule porte, on peut choisir la première remorque parmi 66, puis la deuxième parmi 65, etc. On obtient un problème avec 66 ! solutions possibles, soit $5{,}44.10^{92}$ configurations à tester, à évaluer et à comparer.

Prenons maintenant 10 remorques et 1 porte, avec le même raisonnement, on obtient 10 ! solutions, soit 3 628 800. Si l'on rajoute une deuxième porte à notre exemple et que l'on sépare les 10 remorques équitablement entre les deux portes. On peut choisir 5 remorques parmi 10 pour la première porte et 5 remorques parmi les 5 restantes pour la seconde porte. Ensuite, à la porte numéro un, on peut choisir la première remorque parmi 5, puis la deuxième parmi 4, etc. Soit 5 ! solutions. De même pour la porte numéro deux, nous avons 5 ! possibilités. Nous avons mis sous forme de calcul notre raisonnement dans l'équation ci-dessous :

$$\binom{10}{5}.5!.\binom{5}{5}.5! = 252 * 120 * 1 * 120 = 3\,628\,800 = 10! \tag{3.1}$$

On remarque que l'on retrouve 10! solutions possibles comme avec une seule porte. Nous avons continué de calculer avec 3 et 4 portes, toujours en divisant équitablement les remorques aux portes. Pour le cas avec 3 portes, nous avons fait les groupes suivants : {1,2,3,4}, {5,6,7} et {8,9,10}. Pour le cas avec 4 portes, nous avons fait les groupes suivants : {1,2,3}, {4,5,6}, {7,8} et {9,10}. Pour le cas avec 3 portes, pour le premier ensemble, on peut choisir 4 remorques parmi 10, pour le deuxième, il nous reste 3 remorques à choisir parmi les 6 restantes et enfin les 3 remorques parmi les 3 dernières. Pour ce qui est de l'ordre, on a toujours la factorielle du nombre de remorques affecté à la porte.

$$\binom{10}{4}.4!.\binom{6}{3}.3!.\binom{3}{3}.3! = 210 * 24 * 20 * 6 * 1 * 6 = 3\,628\,800 = 10! \tag{3.2}$$

$$\binom{10}{3}.3!.\binom{7}{3}.3!.\binom{4}{2}.2!.\binom{2}{2}.2! = 120 * 6 * 35 * 6 * 6 * 2 * 2 = 3\,628\,800 = 10! \tag{3.3}$$

$$\prod_{i=10}^{1}\binom{i}{1}.1! = 10! \tag{3.4}$$

Encore une fois, on obtient le même résultat. On a voulu aller au bout du raisonnement avec 10 portes pour 10 remorques. On donc le choix de 1 remorque parmi 10 pour la première porte, puis de 1 parmi 9 pour la deuxième porte, etc. L'équation correspondante à cette suite est écrite ci-dessus et sa valeur est aussi de 10! possibilités. On pourrait penser que le nombre de portes n'est donc pas important dans la combinatoire du problème. Toutefois, on peut autoriser une affectation déséquilibrée du nombre de remorques par porte, ce qui est le cas dans la réalité.

Reprenons notre exemple avec 10 remorques et 2 portes avec cette nouvelle propriété. On peut donc affecter les remorques aux portes librement tant que la somme fait 10. On peut avoir 0 camion à la porte 1 et 10 camions à la porte 2, ou bien 1 seul à la porte 1 et les 9 autres à la porte 2, etc. jusqu'à avoir 10 camions à la porte 1 et 0 à la porte 2. Les équations ci-dessous représentent le nombre de combinaisons d'affectations pour les quatre premiers cas. C'est-à-dire 0, 1, 2 et 3 remorques à la porte 1 et 10, 9, 8 et 7 remorques à la porte 2.

$$\binom{10}{0}.0!.\binom{10}{10}.10! = 0 * 1 * 1 * 10! = 10! \quad (3.5)$$

$$\binom{10}{1}.1!.\binom{9}{9}.9! = 10 * 1 * 1 * 9! = 10! \quad (3.6)$$

$$\binom{10}{2}.2!.\binom{8}{8}.8! = 45 * 2 * 1 * 8! = 10! \quad (3.7)$$

$$\binom{10}{3}.3!.\binom{7}{7}.7! = 120 * 6 * 1 * 7! = 10! \quad (3.8)$$

Au vu de ces résultats, nous en concluons que, quelle que soit la répartition des remorques aux portes, nous aurons toujours un nombre de possibilités égal à la factorielle du nombre de remorque.

Pour en revenir à notre exemple avec deux portes, 10 remorques et une affectation déséquilibrée, on a les combinaisons suivantes d'affectations pour les portes 1 et 2 : {0, 10}, {1,9}, {2,8}, {3,7}, {4,6}, {5,5}, {6,4}, {7,3}, {8,2}, {9,1} et {10,0}. Soit $11 \times 10!$ solutions possibles, soit environ 39 millions de cas différents. Nous avons voulu aller plus loin et nous avons refait ce calcul avec 3 portes. Cela revient à trouver le nombre de combinaisons possibles de trois entiers positifs ou nuls dont la somme vaut 10, sachant que l'ordre est important. Nous avons trouvé 66 combinaisons : {0,0,10}, {0,1,9}, {0,2,8}, etc. Comme pour l'exemple avec deux portes, nous multiplions ce nombre par la factorielle du nombre de remorques qui nous donne un total de près de 239 millions de solutions.

Nous avons trouvé la suite correspondant aux n nombres entiers ou nuls dont la somme vaut 10. Cette suite se nomme hendecaenneon et correspond à la lecture verticale de la 10^{e} colonne du triangle de Pascal [43]. Les autres valeurs de cette suite sont : 1, 11, 66, 286, 1 001, 3 003, 8 008, 19 448… On peut généraliser cette suite pour n'importe quel nombre de remorques avec la formule suivante, où j correspond au nombre de camions et m, au nombre de portes :

$$Combinaison\ d'affectations\ déséquilibrées = \binom{j+m-1}{m} \quad (3.9)$$

Nous concluons cette partie en annonçant que le nombre total de possibilités pour un problème ayant 66 remorques et 40 portes est de $\binom{66+39}{40}.66!$, soit $8{,}75.10^{121}$ façons différentes d'affecter les camions aux portes.

3.2.3 Chromosomes : définition et difficultés

Les algorithmes génétiques représentent la solution d'un problème sous forme d'un vecteur ou d'une matrice appelée chromosome. La forme et le contenu de ce chromosome sont très importants puisqu'ils conditionnent la qualité du résultat final. Nous avons repris la forme utilisée dans les articles de Wizzitapanich et de Assadi [25], [32] (à gauche du Tableau 3.1). Nous l'avons modifiée pour correspondre à notre modèle. Premièrement, nous avons un mode de service mixte, nous ne pouvons donc pas séparer notre chromosome en deux parties avec les entrées et les sorties. Nous avons donc fusionné ces deux parties.

Ensuite, nous nous sommes questionnés sur les paramètres importants pour définir la solution de notre problème. Avec un problème de « *truck-to-door scheduling* », il nous faut savoir quel camion va à quelle porte et dans quel ordre. L'ordre nous suffit, nous pouvons retrouver les paramètres temporels grâce au modèle. Nous avons donc pensé à faire une matrice avec trois lignes qui correspondraient chacun au numéro de la porte, au numéro du camion et à l'ordre de passage. La troisième ligne a été supprimée puisque si l'on construit et lit notre chromosome de gauche à droite, nous avons déjà un ordre d'établi. Nous avons ensuite rassemblé nos deux lignes en une seule, voir à droite du Tableau 3.1. Il se lit comme suit : à la porte 1, nous avons le camion numéro 5. À la porte 2, nous avons le camion 3; en porte 3, le camion 2 puis à nouveau, en porte 1, nous avons le camion 6, qui arrivera après le camion 5, etc.

Tableau 3.1 Représentations des chromosomes

	Entrant							Sortant						
Remorque	1	3	4	2	6	5	7	6	4	7	2	1	3	5
Porte	1	2	3	1	2	3	1	1	2	1	2	1	2	1

Portes							Remorques						
1	2	3	1	2	3	1	5	3	2	6	1	7	4

Durant le début du projet, nous avons utilisé cette forme de chromosome. Toutefois, lors de nos tests sur de petits problèmes, nous avions une multitude de solutions ayant la même performance. La raison est qu'avec notre écriture, l'ordre des portes et des camions est variable. Il arrive donc qu'une même solution physique ait plusieurs écritures de chromosomes possibles, voir l'exemple de la Figure 3.2.

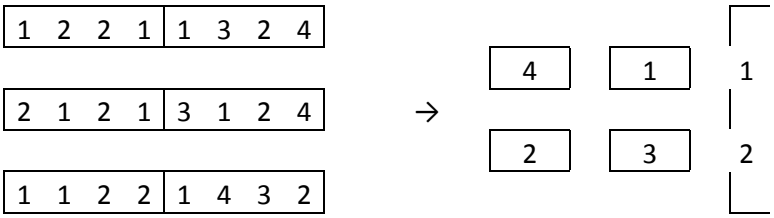


Figure 3.2 Exemple de chromosomes représentant la même situation

La solution que nous avons apportée est de fixer les variables correspondantes à l'ordre des portes, la première partie du chromosome. Ainsi on perd des possibilités, notamment celle qui ne distribue pas l'affectation des camions aux portes de façon équilibrée. Mais on gagne surtout en réduisant le nombre de doublons. Par définition, nous avons fixé l'ordre des portes comme étant la répétition de 1 au nombre de portes total jusqu'à atteindre le nombre de remorques. Soit, si l'on a 10 remorques et 4 camions cela donne : 1, 2, 3, 4, 1, 2, 3, 4, 1, 2.

Nous avons dit que cela nous a permis de réduire le nombre de doublons, mais aussi le nombre total de solutions. En effet, si l'on prend 66 camions et 40 portes, correspondant aux dimensions de l'entreprise, et une seule répartition des camions aux portes, nous avons 66! solutions possibles avec la nouvelle écriture. Avec l'ancienne écriture, on pouvait affecter chacun des 66 camions à l'une des 40 portes, puis les ordonner selon n'importe quel ordre. On avait donc $40^{66} \times 66!$ possibilités anciennement. On a donc réduit de moitié de l'ordre de grandeur des solutions, $5{,}44.10^{92}$ contre $2{,}96.10^{198}$. On rappelle que le nombre total de solutions différentes pour ce problème est de $8{,}75.10^{121}$ ce qui prouve bien que l'ancienne écriture générait des doublons et que la nouvelle ne couvre pas toutes les solutions.

Par ailleurs, avant cette simplification du chromosome, nous faisions le croisement uniquement sur la partie des portes et la mutation uniquement sur les remorques, voir la Figure 3.3. Le croisement sur les portes ne nécessitait pas de reconstruction, autrement dit, ça n'a pas d'importance si l'on retrouvait plusieurs fois la même porte et ce cela de façon déséquilibrée. Nous réalisions la mutation, ou le changement de place des deux gènes, uniquement sur la partie des camions pour ne pas faire apparaître de doublons dans la séquence de camions. Le chromosome n'a plus de sens s'il présente plusieurs fois le même camion et en supprime certains. L'idée, lorsque nous avions commencé à coder l'algorithme, était de proposer la méthode la plus simple. Nous verrons dans la partie suivante comment nous avons amélioré le croisement et la mutation avec la nouvelle écriture fixe du chromosome.

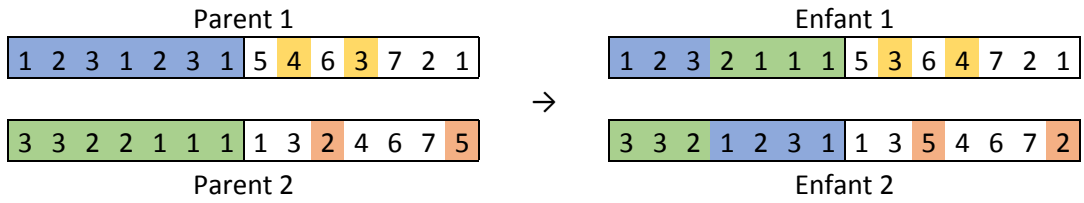


Figure 3.3 Première réalisation du croisement et de la mutation

3.2.4 Structure et explications de l'algorithme

Notre algorithme reprend tous les aspects d'un algorithme génétique. Nous les avons listés ci-dessous et nous les détaillerons par la suite, voir aussi la :

1) Récupération et création des données du problème,
2) Création de la population initiale,
3) Sélection des parents potentiels parmi cette population,
4) Création des enfants par croisement et par mutation,
5) Calcul de la valeur de C_{max} des individus,
6) Tri des enfants et des parents selon la valeur obtenue avec la fonction objective,
7) Suppression des individus les moins performants,
8) Compter une itération et recommencer à l'étape 3,
9) Afficher les résultats.

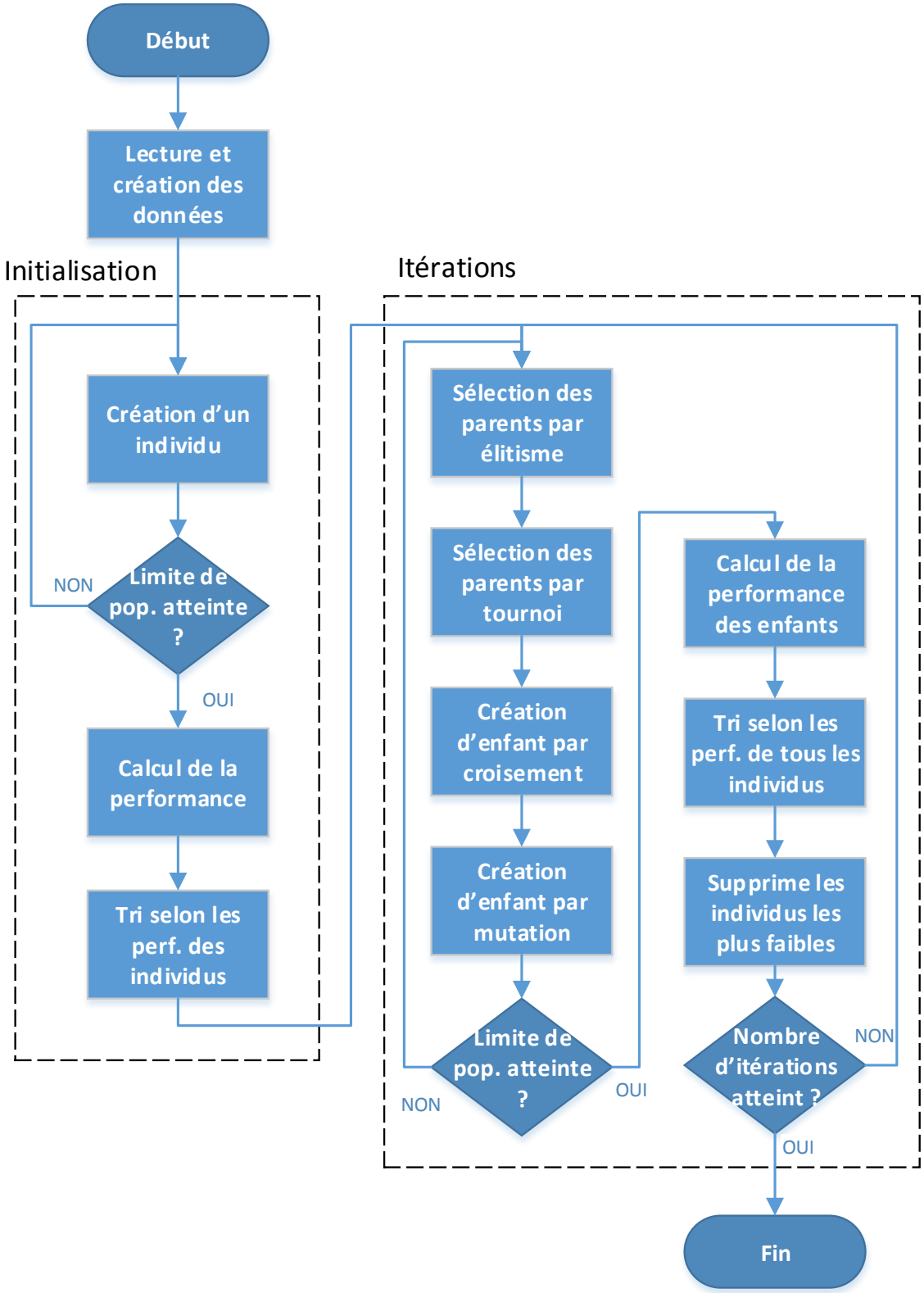


Figure 3.4 Représentation de l'algorithme génétique

1) Récupération et création des données du problème

Dans cette première étape, nous réalisons deux tâches, définir quel type de scénario l'utilisateur veut tester et récupérer les données associées au choix qu'il a fait. Pour la première tâche, l'utilisateur peut soit choisir un scénario préenregistré, soit de nouvelles données créées aléatoirement (en précisant le nombre de remorques, de portes et de commandes), ou bien encore prendre des données à partir d'un fichier texte.

Seuls les paramètres suivants sont nécessaires pour le problème : les nombres de porte, de camions et de commandes ; les matrices U_j et L_j; le temps de changement de camions et les quantités associées aux commandes ainsi que les paramètres pour la fonction des distances entre les portes. La deuxième tâche de cette partie est donc de créer toutes les autres données du problème à partir de celles-ci. C'est le cas des variables suivantes : $t_{mm'}$, Ub et Lb qui sont des matrices binaires qui associent une commande avec un camion, β_p à partir de U_j, $\gamma_{jj'}$ à partir de L_j et enfin Q.

2) Création de la population initiale

Nous créons la première population de chromosome de façon aléatoire. Comme nous l'avons dit dans la partie liée au chromosome, la première partie de celui-ci est toujours la même pour tous les individus. Pour la deuxième partie du chromosome, celle correspondante à l'ordre des camions, nous faisons une liste de nombres de 1 au nombre maximal de remorques que nous mélangeons ensuite. Cela permet de bien avoir tous les camions présents dans le chromosome et de ne pas avoir de doublons. Nous refaisons des individus tant que le nombre d'individus n'ait pas été atteint. Pour faciliter la sélection des parents, nous calculons la fonction objective de chacun des individus créés et nous les trions de la plus faible à la plus importante.

3) Sélection des parents potentiels parmi cette population

Une fois que la première population est créée, nous avons mis en place deux méthodes pour choisir les parents potentiels pour se reproduire et créer des enfants. Les deux méthodes utilisées sont l'élitisme, ou roulette, et le tournoi. Le fonctionnement du premier est d'attribuer

une probabilité de reproduction entre 1 et 100 en fonction de leur valeur avec la fonction objective. Plus cette valeur est faible, plus l'individu est performant et plus sa probabilité de reproduction est élevée. Ensuite, nous sélectionnons un individu au hasard. Nous attribuons une valeur au hasard entre 100 et 1. Si cette valeur est plus faible que la probabilité de reproduction de l'individu alors celui-ci est retenu et fera partie des parents reproducteurs. Nous refaisons ce test jusqu'à avoir sélectionné le nombre de parents désiré.

La deuxième méthode est celle du tournoi. Pour cela, nous attribuons aussi une probabilité de reproduction en fonction de la fonction objective. Nous sélectionnons ensuite trois individus au hasard non encore sélectionnés pour devenir parents. Le meilleur de ses trois individus, celui ayant la probabilité de reproduction la plus élevée, est retenu pour faire partie des parents reproducteurs. Comme pour l'élitisme, nous refaisons des tournois jusqu'à avoir atteint la limite de parents souhaitée.

4) Création des enfants par croisement et par mutation

Pour créer de nouveaux individus, nous sélectionnons deux parents au hasard parmi la liste de parents reproducteurs. S'ils s'accordent, c'est-à-dire s'ils passent la probabilité de croisement alors ils vont pouvoir créer de nouveaux individus. Le croisement a lieu en un seul point, sur la partie camion. Nous reconstruisons la suite du chromosome en fonction des gènes manquants et l'ordre dans lequel ils sont. La Figure 3.5 le montre en couleur, l'enfant 1 récupère la première partie du chromosome du parent 1 et les gènes manquants proviennent selon l'ordre dans lequel ils sont présents chez le parent 2.

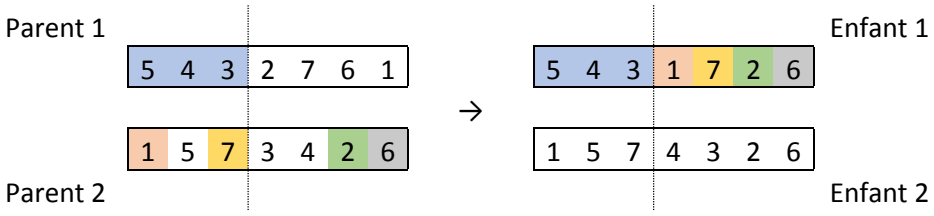


Figure 3.5 Croisement sur la partie camion avec reconstruction

L'autre façon de produire un nouvel individu est la mutation. Elle a lieu après le croisement, avec les deux mêmes parents. Là encore, il y a une probabilité de mutation. Si le couple la respecte alors une mutation aura lieu. La mutation correspond à l'échange de deux gènes d'un des deux parents. La Figure 3.6 illustre le phénomène.

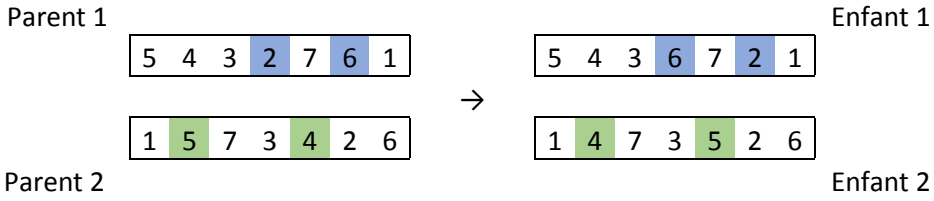


Figure 3.6 Mutation sur la partie camion

5) Calcul de la valeur de C_{max} des individus

Dans cette partie, nous allons voir comment nous avons fait pour calculer la fonction objective. Nous avons repris l'ensemble des fonctions présentes dans le modèle. Toutefois, plusieurs variables ont besoin de connaître la valeur d'autres variables pour être définies. Prenons le camion numéro 2, successeur du camion 1; on ne peut pas calculer l'heure d'arrivée du camion 2 à quai tant que le camion 1 n'a pas fini toutes ses opérations avec l'entrepôt. Cela nous a posé problème puisque la façon dont un ordinateur lit un script est séquentielle. La Figure 3.7 ci-dessous vous montre les interactions entre les variables ainsi que les équations qui les relient entre elles. On peut très clairement voir que les variables sont cycliques, qu'il y a deux boucles différentes et que certaines variables bouclent sur elles-mêmes. Nous avons dû nous reprendre à plusieurs fois avant de réaliser une fonction qui calcule bien la performance d'un individu.

Pour pallier ce problème, nous avons pensé à boucler plusieurs fois sur le modèle. Nous avons aussi rajouté une variable qui compte s'il y a des modifications des variables du modèle. Si celle-ci est nulle alors que l'algorithme vient de boucler sur le modèle, c'est-à-dire que toutes les variables n'ont pas été mises à jour et ont donc atteint leurs valeurs définitives.

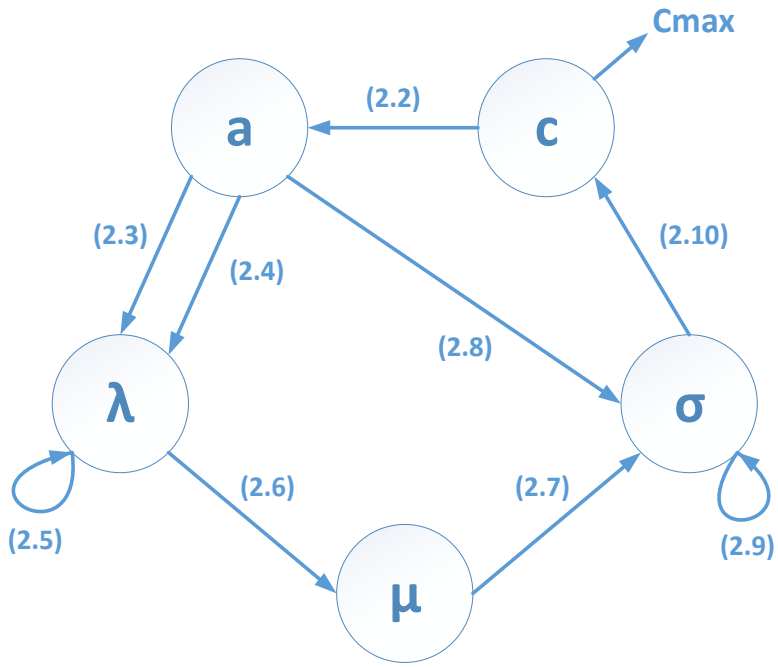


Figure 3.7 Interactions et dépendances des variables

6) Tri des enfants et des parents selon la valeur obtenue avec la fonction objective

Une fois que tous les nouveaux individus connaissent leurs fonctions objectives, on vient les trier de la plus faible valeur à la plus grande. Pour cela, nous cherchons la plus faible valeur parmi toutes les valeurs puis nous la mettons en tête. Puis la seconde à la seconde place, etc. Ainsi, la population dans laquelle le programme cherche la solution diminue à chaque itération. C'est un tri par sélection et sa complexité est égale à $\Theta(n^2)$. Ce n'est pas le tri le plus rapide, toutefois, le gain potentiel de temps est négligeable devant d'autres fonctions du script.

7) Suppression des individus les moins performants

Grâce au tri réalisé juste avant, nous pouvons rapidement supprimer les plus mauvais éléments et ne garder que les meilleurs pour la prochaine itération et la prochaine reproduction.

8) Compter une itération et recommencer à l'étape 3

Une fois que les meilleurs individus ont été mis de côté pour la prochaine itération, nous refaisons l'ensemble des étapes venant après l'initialisation. Plus il y a d'itération et plus

l'algorithme va converger vers une solution efficace qui correspond au meilleur individu, mais plus cela va prendre du temps.

9) Afficher les résultats

Dernière étape de l'algorithme, l'affichage des résultats des valeurs du meilleur chromosome obtenu et du temps de calcul nécessaire à ce calcul.

3.2.5 Organisation et accès à l'algorithme

L'ensemble de l'algorithme a été réalisé en langage Python 3.6 en utilisant PyCharm comme environnement. L'ensemble du code ainsi que tous les résultats et les données sont disponibles, à l'adresse suivante : https://github.com/RemyC/Bourassa-Genetique.

Nous avons découpé l'algorithme en quatre fichiers différents : main.py, qui regroupe l'ensemble des paramètres rentré par l'utilisateur, c'est ce script qui va appeler les autres au fur et à mesure ; utils.data.py, qui génère les données du problème, qui lit les fichiers .txt et qui fait la conversion des rangées de stockage en portes. Le fichier utils.GA.py permet de réaliser l'algorithme génétique en lui-même, c'est-à-dire qu'il réalise l'initialisation et les itérations ainsi que la sélection des parents, le tri et le croisement et la mutation. Enfin, le fichier utils.FF2.py est uniquement dédié au calcul de la fonction objective. Il crée les différentes matrices des paramètres obtenues avec le chromosome proposé. On y retrouve aussi l'intégralité du modèle mathématique et l'astuce de boucle pour bien calculer les différentes variables.

Nous avons représenté graphiquement les liens entre les quatre fichiers ainsi que l'ordre des différentes fonctions créées, voir la Figure 3.8 sur la page suivante. Les pseudo-codes correspondant à la lecture et à la création des données, à l'initialisation ainsi qu'aux itérations sont présents en ANNEXE II, ANNEXE III et ANNEXE IV de ce document. L'objectif de ses pseudo-codes est de faciliter la compréhension du code que nous avons réalisé.

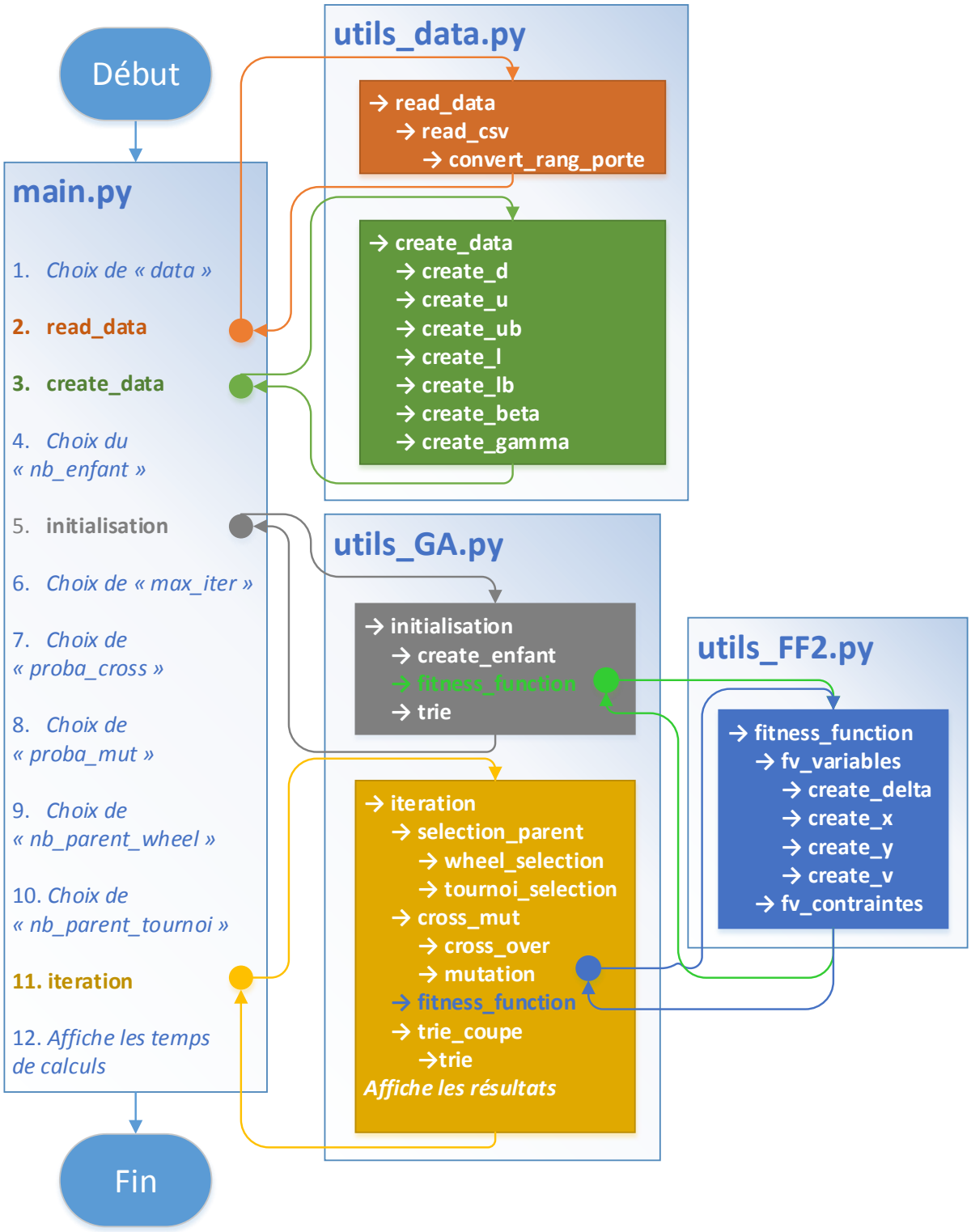


Figure 3.8 Organisation des fichiers et des fonctions Python

3.3 Collecte de données réelles

Afin de rendre tout notre travail plus réaliste et le plus proche du fonctionnement de l'entreprise partenaire, nous avons profité de plusieurs rencontres avec l'entreprise et des journées d'observations pour pouvoir collecter des données réelles. Comme nous l'avons dit précédemment, les données nécessaires pour le modèle sont peu nombreuses. Nous allons vous présenter comment nous avons calculé les temps entre les portes et les matrices U et L.

3.3.1 Temps de déplacement entre les portes

On a vu que le modèle mathématique prenait en compte les temps de déplacement entre les différents quais de déchargement. Étant donné que les manutentionnaires scannent chacune des palettes qu'ils déplacent, on peut récupérer le temps qu'ils ont mis pour aller d'un point à un autre dans l'entrepôt. Pour cela, nous avons récupéré l'historique des mouvements de 11 manutentionnaires durant une période totale de 6 nuits, du 31 janvier au 2 février, du 7 au 9 et du 13 au 15 février. Pour maximiser le nombre de données disponibles, nous avons utilisé les temps mis par les manutentionnaires pour aller d'une porte à une autre et aussi les déplacements entre une porte et une rangée de stockage. Nous avons approximé la porte la plus proche de chacune des rangées.

Cela représente 12 200 déplacements au total. Pour rappel, nous avons un total de 40 portes dans l'entrepôt, donc pour calculer les durées de déplacement, il nous faut calculer 780 durées, soit la somme des entiers entre 39 et 1. Il nous a paru bien plus pratique de trouver une formule capable de synthétiser ces durées plutôt que d'avoir 780 valeurs différentes. D'autant que les données réelles issues des historiques sont sur une plage très étendue, allant de 0 seconde à plus de 21 heures, et sont très bruitées. Nous avons donc essayé de nettoyer les données et de ne conserver que celles intéressantes pour trouver les temps de déplacement. Nous avons gardé uniquement les valeurs comprisses entre 4 secondes et 2 minutes 26, ce qui correspond aux valeurs entre les centiles 20 et 80.

Tableau 3.2 Répartition des données du temps de déplacement

Centile	10	20	30	40	5	60	70	80	90
Temps	00:00:01	00:00:04	00:00:37	00:00:52	00:01:05	00:01:19	00:01:40	00:02:26	00:04:41
Position	1220	2440	3660	4880	6100	7320	8540	9760	10980

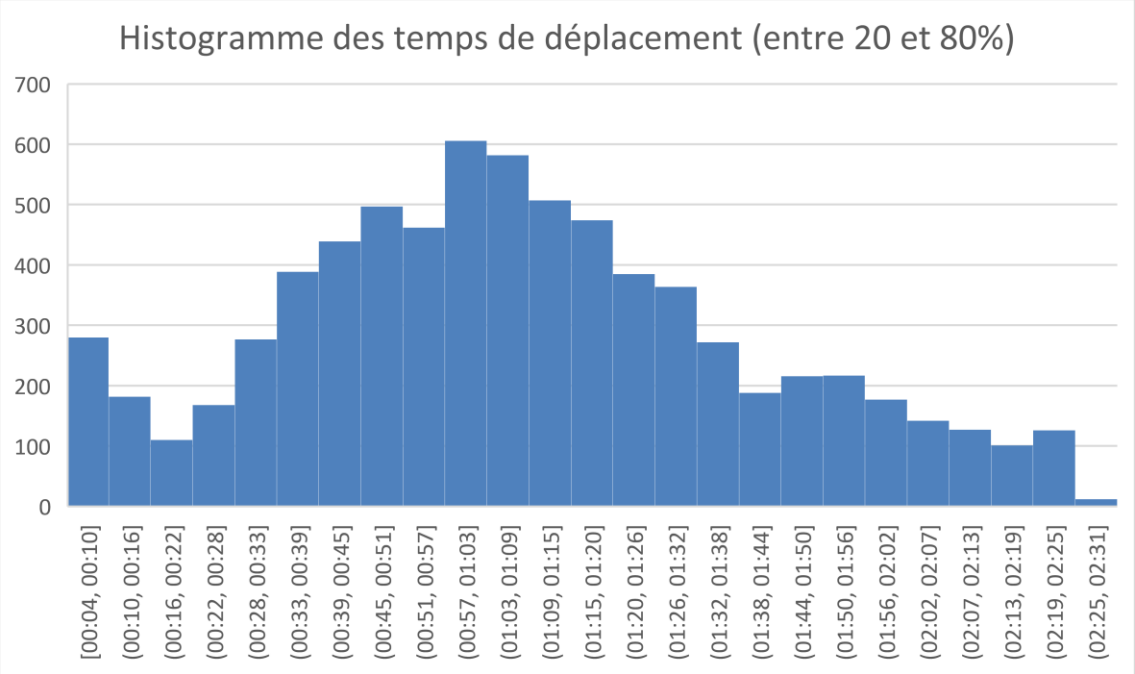


Figure 3.9 Histogramme des temps de déplacements

Pour homogénéiser les temps mis par les manutentionnaires, nous avons estimé que les temps suivaient la formule ci-dessous (3.10). Avec a, une valeur fixe ; b, le temps pour parcourir la distance en longueur séparant les deux portes et c, le temps mis pour traverser le dock d'un côté à l'autre. On peut voir sur la Figure 3.10 ci-dessous la façon dont nous avons calculé les temps.

$$t_{mm'} = a + b.L + c.l \tag{3.10}$$

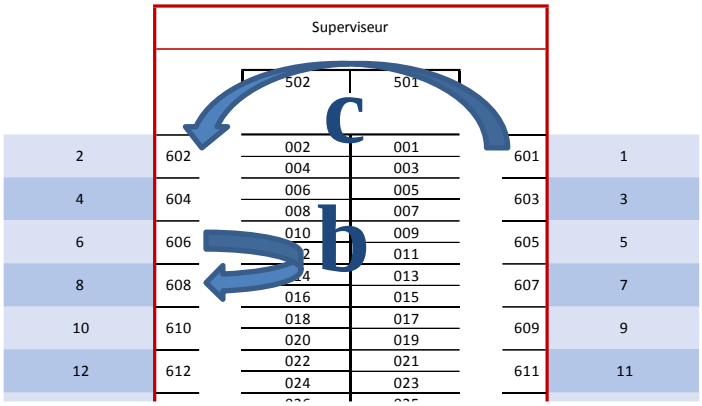


Figure 3.10 Représentation des temps de déplacement

Enfin, pour trouver les meilleures valeurs de nos variables, a, b et c, nous avons utilisé la méthode de la somme des écarts au carré. Autrement dit, nous avons fait varier les trois variables dans notre formule, calculé l'ensemble des 780 temps de déplacements et conservé celles qui minimisent l'erreur avec les données de l'entreprise. Les plages utilisées pour a, b et c sont respectivement de 0 à 20, de 0,25 à 2,5 par pas de 0,25 et de 1 à 20. Les valeurs retenues sont les suivantes : a est égal à 49 secondes, b est égal à 0,25 seconde et c est égal à 1 seconde.

On a donc un distancier qui suit la formule suivante : $t_{mm'} = 49 + 0{,}25.L + l$ avec L, le nombre de portes à parcourir et l, une variable binaire si le déplacement doit changer de coté de l'entrepôt. On a donc l'ensemble des durées des 780 déplacements de connues. On regrette un peu la trop grande précision de la formule, au quart de seconde, par rapport au reste du problème qui est de l'ordre de grandeur de la minute. Toutefois, avec les données disponibles, ce sont les variables qui minimisent le plus l'erreur.

3.3.2 Emplacement initial des commandes et des camions

Nous avons aussi demandé à l'entreprise de nous fournir des informations concernant l'ensemble des chargements à réaliser sur une journée et l'emplacement de toutes les palettes nécessaires à ces chargements. L'entreprise nous a fourni ses informations sous forme de trois fichiers textes. Le premier, nommé voyage.txt, liste l'ensemble des montages à effectuer dans

la soirée, avec le détail de chacune des commandes nécessaires, ainsi que leurs quantités de palettes, leurs emplacements dans l'entrepôt et leur ordre de chargement et de déchargement, β_p, le cas échéant. Le deuxième fichier, porte.txt, recense toutes les remorques accrochées à l'entrepôt à l'instant de la requête. On peut donc savoir quelles remorques sont à quai ainsi que les palettes présentes à l'intérieur de ces remorques. Enfin, le troisième fichier, remorque.txt, dresse une liste de toutes les remorques dans la cour avec leur contenu. Dans ce dernier fichier, nous voyons uniquement les commandes utilisées pour les montages de la soirée ; s'il y a d'autres commandes dans les camions, mais que celles-ci ne font pas partie des montages du fichier voyage.txt alors elles n'apparaissent pas.

Cela représente au total 66 montages et 563 commandes au total, soit 8,5 commandes par chargement. À partir de ces trois fichiers, nous avons essayé de reconstruire l'ensemble des données nécessaires pour notre algorithme génétique. Premièrement, la matrice L_j a été plus simple à recréer puisque, avec le fichier voyage.txt, nous savons ce que chacun des montages comporte. Deuxièmement, pour reconstruire la matrice U_j, il nous a fallu prendre en compte les trois fichiers. En effet, il faut prendre en compte toutes les commandes, qu'elles soient dans un camion dans la cour, dans un camion à une porte ou encore qu'elles soient dans une rangée de stockage dans l'entrepôt.

Notre modèle prend initialement toutes les commandes comme étant dans des camions en attente dans la cour. Nous avons donc dû trouver un moyen de modéliser le fait que certaines commandes sont déjà aux portes ou dans la zone de stockage. Pour résoudre ce problème, nous avons pensé aux choses suivantes : pour les commandes dans la cour, rien de spécial ; pour les commandes présentes à une porte, on fixe la valeur de x_{jm} à 1. Cela oblige l'affectation du camion à la porte en question. On fixe aussi a_j égal à 0, ce qui revient à affecter le camion à sa porte dès le début du problème. Pour les commandes déjà présentes dans une rangée de stockage, nous avons pensé à mettre ses commandes dans un camion fictif affecté à la porte la plus proche des rangées où se situent les commandes. S'il y a déjà un camion d'affecté à une porte, les commandes présentes dans les rangées seraient rajoutées à ce camion. Dans tous les

cas, que le camion soit fictif ou réel, nous lui attribuons des valeurs de x_{jm} et de a_j de 1 et 0 et nous rajoutons que la valeur des commandes déjà déchargées est également nulle, soit $\beta_p = 0$.

Ensuite, nous avons vérifié si l'ensemble des commandes présentes dans le fichier voyage.txt qui correspond à la matrice L_j, étaient présentes dans les deux autres fichiers, qui correspondraient à la matrice U_j. Il s'avère que 7% des commandes sont introuvables, soit près de 40 commandes. Ces 40 commandes ne sont toujours pas arrivées à l'entrepôt et sont sûrement encore sur la route. En plus de cela, nous avons la valeur de β_p dans le fichier voyage.txt et l'ensemble des commandes présent dans les camions dans les deux autres fichiers. La logique voudrait que la valeur du β_p corresponde à l'emplacement de la commande dans la remorque. Or, il n'y a aucune relation entre les ces deux valeurs. Pour terminer, nous n'avons pas réussi à mettre en place informatiquement l'astuce des différentes valeurs de β_p, x_{jm} et de a_j pour les palettes qui ne se situent pas initialement dans la cour.

3.4 Conclusion

Au cours de cette partie, nous vous avons présenté les limites qu'avait la recherche opérationnelle pour résoudre des problèmes de grande taille. L'algorithme génétique est la solution que nous avons choisie pour pouvoir résoudre des problèmes de taille industrielle. Son fonctionnement a été présenté en détail ainsi que la construction du chromosome. Enfin, nous vous avons présenté les données collectées au sein de l'entreprise qui sont nécessaires pour le problème et notre façon de les adapter au fonctionnement du modèle. Enfin, quelques-unes des difficultés ont été abordées comme l'ajout de caractéristiques et de contraintes au modèle ainsi que la résolution avec les données issues de l'entreprise.

CHAPITRE 4

VALIDATION, ANALYSE ET DISCUSSIONS

Dans ce chapitre, nous allons vous présenter plusieurs tests que nous avons réalisés. Nous vous présenterons les résultats obtenus avec notre algorithme génétique ainsi qu'avec la programmation linéaire. Plusieurs scénarios ont été utilisés pour montrer comment réagit notre algorithme génétique si l'on change ses paramètres. Enfin, la seconde partie de ce chapitre sera consacrée à la discussion des résultats obtenus, sur les échecs et les améliorations possibles de ce projet.

4.1 Validation

4.1.1 Validation du modèle mathématique

Dans cette partie, nous allons étudier les résultats obtenus avec la programmation linéaire. Nous allons vous expliquer chacune des valeurs des variables pour nous assurer que le modèle réagit effectivement comme le fonctionnement de l'entreprise. Pour cela, nous allons reprendre le scénario 2. Ci-dessous, les matrices U_j et L_j.

Tableau 4.1 Matrices U_j et L_j du scénario 2

U						
	9	18	104	3	6	275

L						
	69	18	4	327	10	5

Avec ces deux matrices, nous pouvons en déduire les matrices binaires de U_j et de L_j, que nous avons appelées Ub et Lb. Ces deux matrices sont de taille le nombre de remorques par le nombre de commandes. Elles valent 1 si la commande j est déchargée du camion i, 0 sinon. Nous vous avons mis ci-dessous ces deux matrices. Elles vont nous être utiles pour faciliter les calculs du modèle.

Tableau 4.2 Matrices Ub et Lb du scénario 2

Ub	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
j1	0	0	0	0	0	0	0	0	1	0
j2	1	0	0	0	0	0	0	1	0	0
j3	0	0	0	1	0	0	0	0	0	1
j4	0	0	1	0	0	0	0	0	0	0
j5	0	0	0	0	0	1	0	0	0	0
j6	0	1	0	0	1	0	1	0	0	0

Lb	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
j1	0	0	0	0	0	1	0	0	1	0
j2	1	0	0	0	0	0	0	1	0	0
j3	0	0	0	1	0	0	0	0	0	0
j4	0	1	1	0	0	0	1	0	0	0
j5	0	0	0	0	0	0	0	0	0	1
j6	0	0	0	0	1	0	0	0	0	0

Nous avons calculé en avance les valeurs de $\gamma_{pp'}$. Pour rappel, $\gamma_{pp'}$ est une variable binaire égale à 1 si une commande est déchargée avant une autre. Étant donné que l'on a déjà ces informations dans la matrice U_j, on peut alors calculer $\gamma_{pp'}$, voir ci-dessous, le Tableau 4.3. Les valeurs de β_p sont elles aussi calculées en avance puisque connues grâce à la matrice U_j.

Tableau 4.3 Matrice de $\gamma_{pp'}$ pour le scénario 2

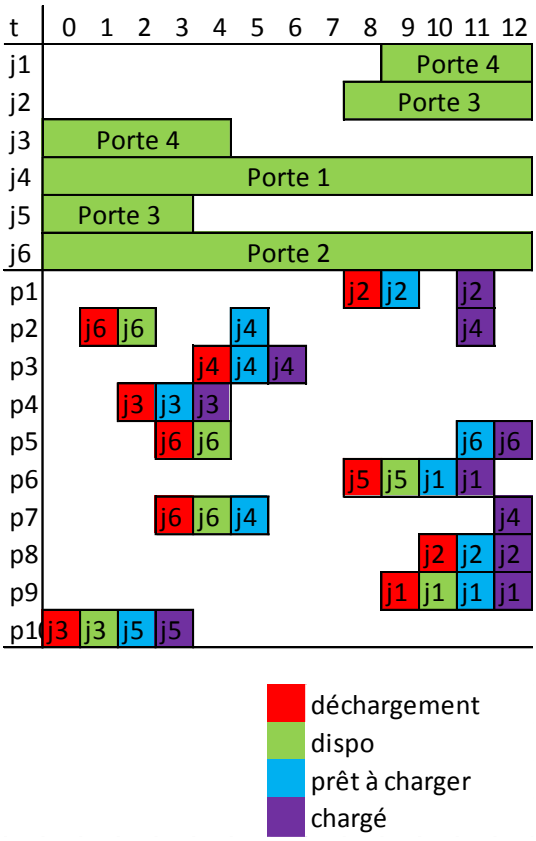
GAMMA	pp1	pp2	pp3	pp4	pp5	pp6	pp7	pp8	pp9	pp10
p1	0	0	0	0	0	0	0	1	0	0
p2	0	0	0	0	0	0	1	0	0	0
p3	0	1	0	0	0	0	0	0	0	0
p4	0	0	0	0	0	0	0	0	0	0
p5	0	0	0	0	0	0	0	0	0	0
p6	0	0	0	0	0	0	0	0	1	0
p7	0	0	0	0	0	0	0	0	0	0
p8	0	0	0	0	0	0	0	0	0	0
p9	0	0	0	0	0	0	0	0	0	0
p10	0	0	0	0	0	0	0	0	0	0

Il faut aussi rajouter au modèle les données concernant le temps mis pour aller d'une porte à une autre. Pour simplifier les tests, nous avons décidé de mettre une durée unitaire pour aller de la porte 1 à 2, de deux unités de temps pour se déplacer de deux portes et de trois pour aller de la porte 1 à la porte 4. Enfin, les dernières données du problème à connaître avant de le résoudre, sont les valeurs de A et D qui correspondent aux heures d'arrivés et de départ des

camions. Encore une fois, pour simplifier les tests, nous avons fixé toutes les valeurs de A à zéro et celle de D à 50, valeurs non contraignantes.

La résolution avec LINGO nous permet de connaître les valeurs de a_j, c_j, λ_p, μ_p, σ_p ainsi que celle de $\delta_{pp'}$, x_{jm} , $y_{jj'}$, $v_{jmj'm'}$ et aussi C_{max}. Les cinq premières variables sont très intéressantes pour vérifier si le modèle est correct puisqu'elles correspondent aux mouvements des palettes et des camions dans l'entrepôt.

Tableau 4.4 Représentation graphique des résultats du scénario 2



	1	2	3	4	5	6	7	8	9	10
a	9	8	0	0	0	0				
c	12	12	4	12	3	12				
Lambda	9	2	5	3	4	9	4	11	10	1
Mu	9	5	5	3	11	10	5	11	11	2
Sigma	11	11	6	4	12	11	12	12	12	3

Tout d'abord, avec les valeurs de a_j, on peut voir que les remorques 1 et 2 n'ont pas été affectées dès le début à l'entrepôt. En effet, nous avons 6 remorques pour 4 portes ce qui correspond bien aux résultats. Deuxième point de contrôle, le temps pour changer de camions à la porte 3 et 4, c'est-à-dire entre les remorques 5 et 3 et 4 et 1, sont bien de 5 minutes ce qui

correspond à l'équation (2.2). Aussi, toutes les valeurs de λ_p sont supérieurs à l'heure d'arrivée de leur remorque plus leur ordre de déchargement, soit le respect de l'équation (2.3). On peut vérifier qu'entre la valeur de λ_p et de μ_p, on retrouve le temps pour déplacer la commande. Par exemple, la commande 7 se déplace de 1 porte (5-4) et en effet elle passe de la remorque 6 à la 4 qui sont respectivement affectées aux portes 1 et 2. On peut observer aussi que toutes les valeurs de σ_p sont supérieures à celles de μ_p + 1. Cela correspond à l'équation (2.7). Enfin, la valeur de C_{max} correspond bien à la plus grande valeur de c_j, comme l'indique l'équation (2.11).

4.1.2 Comparaison entre programme linéaire et algorithme génétique

Premièrement, nous voulons comparer les performances entre notre algorithme génétique et la programmation linéaire. Pour cela, nous avons pris cinq scénarios de tailles relativement faibles pour pouvoir être résolus par LINGO dans des temps acceptables. Les matrices U_j et L_j des cinq scénarios sont présentes en ANNEXE V de ce document.

L'ensemble des tests ont été réalisés sur le même ordinateur pour avoir des résultats comparables. Voici les caractéristiques principales de l'ordinateur, Intel coreI5-3230m cadencé à 2.60Ghz avec 8 Go de mémoire vive.

Le Tableau 4.5 regroupe l'ensemble des résultats obtenus avec les deux méthodes pour des petits problèmes. On peut voir que sur de petites instances, l'algorithme génétique et LINGO trouvent les mêmes résultats concernant la fonction objective. Ce qui est intéressant à voir ici c'est les temps de calcul. On peut remarquer que sur les quatre premiers scénarios, LINGO est plus rapide que notre algorithme génétique. La programmation est jusqu'à trois fois plus rapide. Toutefois, avec les données du scénario 5, le problème le plus complexe, on voit clairement que la programmation linéaire n'est plus suffisante. Le temps de calcul est plus de 5000 fois supérieur à celui de l'algorithme génétique. On peut constater que le nombre de variables dépasse 2000 et que le nombre de contraintes est supérieur à 10 000.

Tableau 4.5 Comparaison des temps de calcul

Scénario	Nombre de portes	Nombre de camions	Nombre de commandes	Meilleur individu	Temps calcul LINGO	Temps calcul GA
1	2	2	4	5	0,14 sec	0,34 sec
2	4	6	10	12	1,97 sec	7,85 sec
3	4	8	12	12	5,76 sec	14,63 sec
4	6	8	15	13	20,48 sec	30,16 sec
5	4	10	18	20	1h 39m 25s	1m 02,5 s

Étant donné que les problèmes de l'entreprise sont bien plus grands que le scénario 5, nous excluons la programmation linéaire de nos solutions en raison de son trop long temps de calcul.

4.1.3 Étude de sensibilité de l'algorithme génétique

Dans cette partie, nous allons vous présenter les différents tests faits sur les paramètres de notre algorithme génétique ainsi que sur les paramètres du modèle. Étant donné le caractère aléatoire d'un algorithme génétique, nous avons réalisé chacun des tests trois fois. Les résultats obtenus sont donc le temps de calcul moyen des trois tests et la performance moyenne du meilleur individu des trois tests.

L'ensemble des tests de cette partie sont présents en ANNEXE V de ce rapport. Ce test compte 10 portes, 20 remorques et une seule combinaison d'affectation. Le nombre de solutions possibles est donc de 20!, soit $2{,}43.10^{18}$, 2,43 milliards de milliards de chromosomes différents. Étant donnée la taille du problème, nous n'avons pas pu trouver sa solution optimale avec LINGO. Nous supposerons que le meilleur résultat de tous les tests est la solution optimale. Les tests de cette partie s'organisent de la façon suivante :

- Variations du nombre d'individus et du nombre d'itérations.
- Variations des probabilités de croisement et de mutation.
- Variations du nombre de parents sélectionné par élitisme et par tournoi.
- Variations du nombre de portes, du nombre de remorques et du nombre de commandes.

4.1.3.1 Sensibilité du nombre d'enfants et d'itérations

Pour ce premier test, nous avons fixé les valeurs suivantes : probabilité de croisement à 80 %, probabilité de mutation à 20 %, 30 % de la population est sélectionnée pour la reproduction par élitisme et 30 % par tournoi. Nous avons utilisé le scénario 6. Nous avons fait varier le nombre d'enfants, qui correspond au nombre d'individus par itération, et le nombre d'itérations parmi les valeurs suivantes : 10, 30, 50 et 100. À titre indicatif, les plus grandes instances de ce test créent et testent 3000 chromosomes différents. Ces chiffres sont obtenus en multipliant le nombre d'individus par le nombre d'itérations. Cela ne représente qu'environ 0,000 000 000 000 1 % de l'ensemble des solutions possibles. Le Tableau 4.6 regroupe les résultats obtenus. À chaque fois, les résultats sont les moyennes des trois tests réalisés. En premier, nous avons la performance moyenne du meilleur individu et en dessous, nous avons le temps de calcul moyen.

Tableau 4.6 Analyse de sensibilité du nombre d'individus et d'itérations

Nombre d'individus	Nombre d'itérations			
	10	30	50	100
10	103,33 00:13:00	106 00:30:48	78,33 00:48:29	73,67 01:42:34
30	80 00:33:39	62,33 01:16:51	61,67 02:19:52	54,67 04:31:58
50	76,33 01:02:35	50,67 02:14:40	53 03:50:30	
100	62 01:54:13	51,67 04:21:45		

Avec les résultats obtenus, on peut faire deux remarques. Premièrement, avec un nombre d'individus identiques, plus on augmente le nombre d'itérations et plus la performance sera faible et le temps de calcul élevé. De même, si l'on fixe le nombre d'itérations et en augmentant le nombre d'individus, on obtient une meilleure performance, mais le temps de calcul est plus élevé. Deuxièmement, les temps de calcul semblent directement liés au nombre d'individus total créé. Par exemple, les temps de calcul avec 10 individus et 30 itérations sont très semblables à ceux obtenus avec 30 individus et 10 itérations. Aussi, si on double le nombre

total d'individus, on double le temps de calcul. On peut le constater, entre autres, lorsque l'on passe de 50 à 100 individus avec 30 itérations : on passe de 2 heures 14 minutes à 4 heures 21 minutes. À l'inverse, la performance se semble pas liée au nombre d'individus total. Lorsque l'on double le nombre d'individus ou d'itération, on ne divise pas par deux la performance. Aussi, le caractère aléatoire d'un algorithme génétique explique les écarts et la non-linéarité des résultats obtenus malgré le fait que nous avons réalisé trois tests pour chaque instance.

Les tests en rouge correspondent aux tests se situant sur le front de Pareto, voir Figure 4.1. Ces tests ont obtenu les meilleurs compromis entre performance et temps de calcul. Pour les départager, il faut qu'une personne choisisse si elle veut privilégier une solution rapide, mais peu performante ou bien une solution très performante, mais très lente à calculer. Pour la suite des tests, nous avons décidé de garder les paramètres suivants : 30 individus et 30 itérations. La raison est que cette configuration se situe sur le front de Pareto et que le temps de calcul nous semble raisonnable pour l'appliquer à l'entreprise.

Nous avons aussi gardé le meilleur individu de chacune des itérations pour tous les tests. Avec ces données, on peut vérifier que l'algorithme converge bien vers un minimum. On peut aussi analyser la vitesse à laquelle il converge et de vérifier s'il n'y a pas trop d'itérations. Sur l'exemple ci-dessous Figure 4.2, on peut voir que la courbe rouge converge plus rapidement que la courbe bleue. Cela s'explique du fait qu'il y a plus d'individus créés avec la courbe rouge ce qui revient à explorer plus de solutions. Aussi, on peut remarquer que les deux courbes ne diminuent que très peu après une trentaine d'itérations. C'est à ce moment qu'il faut choisir : veut-on, peut-on doubler voire quadrupler le temps de calcul pour une amélioration minime de la solution ? L'ensemble des courbes de convergence sont en ANNEXE VI de ce rapport.

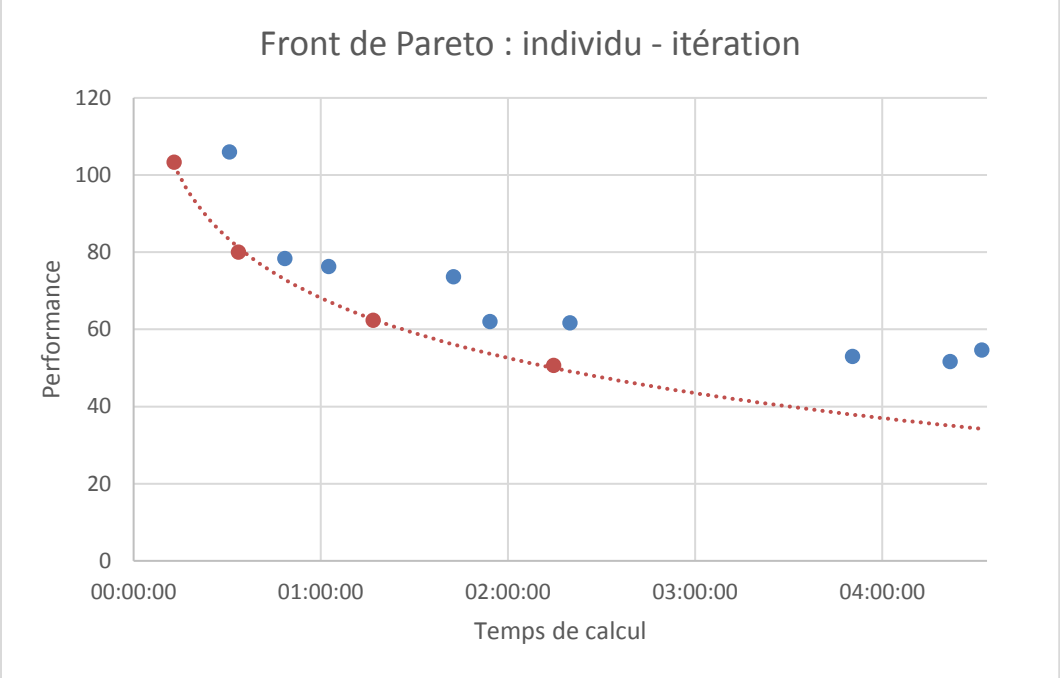


Figure 4.1 Front de Pareto en fonction du nombre d'individus et d'itérations

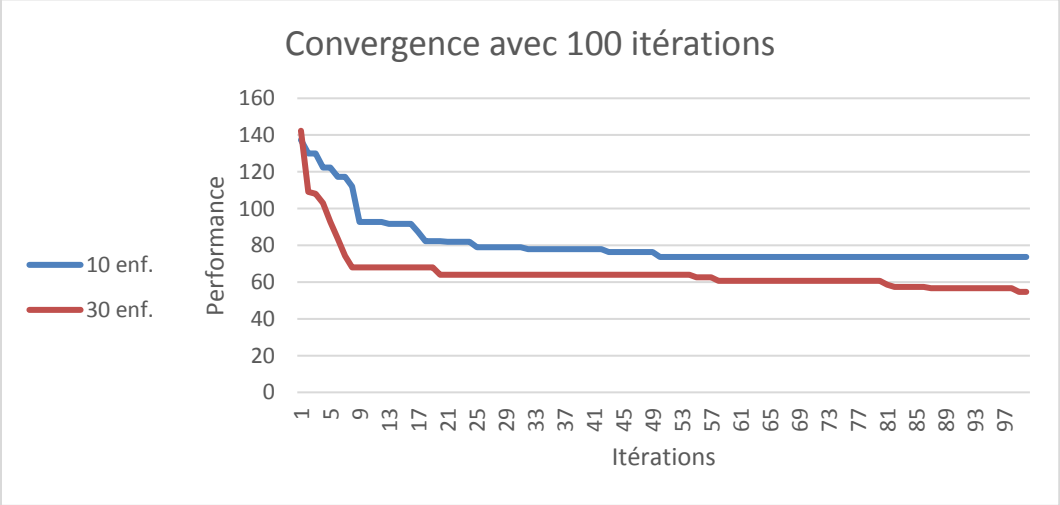


Figure 4.2 Convergence avec 100 itérations

4.1.3.2 Sensibilité des probabilités de croisement et de mutation

Notre deuxième test concerne les probabilités de croisement et de mutation. Nous reprenons le meilleur cas du test précédent, c'est-à-dire 30 individus et 30 itérations. Nous avons utilisé le

scénario 6. Pour ce nouveau test, nous avons pris les probabilités suivantes : 20 %, 50 % et 80 %. Pour rappel, ce pourcentage correspond au taux de réussite d'un croisement ou d'une mutation entre deux parents sélectionnés. Cela n'influe pas le nombre d'enfants créés. Donc, plus le pourcentage est faible et plus il faudra de tentatives pour réaliser un croisement ou une mutation ; et, à l'inverse, plus le pourcentage est élevé et plus les croisements et les mutations auront lieu facilement.

On peut faire l'hypothèse que, pour le croisement ou pour la mutation, plus la probabilité est faible et plus le temps de calcul sera élevé étant donné qu'il faudra sélectionner plus de couples pour créer des enfants et inversement, plus la probabilité est faible et plus le temps de calcul sera faible. Ces paramètres ne devraient pas affecter la performance moyenne.

Tableau 4.7 Analyse de sensibilité des probabilités de croisement et de mutation

	Mutation		
Croisement	20 %	50 %	80 %
20 %	78,67 01:34:07		
50 %	66 01:25:33		
80 %	62,33 01:16:51	71,67 01:26:01	69,33 01:34:21

Le Tableau 4.7 ci-dessus correspond aux résultats que nous avons obtenus. En premier, la performance moyenne des trois meilleurs individus et en dessous, le temps de calcul moyen sur les trois tests. Nous n'avons fait varier qu'une seule probabilité à la fois. On peut voir que le cas avec 80 % de croisement et 20 % de mutation est le meilleur paramétrage, que ce soit pour le temps de calcul et la performance.

On remarque aussi que notre hypothèse sur l'augmentation du temps de calcul lorsque la probabilité diminue est valide pour le croisement, mais pas du tout pour la mutation. À part le côté aléatoire de l'algorithme, nous n'avons pas d'autre explication face à ces résultats. Les courbes de convergences présentes en ANNEXE VI n'apportent pas plus d'informations.

Pour la suite des tests, nous utiliserons une probabilité de croisement de 80 % et une probabilité de mutation de 20 %.

4.1.3.3 Sensibilité de la sélection par élitisme et par tournoi

Pour ce troisième test, nous avons fait varier le nombre d'individus sélectionnés par élitisme et par tournoi pour la reproduction. Nous avons utilisé les valeurs suivantes : 10 %, 30 % et 50 % du nombre d'individus, qui est de 30 pour ce test. Pour rappel, si l'on a 10 % des individus sélectionnés par élitisme, c'est-à-dire que l'algorithme réalisera la sélection par élitisme jusqu'à avoir sélectionné 3 individus. Nous avons utilisé le scénario 6.

Première hypothèse, que l'on modifie le paramètre pour l'élitisme ou pour le tournoi, les deux modes de sélection reposent sur l'aléatoire. Aucun des deux n'est plus performant que l'autre. Le seul impact que nous pensons obtenir est si la somme des probabilités varie, ce qui reviendrait à sélectionner plus ou moins d'individus pour la reproduction.

Notre deuxième hypothèse est que si l'on augmente le nombre d'individus pour la reproduction, étant donné que l'on ne peut pas sélectionner plus d'une fois le même individu, on risque d'avoir sélectionner des individus peu performants qui vont engendrer des enfants eux aussi peu performants. À l'inverse, si le nombre de parents sélectionné est faible alors, grâce au tournoi et à l'élitisme, ces individus seront statistiquement les meilleurs de leurs itérations et engendrons des enfants encore plus performants.

À noter que pour le cas de figure où les deux sélections sont à 50 %, soit un total de 100 % n'est pas réalisable. En effet, l'algorithme n'arrive pas à sélectionner la totalité des individus puisque, par définition, le tournoi qui est le deuxième test de sélection sélectionne le meilleur de trois individus choisis au hasard et recommence si celui-ci est déjà sélectionné. Les deux derniers individus ne seront donc jamais sélectionnés et l'algorithme répète l'opération à l'infini. Nous avons donc corrigé cela en prenant 50 % des individus moins un pour les deux tests. On a donc 28 individus sur 30 de sélectionnés, ce qui est réalisable pour l'algorithme.

Tableau 4.8 Analyse de sensibilité de la sélection par élitisme et par tournoi

Élitisme	Tournoi		
	10 %	30 %	50 %
10 %	62,33 01:27:09	61,33 01:18:01	56 01:22:00
30 %	59 01:20:59	62,3333333 01:16:51	59 01:20:51
50 %	63,33 01:21:43	57 01:25:33	63 01:29:14

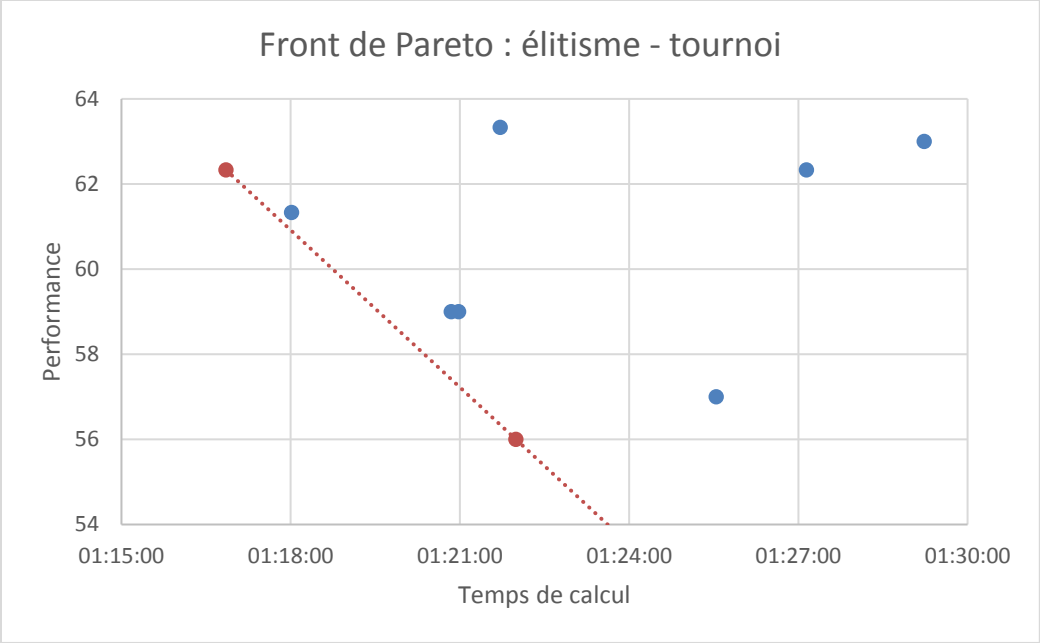


Figure 4.3 Front de Pareto en fonction de l'élitisme et du tournoi

Les résultats obtenus dans le Tableau 4.8 présentent la performance et le temps de calcul moyen. La première remarque que nous faisons est que l'ensemble des résultats est très similaire. Les temps de calcul varient entre 1 heure 14 et 1 heure 31 sur l'ensemble des 27 instances testées, soit seulement 18 % d'écart relatif entre la plus forte et la plus faible valeur. De même pour les performances qui varient entre 76 et 51, soit seulement 30 % d'écart relatif. On peut aussi voir sur la Figure 4.3 que les résultats sont assez proches les uns des autres. La vitesse de convergence est aussi similaire comme on peut le voir sur les courbes présentes en ANNEXE VI. Les deux valeurs en rouge forment un front de Pareto. Aucune des deux

hypothèses que nous avons faites n'est vérifiable avec les résultats obtenus. Pour la suite des tests, nous avons choisi de prendre 10 % des individus avec l'élitisme et 50 % avec le tournoi pour deux raisons, premièrement, cette solution se trouve sur le front de Pareto et deuxièmement elle a obtenu les meilleures performances.

4.1.3.4 Sensibilité des paramètres du problème

Enfin, le dernier test veut analyser comment les dimensions du problème affectent le temps de calcul et la performance obtenue. Pour cela, nous allons prendre un problème avec 10 portes, 20 remorques et 50 commandes et reprendre les meilleurs paramètres obtenus précédemment : 30 individus, 30 itérations, une probabilité de croisement de 80 % et de mutation de 20 % et une sélection des parents à 10 % par élitisme et à 50 % par tournoi. Pour ces tests, nous allons diviser par deux et doubler successivement le nombre de portes, de remorques et de commandes, soit respectivement, 5 et 20, 10 et 40 et 25 et 100.

Les résultats obtenus sont dans le Tableau 4.9 sur la page suivante. Les en-têtes correspondent respectivement au nombre de portes, au nombre de remorques puis au nombre de commandes. Toujours pour éviter le côté aléatoire des solutions, nous avons réalisé trois fois chacun des tests et les résultats sont les moyennes des trois tests. À chaque fois, la colonne du milieu correspond au problème de base. Les courbes de convergence sont en ANNEXE VI.

Le premier tableau correspond à une variation sur le nombre de portes. Lorsque le nombre de portes est divisé par deux, c'est-à-dire que le nombre moyen de camions par porte augmente, on voit que la performance est plus importante. Cela s'explique par le fait que la file d'attente est plus longue devant chaque porte et que donc le temps pour traiter tous les camions augmente. À l'inverse, si l'on double le nombre de porte, cela revient à avoir plus de service pour la file d'attente. La performance est donc basse puisque les camions sont rapidement pris en charge. De plus, on a un nombre de portes égal au nombre de camions, ce qui rend le problème trivial puisque l'affectation aux portes n'est plus un problème.

Tableau 4.9 Analyse de sensibilité selon les paramètres du problème

Portes	5-20-50	10-20-50	20-20-50
Performance	117,67	62,33	15,33
Temps	00:41:07	01:16:51	00:47:24
Scénario	7	6	8

Remorques	10-10-50	10-20-50	10-40-50
Performance	12,33	62,33	144
Temps	00:13:06	01:16:51	02:13:33
Scénario	9	6	10

Commandes	10-20-25	10-20-50	10-20-100
Performance	34,33	62,33	≈180
Temps	00:15:11	01:16:51	> 4h
Scénario	11	6	12

Le deuxième paramètre que nous avons fait varier est le nombre de camions. En le divisant par deux, on retrouve le même cas de figure qu'avec la configuration 20-20-50. C'est-à-dire autant de portes que de camions, ce qui revient à un problème trivial ayant une performance faible. La seule chose qui change est le temps de résolution. En effet avec la configuration actuelle, on n'a que 10! solutions contre 20! pour le test précédent et il faut prendre en compte 20 portes au lieu de 10 pour minimiser les distances parcourues par les marchandises.

Les résultats du scénario 10 sont aussi similaires à ceux obtenus au scénario 7. Puisque le rapport entre le nombre de porte et de camions est le même, on retrouve le même ordre de grandeur pour la performance. Encore une fois, le temps a doublé puisque le nombre de camions et de remorques a lui aussi doublé.

Le dernier tableau correspond aux modifications sur le nombre de commandes. On remarque que plus le nombre de commandes est important et plus le problème est long à résoudre et la performance augmente. La raison évidente à cela est qu'avec plus de commandes, il a y plus d'activités dans l'entrepôt et que la modélisation de tous les mouvements des palettes devient très complexe et rend encore plus complexe l'affectation des remorques aux portes.

4.1.3.5 Conclusion

En regroupant l'ensemble des résultats obtenus, nous savons que notre algorithme est le plus optimal avec les paramètres suivants : une probabilité de croisement de 80 % et de mutation de 20 % et une sélection des parents à 10 % par élitisme et à 50 % par tournoi.

Nous pouvons aussi dire que le nombre d'enfants et le nombre d'itérations affectent le nombre de solutions visitées par l'algorithme. Plus l'on teste de solutions et plus la chance de trouver l'optimum global augmente, mais le temps de calcul augmente lui aussi. Il faut donc trouver un compromis. Les probabilités de croisement et de mutation ainsi que le nombre d'individus sélectionnés par élitisme et par tournoi affectent peu l'algorithme. Enfin, le temps de calcul est directement lié au ratio entre le nombre de camions et le nombre de portes. Plus ce nombre est élevé est plus le temps de calcul augmente. Dernier paramètre, le nombre de commandes, affecte-lui aussi directement le temps de calcul ainsi que la performance de solution trouvée. De plus, l'augmentation du temps de calcul n'est pas proportionnelle à l'augmentation du nombre de commandes, mais plutôt exponentiel. Cela peut devenir inquiétant pour la suite puisque nous savons qu'il y a environ 8,5 commandes par remorque dans le cas réel.

4.2 Résolution d'un scénario réaliste

Maintenant que nous connaissons les meilleurs paramètres pour notre algorithme génétique, nous pouvons passer à la résolution d'un problème de taille industriel. Nous avons justement les données de l'entreprise pour ce type de test. Nous avons aussi développé un script capable de lire ces fichiers, d'extraire les informations importantes et de les mettre en forme pour notre algorithme.

Préalablement, nous avons réalisé des tests avec des scénarios avec le même rapport entre le nombre de portes et le nombre de camions ainsi qu'entre le nombre de camions et le nombre de commandes. Avec les données de l'entreprise, le problème comporte 66 camions pour 40 portes et 564 commandes. Nous avons pris trois scénarios, le premier à 25 % de la taille réelle, le deuxième à 50 % et le troisième à 75 %, voir le Tableau 4.10. Nous avons cette fois-ci mis

les vraies données pour les temps de déplacement et attribué une quantité de palettes par commande entre 1 et 5. Cela fonctionne, mais ralentit l'algorithme.

Tableau 4.10 Paramètres pour les tests liés au cas réel

Ratio	Camions	Portes	Commandes
25 %	16	10	141
50 %	33	20	282
75 %	49	30	423
100 %	66	40	564

Nous avons lancé le premier cas de figure, à 25 % du problème réel. Nous avons vu que dans ce cas-là, l'algorithme met environ une minute pour évaluer la performance d'un seul individu. Sachant que l'on crée 30 individus par itération et que l'on a 30 itérations, on crée donc 900 individus, soit environ 15 heures de calcul. Durant le calcul, on a vu que le temps que l'algorithme à besoin pour réaliser une itération est entre 21 et 25 minutes, soit entre 10,5 et 12,5 heures de calcul total. Cela confirme nos inquiétudes sur la lenteur de l'algorithme pour des problèmes de taille industrielle.

Nous avons laissé calculer l'algorithme. Il a fini de calculer et a proposé une solution après 11 heures 31 minutes en moyenne et affiche des résultats de l'ordre de 430 minutes pour réaliser l'ensemble des chargements du scénario. Les courbes de convergence sont disponibles en ANNEXE VI. Le temps nécessaire pour générer et évaluer un individu dans le cas à 50 % est de 6 minutes et de 20 minutes dans le cas à 75 %. Cela représente 90 heures de calculs soit presque 4 jours pour 50 % et de plus de 12 jours de calcul pour le cas à 75 %. Notre algorithme n'est définitivement pas adapté pour résoudre des problèmes qui se renouvellent quotidiennement.

Enfin, le scénario que nous avons utilisé considère que toutes les palettes sont dans des camions en attente dans la cour. Nous n'avons pas mis en place notre raisonnement pour pouvoir modifier l'emplacement des palettes à l'instant initial du problème. La raison est que nous n'avons pas réussi à intégrer ces modifications au fonctionnement actuel de l'algorithme. Nous

avons eu aussi plusieurs problèmes avec les fichiers de l'entreprise comme des problèmes d'encodages, certains caractères n'étaient pas au format UTF-8, ainsi que le manque d'information ou l'absence de certaines commandes diminuent encore la véracité du résultat obtenu.

4.3 Travaux futurs

Il serait intéressant de continuer les travaux sur certains points pour permettre à ce projet d'être pleinement utile et fonctionnel pour l'entreprise partenaire et aussi pour faire avancer la recherche. Nous allons lister dans cette partie, les étapes clés pour améliorer ce projet.

Tout d'abord, il faudrait continuer d'améliorer le modèle et notamment d'étudier de façon plus approfondie les points suivants :

- Ajouter la préemption dans le modèle.
- Limiter la quantité de stock comme c'est le cas pour l'entreprise.
- Limiter le nombre de manutentionnaires et mieux gérer leurs affectations.
- Rajouter un temps de chargement et de déplacement selon le type de conditionnement.
- Rajouter des remorques vides disponibles pour les chargements.
- Éviter de décharger des commandes si elles restent et repartent dans la même remorque.
- Gérer l'emplacement des marchandises à l'instant zéro (en stockage, dans un camion affecté à une porte ou dans la cour)

Les données réelles que nous avons collectées et que l'entreprise nous a transmisses présentent plusieurs imperfections. Il serait intéressant, pour améliorer la précision de l'algorithme, de chronométrer le temps de déplacement réel entre chaque porte directement à partir d'un chariot élévateur et de rajouter ces nouvelles données aux historiques que nous avons. À propos des fichiers textes, il serait très pratique pour l'entreprise de pouvoir anticiper le manque d'informations sur l'emplacement des marchandises pour pouvoir répondre à son fonctionnement durant la semaine.

Concernant l'algorithme génétique, il y a plusieurs aspects à améliorer. Premièrement, la forme de notre chromosome ne permet pas de représenter toutes les possibilités d'affectations. Il serait intéressant de voir la performance d'un chromosome parcourant toutes les solutions sans pour autant créer de doublons. Aussi, l'aspect des heures de départs et d'arrivées sont présent dans le modèle et dans l'algorithme, mais nous n'avons pas fait de tests avec. La raison est que ces contraintes sont inflexibles et apportent beaucoup d'instabilité dans le modèle. Ensuite, il faudrait revoir le script qui correspond au modèle mathématique dans le fichier utils_FF2.py. Certaines fois, il propose de meilleures solutions, plus performantes que celles obtenues avec LINGO. Ce qui n'est pas réaliste. Aussi, il pourrait être question d'un gain de temps très important si ce même script est repensé et amélioré. En effet, nous avons constaté que la majorité du temps de calcul est consommée par la fonction fv_contraintes. Cela permettrait de traiter des problèmes réalistes dans des temps acceptables et de pouvoir ajouter différents aspects non pris en compte comme les temps d'arrivée et de départ des camions.

CONCLUSION

Les objectifs de ce projet étaient d'étudier le fonctionnement d'entrepôts de cross-docking, de modéliser le fonctionnement de l'entreprise partenaire et de développer un algorithme capable de résoudre ces problèmes d'affectation. Les deux premiers objectifs nous semblent remplit grâce aux explications détaillées du fonctionnement d'un cross dock et à la revue de littérature ainsi qu'à l'élaboration de notre modèle mathématique. Il reste encore de nombreuses pistes d'améliorations pour le modèle si l'on veut être le plus fidèle possible au fonctionnent de l'entreprise. Nous nous sommes arrêtées lorsque le modèle correspondait, dans son ensemble, à l'entreprise et là où nos compétences nous limitaient. Concernant le troisième objectif, nous pouvons dire que l'algorithme développé est fonctionnel et utilisable pour de petits problèmes. Il nous faudrait néanmoins plus de temps et l'aide de programmeur pour pouvoir continuer son développement et optimiser son fonctionnement.

Personnellement, ce projet a été très enrichissant sur plusieurs aspects. Premièrement, la méthodologie suivie dans ce projet très axé sur l'étude de la littérature et sur les travaux déjà réalisés dans le domaine. J'ai passé presque trois mois à étudier des articles scientifiques, ce qui m'a permis de ne pas passer directement dans la phase de développement, de comprendre les différents enjeux et de me resservir des travaux passés. Deuxièmement, ce projet a été riche en termes de programmation informatique. J'ai appris, durant la réalisation de ce projet, les langages de programmation LINGO et Python. Je me suis aussi perfectionné en VBA en réalisant les liens avec le modèle sous LINGO. Une piste d'amélioration majeure pour ce projet est de continuer le programme informatique avec une équipe de développeurs. Cela créerait une synergie entre les membres et pourrait aboutir à un algorithme efficace et à son implantation chez l'entreprise. Enfin, j'ai trouvé très intéressante l'interaction entre le côté scientifique et le côté entreprise. J'ai apprécié à avoir eu à réfléchir sur un problème réel et de trouver des solutions concrètes aux problèmes de l'entreprise.

ANNEXE I

QUESTIONNAIRE POUR DÉFINIR LE TYPE DE CROSS-DOCK

Caractéristique du cross dock

- **Secteur d'activité**
  - Clients
  - Type de conditionnement
  - Type de produits (frais/surgelés, matières dangereuses)

- **Données de la plateforme**
  - Forme : I L U T H E
  - Surface
  - Nombre de docks
  - Nombre d'employés :

	Temps plein	*Temps partiel*
Permanents	...	...
Temporaire	...	...

  - Horaire d'ouverture
  - Horaires d'équipes
  - Horaires de réception/expédition
  - Zone d'attente : □ Oui : Taille □ Non
  - Certifications

- **Flux**
  - Qualification de flux : □ Tendus □ Stockés
  - Quantifications des flux : □ Gros □ Moyen □ Faibles
  - Diversification des flux : □ Très variés □ Moyen □ Réduits
  - Volumes annuels
  - Volumes quotidiens
  - Variabilité
  - Nombre de camions/jours

- **Mode de service**
  - □ Une destination / porte
  - □ Mixte
  - □ Entrée ou Sortie : combien :
  - □ Restriction / porte

- **Transport interne**
  - ☐ Manuel
    Res. Matérielles
    Res. Humaines
    (skills,licenses)
  - ☐ Automatique
    Cadence, fiabilité
  - ☐ Mixte

- **Système d'information**
  - Nom
  - Développement : □ Interne □ Externe
  - Support : □ Interne □ Externe
  - Performance, écart avec le réel, rapidité

Contraintes

- **Temps de déplacement d'une porte à l'autre**
  - ☐ Négligeable
  - ☐ Distance
    Eucl., Manhattan, routes, …
  - ☐ Congestion
  - ☐ RH disponibles
  - ☐ Engins disponibles

- **Planification des ressources**
  - Humaines : □ Tâches/métiers □ Multi-skills □ Horaires flexibles
  - Matérielles : □ Nombre □ Disponibilité

- **Heures d'arrivées des camions entrants et sortants**
  - ☐ Tous en même temps/concentration □ Répartition régulière
  - ☐ Heures estimées : □ Heures exactes □ Tranche horaire

- **Gestions des aléas, des retards ?**

- **Déchargement - Chargement**
  - Composition : □ Mono-Produits □ Multi-Produits
  - Tps de (un)load : □ dépend du camion □ constant □ entre 2 bornes
  - Départs des camions : □ non contraints □ borne sup □ entre 2 bornes

- **Interchangeabilité des produits**

□ Assignée à une destination
□ Assignée à un camion

- **Opérations à valeur ajoutée**
  □ Tri □ Lot sizing □ Etiquettage □ Packaging □ Unpackaging
  ➢ Systématique ? Temps :

- **Stockage intermédiaire**
  □ Totalement interdit

	Au sol sur les quais	En zone de rétention	Sur racks
□ Après chargement :	□	□	□
□ Avant rechargement :	□	□	□

  ➢ Capacités, Coût, Temps, Conséquences si congestion

- **Préemption**
  □ Oui : Coût de changement □ Non

- **Heures de départ des camions sortants**
  □ Non contraintes
  □ Entre 2 bornes □ Borne sup
  □ Dès que chargés □ À heure fixe, pas chargés □ À heure fixe + chargés

Objectifs

- **Mesures des performances actuelles**
  □ Makespan □ Somme pondérée des tps de prise en charge
  □ Retard maximum □ Somme pondérée des retards
  □ Total stocké □ Niveau de stock max
  □ Coups de touches □ Mouvements de camions
  □ Engorgement □ Heures de travail
  □ Utilisation des docks □ Produits non chargés

- **Besoins**
  ➢ Points difficiles, délicats, bloquants dans la planification ?

ANNEXE II

PSEUDO-CODE POUR LES DONNÉES

Début « Lecture et création des données »
Choix des données par l'utilisateur (=1 : scénarios, =2 : aléatoire, =3 : CSV)
Début de la fonction read_data :
Aleatoire = 0
Si choix de données = 1
    Définir nb_porte, nb_mont, nb_dech, nb_cmd, u, l, tc, qte, choix porte
Sinon si choix de données = 2
    Définir nb_porte, nb_mont, nb_dech, nb_cmd, tc, qte, choix porte et aleatoire = 1
Sinon si choix de données = 3
    Définir nb_porte, tc
    Lire le fichier texte voyage
    Définit les tailles des matrices l, beta et qte
    Créer et remplir les matrices l, beta et qte avec le fichier
    Création des matrices u1, u2, u3 et u4
    Remplissage de u1 avec le fichier remorque (palettes dans la cour)
    Remplissage de u2 avec le fichier portes (palettes dans camions à quais)
    Conversion des rangées en portes avec la fonction dédiée
    Remplissage de u3 et de u4 avec le fichier voyage (palettes dans l'entrepôt et absentes)
    Suppression des lignes et des colonnes de zéros
    Concaténation de u1, u2, u3 et u4 en u
    Renvoi nb_porte, tc, nb_cmd, nb_mont, l, u, nb_dech, qte, beta
Fin de la fonction read_data
Début de la fonction create_data
Calcul nb_trailer, d (1. Données de tests, 2. Données réelles) et q
Si aléatoire = 1
    Création de u et de l au hasard
Calcul de ub, lb, beta et gamma
Renvoie de nb_trailer, d, q, u, ub, l, lb, beta, gamma
Fin de la fonction create_data
Fin « Lecture et création des données »

ANNEXE III

PSEUDO-CODE POUR L'INITIALISATION

Début « Initialisation »
Définir le nombre d'enfants
Début de la fonction initialisation
Début de la fonction create_enfant
Boucle for de i allant de 1 au nombre d'enfants
    Construit b, la partie avec l'ordre des camions
    Construit c, la partie avec l'ordre des portes
    Concatenate b et c
    Stock l'individu créer dans all_ind
Fin de la boucle i
Fin de la fonction create_enfant
Début de la fonction fitness_function
Boucle for de i allant 1 au nombre d'enfants
    Début de la fonction fv_variable
    Créer et calcul les variables delta, x, y et v avec les valeurs du chromosome
    Fin de la fonction fv_variable
    Début de la fonction fv_contraintes
    Création des variables lam, mu, sig, cmax
    Boucle tant qu'il y a des modifications, faire
        Calcul des variables selon les contraintes 2, 3, 4, 5, 6, 7, 8, 9, 10
    Fin de la boucle tant que
    Contrainte 11
    Renvoi la valeur de cmax dans fv
    Fin de la fonction fv_contraintes
Fin de la boucle for
Fin de la fonction fitness_function
Début de la fonction tri
Tri les individus créer par ordre décroissant de la fitness value
Fin de la fonction tri
Fin de la fonction initialisation
Fin de « initialisation »

ANNEXE IV

PSEUDO-CODE POUR LES ITÉRATIONS

Début de « itérations »
Définir le nombre d'itérations max, les probabilités de croisement et de mutation
Définir le nombre de parents sélectionné avec la roulette et le tournoi.
Début de la fonction itération
Boucle tant que le nombre d'itérations max n'est pas atteint, faire
Début de la fonction selection_parent
Début de la fonction wheel_selection
Attribut la proba de sélection selon la fitness value des individus
Boucle tant que le nombre de parents sélectionné par la roulette n'est pas atteint, faire
Choisir un individu au hasard
Choisir une valeur de proba de sélection au hasard
Si la fitness value de l'individu sélectionnée est supérieur à la valeur alors
Si l'individu n'est pas déjà sélectionné pour la reproduction
L'individu est sélectionné pour faire partie des parents
Fin du si
Fin du si
Fin de la boucle tant que
Renvoi de la liste des individus sélectionnés
Fin de la fonction wheel_selection
Début de la fonction tournoi_selection
Attribut la proba de sélection selon la fitness value des individus
Boucle tant que le nombre de parents sélectionné par le tournoi n'est pas atteint, faire
Prendre à trois individus au hasard, a, b et c
Si a > b et a > c
Si a n'est pas déjà sélectionné
Alors l'individu a est sélectionner pour la reproduction
Fin boucle si
Si b > a et b > c
Si b n'est pas déjà sélectionné
Alors l'individu b est sélectionner pour la reproduction
Fin boucle si
Si c > a et c > b
Si c n'est pas déjà sélectionné
Alors l'individu c est sélectionner pour la reproduction

Fn boucle si
Fin boucle si
Fin de la boucle tant que
Renvoi de la liste des individus sélectionnés
Fin de la fonction tournoi_selection
Concatenate les deux listes de parents sélectionnés
Fin de la fonction selection_parent
Début de la fonction cross_mut
Sélectionne deux parents dans la liste
Choisir un chiffre au hasard
Si le chiffre est inférieur à la proba de croisement
Début de la fonction cross_over
Création de deux enfants en croisant les gènes des deux parents
Fin de la fonction cross_over
Si le nombre d'enfants < nombre d'enfant max
Stocker les nouveaux enfants
Fin de la boucle si
Fin de la boucle si
Si le chiffre est inférieur à la proba de mutation
Début de la fonction mutation
Création de deux enfants mutés à partir des deux parents
Fin de la fonction mutation
Si le nombre d'enfants < nombre d'enfant max
Stocker les nouveaux enfants
Fin de la boucle si
Fin de la boucle si
Début de la fonction fitness_function
Boucle for de i allant 1 au nombre d'enfants
Début de la fonction fv_variable
Créer et calcul les variables delta, x, y et v avec les valeurs du chromosome
Fin de la fonction fv_variable
Début de la fonction fv_contraintes
Création des variables lam, mu, sig, cmax
Boucle tant qu'il y a des modifications, faire
Calcul des variables selon les contraintes 2, 3, 4, 5, 6, 7, 8, 9, 10
Fin de la boucle tant que
Contrainte 11
Renvoi la valeur de cmax dans fv
Fin de la fonction fv_contraintes

Fin de la boucle for
Fin de la fonction fitness_function
Début de la fonction tri_coupe
Début de la fonction tri
Tri les individus créer par ordre décroissant de la fitness value
Fin de la fonction tri
Supprime les individus les plus mauvais selon leurs fitness value
Fin de la fonction tri_coupe
Affiche les résultats
Fin de la boucle tant que
Fin de la fonction itération
Affiche les résultats
Fin de « itérations »

ANNEXE V

DONNÉES DE TESTS

Scénario 1

U

3	1 4 2

L

3 4	1 2

Scénario 2

U

9	1 8	10 4	3	6	2 7 5

L

6 9	1 8	4	3 2 7	10	5

Scénario 3

U

11	10 6 3	7 1	12	5 8	4	2	9

L

11 9	10	7 1	12 4	5 8	2	3	6

Scénario 4

U

10	1 13	15 8	6	3 2	14 5	12 7 11	4 9

L

9 10 7	13 4	15 5	6	2 14	8 3	12	1 11

Scénario 5

U

1 17 11	18 12 15	5 3	7	8 10	2 9	14 13	16	6	4

L

1	18	3	13 7 8 2	14	9	5 15	16 10 12	6 11	4 17

Scénario 6

U

38 40 14	39 41	49 7	19 13	11 46 12	20 48 5 35	27 22	8 34 6 47 37	24	45 23	32	50 2	9 16	26 36	33 44	25 21 29	4 28	42 31 3 18	15 1 30	43 10 17

L

45 36 12	46 40 9 50	1 15	30	16 49	42 29	7	3 23 8	22 13	5	43 20 6 32	47 26 18	27 39 2	33 38	21 31	35 11 34 24 10	41	17 37 4 14 44 25	48 28	19

Scénario 7

U

38	22 32 26	16 11	50 34 30	8	31 20	1 18 24	28 48	37 27	4	9 36 40	39 2 14 21 12	49 42 43	47 35	44 41 23 15	17 10 3	13 7 45 25	6 33	46 19	5 29

L

33 8 29	37 27 28	24 44 49	36 23	25	10 5 35	32 14	41 7	12 39 34 45	42 17 3	15 19	43 38 48	13 22 47	6 18	26	20 21 11	31	46 50 4	1 9 30 2	40 16

Scénario 8

U

12 9 30	33 10	39 2 49 26	24 25 23 31	29 40	5	21 19 1 7 38 45	48 43	14	16 15 37 34	4	8 47	46 42	41 35 44	36 13 27	28 22	20	3 50	17 18	11 6 32

L

19 7 4 26 24	45 39	14	2 3	18 36 33 17 40	15 5	11 16	23	28	31 1 32 29	37	6 47 12	35 49 13 43	48	42 9	8 34 38	21	44 10 22 30 27	50	25 20 46 41

Scénario 9

U

17 49 45 13	3 25 41 23 2	1 26 48 36 32 35 21	47 28 29	11 46 9 27 39 6	14 31 19 20 43 7	5 34 37 40 24 42	12 33 22	8	4 10 44 50 15 18 38 16 30

L

24 46 36 27	38 2 29	43 25 50 26 19	10 11 8 48 14 12 9	17 23 28 4	35 21 31 6	44 41 15 22 49 39 45 37	1 33 13 47 18 3	5 32 42 40 30 34	16 20 7

Scénario 10

U

9	38 17	28	47	1	49	18 50	10	26	8	16	45 32 46 19	42	43	37	29 4	3	40 11	35	30	5	34	33	41

14 6	13	20	2	27	44 23	22	36	15	31	7 25	21	12	24	39	48

L

11	28	25	4	38	10	16	23 20	46	26 35	41 18	6 36	15	31	12	14	43	13	19 24	48	44	2 9	29 40	3

49	7	45 34	8	42	1	39	22	32	47	27 37	5	21	50	30	17 33

Scénario 11

U

20	14	16 3 8	6	11	17	21	22 9	4	15	19	18 10	24	2	1	12	25	5 23	7	13

L

1	11	17 10	23	25	20	9 18	12	7 15	21	2	24	8	19	3	4 14 6	13	16	22	5

Scénario 12 U

31	45	20	35	32	57	77	42	83	87	75	21	90	74	94	54	8	27	95	44
18	55	30	19	22	14	16	41		67	29	10	52	82	96	98	34	56	38	6
64	12	9	100	59	46	89	49		79	81	63	5	78	25	48	17	70	13	80
	47	43		62		60			76	11	65	61	72			37	91	99	84
	39	58		26		1				2	85	73	3			97	28	40	
	69	53				4				88	92		36			7		71	
	24	68				15					51		66			23		93	
											33					50			
											86								

L

93	54	31	14	70	92	36	71	99	79	78	8	46	86	51	41	66	50	76	67
44	34	40	59	65	77	11		94	47	90	49	61	45	89	30	1	73	62	72
84	5	88	29	85	37	20		22	9	13	56	75	53	48	25	64	100	18	
97	81	57	10	32	58	69		28	3	4	12	26	63	98	6	83			
15		7	87			42			21	38	82	95	24						
80						39			60	33		23	74						
27										91		16	68						
										43		17	52						
										35			55						
													2						
													19						
													96						

Scénario 13 U

32	72	5	115	89	64	103	68	109	88	74	129	67	97	93	76
58	16	135	42	20	86	8	45	120	83	69	140	25	63	84	18
133	31	19	119	125	79	15	117	80	123	99	82	98	141	106	30
9	27	56	28	46	44	110	40	39	111	132	10	107	47	21	121
116	94	114	3	130	26	100	122	65	85	43	51	113	4	1	59
134	70	38	13	105	22		138	127	60	108	41	131	6	112	57
53	118		126	81	137		136	7	71	61	101	62	52	11	55
87	24			2	73		17	48	77	35	12		14	75	34
124	29			66			102		128	91				104	23
	95			36			90							92	96
	49			37			33								50
				139			78								
				54											

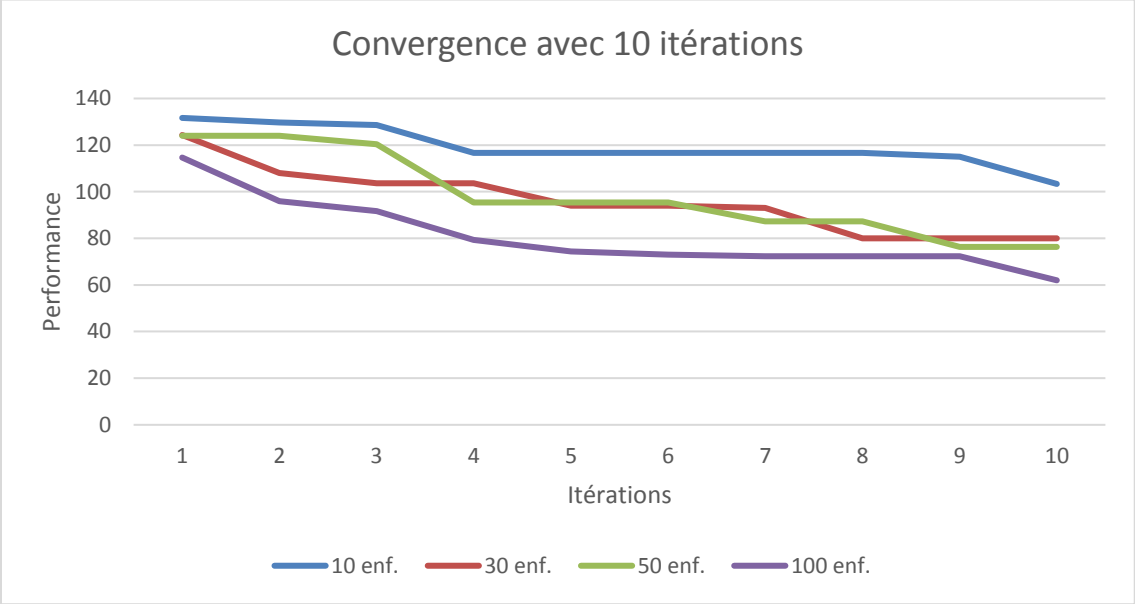
L

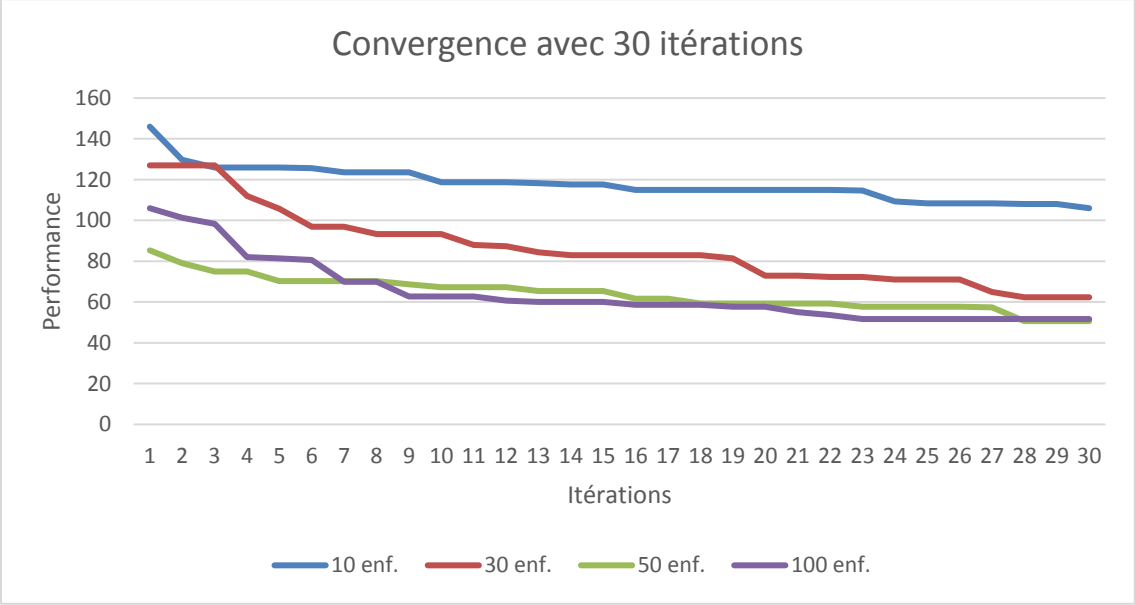
120	91	138	57	19	111	136	98	51	116	56	106	21	8	17	123
87	63	132	53	139	46	92	133	34	78	86	32	118	60	41	33
10	101	81	29	140	93	108	69	126	75	22	23	58	74	72	59
117	42	114	99	109	71	88	121	90	96	141	13	14	82	20	94
9	119	37	128	43	112	36	130	49	137	40	134	15	35	67	85
89	70	26	1	95	38	102	97	110	104	129	115	80	65	25	84
	103	48	122	66	64	131	18	6	76	61			24	47	55
	12		2	83	31		7	27	100	73			54	28	105
	62		113	68	39		79		4	52				77	50
			125	30	11					3					
			135	5	45					44					
			127		16					124					
			107												

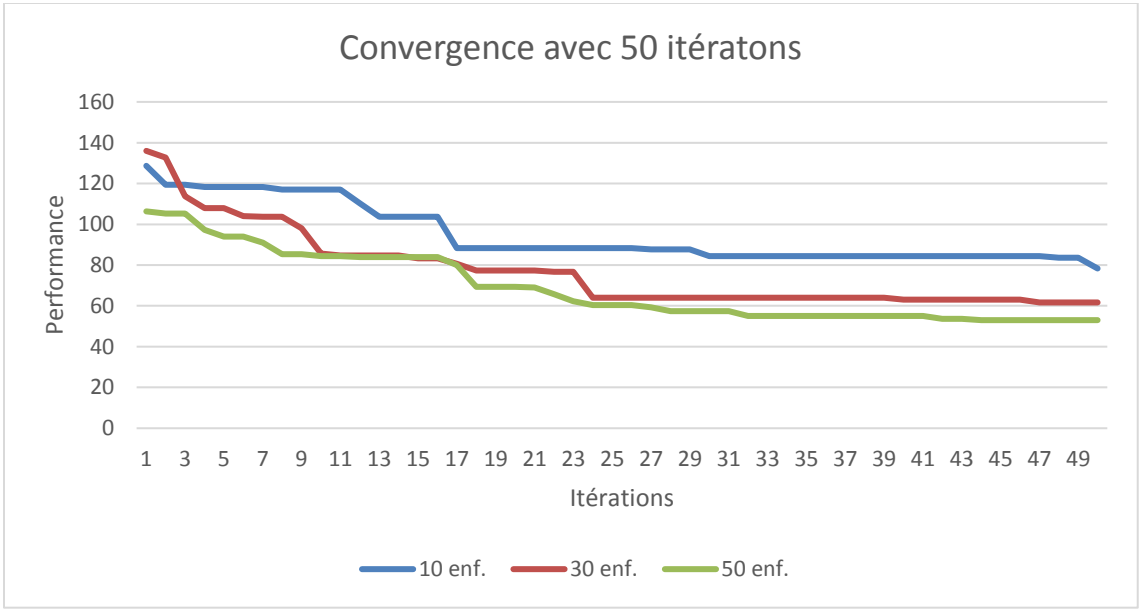
ANNEXE VI

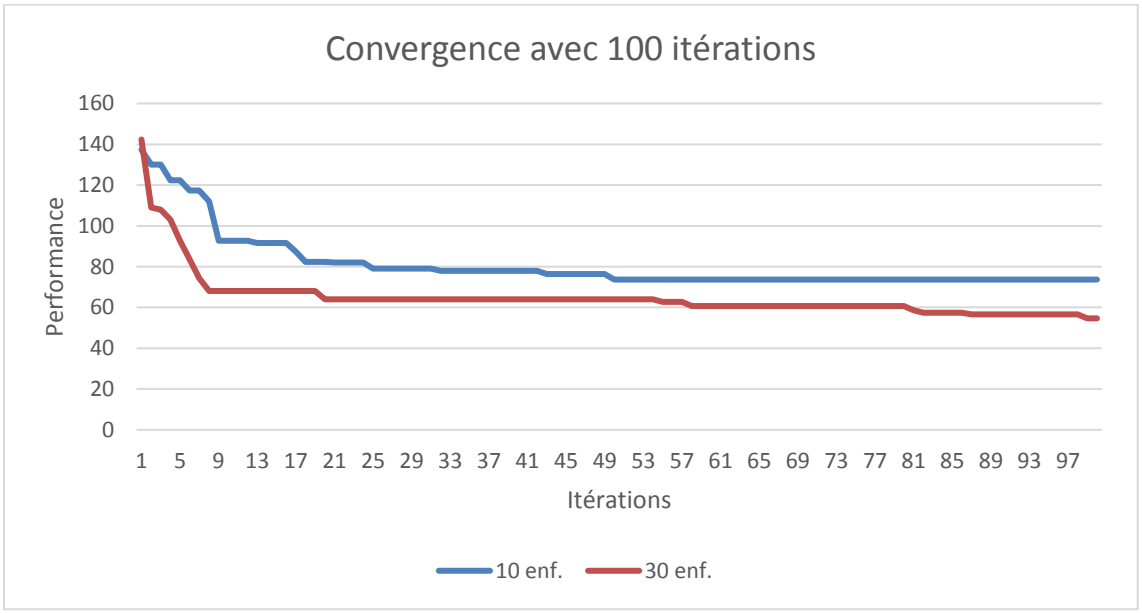
COURBES DE CONVERGENCE

4.1.3.1 Sensibilité du nombre d'enfants et d'itérations

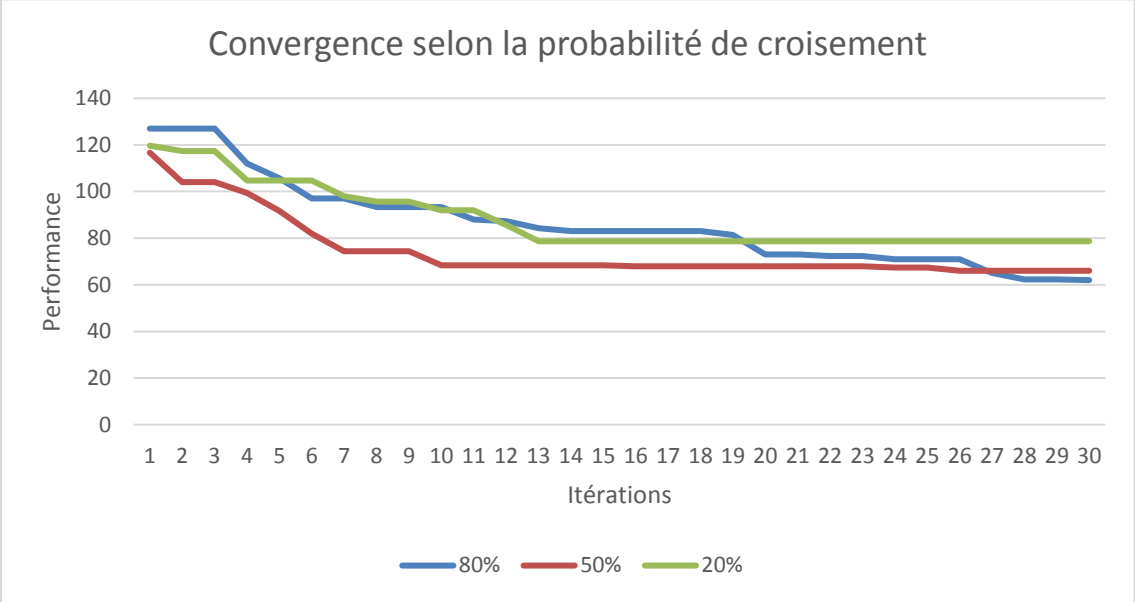


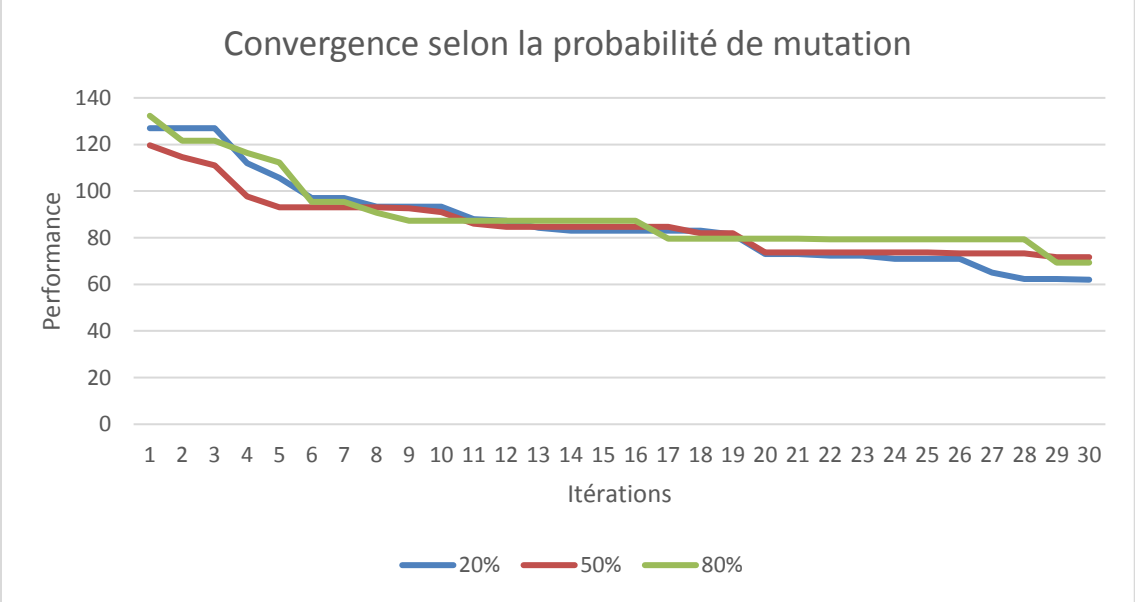



Convergence avec 50 itératons
Performance
160
140
120
100
80
60
40
20
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
Itérations
10 enf.
30 enf.
50 enf.

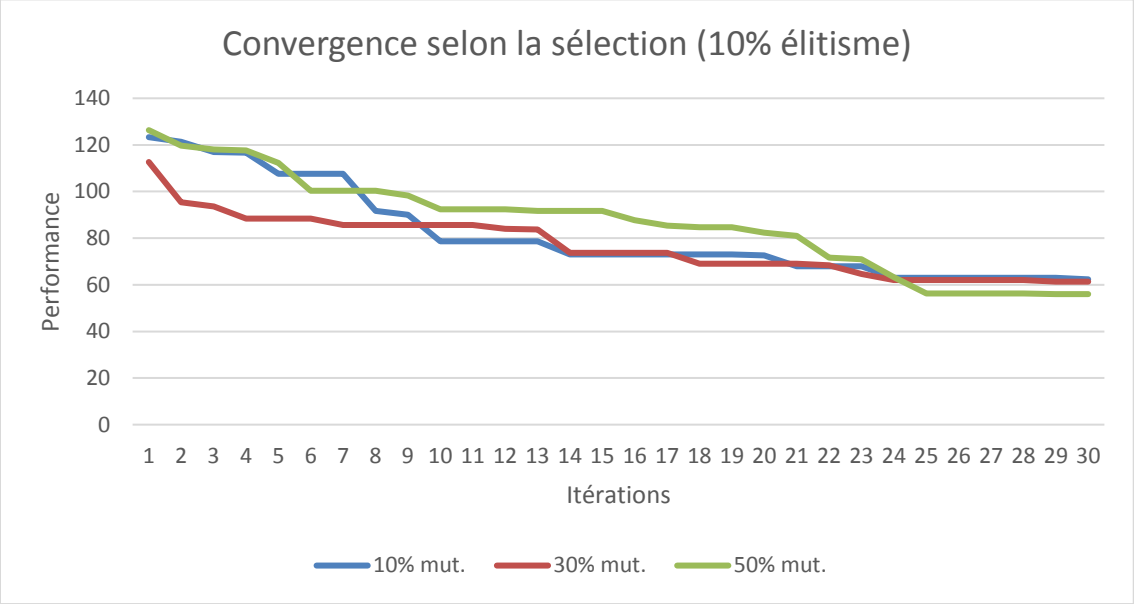

Convergence avec 100 itérations
Performance
160
140
120
100
80
60
40
20
0
1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 93 97
Itérations
10 enf.
30 enf.

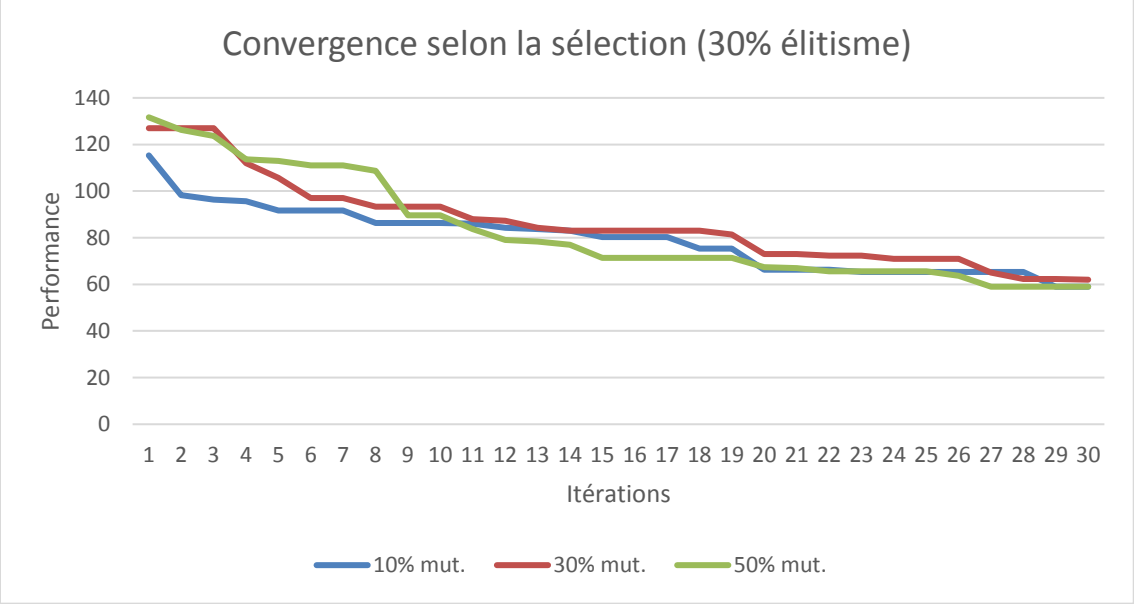
4.1.3.2 Sensibilité des probabilités de croisement et de mutation

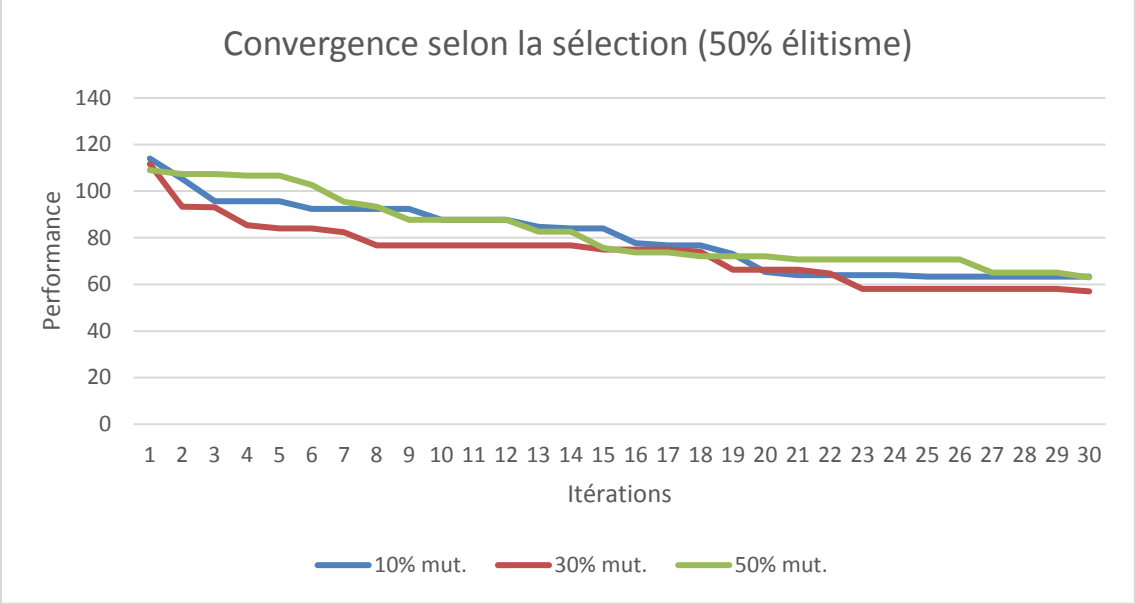




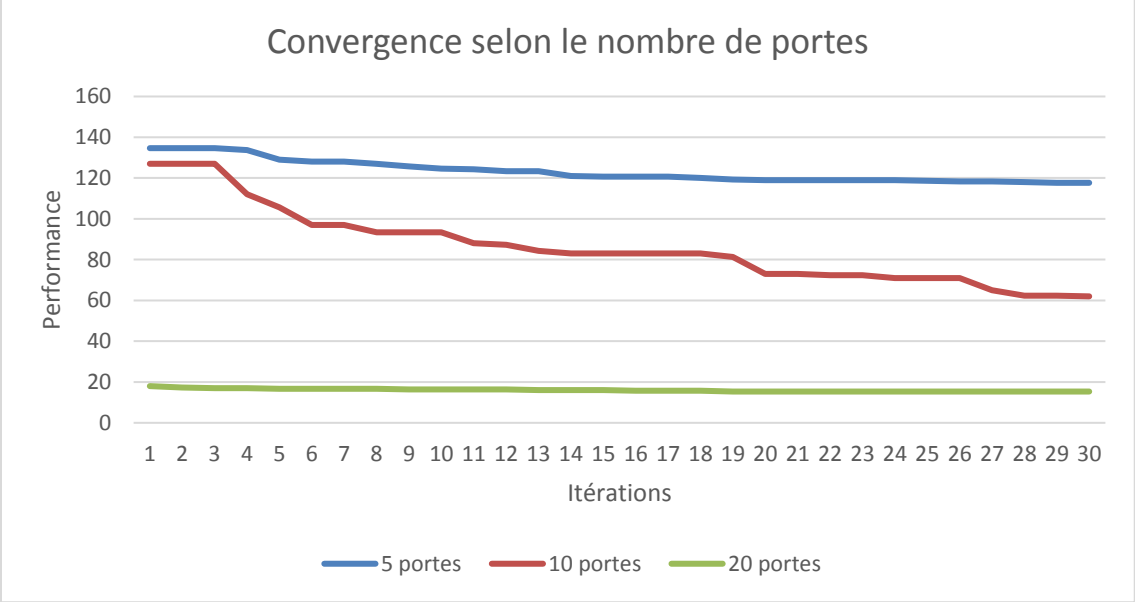
4.1.3.3 Sensibilité de la sélection par élitisme et par tournoi

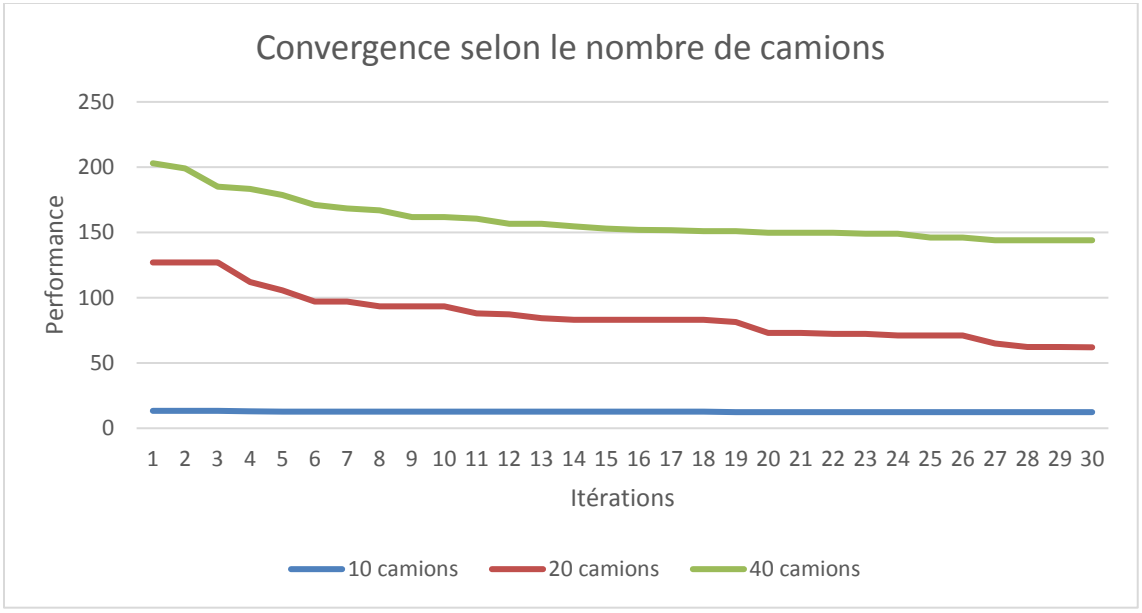


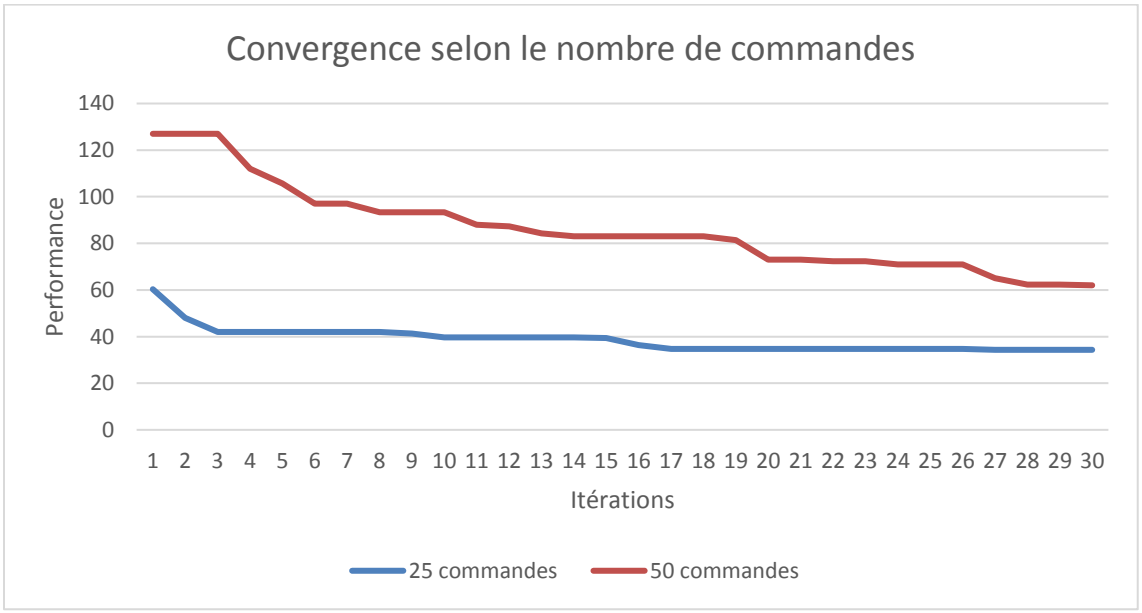




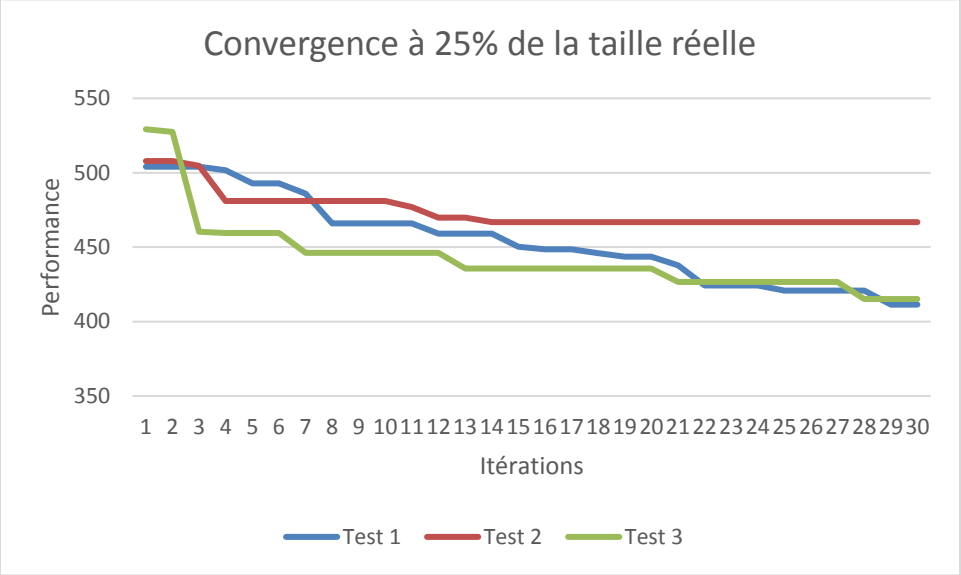
4.1.3.4 Sensibilité des paramètres du problème




Convergence selon le nombre de camions
Performance
250
200
150
100
50
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
Itérations
10 camions
20 camions
40 camions


Convergence selon le nombre de commandes
Performance
140
120
100
80
60
40
20
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
Itérations
25 commandes
50 commandes

4.2 Résolution d'un scénario réaliste



BIBLIOGRAPHIE

[1] « Transport Bourassa – Bienvenue ». [En ligne]. Disponible sur: http://www.bourassa.ca/index_fr.php. [Consulté le: 30-mars-2018].

[2] A.-L. Ladier et G. Alpan, « Cross-docking operations: Current research versus industry practice », Omega, vol. 62, p. 145‑162, juill. 2016.

[3] U. Apte et S. Viswanathan, « Effective Cross Docking for Improving Distribution Efficiencies », Int. J. Logist. Res. Appl., vol. 3, no 3, p. 291‑302, nov. 2000.

[4] K. R. Gue, « Warehouses Without Inventory », Int. Commer. Rev., vol. 7, no 2, p. 124‑132, déc. 2007.

[5] F. Walha, S. Chaabane, A. Bekrar, et T. Loukil, « The cross docking under uncertainty: State of the art », in Advanced Logistics and Transport (ICALT), 2014 International Conference on, 2014, p. 330–335.

[6] N. Boysen et M. Fliedner, « Cross dock scheduling: Classification, literature review and research agenda », Omega, vol. 38, no 6, p. 413‑422, déc. 2010.

[7] J. Van Belle, P. Valckenaers, et D. Cattrysse, « Cross-docking: State of the art », Omega, vol. 40, no 6, p. 827‑846, déc. 2012.

[8] L. Y. Tsui et C.-H. Chang, « A microcomputer based decision support tool for assigning dock doors in freight yards », Comput. Ind. Eng., vol. 19, no 1‑4, p. 309–312, 1990.

[9] M. Rohrer, « Simulation and Cross Docking », in Proceedings of the 27th Conference on Winter Simulation, 1995e éd., Washington, DC, USA: IEEE Computer Society, p. 846--849.

[10] G. Stalk, P. Evans, et L. E. Shulman, « Competing on capabilities : the new rules of corporate strategy », Harvard Business Review, p. 57‑69, avr-1992.

[11] G. Forger, « UPS starts world's premiere cross-docking operation. », 1995, p. 36‑38.

[12] G. F. Schwind, « A systems approach to docks and cross docking », Material Handling Engineering, p. 59‑62, 1996.

[13] J. R. Ross, « Office depot scores major cross-docking gains », Stores, p. 39‑40, 1997.

[14] C. E. Witt, « Cross-docking: Concepts demand choice », Material Handling Engineering, p. 44‑49, 1998.

[15] E. Kinnear, « Is there any magic in cross-docking? », Supply Chain Management: An International Journal, p. 49‑52, 1997.

[16] J. Katz, « Meeting at the crossdock », Industry week, p. 50, 2006.

[17] G. A. Álvarez-Pérez, J. L. González-Velarde, et J. W. Fowler, « Cross-docking— Just in Time scheduling: an alternative solution approach », J. Oper. Res. Soc., vol. 60, no 4, p. 554‑564, avr. 2009.

[18] A. Lim, H. Ma, et Z. Miao, « Truck Dock Assignment Problem with Time Windows and Capacity Constraint in Transshipment Network Through Crossdocks », in Computational science and its applications - ICCSA 2006: international conference, Glasgow, UK, May 8 - 11, 2006; proceedings. Pt. 3: ..., Berlin: Springer, 2006, p. 688‑697.

[19] F. Chen et K. Song, « Minimizing makespan in two-stage hybrid cross docking scheduling problem », Comput. Oper. Res., vol. 36, no 6, p. 2066‑2073, juin 2009.

[20] K. Song et F. Chen, « Scheduling Cross Docking Logistics Optimization Problem with Multiple Inbound Vehicles and One Outbound Vehicle », 2007, p. 3089‑3094.

[21] V. F. Yu, D. Sharma, et K. G. Murty, « Door Allocations to Origins and Destinations at Less-than-Truckload Trucking Terminals », p. 15.

[22] M. H. Fazel Zarandi, H. Khorshidian, et M. Akbarpour Shirazi, « A constraint programming model for the scheduling of JIT cross-docking systems with preemption », J. Intell. Manuf., vol. 27, no 2, p. 297‑313, avr. 2016.

[23] W. Yu et P. J. Egbelu, « Scheduling of inbound and outbound trucks in cross docking systems with temporary storage », Eur. J. Oper. Res., vol. 184, no 1, p. 377‑396, janv. 2008.

[24] A. L. Shiguemoto, U. S. Cavalcante Netto, et G. H. S. Bauab, « An efficient hybrid meta-heuristic for a cross-docking system with temporary storage », Int. J. Prod. Res., vol. 52, no 4, p. 1231‑1239, févr. 2014.

[25] B. Vahdani et M. Zandieh, « Scheduling trucks in cross-docking systems: Robust meta-heuristics », Comput. Ind. Eng., vol. 58, no 1, p. 12‑24, févr. 2010.

[26] A. R. Boloori Arabani, S. M. T. Fatemi Ghomi, et M. Zandieh, « Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage », Expert Syst. Appl., vol. 38, no 3, p. 1964‑1979, mars 2011.

[27] S. Forouharfard et M. Zandieh, « An imperialist competitive algorithm to schedule of receiving and shipping trucks in cross-docking systems », Int. J. Adv. Manuf. Technol., vol. 51, no 9‑12, p. 1179‑1193, déc. 2010.

[28] P. M. Cota, B. M. R. Gimenez, D. P. M. Araújo, T. H. Nogueira, M. C. de Souza, et M. G. Ravetti, « Time-indexed formulation and polynomial time heuristic for a multi-dock truck scheduling problem in a cross-docking centre », Comput. Ind. Eng., vol. 95, p. 135‑143, mai 2016.

[29] Y. Kuo, « Optimizing truck sequencing and truck dock assignment in a cross docking system », Expert Syst. Appl., vol. 40, no 14, p. 5532‑5541, oct. 2013.

[30] M. Keshtzari, B. Naderi, et E. Mehdizadeh, « An improved mathematical model and a hybrid metaheuristic for truck scheduling in cross-dock problems », Comput. Ind. Eng., vol. 91, p. 197‑204, janv. 2016.

[31] M. T. Assadi et M. Bagheri, « Differential evolution and Population-based simulated annealing for truck scheduling problem in multiple door cross-docking systems », Comput. Ind. Eng., vol. 96, p. 149‑161, juin 2016.

[32] A. Golshahi-Roudbaneh, M. Hajiaghaei-Keshteli, et M. M. Paydar, « Developing a lower bound and strong heuristics for a truck scheduling problem in a cross-docking center », Knowl.-Based Syst., vol. 129, p. 17‑38, août 2017.

[33] A.-L. Ladier et G. Alpan, « Robust cross-dock scheduling with time windows », Comput. Ind. Eng., vol. 99, p. 16‑28, sept. 2016.

[34] M. Y. Maknoon, F. Soumis, et P. Baptiste, « Optimizing transshipment workloads in less-than-truckload cross-docks », Int. J. Prod. Econ., vol. 179, p. 90‑100, sept. 2016.

[35] M. Y. Maknoon, F. Soumis, et P. Baptiste, « An integer programming approach to scheduling the transshipment of products at cross-docks in less-than-truckload industries », Comput. Oper. Res., vol. 82, p. 167‑179, juin 2017.

[36] A. Mohtashami, « Scheduling trucks in cross docking systems with temporary storage and repetitive pattern for shipping trucks », Appl. Soft Comput., vol. 36, p. 468‑486, nov. 2015.

[37] S. Rahmanzadeh Tootkaleh, S. M. T. Fatemi Ghomi, et M. Sheikh Sajadieh, « Cross dock scheduling with fixed outbound trucks departure times under substitution condition », Comput. Ind. Eng., vol. 92, p. 50‑56, févr. 2016.

[38] W. Wisittipanich et P. Hengmeechai, « Truck scheduling in multi-door cross docking terminal by modified particle swarm optimization », Comput. Ind. Eng., vol. 113, p. 793‑802, nov. 2017.

[39] M. Shakeri, M. Y. H. Low, S. J. Turner, et E. W. Lee, « A robust two-phase heuristic algorithm for the truck scheduling problem in a resource-constrained crossdock », Comput. Oper. Res., vol. 39, no 11, p. 2564‑2577, nov. 2012.

[40] « Top 500 entreprises québécoises - Les Affaires.com ». [En ligne]. Disponible sur: http://www.lesaffaires.com/classements/les-500/liste. [Consulté le: 15-mars-2018].

[41] A.-L. Ladier, « Planification des opérations de cross-docking: prise en compte des incertitudes opérationelles et de la capacité des ressources internes », Grenoble, 2014.

[42] J. Van Belle, P. Valckenaers, G. Vanden Berghe, et D. Cattrysse, « A tabu search approach to the truck scheduling problem with multiple docks and time windows », Comput. Ind. Eng., vol. 66, no 4, p. 818‑826, déc. 2013.

[43] « Simplicial polytopic numbers - OeisWiki ». [En ligne]. Disponible sur: https://oeis.org/wiki/Simplicial_polytopic_numbers. [Consulté le: 02-avr-2018].