République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A/Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

MÉMOIRE DE MASTER RECHERCHE

En
Informatique

Option
Réseaux et Systèmes Distribués

Thème

Nouvelle heuristique pour la résolution du Problème du Voyageur de Commerce

Présenté par : M. Mounir Hamani

Soutenu le 02 Juillet 2016 devant le jury composé de :

Président	**Dr A. Larbi**	**Maître Assistant A**	**U. A/Mira Béjaia.**
Examinateur	**Dr S. Aissani**	**Maître Assistant A**	**U. A/Mira Béjaia.**
Promotrice	**Dr K. Adel**	**Maître de conf. A**	**U. A/Mira Béjaia.**
Co-Promotrice	**Dr D. Boulahrouz**	**Maître Assistant A**	**U. A/Mira Béjaia.**

Béjaia, Juillet 2016.

⁕ *Remerciements* ⁕

Je remercie mes parents, mes promotrices, la communauté scientifique et Dieu.

⁕ *Dédicaces* ⁕

A mes très chers parents.

M. ***Mounir HAMANI***

Table des matières

Table des figures

Liste des tableaux

Abréviations

ACPM	Arbre Couvrant de Poids Minimal
LK	Lin-Kernighan
LKH	Lin-Kernighan-Helsgaun
NNH	Nearest Neighbor Heuristic
OPT	Tour optimal
PL	Programme Linaire
TSP	Traveling Salesman Problem
TSPLIB	Traveling Salesman Problem Library
VLSI	Very-Large-Scale Integration

Introduction générale

Les routes sont partie intégrante de l'infrastructure d'un pays, elles apportent des bienfaits à la fois économiques et sociaux en jouant un rôle important dans le développement économique local et national et en renforçant et en multipliant les interactions sociales. Les routes améliorent la mobilité et permettent aux personnes de sortir de leur isolement et aux entreprise de multiplier leurs échanges commerciaux. Cependant, les routes constituent une infrastructure lourde, coûteuse et difficile à mettre en place. Multiplier les routes et augmenter leurs capacités serait l'idéal mais leur coût en espace et en argent rend cela prohibitif, c'est pourquoi, il est intéressant d'essayer de maximiser le rendement et les performances des réseaux routiers déjà existants à l'aide de différentes techniques de routage.

Il existe plusieurs types de problématiques de routage routier. On peut citer le routage de point à point qui consiste à trouver depuis un point de départ donné, le chemin le moins coûteux vers un point de destination donné. Il y a aussi le routage d'une tournée de livraison, appelé également, problème du voyageur de commerce, qui étant donné les distances entre chaque paire de points, consiste à trouver le circuit le moins coûteux desservant chacun des points de livraison une fois seulement et réalise un retour au point de départ une fois toutes les destinations visitées. Citons également la problématique de la tournée de véhicules [19] qui consiste à trouver l'ensemble optimal de routes ou circuits à traverser par une flotte de véhicules pour livrer un ensemble de clients, l'objectif ici est de minimiser le coût total de traversée de cet ensemble de circuits. Et enfin la problématique de re-planification de véhicules [42] dont le but est de ré-assigner des véhicules en temps réel après panne ou retard majeur du véhicule assigné préalablement. Chacune de ces problématiques est une extension de la précédente et chacune d'entre-elles possède des variantes selon les contraintes du domaine auquel elle est appliquée. Aussi, à l'exception de la première, toutes les autres appartiennent à la classe de problèmes NP-Difficiles.

La problématique de recherche de chemin ou de circuit routier est une problématique de recherche de chemin ou circuit dans un graphe.

Le routage au sein d'un graphe se fait selon différentes approches tentant de trouver un équilibre ou consensus entre les besoins en mémoire, le temps de réponse et la qualité des chemins trouvés.

Dans les chapitres qui suivent, nous discuterons de la problématique de la tournée de livraison, nous commencerons par une introduction détaillée traitant de l'historique et de l'importance de ce problème ainsi que de ses applications très variées qui ne se limitent pas au domaine routier. Après quoi, nous passerons en revue quelques-unes des heuristiques existantes en expliquant le principe de chacune d'elles. Et puis enfin, nous exposerons notre approche pour la résolution de cette problématique suivie de quelques variations selon les besoins en temps.

Chapitre 1

Introduction sur le Problème du Voyageur de Commerce

1.1 Introduction

Étant donné un ensemble de villes avec les distances ou les coûts de voyage entre chaque paire d'entre elles, le problème du voyageur de commerce, Traveling Salesman Problem en anglais ou TSP pour faire court, est de trouver le circuit le moins coûteux partant d'une ville, visitant toutes les autres puis retournant à la ville de départ. En d'autres termes, il s'agit de trouver l'ordre dans lequel traverser ces villes en minimisant la distance totale parcourue.

Simple en apparence, il s'agit en réalité d'un des problèmes les plus étudiés en mathématiques informatiques, ayant inspiré des recherches en mathématiques, informatique, physique, chimie, biologie et même en neurologie, psychologie [65][46] et neuropsychologie, ainsi que de la part d'amateurs attirés par son apparente simplicité. Et connait également des applications en logistique avec les tournées de livraisons, en génétique et en neurosciences, en télécommunications, dans l'industrie pour le perçage de circuits imprimés ou pour les points de soudure, en astronomie quand il faut observer plusieurs régions du ciel et qu'il faut fair pivoter le téléscope pour celà, et dans de nombreux autres domaines.

L'intérêt porté au TSP en a fait l'un des râres problèmes contemporains à intégrer la culture populaire. Son nom facile à retenir y est probablement pour quelque chose. D'ailleurs, un film est même sorti à son propos, en juin 2012, appelé "Travelling Salesman".

1.2 Historique

1.2.1 Origines du Traveling Salesman Problem

Les origines exactes du problème restent obscures. Un manuel pour voyageurs de commerce, de 1832, fait mention du probème et propose quelques tours d'Allemagne et de Suisse [21]. Les premières formulations mathématiques sont associées au mathématicien anglais Thomas Penyngton Kirkman qui a explicitement posé le problème dans un article qu'il a soumis à la Royal Society ainsi qu'au célèbre

FIGURE 1.1 – Icosian Game [41]

mathématicien irlandais William Rowan Hamilton qui était fasciné par les propriétés géométriques du dodécaèdre. Il présente son travail sous forme d'un jeu qu'il appelle Icosian Game, dans une lettre datée de 1856, puis en fait l'exposition lors d'un meeting de mathématiciens anglais à Dublin l'année qui suit. D'ailleurs, un cycle passant par tous les sommets d'un graphe est appelé Cycle Hamiltonien. Cependant, la tâche de trouver un circuit qui passe seulement une fois par tous les sommets d'un graphe a surgis de létude, deux ans plus tôt, du tour du cavalier d'échecs par Leonhard Euler. Et le probléme du cavalier quand à lui est un exercice qui trouve ses origines encore plus loin dans le temps en remontant jusqu'au joueur et théoricien d'échecs al-Adli ar-Rumi qui en donne une solution vers l'an 840. La première occurrence connue remonte à un traité d'ornement poétique indien. Le principe du problème du cavalier est de visiter à partir d'une case, toutes les autres cases d'un échiquier par une succession de mouvements de cavalier sans visiter deux fois la même case. Malgré la similarité entre le probléme du cavalier et celui du voyageur de commerce, il existe pour le premier des algorithmes permettant de trouver une solution en temps polynomial, voir même linéaire [55][14], alors que le deuxième n'en connait pas pour l'instant.

Après l'episode hamiltonien, c'est lors d'un colloque de mathématiques à Vienne, en Autriche, que le problème refait surface. Dans les notes de Karl Menger [61], il expose ce qu'il appelle alors "Das Botenproblem", ce qui peut se traduire en français par problème du messager ou du facteur, ce qui est décrit par Menger est le passage d'une courbe par un nombre fini de points dans un espace métrique dont on connaîtrait les distances entre chaque paire, de telle manière à minimiser la longueur totale de la courbe. C'est ce que l'on appelle aujourd'hui, chemin Hamiltonien. Il ne manque donc qu'une seule contrainte au Botenproblem pour en faire un TSP, c'est celle du retour au point de départ. Menger a fait l'analogie avec le postier parce que ce dernier a généralement plusieurs destinataires à livrer et qu'il est dans son intérêt de minimiser la distance qu'il a à parcourir. Menger est aussi le premier à attirer l'attention sur la complexité du problème et à poser la question d'un algorithme meilleur que la recherche exhaustive.

Un peu en parallèle, toujours dans les années 1930, mais de l'autre coté de l'atlantique, aux États-Unis, à l'université de Princeton pour être exacte, Hassler Whitney travaillant alors autour de la théorie des graphes, introduit ce qui deviendra le Traveling Salesman Problem à ses collègues, et parmi eux, on trouve Albert Tucker et Merrill Meeks Flood [1]. Ce dernier, qui à son tour, popularisera le problème à Princeton et à la RAND Corporation où il est recruté après la guerre.

Le problème introduit par Whitney à ses collègues était appelé "Problème des 48 états", à l'époque, les États-Unis n'étaient que 48, sans Hawaii et sans l'Alaska. Flood accorde à Tucker le mérite d'avoir attiré son attention à la relation qui existe entre le TSP et les jeux et circuits Hamiltoniens au moment où il trouvait des difficultés en connexion avec l'étude d'un problème de ramassage scolaire.

1.2.2 Début des études autour du TSP

Durant les années 1940, quelques articles étudiant le TSP ont été publiés. Ils seraient les premiers à contenir des résultats mathématiques, parmi lesquels, un article de Fejes [22] qui explore le problème de la plus courte courbe traversant n points

dans le carré unité. En conséquence du travail de ce dernier, Verblunsky [64] montre en 1951 qu'une telle courbe a une longueur inférieure à $2 + \sqrt{2.8n}$.

En 1940, une étude réalisée et publiée par Prasanta Chandra Mahalanobis [47] offre quand à elle une borne inférieure pour un chemin passant par n points aléatoires dans un plan. Son étude a été motivée par le besoin de calculer le coût minimal d'une enquête statistique qu'il menait au Bengale, en Inde, pour le compte du gouvernement. Le pays était à l'époque le plus gros exportateur de jute dans le monde et cette production était répartie sur six millions de fermes, impossible donc de toutes les visiter. Mahalanobis a alors décidé de réaliser des prélèvement et de recueillir des échantillions à travers la région, et les différents déplacements avec le transport du personnel et du materiel étaient le principal coût du projet.

Cette borne calculée par Mahalanobis est $\sqrt{n} - 1/\sqrt{n}$, où n est le nombre de points par lesquels il doit passer.

En 1948, Marks [48] donna une preuve de celle-ci et montra que $\sqrt{\frac{1}{2}A}(\sqrt{n} - 1/\sqrt{n})$, où A est la superficie de la région, est une borne inférieure. Un an plus tard, soit en 1949, Ghosh [24] montra que cette borne est proche de la valeur réelle en donnant une heuristique permettant de trouver un tour avec une borne supérieure de $1.27\sqrt{An}$, et remarqua la complexité du problème : "Après avoir posé n points aléatoires sur une carte de la région, il est difficile de trouver LE chemin le plus court de tous reliant tous les points, à moins que n ne soit très petit, ce qui n'est pas le cas pour une étude à grande échelle."

Mais il est utile de noter qu'à l'époque, le "problème d'affectation" que nous savons aujourd'hui resolvable en temps polynomial, était considéré comme difficile. C'est d'ailleurs une tentative ratée de résolution du TSP qui a conduit Julia Robinson [59] du RAND au développement d'une méthode de réduction de cycle pour le problème d'affectation. Le rapport de Julia Robinson, daté du 5 décembre 1949, est la première référence mathématique répertoriée à utiliser le nom "traveling salesman problem".

Comme c'est le cas avec de nombreux autres problèmes d'optimisation combinatoire et ce qui a attrait à la recherche opérationnelle, la RAND Corporation a joué un role important dans la recherche liée au TSP en amplifiant la promotion

du problème initiée par Merrill Flood. Motivée par l'optique de créer d'autres challenges intellectuels en dehors de la théorie des jeux, un prix a même été offert pour le développement d'un théorème significatif autour du TSP.

Les chercheurs de la RAND, et notamment Koopmans [38], ont considéré le transfert des méthodes ayant fait leurs preuves avec la théorie du transport vers le problème du voyageur de commerce. Tjallings Koopmans et Georges Dantzig, lors de leur rencontre avec Flood à Washington D.C. en Septembre 1947 spéculèrent sur la forte connexion qui existerait entre leurs travaux, qui seront connus plus tard comme le problème de la Programmation Linéaire, et le Traveling Salesman Problem.

Dans une lettre datée du 17 Mai 1983, Flood écrit que le rapport de Robinson a stimulé de nombreuses discussions autour du TSP durant la période 1950-1952 avec Delbert Ray Fulkerson, son assistant de recherches à la RAND. Et en 1952, Beckmann et Koopmans remarquèrent que le TSP pouvait être formulé sous forme d'un problème d'assignement quadratique, mais pour lequel on ne connait pas de méthodes polynomiales. Il ne faudra pas attendre longtemps avant que toute cette agitation scientifique ne commence à produire des résultats significatifs, puisqu'en 1954 l'équipe composée de George Dantzig, Ray Fulkerson et Selmer Johnson publie un papier [18] dans lequel elle introduit plusieurs nouvelles méthodes pour la résolution du TSP, parmi lesquelles, "la méthode des plans sécants". Le papier est qualifié dans le milieu d'évènement majeur dans l'histoire de l'optimisation combinatoire [32] et fournit une solution au TSP des 48 états, contenant en fait 49 villes en comptant Washington D.C., après une réduction astucieuse de celui-ci à 42 villes.

En 1956, c'est Tompkins [62] qui décrit un modèle de Branch-and-Bound pour les problèmes de permutation, y compris le problème de l'assignement et le TSP, mais il insiste sur le fait que cette méthode n'est pas satisfaisante et qu'il existe des cas pour lesquels d'autres méthodes seraient plus avisées. Et c'est le même écho qui se fait entendre du coté de Flood qui juge que les algorithmes existants ne sont pas satisfaisants, tout en avançant qu'au vu des résultats mathématiques les plus récents, le problème pourrait être fondamentalement complexe. Il suggère alors qu'une approche radicalement différente de celles déjà utilisées pourrait être nécessaire pour venir à bout du problème, tout en gardant en perspective qu'il puisse ne pas exister de

méthode générale pour le traitement effectif de celui-ci et qu'une telle démonstration serait tout aussi intéressante.

Les travaux cités ci-dessus ne sont pas les seuls à avoir été réalisés et publiés durant les années 1950. En 1955 G. Morton et A. H. Land [50] contribuent une approche couplant des contraintes (par exemple, le circuit ne doit pas contenir de croisements et ainsi pas de sous-tour) exprimées en programmation linéaire avec une heuristique qui part d'un circuit et fait des remplacements d'arêtes trois par trois. L.L. Barachet [7] de la régie nationale des usines Renault en France, propose en 1957 une méthode graphique itérative, et G. A. Croes [16] de la compagnie Shell, quand à lui présente une méthode de résolution applicable aux instances symétriques tout comme aux instances asymétriques appréciablement plus rapide que celles déjà proposées.

Les travaux de Marks et Ghosh entrepris durant les années 1940 conduisirent à un résultat célèbre de Jillian Beardwood, John H. Halton et John Michael Hammersley [9], publié en 1959 et selon lequel la longueur du tour optimal à travers un ensemble de n points contenus dans une surface v est proportionnelle à $\sqrt{nv}$ quand n est grand, et que la division de cette longueur par $\sqrt{n}$ tend avec une probabilité de 1 vers une constante, estimée récemment [4] autour de 0.7124 ± 0.0002.

Les trois décennies qui ont suivis furent extrêmement riches en contributions.

1.2.3 Popularisation du TSP à grande échelle

L'année 1960 verra l'introduction de la programmation dynamique par Richard Ernest Bellman dans un livre [10] qu'il co-écrit avec Marshall Hall Jr et publié par la Société de Mathématiques Américaine et dans lequel il utilise le TSP comme un exemple d'application de cette méthode à un problème combinatoire.

Au début des années 1960, le problème du voyageur de commerce a été popularisé auprès du public, notamment, grâce à une compagne publicitaire de Procter & Gamble en 1962 offrant un grand prix de $10.000USD pour le tour le plus court d'un TSP de 33 villes en rapport avec une série télévisée du moment. Pour l'époque, le prix était suffisamment important pour permettre l'achat d'une nouvelle maison et a

du coup attiré pas mal d'intérêt y compris celui de mathématiciens qui l'ont cité dans leurs articles. D'ailleurs l'un des gagnants du concours a été Gerald L. Thompson de l'université Carnegie Mellon. Thompson a ensuite publié [36] son tour dans un article de 1964 qu'il a co-écrit avec son collègue Robert L. Karg de l'université Duquesne.

L'année 1962, Martin Held et Richard M. Karp [25], deux noms qui marqueront à jamais le sujet de leur empreinte font leur entrée sur la scène du TSP. Comme beaucoup d'autres au cours de cette décennie, il reprennent l'idée de la programmation dynamique introduite par Bellman deux ans plus tôt et proposent deux approches selon la taille du problème, la première avec un algorithme exact permettant de résoudre des instances pouvant aller jusqu'à 13 villes, et une seconde avec un algorithme approximatif qui a tout de même réussi à trouver la solution optimale au problème réduit de 42 villes de Dantzig, Fulkerson et Johnson lors de deux essais sur cinq.

Il est à noter que jusque-là, la complexité de l'algorithme exact, celui de l'énumération, était de l'ordre de $O(n!)$, ce qui rendait très vite impraticable la résolution des instances même les plus modestes et impossible celles de quelques dizaines de villes. Une instance de 60 villes par exemple possède autant de permutations que le nombre d'atomes dans l'univers connu, et aucun super-ordinateur ou réseau d'ordinateurs au monde ne peut réaliser ou finir un tel calcul même si il s'y exécutait pendant des siècles ou des millénaires. L'algorithme exact proposé par Held et Karp, possède quand à lui une complexité de $n^2 2^n$ et il s'agit de la meilleure borne obtenue pour les algorithmes exacts depuis lors. Malheureusement, ça reste une complexité exponentielle et elle n'améliore rien du tout dans la pratique.

En 1963, J.D.C. Little, K.G. Murty, D.W. Sweeney, et C. Karel ré-inventent un algorithme pour le TSP qui restera dans les annales du domaine de l'optimisation. Cet algorithme, ils l'appelèrent "Branch & Bound" [45]. L'algorithme est exact et il sera utilisé par la suite pour confirmer ou non l'optimalité des tours trouvés par d'autres méthodes mais malheureusement, il reste exponentiel.

L'article de Karg et Thompson publié en 1964 et cité plus haut, décrivait également l'heuristique de l'insertion la moins coûteuse et proposait de l'utiliser pour améliorer des sous-sections de tours. Inspirés par le concours de Procter & Gamble,

FIGURE 1.2 – Concours de Procter & Gamble pour le TSP de 33 villes le plus court [56]

ils appliquèrent leur heuristique à une instance de 57 villes dont ils tirent les données d'un atlas routier.

Une autre figure majeure de l'histoire du Traveling Salesman Problem, Shen Lin, fait son entrée en 1965 avec un article [43] construisant sur le travail de Croes et proposant un schéma de réduction efficace pour l'implémentation de 3-opt sur de grandes instances. La même année, Stanley Reiter et Gordon Sherman [58] proposent un algorithme de recherche locale pour le TSP et le testent sur l'instance de 57 villes de Karg et Thompson ainsi que sur d'autres instances moins importantes. À chaque fois, l'algorithme trouve un tour au moins aussi bon que ceux déjà répertoriés dans d'autres articles jusque là.

En 1966, Eugene Leighton Lawler et David E. Wood publièrent [40] ce qui deviendra une citation classique du domaine de l'optimization combinatoire. Il s'agit d'une revue de la littérature dans laquelle ils montrent que les méthodes d'énumération implicite, de séparation et évaluation progressives ainsi que d'autres, ne sont en réalité que Branch & Bound. Les auteurs apportent également la suggestion d'utiliser les arbres couvrants de poids minimal comme borne inférieure lors de l'application de Branch & Bound.

Beaucoup d'autres contributions et revues ont été publiées durant cette décennie autour de variantes et améliorations de la méthode de Branch & Bound, la méthode des plans sécants et de la programmation linéaire ainsi que quelques méthodes d'énumération.

1.2.4 Emergence de références

Après presque neuf ans de recherche, Held et Karp refont surface en 1970 sur la scène du TSP en reprenant l'idée des arbres couvrants à poids minimal [26]. Ils introduisent la relaxation 1-arbre pour le TSP et l'idée d'utiliser le poids des noeuds pour améliorer la borne donnée par le 1-arbre optimal. Un 1-arbre est un arbre couvrant avec un nœud supplémentaire relié à ce même arbre par deux arêtes. Ils observent qu'un tour est un 1-arbre où tous les nœuds sont de degré 2, qu'il est facile de calculer un 1-arbre à poids minimal et enfin qu'il peuvent opérer la transforma-

tion $c_{ij} \rightarrow C_{ij} + \pi_i + \pi_j$ sur les distances inter-villes impliquant un changement au niveau du 1-arbre mais laissant le TSP invariant. En utilisant ces observations, ils définissent une famille infinies de bornes inférieures $w(\pi)$ sur C^*, le coût d'un tour optimal, et montrent que $max_\pi w(\pi) = C^*$ précisément quand un certain programme linéaire bien défini possède une solution optimale en nombre entiers. Ils donnent une méthode de génération de colonnes, une méthode de sous-gradient pour calculer $max_\pi w(\pi)$ et construisent une méthode de Branch & Bound dans laquelle les bornes inférieures $w(\pi)$ contrôlent la recherche du tour optimal. L'idée est intéressante mais les tests pratiques sont moins bons que prévu et il faudra attendre un an de plus pour que Held et Karp reviennent avec une deuxième partie [27] de leur article dans laquelle ils présentent une méthode itérative efficace pour le calcul d'une borne inférieure satisfaisante pour le 1-arbre. Les bornes calculées sont si proches du tour optimal que les arbres de recherche qui en découlent sont qualifiés de minuscules comparés à ceux habituellement rencontrés dans les problèmes combinatoires de ce type. Ils incorporent alors cette méthode dans un algorithme de Branch & Bound et l'appliquent à la résolution du problème réduit de 42 villes de Dantzig, Fulkerson et Johnson, l'instance de 57 villes de Karg et Thompson ainsi qu'une autre instance euclidienne aléatoire de 64 villes parmi d'autres. Ces résultats étaient alors les meilleurs répertoriés jusque là. Cette méthode qui construit sur les travaux qui l'ont précédé sera à son tour reprise dans un grand nombre d'articles et sera reconnue comme une référence incontournable.

Une autre figure importante fait son entrée en 1972, il s'agit de Nicos Christofides, avec un article [12] proposant une méthode itérative avec garantie de convergence pour le calcul des bornes inférieures dans l'espoir que ça puisse servir aux méthodes de Branch & Bound. Cette méthode a été testée sur 14 exemples d'instances et a produit des bornes inférieures étant en moyenne à seulement 4.7% sous la valeur optimale pour les instances symétriques et à 3.8% pour les instances asymétriques.

En 1973, Shen Lin revient au devant de la scène accompagné cette fois-ci par Brian Wilson Kernighan, tous les deux collègues aux laboratoires Bell au New Jersey aux États-Unis. Leur papier [44] introduit une procédure heuristique très effective produisant des résultats proches de l'optimalité. Leur méthode est si effective qu'elle

est souvent reprise dans les solvers modernes. Dans leur article, ils reviennent sur les travaux de Held et Karp et expliquent que la méthode donnée par ceux-ci résout de manière exacte une certaine classe de TSP, mais que dès qu'un problème est reconnu comme n'appartenant pas à cette classe, alors l'algorithme est couplé à un mécanisme auxiliaire, ici le Branch & Bound, nécessitant du coup des temps de déroulement prohibitifs.

Quelques années de recherche et quelques publications plus tard, Christofides revient de l'autre coté de l'optimalité avec un résultat majeur produit de son algorithme [13] offrant une garantie dans le pire des cas à propos de la borne supérieure. Cet algorithme s'applique aux instances symétriques respectant l'inégalité triangulaire. Il implique comme sous-étapes le calcul d'un arbre couvrant à poids minimal du graphe G définissant le TSP et le fait de trouver un couplage parfait de poids minimal d'un certain sous-graphe induit par G. Une analyse du pire des cas de cet algorithme montre que le ratio de la solution obtenue par rapport à la solution optimale est strictement inférieur à 3/2. Ceci représente une réduction de 50% par rapport à la meilleure des garanties précédentes et qui était de 2.

Durant cette décennie de 1970, plusieurs autres travaux ont été réalisés et publiés autour du TSP par d'autres chercheurs, l'un des thèmes les plus récurrents était comme lors de la décennie précédente, le recours à la méthode des plans sécants.

Durant les années 1980, l'un des chercheurs qui feront le plus de contributions sera Manfred W. Padberg. D'abord en 1980 avec Saman Hong [53], ils tentent de prouver empiriquement l'efficacité de la méthode des plans sécants. Durant la même année, il améliorera ce travail avec l'aide de Harlan Crowder [17] et offriront une solution à une instance de 318 villes décrite par Lin et Kernighan, celle-ci restera la plus large instance résolue jusqu'en 1987 quand Padberg lui même avec Giovanni Rinaldi [51] réussirent à résoudre une instance de 532 villes à l'aide de la méthode de Branch & Cut, dans leur article, ils décrivent brièvement la méthodologie, les algorithmes et les systèmes logiciels qu'ils ont utilisés pour obtenir leurs résulats.

1.2.5 Avancées pratiques substantielles

Bien d'autres chercheurs feront preuve d'intérêt pour le TSP et publieront des articles sur le sujet durant les années 1990, mais Padberg et Rinaldi se distingueront par la continuité de leur collaboration active qu'ils entretiendront durant cette décennie. En 1991, ils repousseront les limites du TSP en publiant un article [52] dans lequel ils traitent à l'aide de Branch & Cut, plusieurs instances allant de 48 jusqu'à 2392 villes.

Toujours en 1991, Gerhard Reinelt de l'institut de mathématiques de l'université d'Augsburg en Allemagne réunit au sein d'une même bibliothèque plus d'une centaine d'instances de TSP issues d'exemples géographique, de l'industrie telle la VLSI (Intégration à très grande échelle) ainsi que d'un bon nombre d'autres sources dans le but de fournir aux chercheurs un large éventail d'instances de test issues de sources différentes et avec des propriétés variées. Cette bibliothèque, il l'appellera tout simplement TSPLIB [57] et elle contiendra des instances allant de 14 villes jusqu'à 85,900. Cette dernière est issue d'une application VLSI des laboratoires Bell, dont le but est de minimiser le temps total de déplacement d'un laser entre les interconnections qu'il doit découper. Les villes correspondent aux positions de ces interconnexions et la distance entre deux villes au temps nécessaire pour un déplacement d'une interconnexion à une autre. La solution définira le meilleur ordre dans lequel découper ces interconnexions.

En 1996, Marco Dorigo, Vittorio Maniezzo et Alberto Colorni définissent un nouveau paradigme de recherche suggéré par une analogie au fonctionnement des colonies de fourmilles [20]. Les principales caractéristiques de ce modèle sont basées sur une recherche distribuée couplée à un système de retour ou feedback positif ainsi que sur une heuristique constructive gloutonne qui permet de trouver des solutions acceptables dès les premières étapes du processus de recherche. Ils appelleront cette méthode Ant System et l'appliqueront au TSP tout en rapportant leurs résultats. Par la suite, pour prouver la robustesse de la méthode, elle est également testée sur le TSP asymétrique, sur le problème de l'assignement quadratique ainsi que d'autres exemples de problèmes combinatoires.

En 1998, David Applegate, Robert Bixby, Vasek Chvátal et William Cook sortirent un livre [3] accompagné d'un programme appelé Concorde [2], basé sur la multitude d'heuristiques et de méthodes développées jusque-là, et plus particulièrement Branch & Cut. Ce solver leur permis alors de s'attaquer et de résoudre la totalité des instances de TSPLIB à l'optimalité pendant la décennie qui suivit, y compris l'instance de 85,900 villes qui fut résolue entre février 2005 et avril 2006. Il est cependant utile de noter que pour réaliser une telle prouesse ils ont eu recours à la parallélisation sur des stations AMD Opteron 250 de 2.4GHz avec un temps CPU total de *136ans*.

Une instance beaucoup plus grande que celle des 85,900 villes est apparue en 2001, réalisée par les créateurs de Concorde, il s'agit du World TSP. Cette instance contient 1,904,711 villes et est trop grande pour Concorde. Le meilleur tour connu sur cette instance a été trouvé grâce à LKH (Lin-Kernighan-Helsgaun), une implémentation effective de l'heuristique Lin-Kernighan publiée et améliorée par Keld Helsgaun en 1998 [28], 2000 [29], 2006 [30] et 2009 [31]. On sait grâce aux bornes inférieures calculées par Concorde que le tour produit par LKH pour le World TSP est au plus 0.058% plus long que le tour optimal. De plus, LKH a généré des solutions optimales pour toutes les instances résolues sur lesquelles elle a été testée et détient le record des tours les plus courts sur toutes les instances du DIMACS TSP Challenge [33] qui n'ont pas de tour optimal prouvé. Les instances du challenge ont des tailles qui varient de 1,000 à 10,000,000 de villes.

Au vu de la simplicité et de la rapidité de l'algorithme de 1976 de Christofides, beaucoup s'attendaient à ce que cela ouvre rapidement la voie à des algorithmes donnant des garanties presque optimales. Cependant, il aura fallu attendre 35 ans pour qu'en 2011 une équipe composée de deux collègues de l'université de Stanford aux États-Unis et un troisième de l'université McGill au Canada améliore cette garantie [23], tenez-vous bien, de quatre centièmes de trilliardième de trilliardième de pour cent. En d'autres termes, ils ont amélioré la borne supérieure de 50% vers 49.99999999999999999999999999999999999999999999996%. Motivées par ces résultats, d'autres équipes ont suivi le pas et ont réussis à contracter la

FIGURE 1.3 – Villes du World TSP [5]

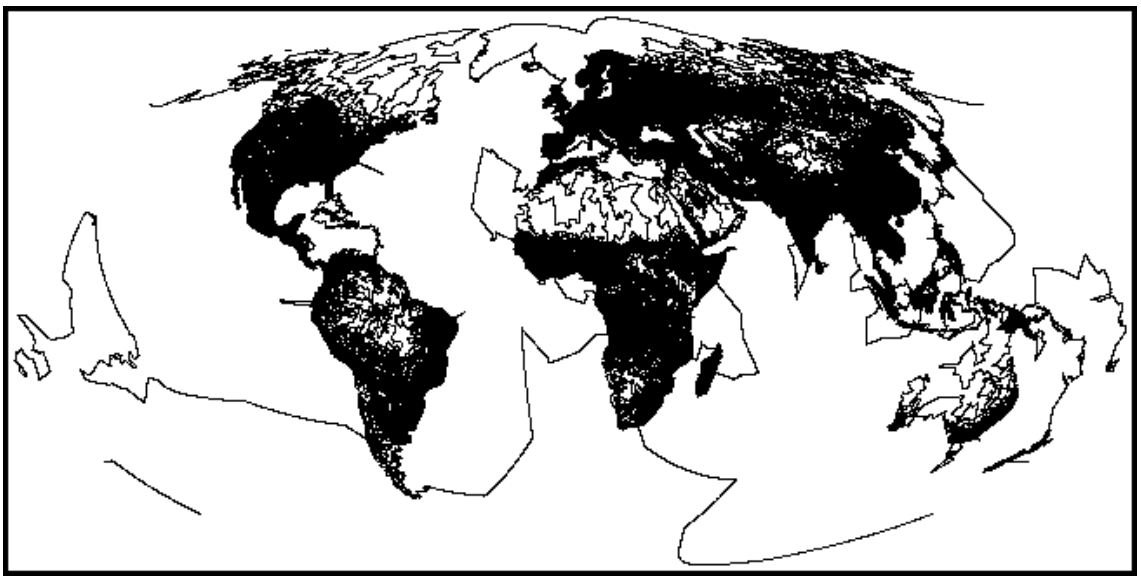

FIGURE 1.4 – Tour du World TSP [5]

borne jusqu'à 46.1% [49], puis un an plus tard, en 2012, à un record de 40% [60] au-dessus de l'optimalité pour la classe des problèmes graphiques.

1.3 Importance du Traveling Salesman Problem

Simple en apparence, il s'agit en réalité d'un des problèmes les plus étudiés en mathématiques informatiques, et sans doute celui ayant les applications les plus diverses et variées. Son intérêt et importance se traduisent tant au niveau théorique et académique qu'au niveau pratique et professionnel.

1.3.1 Intérêt théorique

Le TSP est un prototype de problèmes combinatoires ayant servi de force motrice pour l'emergence et le développement de la plus part des algorithmes, méthodes et heuristiques d'optimisation combinatoire. Et parmi les travaux développés autour du TSP et qui trouvent des applications sur d'autres problèmes, on peut citer le Simplex, Branch & Bound, Branch & Cut, les coupes de Gomory, la programmation linéaire, la programmation linéaire en nombres entiers, la programmation dynamique, la recherche locale stochastique et on en passe. Et ne serait-ce que pour le nombre de travaux qu'il a suscité et les avancées qu'il a provoqué, le TSP justifie amplement de l'intérêt qui lui est accordé.

Aussi, avec l'avènement de la théorie de la complexité, Karp a compilé une série de 21 problèmes NP-Complets, parmi lesquels on retrouve le problème du cycle Hamiltonien. La preuve donnée par Karp de l'NP-Complétude de ce dernier induit que le problème du voyageur de commerce est NP-Difficile. Les versions décisionnelle et euclidienne avec distances en nombres entiers [54] de ce dernier appartiennent quand à elles bel et bien à la classe des problèmes NP-Complets, et de trouver un algorithme de complexité polynomiale à l'un d'entre eux impliquerait que P=NP, ce qui résoudrait donc par la même occasion ce problème du millénaire. La possibilité d'un résultat d'une telle importance suffit à elle seule à justifier de l'intérêt qui est porté au TSP. Sans compter qu'une telle preuve pourrait rapporter à celui ou celle

qu'il l'apporte une récompense d'un montant d'un million de dollars américains ($1M USD) offerte par l'institut de mathématiques Clay [11].

Les problèmes de la classe NP-Complet, ont l'avantage d'appartenir à la fois à la classe NP et à la Classe NP-Difficile. On sait que chaque problème de ces classes peut être réduit en temps polynomial à n'importe quel autre problème de la même classe [37][63]. Alors de trouver une méthode polynomiale exacte pour le TSP, impliquerait grâce à de telles réductions mais aussi comme conséquence de P=NP, avoir trouvé une solution polynomiale à tous les problèmes de ces deux classes.

1.3.2 Importance pratique

La simplicité apparente du problème du voyageur de commerce n'enlève rien à son utilité. En effet, une très grande variété d'applications ont été répertoriées, en voici quelques-unes :

En industrie, les techniques du TSP sont utilisées pour determiner l'ordre de pose des micro-composants (transistors, portes logiques, etc...) pour réduire le temps de production des processeurs et micro-processeurs.

Toujours dans l'industrie, toujours pour les même raisons, mais cette fois-ci au niveau des circuits imprimés, le TSP est utilisé pour determiner l'ordre de perçage et de soudure des trous de connexion.

En astronomie, que ça soit sur terre ou dans l'espace, faire pivoter le télescope pour l'orienter et le pointer vers une succession de cibles à observer, est coûteux en temps et même en carburant dans le cas d'un télescope spatial, il est donc plus que judicieux d'user des méthodes permettant de minimiser ces mouvements.

En logistique, l'intérêt du TSP est évident même pour les non-initiés, en effet, plus courte est la distance parcourue pour livrer un ensemble de clients, moins coûteux est le carburant et plus importante devient la marge bénéficiaire, de même quand l'objectif est de minimiser le temps nécessaire à la réalisation de l'ensemble des livraisons, plus rapidement un livreur termine sa tournée, plus il pourra en faire en une journée ou sur une durée determinée.

En neurologie, le principe du problème est expliqué à des patients victimes de lésions au niveau du cortex pré-frontal, où sont situées les facultés de planification visuo-spatiale [8], à qui l'on demande de résoudre une instance donnée. Les médecins observent alors les délais pris par ces patients pour determiner la prochaine ville à visiter, ces délais servent ensuite à determiner la sévérité de leur lésion cérébrale.

1.4 Conclusion

Pour conclure ce chapitre, il est utile de prendre un instant de réflexion pour mesurer la portée du problème et les implications d'une solution polynomiale à celui-ci. Dans le film "Travelling Salesman" cité en introduction, quatre mathématiciens de génie sont recrutés et coercés par un agent du gouvernement américain pour résoudre secrètement le TSP, qualifié dans la bande-annonce du "problème le plus insaisissable de l'histoire et ayant échappé à toutes les tentatives de résolution". Un problème qui servirait de clé pour la résolution d'autres problèmes mathématiques, en effet, si une solution rapide existait, elle pourrait être convertie pour la résolution d'autres challenges difficiles tel le problème de factorisation des grands entiers en nombre premiers. Sachant qu'une grande partie des méthodes cryptographiques modernes tire sa sécurité de la difficulté de cette tache, on peut facilement imaginer les soucis et les répercussions qu'engendrerait l'apparition, voir le contrôle exclusif d'une méthode aussi innovante. Par solution rapide, nous ne parlons pas d'algorithme polynomial avec un exposant tel que 1000, en effet, un algorithme avec une complexité $O(n^{1000})$ est certes polynomial, mais il reste impraticable. Cependant, à travers l'histoire bien des méthodes sont apparues avec des exposants situés dans les quelques dizaines pour être réduits et améliorés avec le temps vers des valeurs plus accessibles de l'ordre de 2 à 7.

Chapitre 2

État de l'art sur les méthodes de résolution du TSP

2.1 Introduction

Pour résoudre le problème du voyageur de commerce il est utile de le représenter sous forme de graphe. Alors :

Soit $G = (S, A, p)$ un graphe non-orienté, complètement connecté, d'ordre n, tel que :

— S est l'ensemble des n nœuds ou sommets du graphe G.
— A l'ensemble des arêtes reliant chaque paire de nœuds dans S.
— Et $p : A \to \mathbb{R}^+$ une fonction assignant à chaque $a \in A$ un poids $p(a)$.

L'objectif est de trouver un cycle ou circuit Hamiltonien de poids minimal.

2.1.1 TSP symétrique vs. TSP asymétrique

Il existe deux types de TSP, le TSP symétrique (sTSP) et le TSP asymétrique (aTSP). Dans le premier, le coût pour relier une ville A vers une ville B est le même pour relier B vers A, alors que dans le second, il est permis d'avoir des coûts différents, la distance $A \to B$ peut différer de la distance $B \to A$, ce qui peut par exemple se produire sur une carte routière contenant des voies à sens unique ou des voies plus encombrées dans un sens que dans l'autre.

En théorie des graphes, cela se traduit par :

TSP symétrique : $\forall s, s' \in S,\ p(s, s') = p(s', s)$

TSP asymétrique : $\exists (s, s')$, tel que : $p(s, s') \neq p(s', s)$

Un TSP asymétrique peut être réduit vers un TSP symétrique en doublant à peine la taille du problème sans que cela ne change le TSP à résoudre [39][**?**]. Dans ce qui suit nous ne considérerons donc que le cas du TSP symétrique.

2.1.2 Inégalité triangulaire

Dans le cas des TSP métriques et plus particulièrement les TSP euclidiens, il existe une contrainte d'inégalité triangulaire, c'est-à-dire, que la distance directe en

deux villes ne doit pas dépasser la distance qui utilise un détour par une troisième ville. En effet, si pour aller de A à B, il y'a un chemin plus optimal passant par un point tiers C, alors il suffit de mettre à jour la distance ou le coût séparant A et B par la distance ou le coût du chemin passant par ce point C.

En théorie des graphes, cette contrainte se traduit par l'inégalité suivante :

$\forall x, y, z \in S,\ p(x, y) + p(y, z) \geq p(x, z)$

2.1.3 Classification des méthodes de résolution du TSP

Il existe principalement, deux classes de méthodes pour la résolution du problème du voyageur de commerce. Une classe dite exacte, dont les méthodes résolvent le problème à l'optimalité, mais jusqu'à présent tous les algorithmes de cette classe arborent des complexités exponentielles. Et une classe dite approchée, dont le but est de fournir rapidement des tours de bonne qualité, dont le coût ne dépasse pas trop celui d'un tour optimal.

Ces classifications sont détaillées dans la figure 2.1.

2.2 Algorithmes exacts

Le premier algorithme exact proposé a été l'algorithme naif de l'énumération. Sa complexité est de l'ordre de $O(n!)$, ou plus exactement $O(\frac{(n-1)!}{2})$ car une fois avoir choisi la ville départ et d'arrivée, il n'y a plus que $n-1$ villes à faire permuter, ce qui donne $O((n-1)!)$, et comme un tour peut être emprunté dans un sens comme dans l'autre vu que les distances sont symétriques, alors $\frac{(n-1)!}{2}$ permutations suffisent.

En 1962, Held et Karp ont substantiellement amélioré cette complexité en proposant un algorithme de programmation dynamique borné par une complexité de $n^2 2^n$. Ce fut une avancée majeure dans le domaine mais qui reste loin de satisfaire les contraintes et les limites pratiques.

La plus part des algorithmes exacts développés à ce jour sont basés sur Branch & Cut. Ils fournissent une garantie d'optimalité mais aucune garantie de complexité.

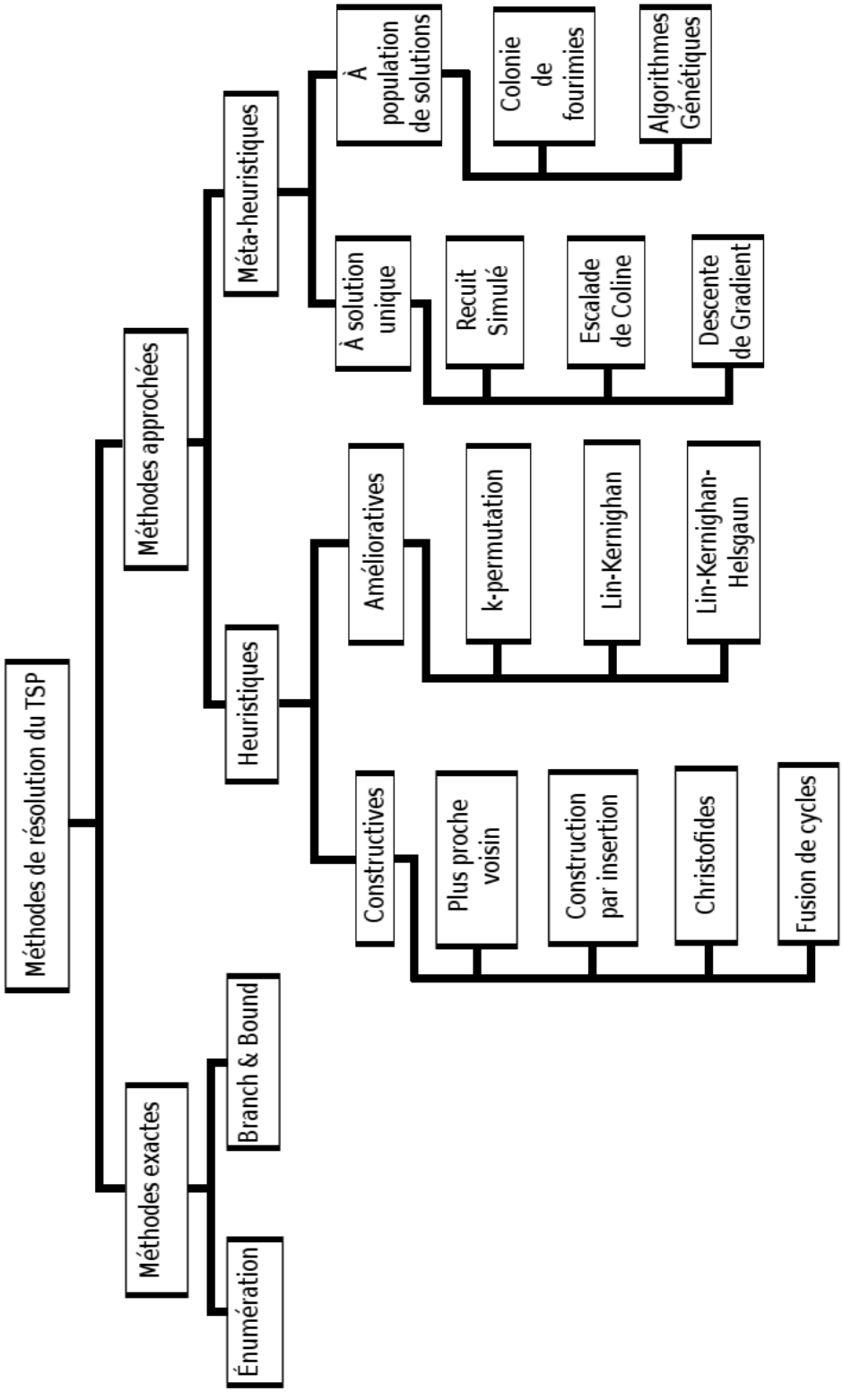

FIGURE 2.1 – Classification des méthodes de résolution du problème du voyageur de commerce

Il existe sur le marché plusieurs programmes de résolution de TSP, parmi ceux-ci nous pouvons citer LKH et Concorde. Ce dernier est justement basé sur une implémentation de Branch & Cut.

2.2.1 Concorde

Concorde est un programme informatique réalisé dans le but de résoudre des instances symétriques du problème du voyageur de commerce. En d'autres termes, Concorde est un solver de TSP. Son code de plus de 130,000 lignes est écrit en langage ANSI C et est disponible pour une utilisation académique. Concorde a été utilisé pour résoudre à l'optimalité la totalité des 110 instances de TSPLIB, y compris celle de 85,900 villes.

Au cœur de Concorde, on retrouve la méthode des plans sécants couplée à ILOG-CPLEX, un solver de programmes linéaires.

Lors de son exécution, le progrès de cette méthode est estimé par l'augmentation de la valeur optimale de son PL, mais plus elle ajoute de coupes, plus la relaxation se ressert et les augmentations se font petites. Quand elles deviennent trop petites, le programme opère une séparation (*Branching*) et se retrouve alors avec deux ensembles de tours. et il applique à nouveau la méthode des plans sécants à un des deux ensemble puis à l'autre. Par la suite, chacun de ces deux sous-problèmes peut à son tour être séparé en sous-sous-problèmes si nécessaire et ainsi de suite. Dans l'arbre binaire de sous problèmes qui en résulte, chaque feuille a soit été résolue par la méthode des plans sécants, soit été ignorée parce que la valeur optimale de sa relaxation s'est révélée être au moins aussi grande qu'un tour déjà connu.

2.2.2 Le tour de Suède

Un exemple d'instance où Concorde a contribué à la résolution et à l'apport d'une preuve d'optimalité est le tour de Suède visitant les quelques 24,978 villes et villages du pays, une instance issue de TSPLIB, et qui a été abordée par des équipes de recherche du plus haut niveau utilisant le top-4 des méthodes de recherche développées à ce jour.

Longueur	Date	Méthode	Équipe
855618	4 Septembre 2001	Tour Merging	Cook and Seymour
855612	20 Septembre 2001	LKH	Helsgaun
855610	30 Septembre 2001	LKH Merge	Helsgaun
855602	16 Mars 2003	Hybrid Genetic	Hung Dinh Nguyen
855597	18 Mars 2003	LKH	Helsgaun

TABLE 2.1 – Tableau résumant les progrès réalisés au fil des tentatives de résolution du Tour de Suède

L'amélioration finale a été réalisée par Keld Helsgaun en utilisant une version de son code LKH. L'optimalité de la valeur 855,597 a été prouvée par Concorde.

Le solver Concorde peut prendre en paramètre le meilleur tour connu pour une instance de TSP si un tel tour est disponible. Mais étant un solver exact, Concorde est conçu pour trouver des solutions optimales quelque soit la qualité de l'estimation donnée en entrée, mais la connaissance d'un bon tour permet d'affiner les paramètres définis dans le code.

Les concepteurs de Concorde ont déclaré que dans le cas du TSP de Suède, les résultats des tentatives répertoriées ont guidé leurs choix dans l'approche de la solution finale au problème. En effet, les étapes finales qui ont amélioré la borne inférieure de 855,595 vers la valeur optimale de 855,597 ont nécessité environ 8 ans de temps de calcul réparti sur un réseau de stations Linux, et sans la connaissance du tour de 855,597, ils n'auraient pas pris la décision d'effectuer la dernière amélioration de la borne.

Au final, la résolution et la vérification exacte de cette instance se seront étalées de Mars 2003 jusqu'en Mai 2004 et auront coûté un total cumulé de *84.8 années* de temps CPU sur un cluster de 96 stations de travail de modèle Intel Xeon 2.8GHz fournies par l'école d'ingénierie industrielle et des systèmes, de l'université Georgia Tech aux États-Unis.

Pour comparaison, l'instance VLSI de 85,900 villes fournie par les laboratoires Bell fut résolue par Concorde entre février 2005 et avril 2006 en ayant recours à la parallélisation sur des stations AMD Opteron 250 de 2.4GHz avec un temps CPU total de *136ans*.

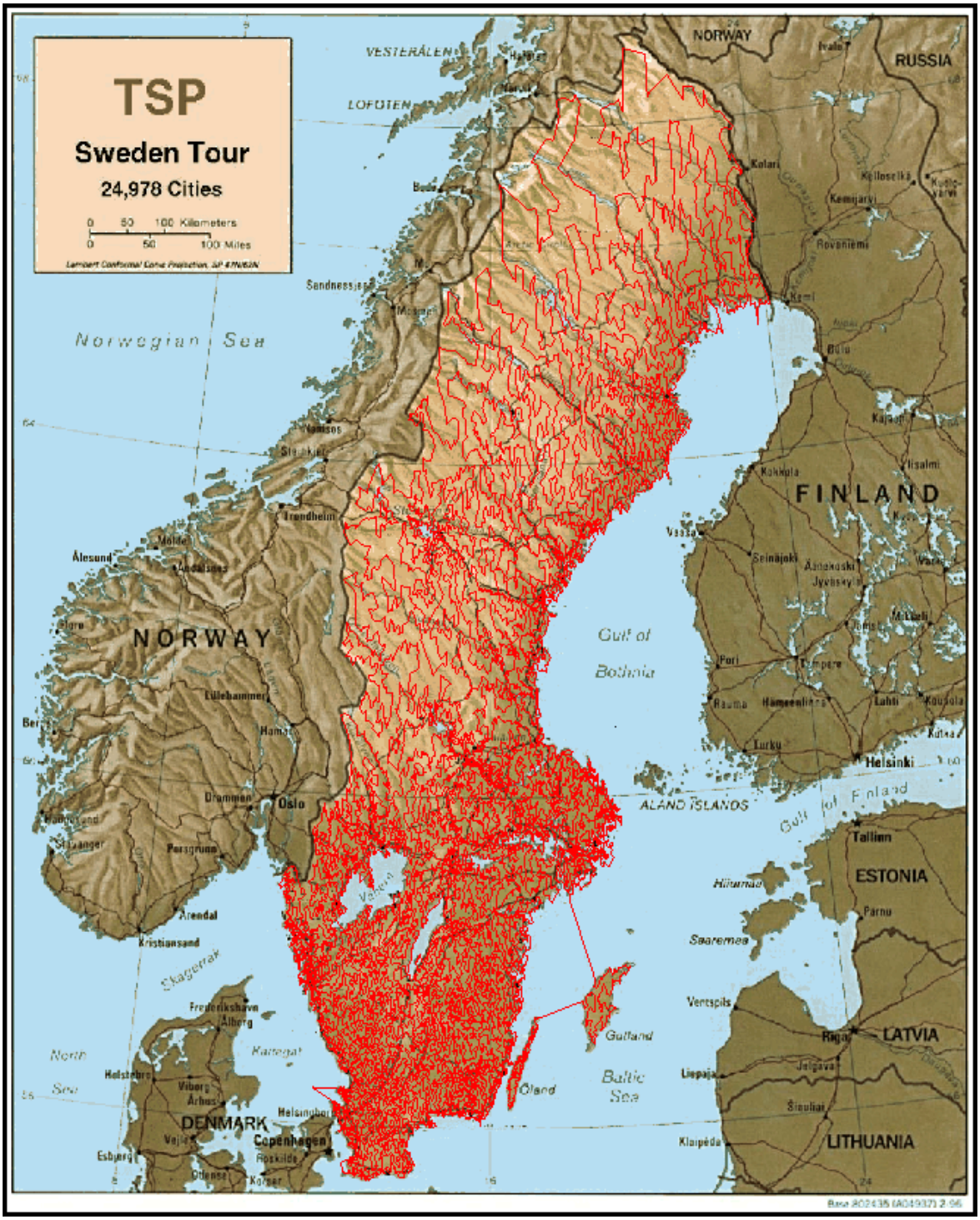

FIGURE 2.2 – Tour de Suède en relief [6]

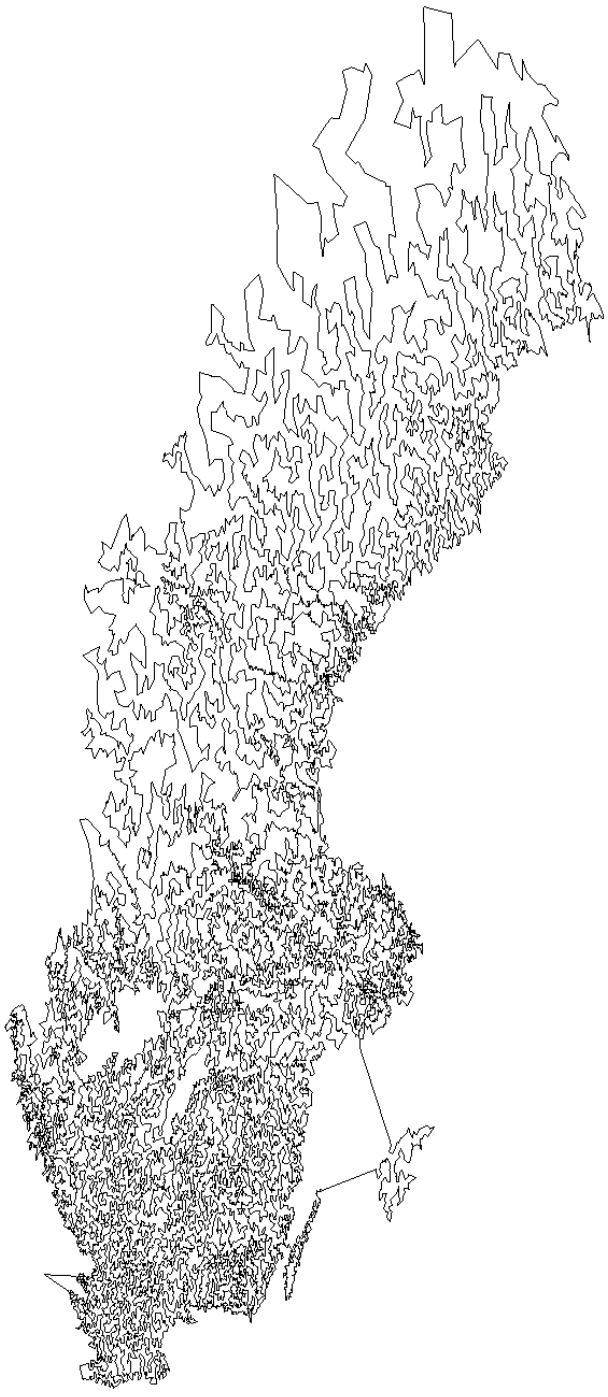

FIGURE 2.3 – Tour de Suède [6]

FIGURE 2.4 – Tour de Suède en courbe pleine [6]

2.3 Méthodes approchées

Les algorithmes exacts ne présentant que des complexités impraticables, les chercheurs se sont orientés en parallèle vers la construction d'heuristiques et de métaheuristiques permettant d'obtenir de bons tours rapidement mais sans garantie d'optimalité.

Ces heuristiques peuvent être classées selon deux catégories, les heuristiques constructives et les heuristiques amélioratives.

2.3.1 Heuristiques constructives

Le principe de base de cette catégorie est de construire un tour en partant de zero et en étendant itérativement des solutions partielles. Il existe principalement trois types d'heuristiques de ce genre :

— Extension itérative d'une solution partielle.

— Construction puis *fusion* de tours partiels.

— Arbre couvrant de poids minimal.

2.3.1.1 Heuristique du plus proche voisin

Les étapes de cette heuristique sont :

- Choisir aléatoirement un nœud source $s \in S$;
- Le pointeur du nœud courant $c = s$;
- Tant que $\exists x \in S$ non visité :
  - Sélectionner v, le voisin le plus proche de c, tel que : $\forall v, v' \in S,\ p(c, v) \leq p(c, v')$;
  - Aller de c vers v ;
  - $c = v$;
- Aller de c vers s ;

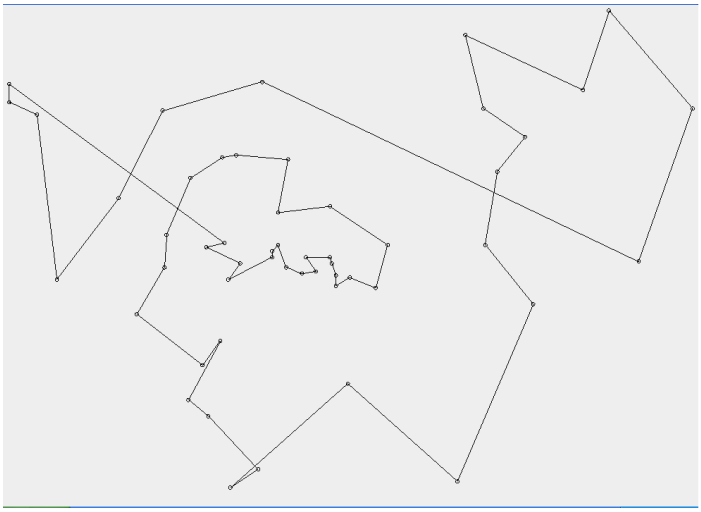

FIGURE 2.5 – Exemple d'exécution de NNH sur l'instance de 52 villes berlin52 issue de TSPLIB

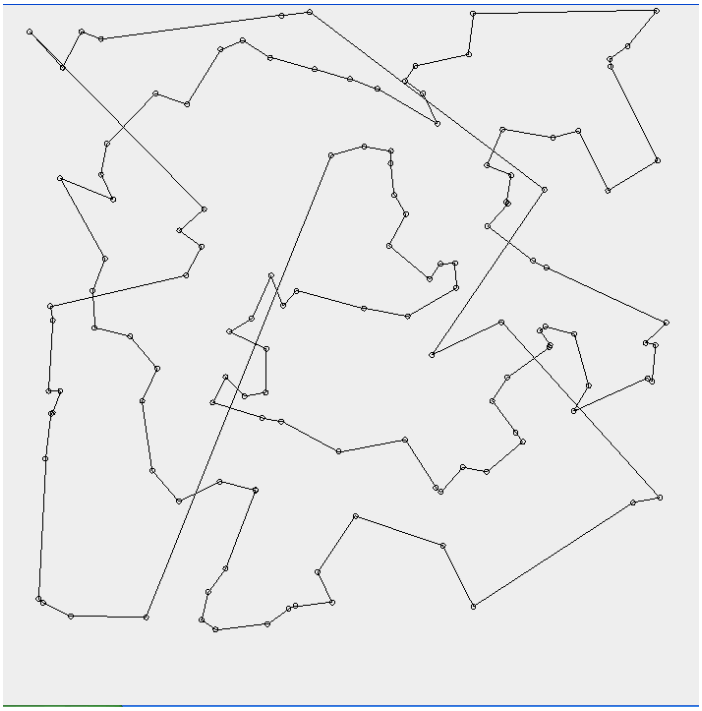

FIGURE 2.6 – Exemple d'exécution de NNH sur l'instance de 130 villes ch130 issue de TSPLIB

L'heuristique du plus proche voisin, Nearest Neighbor Heuristic (NNH) en anglais, est une heuristique très rapide, sa complexité est de l'ordre de $O(n^2)$, mais son ratio d'approximation dans le pire des cas est fonction du nombre de villes ou de nœuds :

$$NNH/OPT \leq 0.5(\lceil log_2 n \rceil + 1)$$

Le défaut de l'heuristique du plus proche voisin c'est qu'elle a un comportement glouton en ne poursuivant que des minima locaux. Dans le cas de l'instance ch130 par exemple, nous pouvons observer dans la figure 2.6, qu'en bas à droite, l'heuristique a épuisé son stock de voisins *proches*, et qu'elle se retrouve alors obligée d'effectuer de grands sauts pour relier les nœuds qu'elle avait raté ou ignoré. Des sauts dont certains lui font croiser le chemin qu'elle a déjà parcouru. D'ailleurs, une fois qu'elle a visité tous les nœuds et qu'elle s'est retrouvée en bas à gauche, elle a du effectuer un saut qui a croisé 5 fois son chemin pour relier le point de départ, comme nous pouvons le voir, toujours dans la même figure.

Une règle informelle utile pour savoir si un tour obtenu par l'heuristique du plus proche voisin est bon, dit qu'il suffit d'observer les derniers sauts réalisés par celle-ci, s'ils ont relativement la même taille que ceux de départ, alors le tour est bon, s'ils sont beaucoup plus grands, il existe probablement de bien meilleurs tours qu'il faut rechercher.

2.3.1.2 Heuristique de l'arbre couvrant de poids minimal

Les étapes de cette heuristique sont :

- Rechercher un arbre couvrant de poids minimal pour le graphe G.
- Effectuer le parcours préfixe de l'arbre trouvé.
- Retourner à la racine de l'arbre.

La complexité de cette heuristique dépend de la méthode utilisée pour trouver l'arbre couvrant de poids minimal. Il existe plusieurs algorithmes polynomiaux pour réaliser cette tache, parmi lesquels on peut citer :

— L'algorithme de Boruvka ayant une complexité en $O(|A|ln(|S|))$.

— L'algorithme de Kruskal ayant une complexité en $O(|A|log(|A|))$.

— Et enfin, l'algorithme de Jarnik, connu aussi sous les noms d'algorithme de Prim ou d'algorithme de Prim-Dijkstra, qui l'ont tous les trois découvert indépendamment, en 1930, 1959 et 1959 respectivement. Un algorithme ayant une complexité en $O(|S|^2)$.

Vu que le graphe G d'un TSP est complet alors $|A| = |S|^2 - |S|$, ce qui implique que l'algorithme ayant la meilleure complexité pour la recherche d'un arbre couvrant de poids minimal dans le cadre d'un problème du voyageur de commerce est l'algorithme de Jarnik.

Aussi, vu que la complexité du parcours de l'arbre est en $O(n)$, alors la complexité de cette heuristique est dominée par celle de l'étape de la recherche de l'arbre couvrant de poids minimal, ce qui implique que la complexité de l'heuristique elle même est en $O(|S|^2)$, ou plus simplement en $O(n^2)$.

Le ratio d'approximation de cette heuristique est de 2, en voici la preuve :

- $OPT \geq ACPM$
- $2 * ACPM \geq Resultat$ (du parcours préfixe)
- $\frac{Resultat}{OPT} \leq \frac{2*ACPM}{ACPM} = 2$

2.3.1.3 Algorithme de Christofides

Les étapes de cette heuristique sont :

- Rechercher un arbre couvrant de poids minimal T pour le graphe G.
- Calculer l'ensemble I des sommets de degré *impair* dans l'arbre T, en d'autres termes, $I = \{s \mid deg(s)\ mod\ 2 = 1\ dans\ T\}$
- Calculer un couplage parfait M de poids minimum dans le sous-graphe $G[I]$ induit par I dans G.
- Combiner les arêtes des deux graphes T et M pour former un multigraphe H dont tous les sommets sont de degré *pair*.
- Calculer un tour eulérien sur H.

- Transformer le tour Eulérien trouvé en un tour Hamiltonien en supprimant les répétitions de sommets dans le circuit.

La complexité de cette heuristique est en $O(n^3)$ et son ratio d'approximation est de 1.5, en voici la preuve :

- $T \leq OPT$
- $M \leq OPT/2$
- $T + M \leq \frac{3}{2}OPT$ $\frac{T+M}{OPT} = \frac{3}{2}$
- $\frac{T+M}{OPT} \leq \frac{3}{2}$

2.3.1.4 Heuristique de construction par insertion

L'heuristique de la construction par insertion possède deux variantes de départ, l'une commence avec *un* nœud seulement, l'autre à partir de l'enveloppe convexe du graphe. La procédure d'insertion quand à elle, consiste à ajouter le reste des nœuds, un par un au tour partiel déjà construit, en choisissant à chaque insertion, la position qui minimise l'augmentation de la longueur du tour.

Le choix du prochain nœud à insérer se fait selon une politique choisie. Et parmi les variantes de politiques déjà découvertes, nous pouvons citer :

— **Plus proche insertion :** Choisir le nœud le plus proche du tour déjà construit.

— **Plus lointaine insertion :** Choisir le nœud le plus loin du tour déjà construit.

— **Meilleure insertion :** Choisir le nœud qui minimise l'augmentation de la longueur du tour déjà construit.

— **Insertion aléatoire :** Choisir aléatoirement le prochain nœud à insérer.

Les politiques de l'insertion aléatoire et de la plus lointaine insertion possèdent un ratio d'approximation de $O(log\ n)$.

Quand aux politiques de la plus proche et de la meilleure insertion, elles possèdent un ratio d'approximation de 2.

Mais dans la pratique, les politiques *aléatoire* et *plus lointaine* produisent des résultats bien meilleurs que ceux des politiques *plus proche* et *meilleure* [35], en effet, ces deux dernières convergent plus rapidement vers les minima *locaux*.

2.3.1.5 Heuristique de construction par fusion de cycles

L'idée de la construction d'un tour par fusion itérative de cycles, est de produire au final un tour ayant un fort degré de similarités locales avec la solution optimale.

Les étapes de cette heuristique sont :

- Commencer avec n cycles contenant *un* sommet chacun.
- Répéter :
  - Trouver les deux cycles les plus proches.
  - Les fusionner en un seul cycle d'une manière minimisant la longueur de l'union.
- Jusqu'à obtenir un tour de n sommets.

Le ratio d'approximation de cette heuristique est de 2.

2.3.2 Heuristiques amélioratives

Le principe de base de cette catégorie est de commencer avec un tour donné, généré par une heuristique constructive, puis de l'améliorer itérativement tant que c'est possible et tant que le temps imparti ne s'est pas écoulé.

2.3.2.1 Heuristiques 2-opt, 3-opt et k-opt

2-opt : est un algorithme itératif de recherche locale plutôt simple, et qui consiste à parcourir le tour courant à la recherche de deux arêtes (a, b) et (c, d) qui peuvent être remplacées par les arêtes (a, c) et (b, d) ou (a, d) et (b, c) pour former un nouveau tour plus court. Si le nouveau tour ne l'est pas, alors nous gardons la configuration précédente et nous recherchons une autre paire d'arêtes.

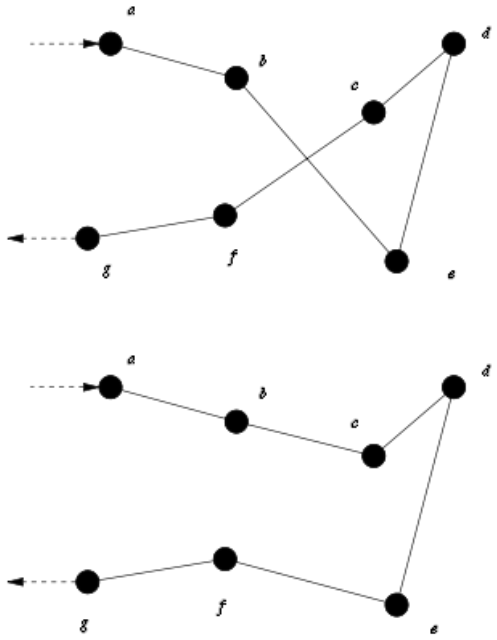

FIGURE 2.7 – Exemple de permutation 2-opt [66]

Il faut noter que lorsque l'on inverse l'ordre de parcours de deux villes, il faut également inverser l'ordre de parcours de toutes les villes qui se trouvent entre elles. Et il s'agit là d'une opération qui peut coûter cher à la longue. Une des améliorations possibles dans ce cas là est d'inverser la partie qui contient le moins de villes.

Il existe également d'autres techniques qui permettent d'améliorer la complexité effective de l'heuristique, comme par exemple le calcul d'une liste des m voisins les plus proches pour chaque sommet, afin de réduire le nombre d'arêtes candidates avec lesquelles comparer chaque arête.

3-opt : ne diffère que très légèrement de 2-opt, dans le mesure où au lieu de choisir une paire d'arêtes à échanger, nous en choisissons un triplet. Ce triplet est remplacé par toutes les combinaisons possibles et la meilleure est gardée.

k-opt : n'est qu'une extension de 2-opt et 3-opt, dans la mesure où elle permet de choisir des k-uplets d'arêtes à tester. Mais dans la pratique, il est rare d'aller au delà de 3-opt puisque les temps d'exécution deviennent rapidement prohibitifs sans que la solution ne soit significativement améliorée.

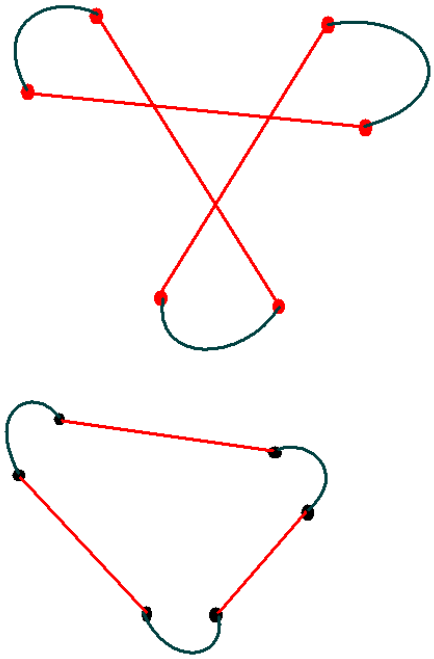

FIGURE 2.8 – Exemple de permutation 3-opt

Un tour est considéré 2-optimal s'il a épuisé toutes les possibilités d'amélioration par 2-opt. Un tour 3-optimal est également 2-optimal, et de manière générale, un tour k-optimal est aussi (k-1)-optimal.

Les procédures 2-opt, 3-opt et k-opt peuvent finir avec une solution optimale mais il se peut également qu'elle terminent dans un minimum local et ne pas atteindre l'optimalité.

Malheureusement, ces heuristiques ne fournissent aucune garantie de complexité, et peuvent théoriquement dériver vers l'exponentielle. Cependant, dans les faits, des complexités tournant autour de n^2, n^3 et n^k pour 2-opt, 3-opt et k-opt, respectivement, sont observées.

2.3.2.2 Lin-Kernighan

Lin-Kernighan : est une généralisation adaptative de k-opt et qui consiste à tester un nombre variable d'arêtes à chaque étape selon des règles prédéfinies.

Cette heuristique fournit de bons résultats et est très populaire dans le milieu. Elle est souvent utilisée comme sous-routine d'heuristiques plus avancées.

2.3.3 Métaheuristiques

Dans le cadre du TSP, quelques métaheuristiques inspirées de la nature ont également été développées, parmi lesquelles nous pouvons citer le recuit simulé, les algorithmes génétiques ou encore les algorithmes de type colonie de fourmis, comme le Ant System par exemple.

2.3.3.1 Optimisation par colonie de fourmis

S'il vous est déjà arrivé d'oublier une sucrerie à l'air libre pendant quelques jours, vous avez certainement pu constater des fourmis venir récupérer ce festin pour l'emmener vers leur colonie. Si vous preniez une minute pour les observer, vous remarqueriez que celles-ci se déplacent généralement en ligne quasi-droite et qu'elles ne dévient que pour contourner un obstacle. La raison pour laquelle tant de fourmis empruntent le même chemin est parce que celles-ci suivent la trace de phéromones laissée par leurs congénères entre leur colonie et l'emplacement de sources de nourriture. Le détail important ici et qui a été repris pour le développement de cette métaheuristique, est que plus le chemin séparant le nid et la source de nourriture est court, plus la trace de phéromones laissée par une fourmi est concentrée. Ainsi, quand une fourmi trouve un raccourci, elle laissera une trace plus importante et plus concentrée que si elle avait suivi un chemin plus long faisant des détours. Et quand d'autres fourmis croisent ce chemin et qu'elles le suivent, elles sécrètent elles aussi des phéromones et renforcent par la même occasion cette trace.

Dans cette métaheuristique, nous commençons donc avec un groupe de fourmis virtuelles, typiquement, une vingtaine, que nous plaçons aléatoirement dans des villes différentes et à qui nous demandons de trouver un tour. Chacune de ces fourmis commence alors à se déplacer de ville en ville sans avoir le droit d'en re-visiter une qu'elle a elle même traversé, à moins que ça ne soit pour compléter le tour. La fourmi ayant trouvé le tour le plus court dépose alors sur les arêtes qu'elle a traversé

une trace de phéromones inversement proportionnelle à la longueur du tour. Lors des prochaines tentatives, cette trace de phéromone sera prise en compte lorsqu'une fourmi choisira vers quelle prochaine ville se déplacer. Cette procédure est répétée jusqu'à avoir trouvé un tour satisfaisant ou jusqu'à ce que le temps imparti se soit écoulé.

La force et l'efficacité des fourmis se trouvent aussi dans leur nombre, mais dans notre cas, l'ajout de fourmis a un coût en complexité et il est donc laissé pour choix à tout un chacun dans son implémentation.

2.4 Conclusion

Dans le chapitre précédent, nous avons pu observer la longue et riche histoire du problème du voyageur de commerce. Dans celui-ci nous avons essayé de présenter succinctement quelques-unes des méthodes les plus populaires issues de ces travaux. La recherche de nouvelles heuristiques ne s'arrête pas là et le chapitre qui suit est notre modeste contribution

Chapitre 3

Proposition d'une nouvelle heuristique

3.1 Introduction

Au vu du nombre et de la variété des heuristiques existantes tentant de résoudre le problème du voyageur de commerce, vous pourriez et vous seriez en droit de vous demander "pourquoi une nouvelle heuristique ?" et "pourquoi ne pas étendre et construire sur celles déjà existantes ?" La raison et la réponse à ces deux questions est que les méthodes existantes, comme celles présentées dans le chapitre précédent, sont suffisamment anciennes et matures pour avoir atteint un stade de *plateau* en terme d'amélioration. Ce que nous voulons dire par là est qu'après toutes ces années d'optimisation et de raffinement, elles restent ce qu'elles sont, c'est-à-dire seulement des heuristiques et par définition seulement des méthodes approchées qui ne garantissent pas l'optimalité et qui pour la plus part, s'en éloigne en proportion avec la taille du problème. C'est là la raison qui nous a poussé à poursuivre cette voie et à proposer cette nouvelle heuristique que nous espérons pouvoir raffiner à l'avenir pour peut-être, pourquoi pas, atteindre l'optimalité.

Dans ce qui suit, nous commencerons par expliquer le principe de cette heuristique puis nous présenteront l'implémentation en Java que nous en avons fait, et puis enfin nous exposerons quelques résultats d'essais que nous avons entrepris et quelques comparaisons que nous avons réalisé avec quelques heuristiques existantes.

3.2 Solution proposée

3.2.1 Description

Notre algorithme est une heuristique constructive dont le principe est de parcourir un graphe condensé du graphe combinatoire. Par graphe combinatoire, nous voulons dire le graphe de l'espace total de solutions.

Le parcours de ce graphe peut se faire soit par Dijkstra soit en largeur (Dijkstra étant évidement meilleur), en ne considérant que le meilleur chemin qui mène à un nœud donné, par exemple, si l'on prend une instance du problème possédant 16 nœuds, alors nous choisirons un nœud qui servira de départ et d'arrivée, puis l'on

placera le reste des nœuds sur une matrice 15*15 où chaque nœud occupera toutes les cases d'une colonne, et tel que chaque nœud est lié par des arcs à tous les nœuds de la ligne suivante, et nous traverserons le graphe avec l'algorithme de Dijkstra à la condition qu'on ne prenne pas en considération les nœuds des colonnes déjà traversées.

Dans la figure 3.1, vous pouvez constater de quoi a l'air un tel graphe condensé dans le cas d'une instance de 5 nœuds. Les arcs verticaux liant les nœuds portant le même numéro ne sont pas représentés dans le graphe et ce de manière délibérée afin d'éviter l'encombrement visuel.

Notons que notre graphe condensé est un graphe pondéré orienté acyclique, et la méthode de parcours étant une variante de l'algorithme de Dijkstra, nous obtenons une méthode dont la complexité est en $O(|A| + |S|)$ [15], où $|A|$ est le nombre d'arcs et $|S|$ est le nombre de sommets.

Notre graphe possède $(n-1)$ nœuds par ligne sur $(n-1)$ lignes, en plus des nœuds de départ et d'arrivée qui sont donc au nombre de 2, ce qui nous donne exactement $(n-1)^2 + 2 = n^2 - 2n + 3$ noœuds au total.

Notre graphe possède également $(n-1)$ arcs partant du nœud de départ et le même nombre arrivant au nœud de destination, ce qui se monte à $2(n-1)$. En plus, comme il y'a $(n-1)$ nœuds par ligne et que de chacun de ses nœuds partent $(n-1)$ arcs vers la ligne suivante, alors il y'a un nombre de $(n-1)^2$ arcs partant de chaque ligne vers sa suivante. Il y'a $(n-1)$ lignes, mais de la dernière, ne sortent que (n-1) arcs, vers la destination, que nous avons déjà comptabilisé, alors il y'a un total de $(n-1)^2(n-2)$ arcs situés dans les interlignes.

La somme de tous ses arcs donne :

$$
\begin{aligned}
&2(n-1) + (n-1)^2(n-2) \\
&= 2n - 2 + (n^2 - 2n + 1)(n-2) \\
&= 2n - 2 + n^3 - 2n^2 + n - 2n^2 + 4n - 2 \\
&= n^3 - 4n^2 + 7n - 4
\end{aligned}
$$

Pour en revenir à la complexité, nous disions donc $O(|A| + |S|)$, ce qui se traduit par :

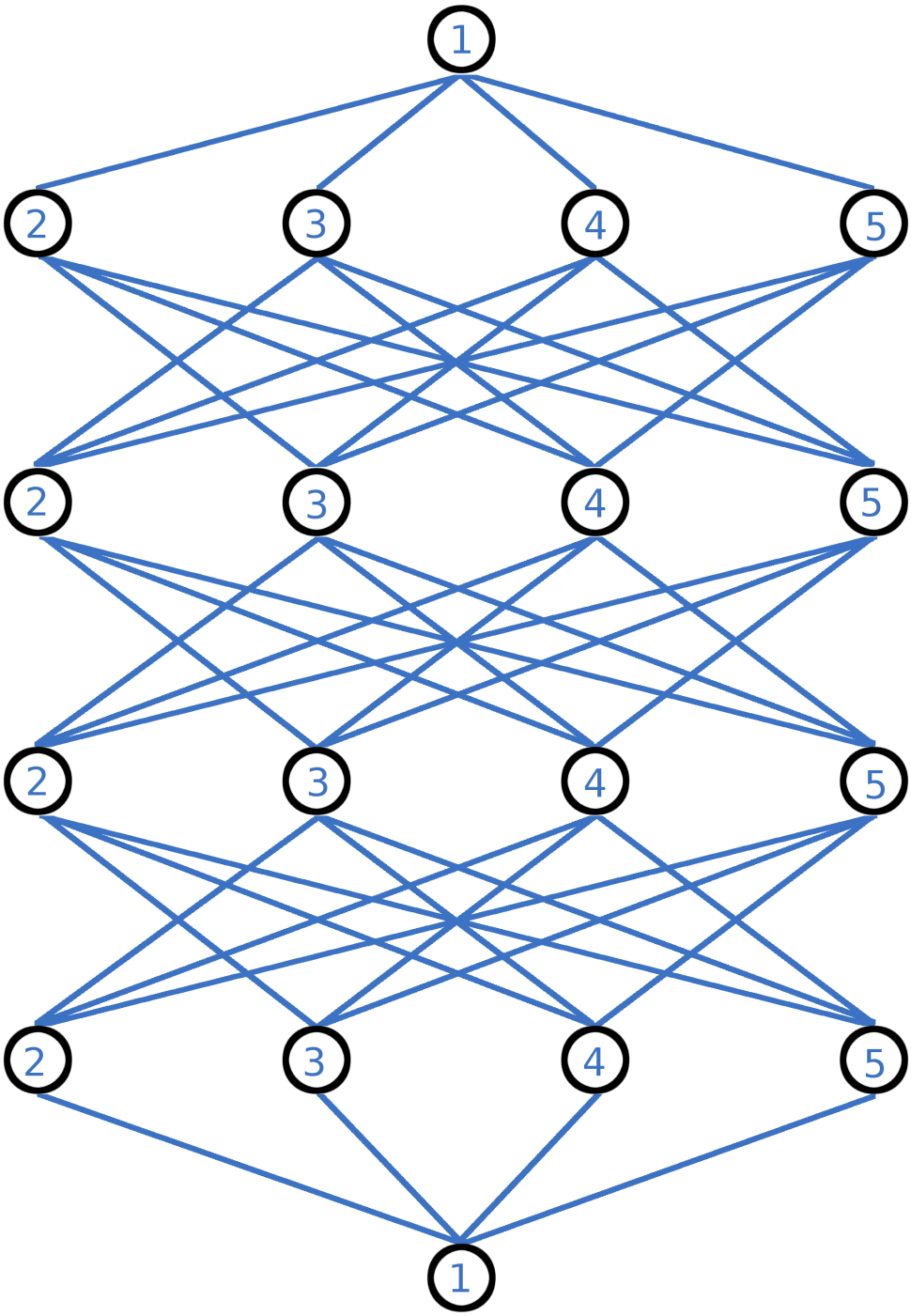

FIGURE 3.1 – Représentation du graphe condensé

$O((n^3 - 4n^2 + 7n - 4) + (n^2 - 2n + 3))$
$= O(n^3 - 3n^2 + 5n - 1)$
$= O(n^3)$

Nous avons donc construit une heuristique dont la complexité est inférieure à $O(n^3$ et qui est capable de traiter des instances comme celle de 130 villes en une seconde environ sur un vieux PC Pentium 4 de près de 10 ans. Sachant que 130 ! (factoriel) est de l'ordre de 10^{219} combinaisons différentes, un nombre bien supérieur au nombre d'atomes dans l'univers connu et qui est de l'ordre de 10^{82}.

Il est aussi possible de réaliser un départ depuis plusieurs ou depuis chacune des n villes au prix d'une complexité augmentée ou en ayant recours à la parallélisation, en effet cette opération est facilement parallélisable sans nécessiter d'adaptation spécifique.

De plus, deux des avantages majeurs de cette heuristique sont qu'elle s'adapte aussi bien aux instances symétriques qu'aux instances asymétriques sans nécessiter de modification et sans nécessiter de réduction asymétrique vers symétrique, et qu'elle s'adapte aussi pour la résolution de la version chemin (Path) du TSP en ne modifiant qu'une seule condition.

Un autre avantage également, est qu'elle fonctionne aussi bien sur des instances métriques et euclidiennes que sur des instances arbitraires ne respectant pas l'inégalité triangulaire.

Et enfin, la version de base tient sur 200 lignes et a donc une faible empreinte sur la mémoire ce qui la rend facile à implémenter sur les appareils mobiles, alors que le programme de référence "Concorde" occupe plus de 100.000 lignes.

Au cours de nos essais nous avons identifié et corrigé deux des défauts de cette heuristique :

— Il existe un cas où l'heuristique produit un tour qui croise son propre chemin comme illustré dans la figure 3.2.

— Il arrive que la méthode traverse un ensemble de villes dans un certain ordre qui peut être amélioré en inversant les positions de la première et de la dernière ville de ce bout de chemin.

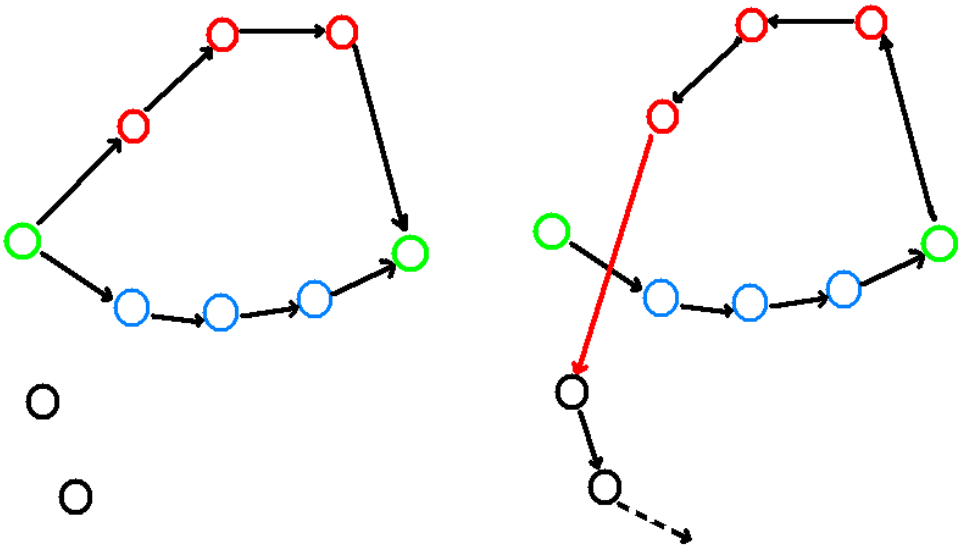

FIGURE 3.2 – Défaut de croisement
Ce type de parcours est produit car en arrivant au second nœud vert, le meilleur chemin *jusque-là* est celui passant par les nœuds bleus.

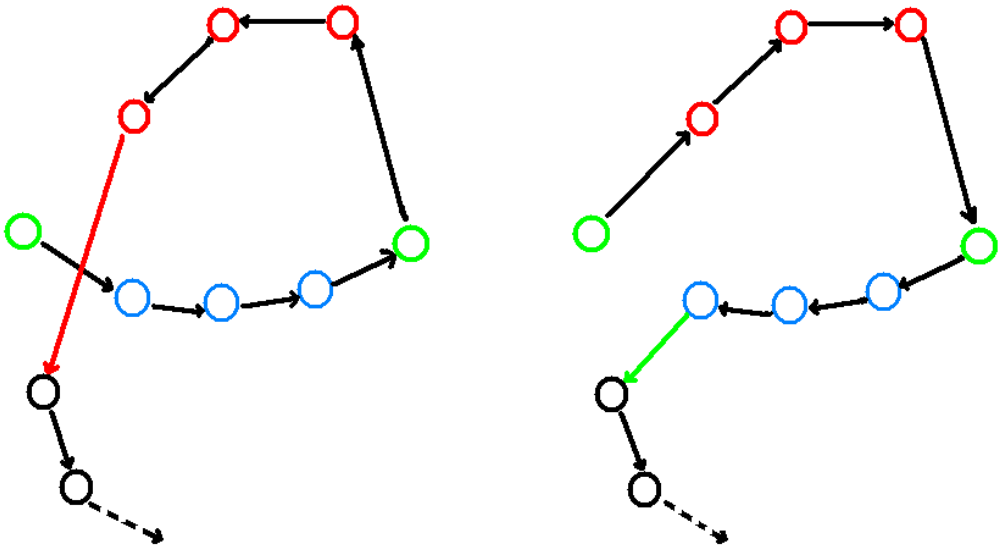

FIGURE 3.3 – Rectification du défaut de croisement
La rectification se fait en éliminant le croisement et en inversant l'ordre de parcours des villes intermédiaires.

La raison de ces deux défauts est qu'un nœud choisi parmi les chemins entrants, le meilleur *jusque-là*, mais emprunter ce chemin peut correspondre à un minimum local et non à un minimum global.

Pour corriger, ces deux défauts, il suffit d'affiner le tour en éliminant les croisements et en testant si l'inversement de positions améliore la longueur du tour. C'est deux opérations sont d'ordre n^2.

Ce qui ne nous a sauté aux yeux que très récemment, c'est que pour notre complexité inférieure à $O(n^3)$, nous arrivons tout de même à obtenir n chemins distincts de bonne qualité qui peuvent servir dans une métaheuristique améliorative à population de solutions comme les algorithmes génétiques ou l'optimisation par colonie de fourmis.

3.2.2 Implémentation

Nous avons réalisé une implémentation simple en Java de cette heuristique. Nous avons prioritisé la simplicité du code dans cette itération mais nous travaillons également sur d'autres versions plus poussées qui éliminent un grand nombre d'instanciations et optimisent certaines des opérations les plus récurrentes et les plus coûteuses à la longue et qui reviennent aussi dans d'autres heuristiques, telle que celle qui vérifie si un chemin contient le nœud suivant ou pas, nous exploitons pour cette tâche les propriétés des nombres entiers.

Notre programme est formé de trois classes, la première est une petite classe appelée *city* qui permet d'instancier et de représenter chacune des villes. En version de base, elle contient un entier identificateur, un entier pour stocker la meilleur distance qui mène à elle, un vecteur d'entiers pour stocker les identificateurs des villes qui forment ce meilleur chemin là, et enfin un mathode *compareTo* utilisée pour comparer les plus courtes distances dans la partie *File de Priorité* de la variante de l'algorithme de Dijkstra.

La seconde classe est la principale, elle se charge d'abord de calculer les distances entre chaque paire de villes à partir des coordonnées de chacune d'elles. Elle peut

aussi prendre en charge une matrice des distances pour pouvoir résoudre les instances asymétrique ou celles ne respectant pas l'inégalité triangulaire.

Optionnellement, elle peut également calculer une liste de k- voisins utilisés pour réduire la complexité du parcours.

Ensuite, nous bouclons pour réaliser un nombre d de départs différents selon le budget en temps. Lors de ces départs, nous parcourons le graphe selon la méthode expliquée dans la partie description.

Selon le choix et selon le budget temps, la procédure d'affinement du tour ou des chemins est exécutée soit de manière imbriquée pour affiner les chemins au fur et à mesure, soit après avoir généré un tour. La version imbriquée donne évidement de bien meilleurs résultats. Un exemple d'exécution sur l'instance *berlin52* issue de TSPLIB est exposé dans la figure 3.4. Le tour optimal quand à lui est montré dans la figure 3.5 pour comparaison. Vous remarquerez que les deux tours sont très proches et quasi identiques.

Et enfin, la troisième classe, appelée *rendering*, se charge simplement d'afficher graphiquement les tours qui lui sont soumis.

3.2.3 Essais et comparaisons

Pour tester notre code, nous avons récupéré quelques instances de type Géo et de type Euclidiennes, issues de TSPLIB, que nous avons transformé en tableau de coordonnées. Pour pouvoir mieux comparer et prendre en compte les erreurs d'arrondissement inhérentes à chaque machine, nous avons sélectionné des instances dont le tour optimal a été publié, et ce afin d'en recalculer la longueur et pouvoir la comparer à celle du tour produit par notre heuristique sur la même machine.

Avec l'instance berlin52, nous vous présentons quelques résultats dans le tableau 3.1.

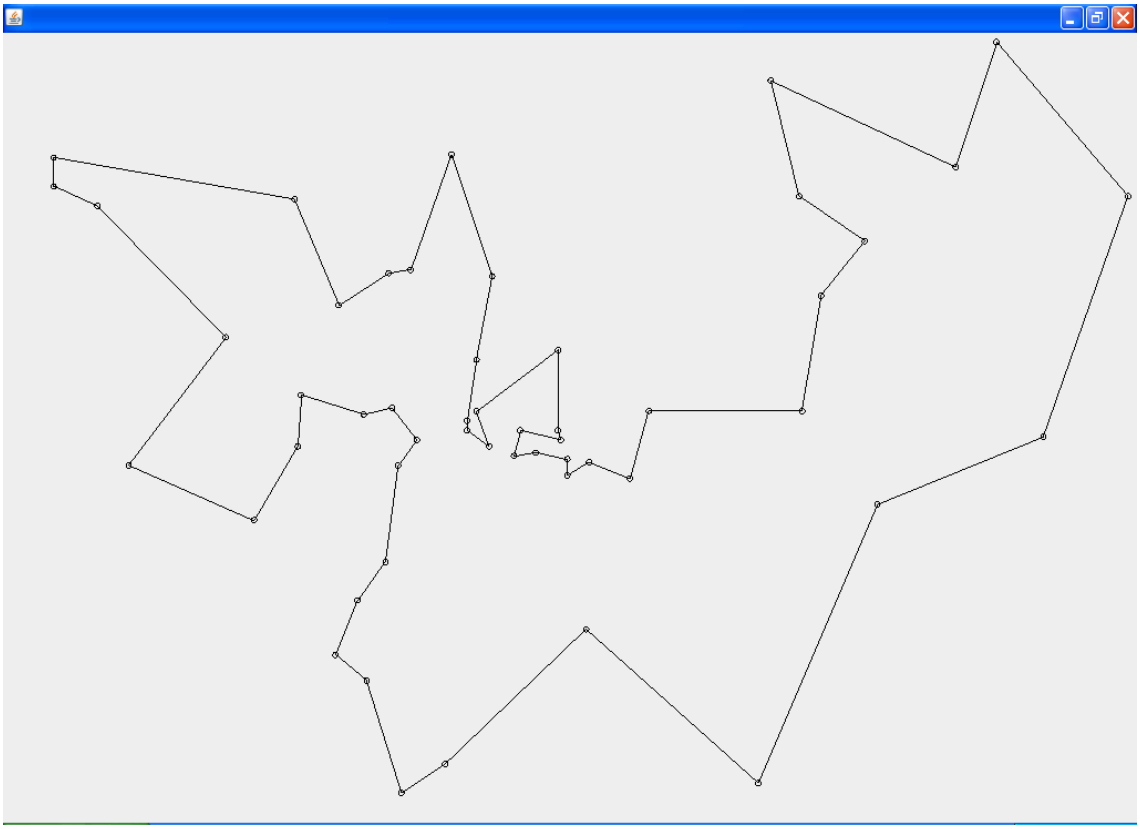

FIGURE 3.4 – Exécution de l'une des versions de notre heuristique sur l'instance berlin52 issue de TSPLIB

Heuristique	**Longueur de tour**	**Tour optimal**	**Ratio d'approximation**
Version de base 1-départ	9409	7526	25%
Version de base n-départs	7911	7526	5.1%
Version affinée	7698	7526	2.2%
H. du plus proche voisin	18296	7526	43.1%

TABLE 3.1 – Tableau comparant entre NNH et quelques résultats obtenus par notre heuristique avec différents paramètres

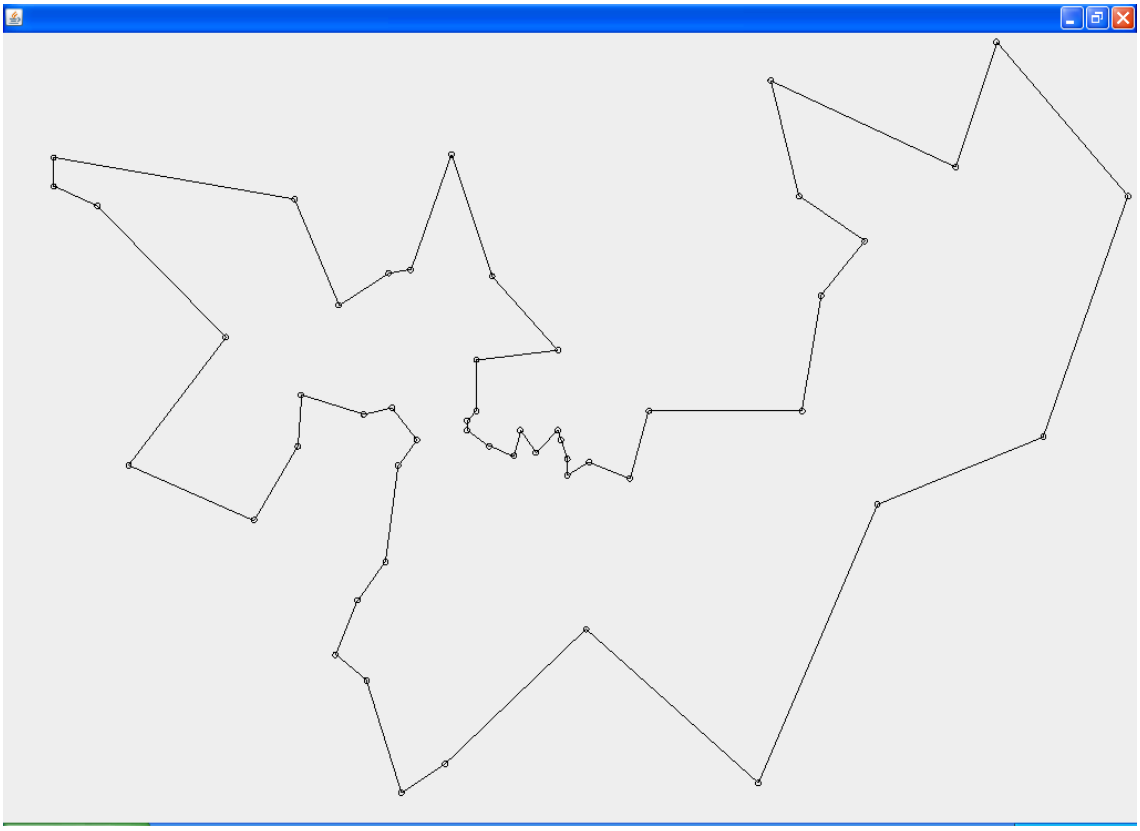

FIGURE 3.5 – Rendu du tour optimal sur l'instance berlin52 issue de TSPLIB

3.3 Conclusion

Bien que notre heuristique reste encore une simple *méthode approchée*, il faut prendre en compte que cette notion de graphe condensé n'en est qu'à ses débuts et qu'elle est loin de nous avoir livré tous ses secrets. Nous mêmes, avons encore du travail devant nous afin d'approfondir notre analyse et notre exploration de cette nouvelle structure. L'amélioration de notre compréhension de celle-ci nous mènera à itérer avec des versions de plus en plus proches de l'optimalité. Le problème $P{=}NP$ reste encore ouvert, et nous restons optimistes à son propos.

Conclusion et perspectives

L'heuristique présentée dans ce Mémoire est encore toute jeune et n'en est qu'à ses débuts. Les essais que nous avons entrepris ont démontré son efficacité et les optimisations sur lesquelles nous travaillons nous donnent déjà beaucoup d'espoir sur les possibles améliorations que nous pouvons y apporter.

De plus, certaines des propriétés de notre heuristique, comme le nombre, la variété et la qualité des tours qu'elle peut produire en une seule exécution donnent des signes prometteurs sur la possibilité de l'utiliser comme routine d'initialisation dans certaines métaheuristques comme les algorithmes génétiques ou l'optimisation par colonie de fourmis.

Pour finir, nous espérons que les contributions qui sont et seront développées à l'avenir dans le cadre de ce travail, puissent trouver utilité dans d'autres domaines également comme c'est le cas pour les autres grandes contributions développées autour du Traveling Salesman Problem.

Bibliographie

[1] Karen Aardal, George L Nemhauser, and Robert Weismantel. *Handbooks in Operations Research and Management Science : Discrete Optimization*, volume 12. Elsevier, 2005.

[2] David Applegate, Robert Bixby, Vasek Chvátal, and William Cook. Concorde tsp solver, 2006.

[3] David Applegate, Robert Bixby, William Cook, and Vasek Chvátal. *On the solution of travelling salesman problems*. Universität Bonn. Institut für Ökonometrie und Operations Research, 1998.

[4] David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem : a computational study*. Princeton university press, 2011.

[5] Applegate, David and Bixby, Robert and Chvátal, Vasek and Cook, William. World tsp. Site web : http ://www.math.uwaterloo.ca/tsp/world/, 2001. [Accédé le 27 Juin 2016].

[6] Applegate, David and Bixby, Robert and Chvátal, Vasek and Cook, William. Sweden tour of 24,978. Site web : http ://www.math.uwaterloo.ca/tsp/sweden/tours/tours.htm, 2003. [Accédé le 27 Juin 2016].

[7] LL Barachet. Letter to the editor-graphic solution of the traveling-salesman problem. *Operations Research*, 5(6) :841–845, 1957.

[8] Demis Basso, Martin Lotze, Lavinia Vitale, Florinda Ferreri, Patrizia Bisiacchi, Marta Olivetti Belardinelli, Paolo Maria Rossini, and Niels Birbaumer. The role of prefrontal cortex in visuo-spatial planning : a repetitive tms study. *Experimental brain research*, 171(3) :411–415, 2006.

[9] Jillian Beardwood, John H Halton, and John Michael Hammersley. The shortest path through many points. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 55, pages 299–327. Cambridge Univ Press, 1959.

[10] Richard Bellman and Marshall Hall Jr. *Combinatorial analysis*, volume 10. American mathematical society, 1960.

[11] James A Carlson, Arthur Jaffe, and Andrew Wiles. *The millennium prize problems*. American Mathematical Soc., 2006.

[12] Nicos Christofides. Technical note - bounds for the travelling-salesman problem. *Operations Research*, 20(5) :1044–1056, 1972.

[13] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976.

[14] Axel Conrad, Tanja Hindrichs, Hussein Morsy, and Ingo Wegener. Solution of the knight's hamiltonian path problem on chessboards. *Discrete Applied Mathematics*, 50(2) :125–134, 1994.

[15] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, et al. Dijkstra\'s algorithm. 2001.

[16] Georges A Croes. A method for solving traveling-salesman problems. *Operations research*, 6(6) :791–812, 1958.

[17] Harlan Crowder and Manfred W Padberg. Solving large-scale symmetric travelling salesman problems to optimality. *Management science*, 26(5) :495–509, 1980.

[18] George Dantzig, Ray Fulkerson, and Selmer Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4) :393–410, 1954.

[19] George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1) :80–91, 1959.

[20] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system : optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1) :29–41, 1996.

[21] Von einem alten Commis-Voyageur. Der handlungsreisende wie er sein soll und was er zu tun hat, um aufträge zu erhalten und eines glücklichen erfolgs in seinen geschäften gewiß zu sein. Manuel pour voyageurs de commerce, 1832. Le voyageur de commerce, comment il se doit d'être et ce qu'il devrait faire afin de gagner ses commssions et prospérer dans son affaire, par un vieux commis-voyageur.

[22] L Fejes. Über einen geometrischen satz. *Mathematische Zeitschrift*, 46(1) :83–85, 1940.

[23] Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. A randomized rounding approach to the traveling salesman problem. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 550–559. IEEE, 2011.

[24] HW Ghosh. Expected travel among random points. *Bull. Calcutta Statist. Assoc*, 2 :83–87, 1949.

[25] Michael Held and Richard M Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1) :196–210, 1962.

[26] Michael Held and Richard M Karp. The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18(6) :1138–1162, 1970.

[27] Michael Held and Richard M Karp. The traveling-salesman problem and minimum spanning trees : Part ii. *Mathematical programming*, 1(1) :6–25, 1971.

[28] K HELSGAUN. Year effective implementation of the flax-kernighan traveling salesman heuristic. *Department of Computer Science, University of Roskilde, Denmark*, 1998.

[29] Keld Helsgaun. An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1) :106–130, 2000.

[30] Keld Helsgaun. *An effective implementation of K-opt moves for the Lin-Kernighan TSP heuristic*. PhD thesis, Roskilde University. Department of Computer Science, 2006.

[31] Keld Helsgaun. General k-opt submoves for the lin–kernighan tsp heuristic. *Mathematical Programming Computation*, 1(2-3) :119–163, 2009.

[32] AJ Hoffman, J Wolfe, RS Garfinkel, DS Johnson, CH Papadimitriou, PC Gilmore, EL Lawler, DB Shmoys, RM Karp, JM Steele, et al. *The traveling salesman problem : a guided tour of combinatorial optimization*. J. Wiley & Sons, 1986.

[33] David S Johnson, L McGeogh, Fred Glover, and César Rego. 8th dimacs implementation challenge : the traveling salesman problem, 2000.

[34] Roy Jonker and Ton Volgenant. Transforming asymmetric into symmetric traveling salesman problems : erratum. *Operations research letters*, 5(4) :215–216, 1986.

[35] Michael Jünger, Gerhard Reinelt, and Giovanni Rinaldi. The traveling salesman problem. *Handbooks in operations research and management science*, 7 :225–330, 1995.

[36] Robert L Karg and Gerald L Thompson. A heuristic approach to solving travelling salesman problems. *Management science*, 10(2) :225–248, 1964.

[37] Donald Ervin Knuth. Postscript about np-hard problems. *ACM SIGACT News*, 6(2) :15–16, 1974.

[38] Tjalling C. Koopmans. Optimum utilization of the transportation system. *Econometrica*, 17 :136–146, 1949.

[39] Ratnesh Kumar and Haomin Li. On asymmetric tsp : Transformation to symmetric tsp and performance bound. *Journal of Operations Research*, 1996.

[40] Eugene L Lawler and David E Wood. Branch-and-bound methods : A survey. *Operations research*, 14(4) :699–719, 1966.

[41] Leonhard Euler. Icosian game. Site web : http ://www.daviddarling.info/encyclopedia/I/Icosian_Game.html, 1856. [Accédé le 26 Juin 2016].

[42] Jing-Quan Li, Pitu B Mirchandani, and Denis Borenstein. The vehicle rescheduling problem : Model and algorithms. *Networks*, 50(3) :211–229, 2007.

[43] Shen Lin. Computer solutions of the traveling salesman problem. *The Bell System Technical Journal*, 44(10) :2245–2269, 1965.

[44] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2) :498–516, 1973.

[45] John DC Little, Katta G Murty, Dura W Sweeney, and Caroline Karel. An algorithm for the traveling salesman problem. *Operations research*, 11(6) :972–989, 1963.

[46] James N MacGregor and Tom Ormerod. Human performance on the traveling salesman problem. *Perception & Psychophysics*, 58(4) :527–539, 1996.

[47] Prasanta Chandra Mahalanobis. A sample survey of the acreage under jute in bengal. *Sankhyā : The Indian Journal of Statistics*, pages 511–530, 1940.

[48] Eli S Marks. A lower bound for the expected travel among m random points. *The Annals of Mathematical Statistics*, 19(3) :419–422, 1948.

[49] Tobias Mömke and Ola Svensson. Approximating graphic tsp by matchings. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 560–569. IEEE, 2011.

[50] G Morton and AH Land. A contribution to the "travelling-saleman" problem. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 185–203, 1955.

[51] Manfred Padberg and Giovanni Rinaldi. Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters*, 6(1) :1–7, 1987.

[52] Manfred Padberg and Giovanni Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, 33(1) :60–100, 1991.

[53] Manfred W Padberg and Saman Hong. On the symmetric travelling salesman problem : a computational study. In *Combinatorial Optimization*, pages 78–107. Springer, 1980.

[54] Christos H Papadimitriou. The euclidean travelling salesman problem is np-complete. *Theoretical Computer Science*, 4(3) :237–244, 1977.

[55] Ira Pohl. A method for finding hamilton paths and knight's tours. *Communications of the ACM*, 10(7) :446–449, 1967.

[56] Procter & Gamble. Help "car 54" and win cash. Site web : http ://www.math.uwaterloo.ca/tsp/history/pictorial/car54.html, 1962. [Accédé le 27 Juin 2016].

[57] Gerhard Reinelt. Tsplib-a traveling salesman problem library. *ORSA journal on computing*, 3(4) :376–384, 1991.

[58] Stanley Reiter and Gordon Sherman. Discrete optimizing. *Journal of the Society for Industrial and Applied Mathematics*, 13(3) :864–889, 1965.

[59] Julia Robinson. On the hamiltonian game (a traveling salesman problem). Technical report, DTIC Document, 1949.

[60] András Sebo and Jens Vygen. Shorter tours by nicer ears. *arXiv preprint arXiv :1201.1870*, 2012.

[61] Friedrich Stadler. Karl menger, karl s wiener kreis : Das mathematische kolloquium 1928–1936. In *Der Wiener Kreis*, pages 201–223. Springer, 2015.

[62] C Tompkins. Machine attacks on problems whose variables are permutations. In *Proceedings of Symposia in Applied Mathematics*, volume 6, pages 195–211, 1956.

[63] Jan Van Leeuwen. *Handbook of theoretical computer science (vol. A) : algorithms and complexity*. Mit Press, 1991.

[64] S Verblunsky. On the shortest path through a number of points. *Proceedings of the American Mathematical Society*, 2(6) :904–913, 1951.

[65] Douglas Vickers, Marcus Butavicius, Michael Lee, and Andrei Medvedev. Human performance on visually presented traveling salesman problems. *Psychological Research*, 65(1) :34–45, 2001.

[66] Wikipedia, l'encyclopedie libre. Exemple de permutation 2-opt. Site web : https ://fr.wikipedia.org/wiki/2-opt. [Accédé le 27 Juin 2016].

RÉSUMÉ

Dans ce mémoire, nous jetons les bases et ouvrons la voie à une nouvelle heuristique de complexité polynomiale pour la résolution du célèbre problème du voyageur de commerce en introduisant une nouvelle représentation graphique de l'espace total de solutions, sous forme d'un graphe condensé, pondéré, orienté et acyclique, ainsi qu'une méthode de parcours de ce nouveau graphe inspirée de l'algorithme de recherche de chemin point-à-point de Dijkstra.

Mots clés : Problème du voyageur de commerce, circuit Hamiltonien, heuristique, graphe condensé, pondéré, orienté, acyclique.

ABSTRACT

In this document, we lay the foundations and pave the way for a new polynomial heuristic for the solution of the famous Traveling Salesman Problem by introducing a new graph structure representing the total solution space in the form of a compact, weighted, directed and acyclic graph, along with an exploration method inspired by the point-to-point path search algorithm of Dijkstra.

Key words : Traveling salesman problem, Hamiltonian circuit, heuristic, compact graph, weighted, directed, acyclic.