

2016

UFR Sciences Angers
Master 1 Informatique



Placement de routes

Urbanisme

David Florian

Sous la direction de M. Da Mota Benoit et M.
Goëffon Adrien

Membres du jury

Soutenu publiquement le :
23 juin 2016



L'auteur du présent document vous autorise à le partager, reproduire, distribuer et communiquer selon les conditions suivantes :



- Vous devez le citer en l'attribuant de la manière indiquée par l'auteur (mais pas d'une manière qui suggérerait qu'il approuve votre utilisation de l'œuvre).
- Vous n'avez pas le droit d'utiliser ce document à des fins commerciales.
- Vous n'avez pas le droit de le modifier, de le transformer ou de l'adapter.

Consulter la licence creative commons complète en français :
<http://creativecommons.org/licences/by-nc-nd/2.0/fr/>

+

Ces conditions d'utilisation (attribution, pas d'utilisation commerciale, pas de modification) sont symbolisées par les icônes positionnées en pied de page.



REMERCIEMENTS

Je tiens à remercier mes enseignants référents, Adrien Goëffon et Benoit Da Mota, pour leurs conseils et leur accompagnement.

Il me faut également remercier mes camarades de promotion, plus particulièrement ceux présents à l'Université au cours de ces deux mois et demi de travail. Parmi eux, les présences fréquentes de Jérôme Fourmond, Morgane Troysi et Ugo Rayet ont été particulièrement appréciables et appréciées.

Table des matières

INTRODUCTION.....	4
1 Présentation du contexte.....	4
2 Présentation du projet.....	4
3 Introduction au sujet du projet.....	4
4 Contextualisation.....	4
5 Etudes préliminaires.....	5
6 Choix des outils.....	5
DÉMARCHE.....	6
1 Organisation.....	6
2 Représentation des données du problème.....	7
3 Evaluation des solutions.....	8
3.1. Nombre de parcelles exploitables.....	8
3.2. Calcul du ratio entre distance directe et distance par les routes.....	8
4 Recherche de solutions améliorant un objectif.....	8
5 Persistance des solutions non dominées.....	9
6 Affichage et Interactions.....	9
6.1. Affichage.....	9
6.2. Interactions.....	9
7 Création du front Pareto.....	10
CONCLUSION.....	11
SITOGGRAPHIE.....	12
TABLE DES ILLUSTRATIONS.....	14

Introduction

1 Présentation du contexte

Dans le cadre du Master Informatique, il nous est demandé de réaliser un projet de recherche sur 10 semaines, afin de valider nos acquis, d'approfondir certaines connaissances et compétence et d'avoir une première approche d'un travail sur un projet de recherche, réalisé en quasi autonomie, puisque encadré par nos enseignants-chercheurs.

2 Présentation du projet

Ce projet a été proposé par deux enseignants : Benoit Da Mota et Adrien Goëffon. Il utilise principalement des principes étudiés dans le cadres de l'Optimisation Combinatoire, et de l'option Résolution de Problèmes. De plus, la combinaison de ces deux domaines, et compte tenu de la complexité relative du sujet, un bon niveau en programmation performante. De plus, la visualisation d'une solution permet de mieux se rendre compte des résultat, la maîtrise de bibliothèques graphiques pouvait être un avantage.

3 Introduction au sujet du projet

Le but est de pouvoir fournir à un utilisateur des dispositions possibles de routes sur une surface. Elles doivent respecter plusieurs contraintes. La première stipule que toutes les routes doivent être reliées entre elles et aux entrées-sorties.

<http://www.info.univ-angers.fr/~goeffon/TER2016/F-routes.pdf>

4 Contextualisation

Le sujet peut être appliqué sur différents domaines. Une des applications est l'organisation et l'accès à des parcelles d'une exploitation agricole, selon le placement des routes et chemins desservant les parcelles. Des ajouts pour se rapprocher de ce modèle pourrait être l'ajout de la gestion des entrées et sorties entre les parcelles et les accès ou des tailles de parcelles différentes, au lieu d'une unité fixe dans le cas présent (une route a la même dimension qu'une parcelle).

Une autre application serait la planification de l'urbanisation d'une ville, afin d'optimiser les emplacements pour habitations et bâtiments, tout en conservant une bonne circulation générale sur l'ensemble du réseau routier. Pour améliorer ce modèle, on pourrait ajouter des priorités sur certains bâtiments.

D'autres utilisations peuvent être imaginées, il suffit de modifier l'application afin d'ajouter les contraintes du problèmes que l'on veut étudier.

5 Etudes préliminaires

1. Réflexions sur les applications et enjeux du problème
2. Recherche de travaux similaires
3. Réflexion sur les langages, outils et approches du problème

6 Choix des outils

Développement orienté multiplate-formes, j'ai donc choisi des outils fortement universels :

Langage C++, avec Kdevelop, Cmake et Make, puis Qt pour l'interface graphique, avec Qt Creator, davantage adapté à ce framework. Des tests unitaires ont été réalisés avec la bibliothèque CppUnit.

Les outils de débogage utilisés sont Valgrind et ceux intégrés à Kdevelop et QtCreator. Valgrind a également été utilisé afin de supprimer les fuites mémoires. De plus, couplé à Callgrind, il permet de générer des fichiers lisibles par Kcachegrind, pour l'analyse des performances de l'application. J'ai également utilisé l'outil CppCheck afin d'effectuer certaines micro-optimisations.

Le versionnage est géré à l'aide d'un dépôt Git, hébergé sur github.com. La documentation est créée grâce à Doxygen.

Mes plateformes de développement et de test ont été Linux Mint et Windows 10.

Démarche

1 Organisation

J'ai décidé de séparer le programme et ses fichiers sources en plusieurs parties. Ceci a pour but de pouvoir exécuter le programme sans utiliser obligatoirement une interface graphique, d'améliorer sa portabilité avec d'autres outils ou plateforme de destination. Cela améliorant également la maintenabilité et la poursuite éventuelle du programme par d'autres personnes.

Ainsi, le code du programme est divisé en trois parties, chacune représentant un niveau de fonctionnalités.

- Le cœur de l'application est appelé le moteur (« Engine »), il est chargé de représenter et de manipuler les données représentant les solutions ainsi que les paramètres d'une instance du problème. De plus, une classe « Coordinates » permet de pouvoir stocker et comparer des coordonnées lors des manipulations d'une surface.
- La partie algorithmique du projet est placée à la racine de celui-ci, car c'est elle qui est chargée de résoudre les instances du problèmes, de chercher des solutions, de les stocker et de évaluer pour les comparer. Nous avons donc une classe représentant une surface et son évaluation, « FieldEvaluation ». L'évaluation de la surface doit être à tout moment à jour, à partir du moment où la solution est réalisable. Il y a ensuite une classe permettant de trouver des voisins d'une solutions, « LocalSearch » ; Il s'agit donc d'une recherche locale. Il serait envisageable de créer et d'utiliser d'autres méthodes de recherche de solutions telles que des algorithmes génétiques ou de la programmation par contrainte. Enfin, puisque le problème est multi-objectifs, il nous faut stocker et mettre à jour les solutions ayant une évaluation non dominée. Ceci est pris en charge par la classe « Resolution » qui est chargée de gérer quelle est la solutions sur laquelle on travaille et de gérer les solutions trouvées précédemment, ainsi que d'exporter les résultats sous la forme d'un front Pareto.
- La dernière partie est la partie interface graphique. Elle comprend l'affichage et les interactions pouvant être effectuées par l'utilisateur de l'application.

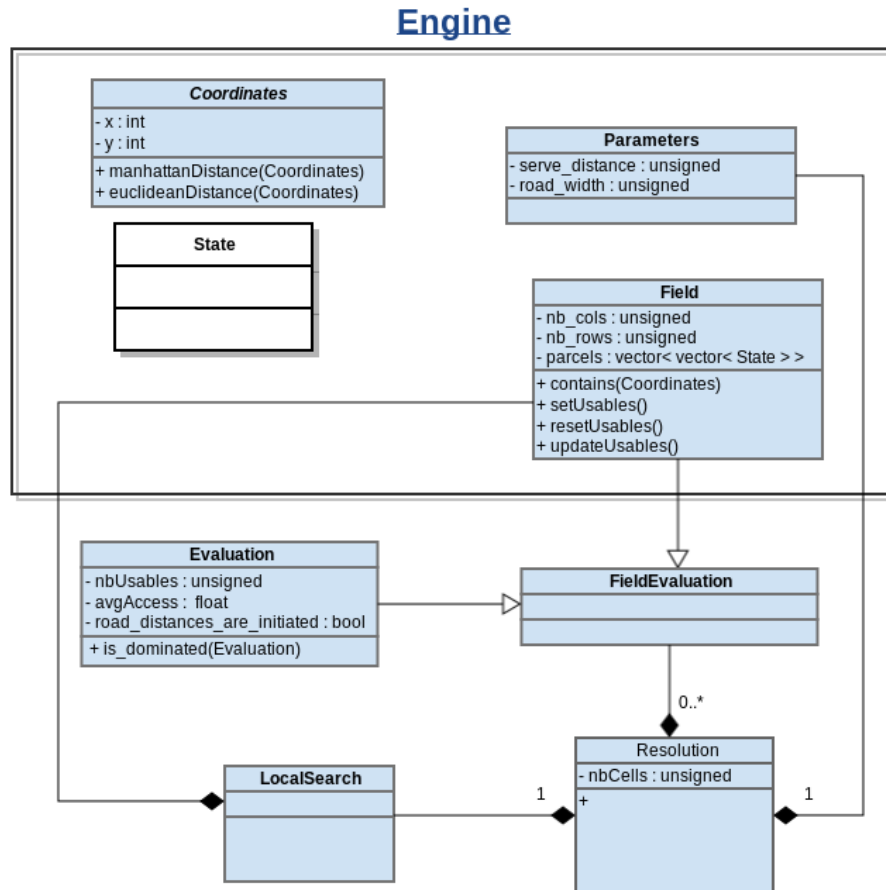


Illustration 1: Diagramme de classes du projet

2 Représentation des données du problème

La surface du problème étant un rectangle de cases rectangulaires, la structure de donnée la plus évidente est de la stocker sous forme de matrice à deux dimensions. C'est le choix qui a été fait, afin de faciliter la représentation du problème, et donc sa compréhension et la réflexion sur ses spécificités et enjeux. Chaque cellule est donc représentée par un état dans la matrice.

Cependant, s'est posée la question de l'intérêt de l'utilisation d'une représentation à base de graphes, plutôt que matricielle. Cette solution paraît davantage indiquée dans le cas de l'utilisation d'un algorithme génétique, du fait du croisement possible de « branches » du graphe. De plus, cela aurait permis de plus facilement changer la forme des cellules, avec une forme hexagonale par exemple. Malgré cela, cette représentation pouvait apporter une certaine confusion dans ce problème déjà complexe et vaste ; c'est pourquoi cette possibilité n'a pas été retenue.

3 Evaluation des solutions

3.1. Nombre de parcelles exploitables

Il s'agit de vérifier qu'elles sont les parcelles accessibles, et donc exploitable. On applique cette règle : *Si une cellule, qui n'est pas une route, a une route à une distance inférieure ou égale à D cellules (D le paramètre de desserte, en distance Manhattan), elle est alors exploitable.* On compte ensuite le nombre de parcelles exploitables total.

3.2. Calcul du ratio entre distance directe et distance par les routes

Le ratio moyen permettant d'évaluer l'accessibilité est plus difficile à calculer. En effet, il s'agit de trouver, pour chaque parcelle exploitable, on cherche la distance par les routes pour aller à chaque autre parcelle exploitable, pour pouvoir diviser leur distance directe – Manhattan ou euclidienne – le ratio entre ces deux parcelles. Il faut ensuite faire la moyenne des tous les ratios obtenus ainsi. Il y a donc $N \times (N-1)$ ratios à calculer, N étant le nombre de parcelles exploitables, pour obtenir le ratio total.

Or, pour calculer ce ratio, il nous faut, pour chaque parcelle, la distance vers chaque autre parcelle. Pour les obtenir, il nous faut donc trouver les chemins les plus courts pour aller d'une parcelle aux autres. Il s'agit donc d'un « pathfinding », opération coûteuse, surtout lorsque l'on travaille avec les valeurs exactes et minimales, comme c'est le cas ici.

De plus, on était amené à évaluer très fréquemment de nouvelles solutions, avec de légers changements. Une évaluation aussi performante aurait donc compromis la bonne progression du développement. La solution a été d'utiliser le concept de programmation dynamique, vu cette année. En effet, lors du calcul des plus courts chemin, il arrive fréquemment que l'on calcul plusieurs fois le chemin minimal entre deux mêmes routes. Cela est dû fait que si deux routes sont coller, il y a de forte chances que le chemin le plus court vers une troisième route soit le même, à une route près.

Cette amélioration a permis de grandement améliorer les performances de l'application. L'évaluation reste malgré tout une part importante des ressources utilisées par l'application. La mise en place d'une potentielle évaluation incrémentale, pour calculer l'accessibilité d'une solution, dérivée voisine d'une autre dont on connaît l'accessibilité, s'est avéré bien plus restreinte et difficile à mettre en place qu'escompté. Les tentatives sur ce sujet n'ont pas abouti. Cela vient du fait que l'ajout d'un chemin entre deux routes peut modifier la distance entre deux autres routes, qu'elles soient proches ou non du nouveau chemin.

4 Recherche de solutions améliorant un objectif

Le choix du type d'algorithme lors de la recherche de solutions améliorantes s'est porté sur la recherche locale. Ce choix est justifié par le fait que la structure d'une surface simplifiait ce procédé. De plus, il ne restreindrait pas l'application future des évolutions évoquées.

La première étape consiste à maximiser le nombre de parcelles exploitables. Pour ceci, un algorithme glouton a été utilisé. Pour chaque route ajoutée, on cherche celle qui augmentera le plus le nombre de routes exploitables en la plaçant. En revanche, les routes qui auraient plusieurs autres routes voisines seront défavorisées. Il n'y a donc pas de stratégie de placement sur plusieurs routes à placer. Ainsi, l'ajout d'une route peut diminuer l'intérêt d'une route précédemment placée. Ce placement n'est donc pas optimal, mais puisqu'on est sur un problème multi-objectifs, il ne serait pas judicieux de se concentrer sur un seul des ces objectifs.

La seconde étape va donc chercher à maximiser le second objectif : l'accessibilité. On va chercher à ajouter des « chemins » de routes entre deux routes existantes, afin de créer des passages améliorant globalement la circulation, à l'aide d'un changement local.

...

On peut alterner les étapes en fournissant un nombre de routes à ajouter pour le nombre d'exploitables – sachant que la valeur 0 permet d'ajouter des routes tant qu'on améliore l'objectif – et le nombre de chemins pour l'accessibilité.

5 Persistance des solutions non dominées

Lors de l'évaluation d'importantes quantités de données sont générées lors de la recherche de chemin. Les valeurs de distance entre toutes les routes et toutes les autres routes sont stockées dans une matrice à quatre dimensions (2 pour les coordonnées de départ et deux pour les coordonnées d'arrivée).

6 Affichage et Interactions

6.1. Affichage

L'affichage est donc géré avec le framework Qt. Une surface est affichable à l'aide d'un widget, représenté par une classe. Cette classe nécessite la surface et son évaluation, afin de pouvoir dessiner une « Hotmap ». Les E/S sont rouges, les routes grises, les parcelles non exploitables bleu clair et les exploitables blanches. Cet affichage doit être mis à jour lors d'une interaction par l'utilisateur et à la suite de l'exécution d'un algorithme de résolution. La solution affichée par l'application est la dernière solution trouvée.

Un affichage alternatif est possible. En effet, une fois qu'une solution est réalisable, l'utilisateur peut avoir la possibilité d'afficher une « Hotmap », afin de le guider dans ses prises de décisions. Ce terme désigne l'affichage de couleurs, allant de vers à rouge, selon l'écart avec la moyenne de l'accessibilité de la parcelle.

6.2. Interactions

... l'application étant pensée comme une aide à la décision, elle gère différentes interactions avec l'utilisateur.

L'utilisateur peut ajouter ou supprimer des entrées-sorties (E/S) et des routes. Il peut choisir de créer une solutions réalisable manuellement, puis de l'évaluer. Il peut également choisir de placer uniquement des E/S et demander à l'application de les relier automatiquement. Il a ensuite la possibilité de lancer une résolution sur une solution initiale générée ou créée manuellement.

7 Création du front Pareto

La dernière étape de la part de l'utilisateur et de choisir une des solutions qui lui convient.

Conclusion

Le sujet demandait des compétences dans divers domaines. Ceux ci sont : l'optimisation combinatoire, la résolution de problèmes, la bonne utilisation – organisation et optimisation – du développement dans un langage adapté et performant, et la création d'une interface graphique complète.

Le développement de l'application a fait appel à plusieurs notions, types d'algorithmes et approches, étudiées durant l'année du Master 1 :

- ◆ La programmation dynamique
- ◆ La recherche de voisins grâce à une recherche locale
- ◆ Une utilisation importante de structures de données complexes : matrices à 4 dimensions, création et utilisation de graphes, listes d'éléments contenant d'importantes données calculées
- ◆ Gestion du partage des données, contenues dans les structures, entre les différents niveaux de l'application : affichage, interactions, résolution, export et sauvegarde ...

Il à été nécessaire de mettre en place et de respecter certaines méthodologies et de faire des choix : déterminer l'intérêt des algorithmes génétiques dans ce cas précis, la gestion du multi-objectifs et création d'un front Pareto à partir d'une évaluation des solutions, l'utilisation d'une ancienne solution, sans devoir recalculer ses données, gestion de la mémoire, mettre à jour les données lorsque c'est pertinent et seulement lorsque c'est pertinent pour éviter d'effectuer des longs calculs si ils ne sont pas utiles dans certains cas,

Il aurait été vain de tenter de traiter l'ensemble de l'étendue du problème. Il a donc fallu déterminer les éléments primordiaux à mettre en place et sélectionner ceux à traiter lors de la résolution. Il est donc possible de continuer le projet afin de comparer différents algorithmes et implémentations, ajouter des interactions et informations fournies à l'utilisateur.

La mise en place de tests et devoir maintenir un code clair et commenté était indispensable pour le bon déroulement du projet sur une période de temps plus étendue qu'à l'accoutumée.

Sitographie

Table des illustrations

Illustration 1: Diagramme de classes du projet.....7

Tableau 1: Inscrire une légende

mots-clés : Entrée-sortie ; Solution réalisable

ENGAGEMENT DE NON PLAGIAT

Je, soussigné(e)
déclare être pleinement conscient(e) que le plagiat de documents ou d'une
partie d'un document publiée sur toutes formes de support, y compris l'internet,
constitue une violation des droits d'auteur ainsi qu'une fraude caractérisée.
En conséquence, je m'engage à citer toutes les sources que j'ai utilisées
pour écrire ce rapport ou mémoire.

signé par l'étudiant(e) le

**Cet engagement de non plagiat doit être signé et joint
à tous les rapports, dossiers, mémoires.**

Présidence de l'université
40 rue de rennes – BP 73532
49035 Angers cedex
Tél. 02 41 96 23 23 | Fax 02 41 96 23 00

