Master 1 Informatique
Optimisation Combinatoire
2020-2021

Projet : algorithmes gloutons pour le placement de bâtiments.

Le projet pourra être réalisé en binôme. En plus de votre code, vous remettrez un rapport de plusieurs pages pour présenter le projet et les résultats obtenus de manière analytique. Des consignes plus précises sont données plus bas dans le projet. *Attention, le rapport ne devra contenir de code (pseudo-code néanmoins autorisé).* Vous pouvez choisir le langage parmi C, C++, Python et Java.

On dispose d'un ensemble de bâtiments rectangulaires, dont un hôtel de ville, ainsi que d'un terrain lui aussi rectangulaire. On veut placer une partie des bâtiments sur ce terrain de telle sorte que tous les bâtiments placés soient reliés à l'hôtel de ville par une route, en cherchant à maximiser l'aire du terrain occupée par les bâtiments. Le projet consiste à implémenter le problème et des variantes d'algorithmes glouton que vous comparerez pour la résolution du problème.

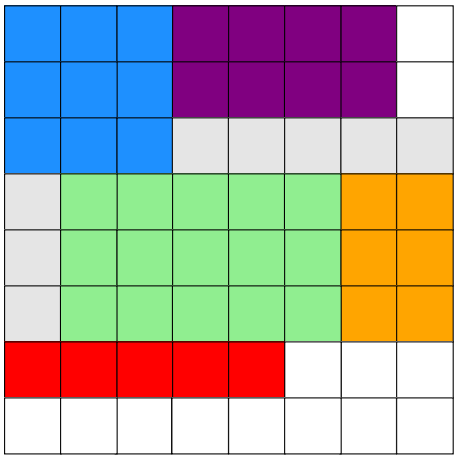

FIGURE 1 – Exemple de solution valide obtenue en plaçant les bâtiments de l'instance exemple ci-après dans leur ordre d'apparition dans les fichiers. L'hôtel de ville est en bleu, les routes reliant les bâtiments placés à l'hôtel de ville sont en gris.

Une instance de ce problème contient, dans l'ordre, la taille du terrain (longueur, largeur), le nombre de bâtiments, puis les dimensions des différents bâtiments. Le premier bâtiment dont on donne les coordonnées correspond à l'hôtel de ville. L'instance de l'exemple ci-dessus sera un fichier texte de la forme suivante :

```
8 8 % taille du terrain
8 % nombre de batiments
3 3 % taille du batiment 1 (= hotel de ville)
2 4 % taille du batiment 2
3 5 % ...
3 2 % ...
1 5 % ...
4 4 % ...
4 3 % ...
2 2 % taille du batiment 8
```

Une solution du problème consiste en une affectation de coordonnées à chaque bâtiment. En plus de sa représentation matricielle, la solution représentée sur l'exemple pourrait correspondre à l'affectation des coordonnées suivantes aux bâtiments de l'instance : (1,1), (1,4), (4,2), (4,7), (7,1). Chaque coordonnée correspond à l'emplacement du coin supérieur gauche du bâtiment correspondant (par exemple, le premier bâtiment 3*3 de l'instance commence aux coordonnées (1,1)). Vous utiliserez les deux formes de représentation afin de gérer efficacement le placement des bâtiments.

Les algorithmes gloutons implémentés placeront itérativement des bâtiments de manière valide jusqu'à ce qu'il ne soit plus possible de placer aucun bâtiment.

Remarque : vous décomposerez le problème en plusieurs fonctionnalités : lecture de l'instance, test de superposition de deux bâtiments, test de l'existence d'un chemin entre un bâtiment et l'hôtel de ville, test de validité de placement... Des indications, ainsi que les variantes d'algorithmes gloutons, sont données dans les questions ci-dessous.

Question 1

Implémenter les structures de données nécessaires pour la suite du projet. Vous devrez notamment stocker la taille du terrain, les bâtiments et leur taille, indiquer quels bâtiments sont placés au non (avant d'appeler un quelconque algorithme glouton, aucun bâtiment ne sera placé), pouvoir représenter une solution au moyen de coordonnées et pouvoir représenter la solution dans la matrice correspondant au terrain (les cases occupées par le bâtiment numéro x contiendront la valeur x et les cases vides la valeur 0).

Question 2

Pour tester les différentes parties du projet, il est crucial de pouvoir afficher les solutions produites. Faites en sorte d'afficher le terrain (dans le terminal). Lors de l'affichage, le terrain sera une matrice, les cases occupées par un bâtiment x (x étant le numéro du bâtiment) contiendront le nombre x et les routes (pour simplifier, les routes seront toutes les cases inoccupées) un caractère spécial de votre choix.

Question 3

Les algorithmes gloutons construisent une solution incrémentalement. Ainsi, dans la suite, un bâtiment sera ajouté sur le terrain à chaque itération. Cependant, certains placements de bâtiments ne sont pas valides : pour placer un nouveau bâtiment, il faut que ce dernier soit dans les limites du terrain, mais aussi qu'il ne se superpose pas avec d'autres bâtiments déjà placés (les routes seront gérées dans la question suivante).

Plus précisément, pour qu'un bâtiment ne soit pas en conflit avec un autre, il suffit que son point de départ soit situé après l'autre bâtiment (ligne, colonne, ou les deux) ou que son extrémité inférieure droite soit située avant l'autre bâtiment (ligne, colonne, ou les deux). Les deux cas sont illustrés dans la figure 2.

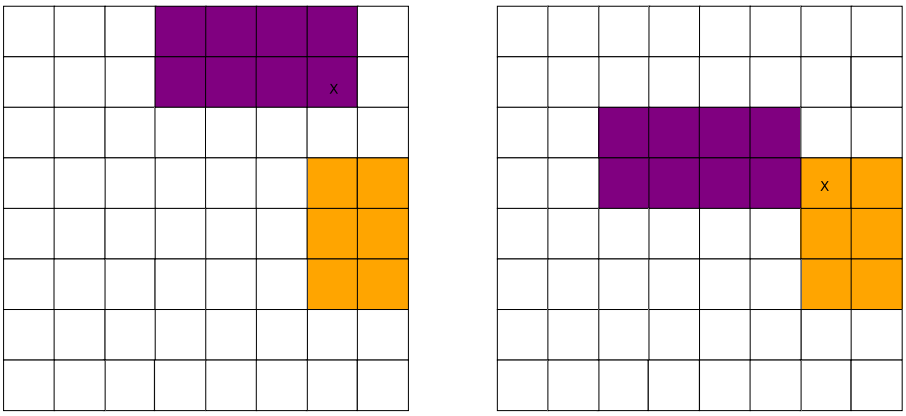

FIGURE 2 – Illustration de la vérification de la superposition entre deux bâtiments.

Question 4

Lors de la construction d'une solution par l'algorithme glouton, si le nouveau bâtiment placé n'est en conflit avec aucun autre et n'est pas hors limites du terrain, il faut vérifier qu'il est bien relié à l'hôtel de ville mais aussi que son placement ne coupe pas les chemins entre l'hôtel de ville et les autres bâtiments placés.

Vous pouvez effectuer cette vérification au moyen d'un système de pile, cela nécessitera de changer temporairement la valeur des cases vides (à -1 par exemple) pour que la fonction puisse s'arrêter. Dans un premier temps, toutes les cases vides adjacentes à l'hôtel de ville seront empilées (et marquées comme parcourues). Ensuite, tant que la pile n'est pas vide, l'algorithme fera les étapes suivantes :

- dépiler un élément, et regarder les cases adjacentes dans les 4 directions,
- pour les cases adjacentes, celles qui sont vides (et non parcourues) seront empilées (et marquées), celles correspondant à des bâtiments permettront de marquer ce bâtiment comme étant relié à l'hôtel de ville.

Il faut que tous les bâtiments du terrain soient marqués comme reliés à l'hôtel de ville une fois que la pile est vide.

Question 5

Ce problème étant un problème d'optimisation, il faut pouvoir associer un score à chaque solution. Ici, la fonction d'évaluation à maximiser correspond à l'aire (nombre de cases) du terrain occupée par des bâtiments (on ne compte pas les routes). Mettez en place une fonction qui calcule ce score pour une solution.

Question 6

L'algorithme glouton va ajouter itérativement un bâtiment sur le terrain à chaque étape jusqu'à ce que cela ne soit plus possible. Attention, afin de pouvoir gérer les routes dès le début, il sera primordial de placer l'hôtel de ville en premier sur le terrain.

Voici l'algorithme glouton (basique) proposé :

- On place d'abord l'hotel de ville aléatoirement sur le terrain (en étant équitable pour chaque positionnement possible).
- On parcourt les cases une à une (en partant d'un coin). Si la case est libre, qu'un bâtiment peut y être placé à cette coordonnée, qu'il est reliable à l'hotel de ville, et qu'il n'empêche

pas les autres bâtiments placés d'être reliés à l'hotel de ville, alors on place ce bâtiment.
— On passe à la case suivante, que l'on ait placé un bâtiment ou non. Une fois toutes les cases parcourues, l'algorithme s'arrête.

Question 7

Proposer plusieurs variantes de l'algorithme glouton et les implémenter.

Le résultat de l'algorithme glouton dépend entièrement du placement de l'hotel de ville et de l'ordre dans lequel apparaissent les bâtiments au moment du placement.

Voici les variantes à implémenter pour les placements des bâtiments. Ils seront placés en fonction de leur ordre dans un tableau qui sera déterminé selon la variante mise en place. Les variantes consistent à :

— placer prioritairement les bâtiments d'aire maximale en premier,
— placer les bâtiments d'encombrement (longueur + largeur) maximal en premier,
— placer les bâtiments en fonction d'un ordre aléatoire.

Il est conseillé de trier le tableau contenant les bâtiments en fonction du critère en début d'algorithme glouton.

Voici une variante pour le placement de l'hôtel de ville : on peut appliquer l'algorithme glouton pour chaque placement possible de l'hôtel de ville, et retourner au final la solution de qualité maximale.

Question 8

Vous pouvez proposer une ou plusieurs variantes gloutonnes supplémentaires. L'objectif est d'obtenir un algorithme plus efficace, cependant toute solution proposée est intéressante (même si elle est inéfficace, une variante proposée ne sera pas sanctionnée).

Vous pouvez proposer un algorithme exact de type branch & bound. Voici quelques indication pour vous aider à le concevoir :

— Une borne inférieure peut-être obtenue initialement en lançant un algorithme glouton (mais on peut aussi l'initialiser à 0).
— Une borne supérieure peut être calculée sur une solution partielle, en comptant le nombre de case déjà inoccupées par des bâtiments.
— Le problème étant doublement symétrique (en longueur et en largeur), les postions de l'hotel de ville à envisager peuvent se limiter au quart supérieur gauche du terrain.
— La construction de l'arbre de branche & bound suit sensiblement le même processus que les algorithmes gloutons (placmeent hôtel de ville, puis case par case), sauf que l'on considère tous les placements possibles.

La proposition d'un algorithme de branch & bound fonctionnel sera fortement apprécié !

Question 9

Vous fournirez un rapport décrivant le problème, les différents algorithmes implémentés et les résultats obtenus. Pour la partie résultat, vous reporterez les résultats de chaque variante pour chaque instance (disponibles sur discord) et vous discuterez des résultats obtenus. Attention, les variantes stochastiques devront être exécutées plusieurs fois, et vous indiquerez leur moyenne ainsi que l'écart type à cette moyenne. Une section dédiée à la répartition du travail (quand réalisé en binôme) est la bienvenue dans le rapport.