INF8215 – Intelligence artif.: méthodes et algorithmes

Automne 2020

TP No.2

Groupe 2

1849263– Emmanuelle Roberge

1907807 – Samuel Aguilar Lussier

1904141 – Thomas McAdams

Soumis à : Thiago Correia Pereira

Mardi 3 novembre 2020

Exercice 1

Solution initiales : Pour ce problème, il semblait assez clair que la solution naïve était la meilleure solution initiale, alors nous l'avons utilisée. Par contre, nous utilisons aussi des solutions initiales qui connectent de manière aléatoire des machines aux génératrices afin d'élargir l'ensemble de nos états explorés.

Voisinage : Notre voisinage est d'une taille de 25 mouvements. Ceci peut être expliqué par le fait qu'à chaque nouvelle itération, notre algorithme recherche une nouvelle solution en changeant l'état d'une seule génératrice. Ceci veut donc dire que si la génératrice sélectionnée est éteinte elle sera allumée et dans le cas contraire elle sera éteinte. Nous ne prenons pas en compte les machines lors du calcul de notre voisinage puisque dans tous les cas, la solution optimale sera d'attacher chaque machine à la génératrice allumée la plus proche, cela n'augmente donc pas la taille du voisinage.

Fonction d'évaluation : Tel que mentionné dans l'énoncé du TP2, notre fonction d'évaluation est le coût total. Ce coût peut être divisé en deux parties distinctes. La première partie s'agit de la somme des génératrices activées et la deuxième partie représente la somme de la distance euclidienne entre chaque machine et la génératrice à laquelle elle est attachée.

Fonction de validité : Dans tous les cas nous ne considérons que les solutions qui améliorent notre solution courante et donc qui diminuent le coût total calculé par notre fonction d'évaluation.

Fonction de sélection : Pour notre algorithme de recherche locale, nous avons utilisé deux fonctions de sélection différentes. La première garde en mémoire la modification du premier noeud trouvé diminuant le coût. Celui-ci est remplacé dès qu'on trouve un noeud diminuant davantage le coût. Ceci permet ainsi de trouver la génératrice à allumer ou à éteindre qui diminue le plus la fonction d'évaluation (hill climbing). La seconde fonction de sélection a une chance sur deux de remplacer la solution actuelle par une génératrice explorée, si cette dernière diminue le coût.

Critère d'arrêt : Puisque notre façon d'échapper aux minimums locaux sont les restarts, notre critère d'arrêt est lorsqu'aucune possibilité de génératrices à allumer ou éteindre nous permet d'améliorer la solution.

Restarts : Puisque nous avions une limite de temps, nous avons utilisé la librairie time pour maximiser le nombre de restarts que nous faisions lors de l'exécution de notre algorithme.

Heuristiques et algorithme complet : Lors de nos tests, nous n'avons jamais réussi à atteindre une meilleure solution qu'avec une solution initiale naïve et une fonction de sélection sans composante aléatoire. Cependant nous savons que cela pourrait nous laisser bloqués dans un minimum local. Nous avons donc essayé différentes méthodes pour explorer plus de solutions. Pour ce faire, on vérifie tout de même la première solution, mais on essaie alors de la battre avec une première boucle qui effectue des restarts à partir d'une solution initiale naive et des décisions aléatoires. La seconde boucle le fait avec une solution initiale aléatoire (connecter machines à des génératrices aléatoires) et des décisions aléatoires.