Projet de résolution de problèmes

Metaheuristiques pour la résolution du problème de l'arbre de Steiner de poids minimum

Alexandre Bontems, Gualtiero Mottola

Table des matières

1. Introduction

Ce projet aborde la résolution du problème de l'arbre de Steiner de poids minimum avec deux méthodes : un algorithme génétique et un algorithme de recherche locale. Le problème d'optimisation combinatoire consiste en la recherche d'un arbre de poids minimum couvrant les nœuds dits terminaux d'un graphe. La solution peut comprendre des nœuds facultatifs, dits de Steiner, pour être fortement connexe. La Figure 1 montre une solution optimale en bleu qui comprend les noeuds de Steiner u_1 et u_2.

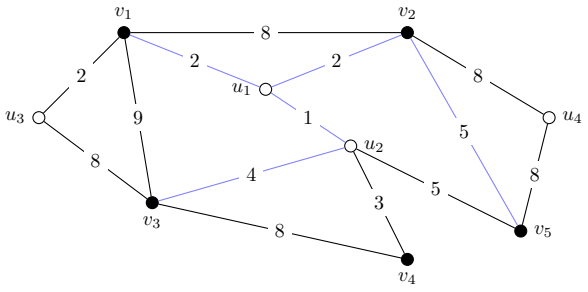

Figure 1 – Exemple de Steiner Tree Problem

L'algorithme génétique a été conçu de façon modulaire afin de pouvoir accueillir différentes fonctions d'initialisation, différents opérateurs de croisement et de mutation, etc. Il a ainsi été possible d'étudier les performances de chaque composante pour les différentes types d'instances considérés. L'algorithme de recherche locale réutilise les mêmes fonctions d'initialisation.

Dans ce rapport, tous les algorithmes sont testés sur les ensembles d'instances disponibles à l'adresse `http://steinlib.zib.de/testset.php`. Les instances sélectionnées parmi les ensembles B, C, D et E, sont répertoriées en Table 1.

Nom	Nb de sommets	Nb d'arêtes	Sommets terminaux	Opt
b02	50	63	13	83
b08	75	94	19	104
b10	75	150	13	86
b17	100	200	25	131
b18	100	200	50	218
c02	500	625	10	144
c05	500	625	250	1579
c12	500	2500	10	46
c18	500	12500	83	113
d02	1000	1250	10	220
d04	1000	1250	250	1935
d10	1000	2000	500	2110
e02	2500	3125	10	214

TABLE 1 – Instances de tests

2. ALGORITHME GÉNÉTIQUE

Un algorithme génétique consiste généralement en l'enchaînement des actions suivantes sur plusieurs générations : *Initialisation, Sélection, Crossover, Mutation*. La phase d'initialisation permet de générer une population d'individus qui seront ensuite sélectionnés, croisés, et mutés afin de construire la génération suivante.

2.1. POPULATION INITIALE

Les individus solutions sont représentés par un vecteur de variables binaires pour chaque sommet facultatif, prenant la valeur 1 si le sommet est présent dans la solution et 0 sinon. Par exemple, la solution de l'exemple en Figure 1 est représenté par le vecteur suivant.

u_1	u_2	u_3	u_4
1	1	0	0

INITIALISATION ALÉATOIRE Une première approche pour la phase d'initialisation a été de produire des individus en choisissant, de manière aléatoire, les sommets pris dans la solution. Les proportions de sommets pris dans une solution sont tirées aléatoirement entre 5 et 20% lors de la génération d'un individu. On essaye ainsi de se rapprocher le plus possible de l'optimum tout en s'assurant une certaine diversité dans la population de départ.

La Table 2 montre des coûts de départ pour quelques instances. On se rend vite compte qu'ils se trouvent souvent bien loin des solutions optimales et c'est pourquoi on utilisera par la suite des heuristiques de construction. Le temps de résolution étant borné à 5 minutes, on essaye ainsi d'améliorer les performances de nos algorithmes. Cette méthode de génération reste cependant intéressante pour introduire de la diversité dans nos population initiales.

HEURISTIQUE DU PLUS COURT CHEMIN L'heuristique de construction du plus court chemin reste la plus efficace en terme de coût initial. Elle consiste en la construction d'un arbre couvrant depuis le graphe des distances des sommets terminaux. Puisqu'elle est déterministe, on procède avant la génération à une randomisation du graphe problème pour obtenir davantage d'individus différents.

Génération	Instance	Proportion	Coût	Opt
1	b02	0.31	5075	83
2	b02	0.45	3602	83
3	b02	0.35	3102	83
1	b08	0.40	5112	104
2	b08	0.47	5180	104
3	b08	0.36	3174	104
1	c18	0.46	312	113
2	c18	0.44	302	113
3	c18	0.31	238	133

TABLE 2 – Exemple de population initiale aléatoire

La Table 3 présente les coups obtenus grâce à cette heuristique pour quelques instances. Le graphe d'origine est altéré de 5 à 20% pour chaque poids mais il est apparent que la perturbation de l'heuristique est faible : les solutions générées sont souvent identiques.

Génération	Instance	Coût	Opt
1	b02	92	83
2	b02	94	83
3	b02	92	83
1	b08	113	104
2	b08	113	104
3	b08	113	104
1	c18	137	113
2	c18	142	113
3	c18	139	133

TABLE 3 – Génération avec heuristique du plus court chemin

HEURISTIQUE DE L'ARBRE COUVRANT MINIMUM Cette heuristique nous permet d'améliorer la diversité de la population initiale. Quelques générations sont présentées en Table 4.

Génération	Instance	Coût	Opt
1	b02	93	83
2	b02	97	83
3	b02	96	83
1	b08	111	104
2	b08	107	104
3	b08	107	104
1	c18	224	133
2	c18	210	133
3	c18	209	133

TABLE 4 – Génération avec heuristique de l'arbre couvrant minimum

Notre algorithme génétique

2.2. Processus de sélection

Fitness proportionate selection

Stochastic Universal Sampling

Tournament Selection

2.3. Opérateurs de croisement

Croisement à point

Croisement uniforme

3. Recherche locale

4. Conclusion