Étude comparative sur les heuristiques de résolution du voyageur de commerce

SAE2.02

Evan Damas et Élia Gautier | SAÉ 2.01 | avril-mai 2022

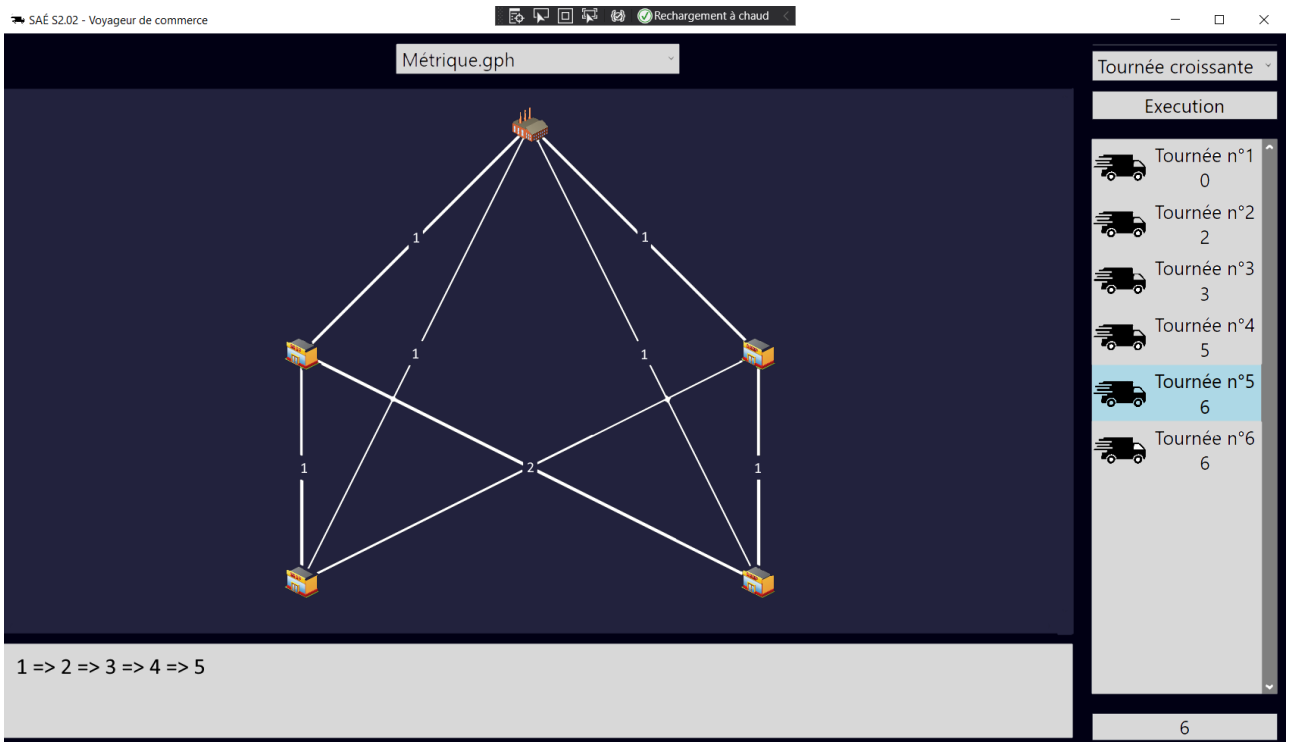

Table des matières

I. Heuristiques

HEURISTIQUES GLOUTONNES

Plus proche voisin

Cet algorithme a pour principe de toujours ajouter à la tournée le sommet non visité le plus proche du dernier ajouté à la tournée. Pour l'initialisation on choisit arbitrairement l'usine comme point de départ, mais la tournée étant cyclique on peut théoriquement démarrer à n'importe quel sommet.

Cet algorithme est très simple algorithmiquement ce qui le rend très rapide sur la plupart des graphes tout en maintenant des résultats tout à fait satisfaisants. Il dévoile tout son potentiel lorsqu'il s'agit de traiter des graphes bipartis, ce qui s'explique par leur nature qui fait que l'algorithme alterne entre les deux parties du graphe assez facilement.

Christofides

Nous appellerons le graphe de départ G.

Pour commencer nous avons besoin de faire un arbre couvrant de notre graphe de poids minimum, pour ce faire j'ai décidé d'utiliser l'algorithme de Kruskal (consiste à prendre les arêtes par ordre croissant selon leur poids tant qu'elles ne créent pas de cycle, et on s'arrête lorsqu'on a qu'une composante connexe). On appellera cet arbre couvrant de poids minimum T. Kruskal()

Ensuite on récupère les sommets de degrés impair de T dans un ensemble I. ListeLieuDegreImpair()

On fait un graphe induit de G par les sommets de I. Pour cela, on prend les sommets de I et les arêtes de G, et chaque arête à laquelle il manque un sommet de départ ou un sommet d'arrivée est supprimée. On appellera ce nouveau graphe induit V. SupprimeRouteEnTrop()

Par la suite, on fait un couplage minimal du graphe V. Couplage()

On fusionne l'arbre couvrant T avec le couplage obtenu Union(), et on fait un tour hamiltonien de ce nouveau graphe, et on a enfin de notre résultat (en supprimant les doublons). Final() insertion au plus proche

Le principe de cette heuristique est de partir des deux sommets les plus éloignés l'un de l'autre puis de toujours ajouter le sommet le plus proche de la tournée à la position qui rallongerait le moins la tournée, et ce jusqu'à avoir intégré tous les sommets désirés à la tournée.

Dans la plupart des cas cet algorithme est comparable en termes de temps d'exécution à celui du plus proche voisin, mais avec des résultats qui sont la plupart du temps en deçà de ceux de ce dernier. En outre sur certains graphes on parvient à un temps d'exécution ou un

résultat bien plus efficient, cependant nous ne sommes pas parvenus à en identifier les causes.

Cependant, l'algorithme réalisé n'est pas au point, en effet la fonction du couplage n'est pas parfaite. Une approche possible pour cette fonction est l'algorithme de Hongrois, cependant il ne fonctionne seulement sur les graphes bipartis, et le graphe auquel s'applique cette fonction ne l'est pas forcément. Donc il est resté préférable de faire une fonction qui donne des résultats que d'abandonner tout l'algorithme pour cette fonction.

Insertion au plus loin

Cette heuristique fonctionne selon le même principe que la précédente, à la différence près qu'au lieu d'ajouter systématiquement le sommet le plus proche de la tournée on ajoute celui le plus éloigné de cette dernière.

On s'attend naturellement à ce que dans la plupart des cas cet algorithme donne des résultats moins efficaces que l'insertion au plus proches, et les faits donnent raison à notre intuition première sur ce point, mais on constate que ces deux algorithmes obtiennent régulièrement des résultats similaires voire parfois en faveur de l'insertion au plus loin.

HEURISTIQUES LOCALES

Voisinage d'une tournée

Ici on part d'une tournée initiale et on cherche parmi ses voisins si l'un d'entre eux est plus optimisé qu'elle. Une fois qu'on en a trouvé un plus performant on remplace la tournée initiale et on réitère jusqu'à ne plus trouver d'améliorations.

On définit ici une voisine d'une tournée comme étant une tournée différente pour laquelle il suffirait d'inverser deux éléments pour obtenir la tournée originale.

HEURISTIQUES GENETIQUES

Les algorithmes génétiques se basent sur les principes biologiques de l'évolution, ils intègrent pour la plupart d'entre eux une part de hasard. Ici nous en avons créé deux similaires à la différence de la notion de « sélection » qui varie de l'un à l'autre.

On définit ici un individu comme étant une tournée de sommets et une population comme étant un groupe d'individus.

Dans un premier temps l'algorithme s'initialise en générant une population d'individus composés aléatoirement. A partir de ce moment il va faire évoluer sa population jusqu'à ce que le meilleur individu de celle-ci ne s'améliore pas lors de l'évolution de la population n vers la population n + 1 pendant un nombre de générations défini dans le constructeur.

L'évolution comporte trois étapes : la sélection, la reproduction et la mutation. Lorsqu'on effectue une sélection on cherche à sélectionner les individus les plus performants de la population précédente sans détruire la diversité de la population ce qui empêcherait son

évolution dans le futur. Une fois la sélection terminée on passe à la mutation en faisant se reproduire entre eux les couples d'individus sélectionnés.

La reproduction entre deux individus consiste au fait de « mélanger » leurs deux listes de lieux. Pour cela on sélectionne un indice aléatoire dans le premier parent et on prend tous ses sommets jusqu'à l'indice choisi. Une fois cela fait on retire du second parent les sommets déjà ajoutés à l'enfant et on insère ceux restants à la suite de ceux déjà présents chez l'enfant.

Cette reproduction faite on applique une mutation, définie ici comme l'inversion de deux sommets sélectionnés aléatoirement au sein d'un individu, sur certains individus choisis au hasard.

Sélection par roulette

La sélection par roulette consiste à donner à chaque individu une probabilité d'être sélectionné proportionnelle au fitness de l'individu. On appelle en biologie « fitness » le taux d'adaptation d'un individu.

Sélection par rang

La sélection par rang consiste à donner à chaque individu une probabilité de sélection proportionnelle à son rang dans le classement des individus composant la population.

II. Choix des graphes

BIPARTI

Simple

Le graphe biparti possède des propriétés intéressantes, c'est pourquoi nous avons testé nos algorithmes dessus afin de voir comment réagissais le voyageur d'avoir des magasins séparés en deux groupes distincts.

Complet

Le graphe biparti complet à tous ses magasins d'un groupes reliés à tous ceux de l'autre. Nous avons mis la valeur des routes à 1.

Complet à valeur aléatoire

Ce graphe est un biparti complet, cependant la valeur de ses routes est de valeurs aléatoires.

METRIQUE

La propriété de ce graphe est qu'il vérifie l'inégalité triangulaire, c'est-à-dire que pour chacun de ses magasins accède à l'un de ses voisins, c'est la route qui relit les deux le plus cours chemins, et non faire un détour. On a d'autres graphes qui vérifie cette inégalité triangulaire, mais si on a choisi lui c'est qu'il était simple et qu'il a surtout servit de test pour la création de l'algorithme de Christofides qui est capricieux.

SIMPLE 1

Ce graphe est l'un des graphes fournis dans le sujet, il a été très utile à comprendre le mécanisme du framework et à la création de premier algorithme.

SIMPLE 2

Ce graphe était également fourni mais il nous donnait une idée plus étendue de ce qu'on devait faire comme concevoir nos algorithmes en considérant qu'on les utilisera sur de grand graphe.

PEIGNE

Pourquoi faire compliqué quand on peut faire simple ? C'est un peu un échauffement pour nos algorithmes, voir s'ils arrivent à retourner la valeur la plus courte qui est 2 fois la taille du chemin reliant les deux extrémités.

TRIBU

Le graphe tribu est une sorte de symétrie centrale, il est intéressant de voir l'effet de nos heuristiques sur ce genre de graphes.

EDOUARD

Ce graphe est plus une blague qu'autre chose, il représente le logo de notre équipe, mais il est quand même intéressant puisqu'il possède un très grand cycle (oui on s'est amusé à le faire).

PETERSEN

Le graphe Petersen est le graphe qu'on a vu lors du TP sur la coloration, l'utilisation de nos heuristiques peut peut-être avoir un résultat surprenant dessus.

KITTELL

Graphe qu'on a jugé sympa vu sa forme.

III. Comparaisons

		Temps d'execution (en ms)										
Type		Biparti	Biparti C.	BC val Alea	Métrique	Simple1	Simple2	Peigne	Tribu	Edouard	Petersen	Kittel
Glouton	PlusProcheVoisin	6	92	105	0	0	2088	0	0	497	0	2
Glouton	TourneeCroissante	15	96	0	0	0	2231	0	0	483	0	2
Recherche Locale	VoisinageTournee	7	99	132	0	0	2145	0	0	486	0	6
Glouton	Christofides	x	114	x	0	0	x	0	0	x	x	x
Glouton	InsertionAuPlusLoin	6	99	107	0	0	2100	0	0	499	0	3
Glouton	InsertionAuPlusProche	6	100	106	0	2	2283	0	0	496	0	7
Génétique	Roulette	112854	291160	319954	107751	106613	temps trop long	101475	97565	temps trop long	94233	102501
Génétique	Par Rang	256762	461318	461544	247111	245658	temps trop long	251813	238000	temps trop long	288951	258804

		Taille du chemin trouvé										
Type		Biparti	Biparti C.	BC val Alea	Métrique	Simple1	Simple2	Peigne	Tribu	Edouard	Petersen	Kittel
Glouton	PlusProcheVoisin	274	80	112	6	20	252	20	12	187	30	165
Glouton	TourneeCroissante	687	158	232	6	20	448	20	12	203	44	303
Recherche Locale	VoisinageTournee	582	154	206	6	20	448	20	12	201	38	251
Glouton	Christofides	x	118	x	6	20	x	20	12	x	x	x
Glouton	InsertionAuPlusLoin	605	158	211	7	30	448	20	18	1811	51	336
Glouton	InsertionAuPlusProche	587	158	196	8	30	318	36	19	559	53	368
Génétique	Roulette	639	114	235	6	20	temps trop long	56	29	temps trop long	44	388
Génétique	Par Rang	629	122	233	7	20	temps trop long	42	24	temps trop long	48	357

GRAPHE BIPARTI

L'heuristique la plus efficace pour ce graphe est celle du plus proche voisin, en effet celui-ci peut passer d'une partie à l'autre et est donc très efficace dans un graphe où chaque partie contient le même nombre de sommets comme celui que nous avons utilisé pour les tests.

GRAPHE BIPARTI COMPLET

Les résultats pour ce graphe sont similaires à ceux du biparti simple à la différence du fait que les algorithmes génétiques gagnent beaucoup en efficacité et passent au-dessus de toutes les autres heuristiques à l'exception de celle du plus proche voisin. On peut supposer que cette amélioration des algorithmes génétiques est liée à la grande quantité d'arêtes dans le graphe qui rend alors plus utile leur capacité à effectuer de nombreux tests. Cependant leur temps d'exécution de plus de 7 minutes les rend peu avantageux en comparaison d'autres algorithmes comme celui de Christofides ou du plus proche voisin.

GRAPHE BIPARTI COMPLET A DISTANCES ALEATOIRES

On constate que les résultats sont proportionnellement très similaires à ceux obtenus sur le graphe biparti complet à chemins égaux, on peut donc supposer que le poids des arêtes d'un graphe dans le cadre d'un biparti n'impacte que très peu le type d'algorithme qui sera le plus efficace.

GRAPHE METRIQUE

On constate des résultats très similaires entre les différents algorithmes, à la différence notable des temps d'exécution des algorithmes génétiques et des résultats de ces derniers et des algorithmes d'insertion qui sont en deçà des autres heuristiques.

GRAPHE SIMPLE 1

De nouveau peu d'algorithmes se démarquent, comme usuellement les algorithmes génétiques prennent les temps d'exécution les plus longs mais obtiennent des résultats satisfaisants mais cette fois-ci les algorithmes d'insertion on des résultats équivalents à environ 3/2 ceux des autres algorithmes.

GRAPHE SIMPLE 2

Beaucoup d'algorithmes semblent atteindre leurs limites dans ce graphe très simple, cela semble être dû au grand degré du graphe. Seuls l'insertion au plus proche et l'algorithme du plus proche voisin semblent ne pas être réellement affectés par ces conditions.

GRAPHE EN PEIGNE

A la façon du graphe métrique cet algorithme très simple pose étonnamment beaucoup de problèmes à certaines heuristiques. En effet seul le voisinage d'une tournée ainsi que l'algorithme de Christofides et le plus proche voisin parviennent à des résultats réellement satisfaisants. On peut s'attendre à ce que ces problèmes soient causés par la petite quantité d'arêtes dans le graphe.

GRAPHE TRIBU

Comme sur le graphe précédent, les algorithmes génétiques et d'insertion sont mis à mal par ce graphe pour lequel leurs résultats sont très insatisfaisants. On peut supposer que le point de passage central de la symétrie du graphe peut provoquer en partie ces problèmes.

GRAPHE EDOUARD

Encore une fois l'algorithme du plus proche voisin est le plus efficace et le plus économe en temps d'exécution. En outre les algorithmes d'insertion semblent encore une fois mis à mal par le très grand cycle qui compose le graphe.

GRAPHE DE PETERSEN

Les observations sur ce graphe n'ont rien de réellement notable par rapport aux précédents.

GRAPHE DE KITTELL

Les résultats sur ce graphe sont légèrement différents des précédents du fait de l'efficacité de l'algorithme recherche locale qui est notablement plus efficace que les autres, on peut supposer que cette différence d'efficacité est due à la quantité de chemins qui apporte plus de potentielles améliorations à chaque inversion d'éléments.