Algorithme HDS pour le Problème du Voyageur de Commerce

Mohamed Elakef Zenagui

1 Introduction

Le problème du voyageur de commerce (*Travelling Salesman Problem*, TSP) consiste à déterminer une tournée hamiltonienne de coût minimal passant exactement une fois par chaque sommet d'un graphe complet valué. Ce problème est NP-difficile.

Dans ce travail, nous présentons un algorithme exact basé sur le principe du *Branch and Bound*, utilisant une borne inférieure obtenue par l'heuristique de la demi-somme, noté **HDS**.

2 Structures de données

CONSTANTE **n** = 10

TYPE **Arrete** = enregistrement
sommet : entier
suiv : ↑ Arrete
FIN

TYPE **GrapheM** = enregistrement
n : entier
M : Tableau[1..n][1..n] de **réel**
FIN

TYPE **NoeudTas** = enregistrement
borne : réel
cout : réel
cycle : ↑ Arrete
suiv : ↑ NoeudTas
FIN

TYPE **Tas** = ↑ **NoeudTas**

3 Heuristique de la demi-somme

L'heuristique de la demi-somme fournit une borne inférieure du coût restant à parcourir. Pour chaque sommet on considère la demi-somme des deux plus petites arêtes incidentes admissibles.

Input: GrapheM G, $\uparrow Arrete\ c$
Output: Entier borne
Var k, i : Entier
distances : Tableau[1..n] de réels
p, q, r : $\uparrow Arrete$

$borne \leftarrow 0$
$k \leftarrow |c|$
for $i \leftarrow 1$ **to** $G.n$ **do**
  if $k = 1$ ***or not*** $i \in sommets(c)$ **then**
    $distances \leftarrow tri(G.M[i])$
    $borne \leftarrow borne + distances[2] + distances[3]$
  end
end
$p \leftarrow c$
$q \leftarrow p \uparrow suiv$
while $q \neq NIL$ **do**
  if $p = c$ ***ou*** $q \uparrow suiv = NIL$ **then**
    $borne \leftarrow borne + G.M[p \uparrow sommet, q \uparrow sommet]$
    if $q \uparrow suiv = NIL$ **then**
      $distances \leftarrow tri(G.M[q \uparrow sommet])$
    end
    else
      $distances \leftarrow tri(G.M[p \uparrow sommet])$
    end
    if $distances[2] = G.M[p \uparrow sommet, q \uparrow sommet]$ **then**
      $borne \leftarrow borne + distances[3]$
    end
    else
      $borne \leftarrow borne + distances[2]$
    end
  end
  else
    $r \leftarrow q \uparrow suiv$
    $borne \leftarrow borne + G.M[p \uparrow sommet, q \uparrow sommet] + G.M[q \uparrow sommet, r \uparrow sommet]$
  end
  $p \leftarrow q$
  $q \leftarrow q \uparrow suiv$
end
return $\frac{borne}{2}$

Algorithm 1: Heuristique de la demi-somme

4 Algorithme HDS

L'algorithme HDS explore l'espace des solutions partielles en privilégiant les nœuds possédant la plus petite borne inférieure.

```
Input: GrapheM G
Output: ↑ Arrete best_solution, Réel best_cost
Var u, v, n : Entier
    borne, h : Réel
    T : Tas
    c, new_c : ↑ Arrete
n ← |M|
best_cost ← +∞
best_solution ← NIL
T ← ∅
c ← ∅
ajouter(c, 1)
cost ← 0
borne ← Heuristique_Demi_Somme(M, c)
Entasser(T, ⟨borne, cost, c⟩)
while T ≠ NIL do
  | Détasser(T, ⟨borne, cost, c⟩)
  | if borne < best_cost then
  |   | if |c| = n then
  |   |   | total_cost ← cost + M[dernier(c)][1]
  |   |   | if total_cost < best_cost then
  |   |   |   | best_cost ← total_cost
  |   |   |   | best_solution ← c
  |   |   | end
  |   | end
  |   | else
  |   |   | u ← dernier(c)
  |   |   | for v ← 1 to n do
  |   |   |   | if not v ∈ sommets(c) then
  |   |   |   |   | new_cost ← cost + M[u][v]
  |   |   |   |   | new_c ← c
  |   |   |   |   | ajouter(new_c, v)
  |   |   |   |   | h ← Heuristique_Demi_Somme(M, new_c)
  |   |   |   |   | if h < best_cost then
  |   |   |   |   |   | Entasser(T, ⟨h, new_cost, new_c⟩)
  |   |   |   |   | end
  |   |   |   | end
  |   |   | end
  |   | end
  | end
end
return (best_solution, best_cost)
```

Algorithm 2: Algorithme HDS

5 Conclusion

L'algorithme HDS est un algorithme exact pour la résolution du TSP. L'utilisation de l'heuristique de la demi-somme permet d'obtenir une borne inférieure admissible, réduisant considérablement l'espace de recherche par élagage.