Méthodes Heuristiques

ou comment résoudre efficacement les problèmes difficiles.

Dr Jean-Philippe Préaux,

2010

Comment résoudre un problème NP-difficile ?

- On a vu en optimisation combinatoire plusieurs classes de problèmes NP-difficiles, dont :
  - Coloration de graphe,
  - Problème du sac à dos,
  - Problème du voyageur de commerce,
  - Problème de satisfaisabilité des clauses.
- On a vu pourquoi (P=NP?), sauf à être éxagéremment optimiste (pessimiste?) on ne pouvait espérer en trouver une solution vraiment efficace, *i.e.* un algorithme de complexité polynomiale.

Comment résoudre un problème NP-difficile ?

- Cependant, la technologie moderne impose tous les jours de résoudre aussi efficacement que possible de tels problèmes avec des tailles de données gigantesques :
  - Réseaux modernes de communication,
  - Gestion du trafic aérien,
  - Génie logiciel, Intelligence artificielle,
  - Organisation logistique,
  - Optimisation des coûts, des bénéfices, des stocks, dans l'industrie, l'économie,
  - etc...
- Comment alors doit-on procéder ?... C'est le sujet de ce cours.

Comment résoudre un problème NP-difficile ?

On distingue deux types de méthodes :

- Les méthodes exactes, qui retournent une solution optimale -de complexité exponentielle en théorie- mais qui, en moyenne, s'avèrent efficientes.
- Les méthodes approchées ou ***heuristiques***, qui construisent en un temps acceptable une solution raisonnable sans aucune garantie d'optimalité.

Les méthodes exactes

- **Méthodes arborescentes** : (ou branch & bound. ex: algo. de Little pour le PVC).

  Elles cherchent ‘intelligemment’ parmi l’ensemble des solutions à déterminer une solution optimale sans analyser tous les cas.
- **Programmation dynamique** :

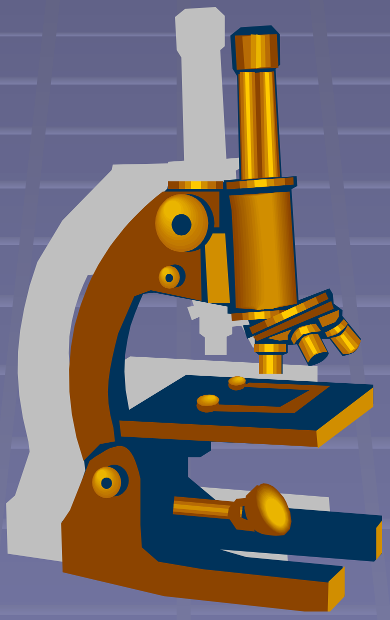
  Elles déterminent inductivement une solution en ramenant un problème sur N données à un problème sur N-1 données.
- La **programmation linéaire en nombres entiers** (PLNE)

  Elle développe principalement 2 types d’approche dont l’une n’est autre qu’une méthode arborescente.

Les heuristiques

- **Méthodes constructives :**
  Elles construisent une solution par une suite de choix partiels et définitifs, c’est à dire sans retour arrière. Lorsque à chaque itération elles prennent le choix le plus avantageux, on parle de méthodes *gloutonnes.*
- **Recherches locales :**
  On part d’une solution initiale et par transformations successives on construit une suite de solutions de coûts décroissants.
- **Métaheuristiques :**
  Il s’agit d’une méthode de recherche locale ou l’on s’autorise d’augmenter temporairement la fonction économique pour éviter de rester piégé en un minimum local. En général elles convergent en probabilité.

Evaluation des Heuristiques



Jean-Philippe Préaux

http://www.i2m.univ-amu.fr/~preaux

Evaluation des Heuristiques

- Le problème de l’évaluation des heuristiques est crucial : en effet n’ayant aucune garantie d’optimalité on peut dans une implémentation être proche ou très éloigné d’un optimum.
- Pour une donnée *D* on note *H(D)* le coût d’une heuristique *H* et *Opt(D)* le coût optimal. On appelle *performance relative* le quotient :

$$R_H(D) = \frac{H(D)}{Opt(D)}$$

  qui est ≥ 1 pour un problème de minimisation et ≤ 1 pour un problème de maximisation.
- Elle peut être parfois évaluée à priori ou être imprévisible et constatée à postériori.

Evaluation à priori

- On appelle *performance relative au pire* d'une heuristique *H* :

$$P_H = \max_D \left\{ R_H(D) \right\}$$

c'est à dire sa plus mauvaise performance relative sur l'ensemble des données.

- Il s'agit d'une garantie de performance obtenue mathématiquement par l'analyse de l'heuristique *H.*
- C'est difficile à obtenir et l'on n'en connaît que pour quelques heuristiques.

Exemples de performances relatives au pire

- on connaît une heuristique de PRP 1.5 pour le PVC (Δ-PVC) lorsque les coûts vérifient l'inégalité triangulaire $C_{ij} \leq C_{ik}+C_{kj}$ et sont symétriques $C_{ij}=C_{ji}$.

  => Le résultat de cet heuristique n'est jamais à plus de 50% de l'optimum.
- On ne connaît pas d'heuristique pour le problème de coloration de graphe avec une PRP meilleure que O(N/log N), où N=card(*D*), en particulier bornée.

- Il est bien rare de pouvoir déterminer la PRP. En voici cependant un exemple :
- Problème du transversal minimal : dans un graphe non orienté trouver un ensemble de sommets *T*, de cardinal minimal tel que toute arête en soit incidente.

<u>Heuristique :</u>

T est vide

Tant qu'il reste des arêtes

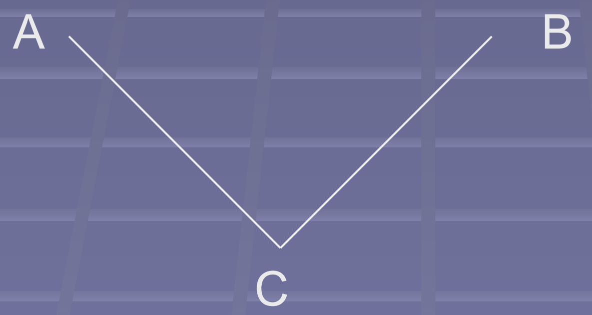
choisir une arête, ajouter ses extrémités à *T*

Les supprimer du graphe, ainsi que toute arête incidente.

Fin Tant que

- Le résultat obtenu est clairement un transversal.

- Tout transversal du graphe, y compris un transversal minimal, doit contenir au moins un sommet de chaque arête choisie dans l'heuristique. Aussi aucun ne peut contenir moins de la moitié des sommets de *T*. Aussi $P_H \leq 2$.
- Voici un exemple où $R_H = 2$:



Choisir [A,C]. On obtient un transversal {A,C} de cardinal 2, alors que celui minimal, {C}, est de cardinal 1. Ainsi, $P_H = 2$.

Evaluation à posteriori

- Le plus souvent, on ne sait pas évaluer la performance relative au pire. On ne peut alors qu'évaluer la performance relative sur les résultats obtenus après exécution de l'heuristique, et encore, à condition de connaître l'optimum.
- Mais en général le calcul de l'optimum ne peut se faire en temps raisonnable.
- On peut obtenir une évaluation moins fine si l'on dispose d'une évaluation de l'optimum : par défaut (minorant) pour un minimum ou par excès (majorant) pour un maximum.
- Exemple : Dans la recherche d'un <u>minimum</u>, pour une donnée *D*, en notant *B(D)* une évaluation par défaut de *Opt(D)* :

$$1 \leq R_H(D) = \frac{H(D)}{Opt(D)} \leq \frac{H(D)}{B(D)}$$

- En particulier si *H(D)/B(D)*=1, on est sûr d'avoir atteint un optimum.

Exemples de Bornes min *B(D)* pour le PVC

Soit D la matrice nxn des coûts :

- borne inf naïve :

  la somme des n plus petits éléments de D.
- borne inf moins naïve :

  la somme des minima de chaque ligne.
- borne inf de Little :

  la somme des nombres soustraits aux lignes & colonnes dans la ‘réduction de la matrice’ (cf. algo de Little).

Exemples de Bornes min *B(D)* pour le PVC

- Si D est symétrique (graphe non orienté) :
- Si dans un circuit hamiltonien on supprime une arête on obtient un arbre recouvrant (i.e. contenant chaque sommet) de coût inférieur.
- <u>borne inf : coût minimal d'un arbre recouvrant</u>.
- C'est un problème facile (en $O(N^2)$), dont voici un algorithme :

Initialement l'arbre T est réduit à un sommet

Pour k variant de 1 jusqu'à N-1

  Chercher l'arête [i , j] de coût min t.q. $I \in T$ et $j \notin T$

  Ajouter [i , j] à T

- **Exemple :**
- Borne naïve :
  $B_1(D)=6$.
- Borne moins naïve :
  $B_2(D)=7$.
- Borne de Little :
  $B_3(D)=9$.
- Coût minimal d'un arbre recouvrant :
  $B_4(D)=7$ (arbre : [A,B],[B,E],[E,C],[C,D]).

	A	B	C	D	E
A	-	3	4	5	4
B	3	-	2	2	1
C	4	2	-	1	2
D	5	2	1	-	3
E	4	1	2	3	-

- **Heuristique du plus proche voisin :**
  Partir du sommet A et construire un cycle en reliant le dernier sommet visité au plus proche sommet libre. On obtient A-B-E-C-D-A de coût 3+1+2+1+5=12. La performance relative au pire sur C est au plus 12/9=1.33, soit une erreur d'au plus 33%. En fait le coût optimal est 12 (appliquer l'algo de Little). Par chance c'est optimal...

Evaluation statistique(à posteriori)

- On est souvent contraint à cette extrémité pour les problèmes sur de grande taille de données.
- On compare diverses méthodes sur une série de problèmes générés aléatoirement (ex : pour le PVC sur N sommets attribuer aléatoirement les coûts à l'aide d'une loi de probabilité uniforme sur $[0,C_{max}]$, TSPLIB) aux méthodes existantes.
- On compare leur résultat, leur temps d'exécution minimaux, maximaux, moyens, les écarts types, à un résultat optimal (si c'est possible) ou à une borne inférieure.
- Cela présente par ailleurs un avantage : même si l'on connaît la PRP, elle peut n'être atteint que dans des cas pathologiques. Une performance 'en moyenne' présente un grand intérêt.
- Toute amélioration substantielle est intéressante, comme par exemple la meilleure heuristique 'en moyenne ' selon la taille des données.