Optimisation en informatique
RCP104
Chapitre 1 : Introduction

Sourour Elloumi

Plan

- Problèmes d'Optimisation Combinatoire : définition et exemples
- Un POC est-il toujours facile à résoudre ? Élèments de théorie de la complexité
- Introduction aux approches que nous adopterons dans la suite de l'UE :
  - Méthodes de résolution exacte : Programmation mathématique, Algorithmes de résolution, résolution à l'aide d'un logiciel
  - Méthodes de résolution approchée :Heuristiques

Problèmes d'Optimisation Combinatoire : définition et exemples

- Wikipédia : L'optimisation combinatoire est une branche de l'optimisation en mathématiques appliquées et en informatique, également liée à la recherche opérationnelle, l'algorithmique et la théorie de la complexité.
- On parle également d'optimisation discrète

Exemple 1: Plus court chemin

- Problème du plus court chemin entre 2 points d'une ville représentée par un graphe.
- Solution réalisable (admissible) : un chemin -ensemble d'arcs- qui relie les deux points.
- Coût d'un chemin : somme des longueurs de ses arcs

Exemple 2 : Achat de billets d'avion

Un homme d'affaires résidant à Paris (P) doit aller à Rome (R) le lundi et rentrer le mercredi pendant 5 semaines consécutives. Le prix d'un billet aller-retour : 400 euros. Réduction de 20% si un week-end est inclus. Aller simple : 75 % du prix aller-retour.

- Solution 1 : 5 A/R lundi-mercredi de la même semaine ; coût 5*400=2000 euros
- Solution 2 : 1 P-R lundi de la première semaine, 4 A/R mercredi-lundi de la semaine suivante, 1 R-P le mercredi de la dernière semaine ; coût 400*(0.75+4*0.8+0.75)= 1880 euros
- Solution 3 : 1 A/R lundi de la première semaine- mercredi de la dernière semaine et 4 A/R mercredi-lundi de la semaine suivante ; coût 400*5*0.8=1600 euros

On ne peut pas faire mieux que la Solution 3. Elle est optimale.

Exemple 3 : Localisation de concentrateurs (notre exemple de base)

Données : terminaux, sites potentiels pour concentrateurs,

- distances (en km) sites-terminaux et coût d'un km de raccordement
- coût d'installation d'un concentrateur sur un site

dist (km)	S1	S2	S3	S4
T1	1	2	1	4
T2	6	1	6	3
T3	5	2	3	1
T4	3	3	7	8
T5	4	5	3	2

	S1	S2	S3	S4
Coût inst (euro)	110	200	100	100

Coût d'un km de raccordement : 15 euros

Exemple 3 : Localisation de concentrateurs

dist (km)	S1	S2	S3	S4
T1	1	2	1	4
T2	6	1	6	3
T3	5	2	3	1
T4	3	3	7	8
T5	4	5	3	2

	S1	S2	S3	S4
Coût inst (euro)	110	200	100	100

Coût d'un km de raccordement : 15 euros

Solution 1 : S1, S2 et S3 ouverts

On a intérêt à raccorder T1 à S1 (ou S3), T2 à S2, T3 à S3, T4 à S1 (ou S2) et T5 à S3

Coût d'installation : 110+200+100 = 310

Coût de raccordement : 15(1+1+2+3+3)=150

Coût total : 310+150= 460

7

Exemple 3 : Localisation de concentrateurs

dist (km)	S1	S2	S3	S4
T1	1	2	1	4
T2	6	1	6	3
T3	5	2	3	1
T4	3	3	7	8
T5	4	5	3	2

	S1	S2	S3	S4
Coût inst (euro)	110	200	100	100

Coût d'un km de raccordement : 15 euros

Solution 2 : S3 ouvert

On raccorde tous les sites à S3

Coût d'installation : 100

Coût de raccordement : 15(1+6+3+7+3)=300

Coût total : 100+300= **400**

8

Point commun aux POC

- Espace discret de ***solutions réalisables***
- On veut trouver une ***solution optimale*** (minimise ou maximise une ***fonction objectif***)
- Peut-on énumérer tous les cas possibles pour déduire une solution optimale ?
- **NON**- en général beaucoup trop de cas possibles
- La **Théorie de la complexité** donne un cadre de classification des problèmes en « faciles » ou « difficiles »

9

Complexité des Algorithmes

Instance d'un problème :
Ensemble de valeurs fixant les données d'un problème

Algorithme :
Séquence finie d'étapes pour résoudre un problème (l'algorithme est exécuté sur une instance pour la résoudre)

Analyser un algorithme, c'est :
- Prouver sa correction
- Donner une estimation de son **temps d'exécution**

10

Complexité des Algorithmes

estimer le temps d'exécution d'un algorithme

=

donner un ordre de grandeur du nombre d'opérations élémentaires qu'il nécessite au pire des cas, en fonction de **la taille des données**

=

« calculer sa complexité »

11

Complexité des Algorithmes

Notation :
$g(n)=O(f(n)) \Leftrightarrow$ il existe c et n_0 tels que
$\forall\ n>n_0,\ g(n) \leq c*f(n)$

Interprétation : en comportement asymptotique, g(n) ne croît pas plus vite que f(n)

Usage : un algorithme qui nécessite $5n^2 + 45n+230$ opérations où n est la taille du problème est dit de complexité $O(n^2)$

12

Complexité des Algorithmes

Estimation du nombre d'opérations étant donnés n et la complexité

n	10	100	1000
$O(n)$	10	100	1000
$O(n\log(n))$	33	664	9666
$O(n^3)$	10^3	10^6	10^9
$O(2^n)$	1024	$\approx 10^{30}$	$\approx 10^{300}$
$O(n!)$	3628800	$\approx 10^{158}$	$\approx 10^{2567}$

13

Complexité des Algorithmes

- Supposons qu'un algorithme de complexité $O(n^3)$ soit capable de résoudre, en 1 heure, un problème de taille au maximum n=100 sur une machine M. Quelle est est la nouvelle taille maximale n' résolue en 1 heure sur une machine M' deux fois plus rapide que M ?
- Même question avec un algorithme de complexité $O(2^n)$

Réponse : ≈126 puis 101

⇒ Les algorithmes polynomiaux sont bien plus efficaces que les algorithmes exponentiels

14

Complexité des Algorithmes

Mais, on ne trouve pas toujours un algorithme polynomial pour un problème donné

⇒ Théorie de la complexité

Classer ***les problèmes*** *de décision* par type d'algorithme qu'on peut mettre en œuvre pour les résoudre

15

Théorie de la Complexité (complexité des problèmes)

- **Problème de décision** : Problème dont les 2 seules réponses possibles sont oui et non

  Exemple : Etant donné un graphe, existe-t-il un circuit hamiltonien ?

- Autres : **Problèmes d'optimisation**

  Exemple : Etant donné un graphe valué, trouver un circuit hamiltonien de coût minimal ?

16

Théorie de la Complexité

- un problème d'optimisation peut toujours être transformé en un problème de décision (version de décision d'un problème d'optimisation)

  Exemple : Etant donné un graphe valué et un entier k, existe-t-il un circuit hamiltonien de coût ≤ k ?

je ne peux pas résoudre le problème de décision en temps polynomial

⇒

je ne peux pas résoudre le problème d'optimisation en temps polynomial

On peut se restreindre à la classe des problèmes de décision

17

Théorie de la Complexité

Classe P : ensemble des problèmes de décision pour lesquels il existe un algorithme polynomial

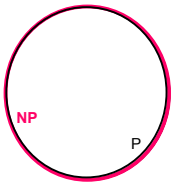
Un problème de décision possède **la propriété de certificat affirmatif** s'il existe un algorithme polynomial qui permet de vérifier la validité d'une réponse oui à ce problème. Ex : soit un ordre des sommets d'un graphe, vérifier qu'il constitue un circuit hamiltonien.

Classe NP : ensemble des problèmes de décision possédant la propriété de certificat affirmatif

18

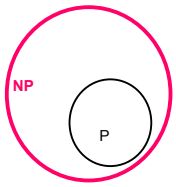
Théorie de la Complexité

Évidemment, NP contient P. Mais laquelle de ces deux configurations est la bonne ?



P = NP ?

ou



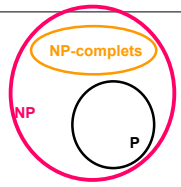
P ≠ NP

Conjecture

Théorie de la Complexité

On a défini une autre classe de problèmes dans NP : les problèmes **NP-complets**, tous liés entre eux par la notion de réduction polynomiale

Le problème R est **réductible** en P ($R \propto P$) si, pour toute instance I_R de R, on peut construire une instance I_P de P telle que réponse de I_R=réponse de I_P. De plus, la transformation de I_R en I_P doit se faire par un algorithme polynomial



P ≠ NP

Conjecture

Théorie de la Complexité

Un problème P de NP est **NP-complet** ssi
$\forall R \in NP, R \propto P$

Théorème : Cook (1971)

Le problème SAT est NP-complet

SAT : Etant donnée une Forme Normale Conjonctive (FNC) contenant n variables booléennes, existe-t-il une affectation des valeurs vrai ou faux aux variables telle que la FNC ait la valeur vrai ?

Exemple : $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee x_4) \wedge x_5$

Théorie de la Complexité

Exercice

Soit R un problème NP-complet. Supposons qu'il existe un algorithme polynomial pour résoudre R. Montrer alors que P=NP

A ce jour, personne n'a encore trouvé un tel problème R

Et personne n'a pu montrer qu'un tel problème n'existait pas

Voir http://www.claymath.org/millennium/P_vs_NP/

Théorie de la Complexité

Etant donné un problème P qui n'a pas encore été classé :

- soit on essaie de trouver un algorithme polynomial pour le résoudre
- soit on montre qu'il est NP-complet, donc ce n'est plus la peine de chercher un algorithme polynomial
- soit on cherche encore …

Pour montrer qu'un problème est NP-complet,

- montrer qu'il est dans NP
- trouver un problème R dont on sait qu'il est NP-complet et prouver $R \propto P$ (car la relation $\propto$ est transitive)

Théorie de la Complexité

Collections de problèmes :

- Livre de Garey et Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, 1979

- site Internet :

http://www.nada.kth.se/~viggo/wwwcompendium/

Théorie de la Complexité

Un problème d'optimisation est **NP-difficile** ssi le problème de décision qui lui est associé est NP-complet

Je sais que mon problème est NP-difficile. Qu'est-ce que je fais :

1. Je veux quand même une solution optimale (méthodes exactes) : elle sera coûteuse en temps de calcul et impossible à calculer à partir d'une certaine taille
2. Je cherche une solution avec une garantie a priori sur l'erreur (méthodes approchées). De telles méthodes n'existent pas pour tous les problèmes
3. Je cherche une solution admissible si je peux (méthodes heuristiques), sans garantie sur la qualité de cette solution

Retour vers le problème de localisation de concentrateurs

- **Exercice** : Montrer que le problème de localisation de concentrateurs est NP-difficile
- Résolution exacte : formulation par un programme mathématique
- Résolution par une heuristique « gourmande » ou autre

Exemple 3 : Localisation de concentrateurs (notre exemple de base)

Données : terminaux, sites potentiels pour concentrateurs,

- distances (en km) sites-terminaux et coût d'un km de raccordement
- coût d'installation d'un concentrateur sur un site

dist (km)	S1	S2	S3	S4
T1	1	2	1	4
T2	6	1	6	3
T3	5	2	3	1
T4	3	3	7	8
T5	4	5	3	2

	S1	S2	S3	S4
Coût inst (euro)	110	200	100	100

Coût d'un km de raccordement : 15 euros

Localisation de concentrateurs- Formulation par un programme mathématique

- **Variables :**

y_j= 1 si un concentrateur est installé sur le site Sj
= 0 sinon
Cela nous fait 4 variables y_1, y_2, y_3 et y_4

x_{ij}= 1 si le terminal Ti est raccordé au site Sj
= 0 sinon
Cela nous fait 20 variables x_{11}, x_{12}, x_{13}, x_{14}, x_{21}, x_{22}… et x_{54}

Décrivent toutes les possibilités

Localisation de concentrateurs- Formulation par un programme mathématique

- **Contraintes :**

$x_{11} + x_{12} + x_{13} + x_{14} = 1$ T1 est raccordé à exactement 1 site

$x_{21} + x_{22} + x_{23} + x_{24} = 1$ T2 est raccordé à exactement 1 site

…

et

$x_{11} \leq y_1$ T1 n'est raccordé à S1 que s'il est équipé d'un concentrateur

$x_{12} \leq y_2$ T1 n'est raccordé à S2 que s'il est équipé d'un concentrateur

…

et x et y ne peuvent valoir que 0 ou 1

Définissent l'ensemble des solutions réalisables

Localisation de concentrateurs- Formulation par un programme mathématique

- **Coût d'une solution :**

$$110y_1 + 200y_2 + 100y_3 + 100y_4$$

Coût d'installation des concentrateurs sur les sites

+

$$15*(1x_{11} + 2x_{12} + 1x_{13} + 4x_{14} + 6x_{21} + \cdots + 2x_{54})$$

Coût de raccordement des terminaux aux sites

On veut trouver une solution de coût minimum : **fonction objectif**

Localisation de concentrateurs- Formulation par un programme mathématique

- On a obtenu un *programme mathématique*
- Résoudre ce programme revient à
  - Associer des valeurs aux variables
  - Qui satisfont les contraintes
  - Et minimisent la fonction objectif

31

Localisation de concentrateurs- Formulation par un programme mathématique

- Pour **résoudre**, on dispose de méthodes mathématiques : programmation linéaire et programmation linéaire en nombres entiers ; vus plus tard dans le cours
- Qui sont implémentées dans des logiciels

Ci-après une façon de faire en utilisant le logiciel libre glpk :

32

Localisation de concentrateurs- Formulation par un programme mathématique

- On écrit un fichier au format « lp » `loc_conc1.lp`
- On résout par la commande :

```
glpsol --cpxlp loc_conc1.lp --output loc_conc1.sol
```

- On obtient le fichier `loc_conc1.sol`

33

Localisation de concentrateurs- Formulation par un programme mathématique

- J'en déduis le coût d'une solution optimale :

```
Objective:  f = 360 (MINimum)
```

- Et une solution optimale :

Sites S1 et S4 équipés de concentrateurs
(y_1=1, y_2=0, y_3=0, y_4=1)

T1, T2, T3, T4 et T5 raccordés respectivement à S1, S4, S4, S1 et S4
(x_{11}=1, x_{24}=1, x_{34}=1, x_{41}=1 , x_{54}=1) les autres x valent 0

34

Localisation de concentrateurs- Résolution par une heuristique gourmande

• La résolution exacte par la programmation mathématique peut se révéler coûteuse en temps de calcul, en particulier pour les instances de grande taille (complexité non polynomiale)

• On peut être moins ambitieux : rechercher une solution admissible non forcément optimale en se laissant guider par « le bon sens ». Cela donne lieu à des **heuristiques** plus ou moins sophistiquées.

•Les heuristiques **gourmandes** sont peu sophistiquées

35

Localisation de concentrateurs- Résolution par une heuristique gourmande

On peut se poser la question : si un seul site est équipé ?

dist (km)	S1	S2	S3	S4
T1	1	2	1	4
T2	6	1	6	3
T3	5	2	3	1
T4	3	3	7	8
T5	4	5	3	2

	S1	S2	S3	S4
Coût inst (euro)	110	200	100	100

Coût d'un km de raccordement : 15 euros

Coût (S1) : 110+ 15(1+6+5+3+4)= 395
Coût (S2) : 200+ 15(2+1+2+3+5)= 395
Coût (S3) : 100+ 15(1+6+3+7+3)= 400
Coût (S4) : 100+ 15(4+3+1+8+2)= 395

• Site le moins cher, S1 ou S2 ou S3. Choisissons au hasard d'équiper S2. Solution courante, S2 équipé, les autres non, de coût 395

36

Localisation de concentrateurs- Résolution par une heuristique gourmande

Maintenant que S2 est équipé, quel site nous ferait le plus baisser le prix ?

dist (km)	S1	S2	S3	S4
T1	1	2	1	4
T2	6	1	6	3
T3	5	2	3	1
T4	3	3	7	8
T5	4	5	3	2

	S1	S2	S3	S4
Coût inst (euro)	110	200	100	100

Coût d'un km de raccordement : 15 euros

Coût (S2, S1) : 200+ 110+ 15(1+1+2+3+4)=475

Coût (S2, S3) : 200+100+ 15(1+1+2+3+3)= 450

Coût (S2, S4) : 200+100+ 15(2+1+1+3+2)= 435

• Aucun autre site ne peut améliorer la solution courante. Arrêt.

37

Localisation de concentrateurs- Résolution par une heuristique gourmande

- Algorithme de cette heuristique :
  - Trouver le site le moins cher s'il est le seul équipé
  - Arret = faux
  - Tant que (Arret = faux) faire
    - Si possible, ajouter à la solution courante un site qui fait baisser au plus le coût
    - Sinon, Arret = vrai
  - fin

Algorithme de complexité polynomiale, qu'il faut coder dans un langage de programmation (C, java, …)

Algorithme « gourmand » car on ne revient jamais sur des choix précédemment effectués

38

Suite du cours

- Résolution exacte :
  - Modélisation d'un POC par un programme mathématique
  - Rappels de programmation linéaire
  - Programmation Linéaire en nombres entiers
  - Utilisation des logiciels de programmation mathématique
- Résolution approchée :
  - Heuristiques de recherche locale
  - Métaheuristiques
- Mise en pratique à travers un projet

39