

M1if09 – Modèles de calcul & complexité

sylvain.brandel@univ-lyon1.fr



COMPLEXITÉ

NP-COMPLÉTUDE

NP-complétude

- Etalons ou références
 - Calculabilité : problème de l'arrêt
 - Complexité ?
 - problèmes **NP-complets** (les plus compliqués de la classe NP)
- Problèmes NP-complets
 - Les problèmes NP-complets $\in NP$
 - Les problèmes de la classe $P \in NP$
 - Il existe des problèmes NP
 - Qui n'ont pas été montrés $\in P$
 - Et qui n'ont pas été montrés NP-complets
- Théorème

S'il existe un problème NP-complet décidé par un algorithme polynomial, alors tous les problèmes de NP sont décidables en temps polynomial

(c-à-d $P = NP$)

Réduction polynomiale

- Réduction
 - Décidabilité : fonction récursive
 - Complexité : fonction polynomiale
- Une fonction $\Sigma^* \rightarrow \Sigma^*$ est dite **calculable en temps polynomial** ssi il existe une MT (déterministe) polynomialement bornée qui la calcule
- Soient L_1 et $L_2 \subseteq \Sigma^*$.
Une **réduction polynomiale** (ou **transformation polynomiale**) de L_1 à L_2 est une fonction $\tau : \Sigma^* \rightarrow \Sigma^*$, calculable en **temps polynomial**, telle que
$$x \in L_1 \text{ ssi } \tau(x) \in L_2$$
- Lemme *Si τ_1 est une réduction polynomiale de L_1 vers L_2 et τ_2 est une réduction polynomiale de L_2 vers L_3 alors $\tau_1 \circ \tau_2$ est une réduction polynomiale de L_1 vers L_3*

Exemples de problèmes

- Planification de 2 machines (*2-machine scheduling*)
 - Soit un ensemble de n tâches avec des durées $S = \{a_1, a_2, \dots, a_n\}$
 - Répartir S sur 2 machines de sorte qu'une deadline D soit respectée
- Problème du sac-à-dos (*Knapsack*)
 - Soient un ensemble de n objets avec des poids $S = \{a_1, a_2, \dots, a_n\}$ et un entier K
 - Trouver un sous-ensemble $P \in S$ tel que $\sum_{a_i \in P} a_i = K$
- Problème de la partition (*Partition*)
 - Soit un ensemble de n entiers positifs $S = \{a_1, a_2, \dots, a_n\}$
 - Existe-t-il $P \in \{1, 2, \dots, n\}$ tel que $\sum_{a_i \in P} a_i = \sum_{a_i \notin P} a_i$?
- Problème du cycle hamiltonien (*HAM*)
 - Soit G un graphe
 - Existe-t-il dans G un cycle passant par chaque **sommet** une fois et une seule ?
- Satisfiabilité de formules booléennes (*SAT*)
 - Soit F une FNC (Forme Normale Conjonctive)
 - F est-elle satisfiable ?
- Coloriage d'un graphe (*COL*)
 - Soient G un graphe et K un entier
 - Est-il possible de colorier les sommets de G avec K couleurs de manière à ce que deux sommets adjacents n'aient pas la même couleur ?
- Ces problèmes sont équivalents du point de vue de la complexité polynomiale
 - On peut réduire polynomialement chacun d'eux dans les autres

Exemples de réduction

- Réduction HAM vers SAT
 - À partir d'un graphe G , on construit une FNC F telle que G hamiltonien ssi F est satisfiable
- Réduction SAT vers HAM
 - À partir d'une FNC F , on construit un graphe G tel que F est satisfiable ssi G est hamiltonien
- Réduction SAT vers 3-COL
 - À partir d'une FNC F , on construit un graphe G tel que F est satisfiable ssi G est 3-coloriable

NP-complétude

- Un langage L est **NP-complet** si
 - $L \in NP$
 - Pour **tout** langage $L' \in NP$, il existe une réduction polynomiale de L' dans L
- Le premier point est facile à établir
 - Algorithme non déterministe polynomial
- Le second point est plus délicat
 - Réduction depuis tout langage de NP
 - Ou alors, si on connaît un langage L'' NP-complet, il suffit de démontrer qu'il existe une réduction polynomiale de L'' dans L
- Un langage L est **NP-complet** si
 - $L \in NP$
 - Il existe **un** langage L'' **NP-complet** tel que il existe une réduction polynomiale de L'' dans L

Exemples de déduction

- Réduction HAM vers SAT
 - À partir d'un graphe G , on construit une FNC F telle que
 G hamiltonien ssi F est satisfiable
 - On en **déduit** que **si** HAM est NP-complet **et** $SAT \in NP$, **alors** SAT est NP-complet
- Réduction SAT vers HAM
 - À partir d'une FNC F , on construit un graphe G tel que
 F est satisfiable ssi G est hamiltonien
 - On en **déduit** que **si** SAT est NP-complet **et** $HAM \in NP$, **alors** HAM est NP-complet
- Réduction SAT vers 3-COL
 - À partir d'une FNC F , on construit un graphe G tel que
 F est satisfiable ssi G est 3-coloriable
 - On en **déduit** que **si** SAT est NP-complet **et** $3-COL \in NP$, **alors** 3-COL est NP-complet

Exemple de réduction : HAM vers SAT

- *HAM*
 - Soit G un graphe
 - Existe-t-il dans G un cycle passant par chaque **sommet** une fois et une seule ?
- Traduction en variables propositionnelles exprimant que :
un sommet i possède une certaine propriété vis-à-vis d'un **possible** cycle hamiltonien dans G
- On considère n^2 **variables** notées $x_{i,j}$ avec la **sémantique intuitive** :
 $x_{i,j}$ devrait être vraie si le sommet i de G est à la $j^{\text{ème}}$ position d'un cycle hamiltonien
- On cherche ensuite des équations logiques sous forme de disjonction exprimant formellement et globalement ce fait
- On doit exprimer
 - 1) À la $j^{\text{ème}}$ position doit apparaître exactement un sommet ← **cohérence cycle**
 - 2) Chaque sommet apparaît dans le cycle exactement une fois ← **HAM**
 - 3) On ne peut pas passer dans G du sommet j au sommet $(j + 1)$ s'il n'y a pas d'arête

(Seul le 3^{ème} point dépend de G)

Exemple de réduction : HAM vers SAT

1) À la $j^{\text{ème}}$ position doit apparaître exactement un sommet

a) À la $j^{\text{ème}}$ position doit apparaître au moins un sommet

$$f_j = x_{1,j} \vee x_{2,j} \vee \dots \vee x_{n,j}$$

1 clause par place j
→ n clauses

b) À la $j^{\text{ème}}$ position doit apparaître au plus un sommet (si c'est i ça n'est pas un autre)

$$\text{pour tout } i : x_{i,j} \Rightarrow \overline{x_{1,j}} \wedge \overline{x_{2,j}} \wedge \dots \wedge \overline{x_{(i-1),j}} \wedge \overline{x_{(i+1),j}} \wedge \dots \wedge \overline{x_{n,j}}$$

$$\text{ou } (x_{i,j} \Rightarrow \overline{x_{1,j}}) \wedge \dots \wedge (x_{i,j} \Rightarrow \overline{x_{(i-1),j}}) \wedge (x_{i,j} \Rightarrow \overline{x_{(i+1),j}}) \wedge \dots \wedge (x_{i,j} \Rightarrow \overline{x_{n,j}})$$

$$f_{i,j,k} = \overline{x_{i,j}} \vee \overline{x_{k,j}} \text{ pour tous les triplets } (i, j, k) \text{ tels que } i \neq k$$

→ $O(n^3)$ clauses

$$F_f = \bigwedge_{j=1}^n f_j \wedge \bigwedge_{(i,j,k) \in \{1,\dots,n\}^3, i \neq k} f_{i,j,k}$$

2) Chaque sommet apparaît dans le cycle exactement une fois

a) Chaque sommet apparaît au moins une fois

$$g_i = x_{i,1} \vee x_{i,2} \vee \dots \vee x_{i,n}$$

→ n clauses

b) Chaque sommet apparaît au plus une fois (si c'est i ça n'est pas un autre)

$$x_{i,j} \Rightarrow \overline{x_{i,k}} \text{ pour } k \neq j \quad (\text{si } i \text{ apparaît à la } j^{\text{ème}} \text{ place, il n'apparaît pas à la } k^{\text{ème}})$$

$$g_{i,j,k} = \overline{x_{i,j}} \vee \overline{x_{1,k}} \text{ pour tous les triplets } (i, j, k), k \neq j$$

→ $O(n^3)$ clauses

$$F_g = \bigwedge_{i=1}^n g_i \wedge \bigwedge_{(i,j,k) \in \{1,\dots,n\}^3, j \neq k} g_{i,j,k}$$

Exemple de réduction : HAM vers SAT

3) On ne peut pas passer dans G de j à $(j + 1)$ s'il n'y a pas d'arête

$$x_{i,j} \Rightarrow \overline{x_{k,(j+1)}} \text{ si l'arête } (i,k) \notin G$$

$$h_{i,j,k} = \overline{x_{i,j}} \vee \overline{x_{k,(j+1)}} \text{ pour tous les triplets } (i,j,k), (i,k) \notin G$$

→ $O(n^3)$ clauses

$$F_h = \bigwedge_{(i,j,k) \in \{1,\dots,n\}^3, \forall j, \forall (i,k), (i,k) \notin G} h_{i,j,k}$$

- On obtient $F = F_f \wedge F_g \wedge F_h$ qui est une FNC
- Cette traduction peut se faire à l'aide d'un algorithme polynomial
 - $O(n^3)$ clauses
 - $O(n^3)$ littéraux
- Proposition

*Soient G un graphe et F la FNC obtenue par la traduction précédente.
Alors G est hamiltonien ssi F est satisfiable.*

- Preuve

Exemple de réduction : HAM vers SAT

- Si G est hamiltonien, alors F est satisfiable
 - G hamiltonien : il existe une permutation π de $[1, \dots, n]$ telle que $(\pi^{-1}(i), \pi^{-1}(i + 1)) \in G$
 - On considère l'interprétation $I : x_{i,j} = \top$ si $j = \pi(i)$, \perp sinon
 - $I(x_{\pi^{-1}(j),j}) = \top$ donc $I(f_i) = \top$
 - Si $j = \pi(i)$ alors $k \neq \pi(i)$, donc $\overline{x_{i,k}} = \top$
 - Si $k = \pi(i)$ alors $j \neq \pi(i)$, donc $\overline{x_{i,j}} = \top$
 - Si $j \neq \pi(i)$ et $k \neq \pi(i)$, donc $\overline{x_{i,k}} = \top$ et $\overline{x_{i,j}} = \top$
 - Pareil pour g
 - Si $j = \pi(i)$ alors $j + 1 \neq \pi(k)$ (pas d'arête de i à k) donc $\overline{x_{k,(j+1)}} = \top$ donc $I(h_{i,j,k}) = \top$
 - Si $j \neq \pi(i)$ alors $I(h_{i,j,k}) = \top$

$\Rightarrow I(F) = \top$

- Si F est satisfiable, alors G est hamiltonien
 - F satisfiable : il existe une interprétation I telle que $I(F) = \top$
 - Un littéral est interprété à \top dans chaque clause de f , I induit une bijection sur les sommets de G :
 - Pour chaque i , un et un seul $x_{i,j}$ est tel que $I(x_{i,j}) = \top$
 - Pour chaque j , un et un seul $x_{i,j}$ est tel que $I(x_{i,j}) = \top$
 - Notons $\pi(i)$ l'unique j pour lequel $I(x_{i,\pi(i)}) = \top$
 - Pareil pour g
 - Les clauses de h sont aussi satisfaites :
 - Si $i = \pi(j)$ et $k = \pi(j + 1)$ alors (j, k) est une arête de G , et donc $\pi(1), \pi(2), \dots, \pi(n)$ est un cycle hamiltonien de G

$\Rightarrow G$ est hamiltonien

Théorème de Cook

- Jusqu'ici on ne peut que faire des **déductions**
 - Si P_1 est NP-complet et réduction de P_1 à P_2 et $P_2 \in NP$, alors P_2 est NP-complet
- Il manque un problème montré NP-complet directement
 - On pourra en déduire tous les autres
- **SAT**
 - Soit F une formule propositionnelle en forme normale conjonctive
 - Existe-t-il une interprétation qui rend F vraie ?
(Satisfiabilité d'une FNC)
- Théorème

Le problème SAT est NP-complet
- Preuve (Stephen A. Cook, Canada, 1939 -)
 - Premier problème NP-complet prouvé (1971)

Théorème de Cook

Preuve

Pour prouver SAT NP-complet, on doit montrer

- 1) SAT $\in NP$
- 2) **Tout** problème NP peut être réduit à SAT

1) SAT $\in NP$

- On génère de façon non déterministe une interprétation
- On vérifie que cette interprétation rend la formule vraie

Théorème de Cook

Preuve

2) On peut réduire polynomialement tout problème NP à SAT

- Soit $L \in NP$
 - L décidé par $M = (K, \Sigma, \Gamma, \Delta, s, F)$, non déterministe polynomialement bornée par p
 - $\tau : M, w \rightarrow$ une instance de SAT positive ssi M accepte w
 - M accepte w ssi il existe une **exécution** de M sur w d'au plus $p(n)$ étapes ($n = |w|$)
- **Exécution** : suite d'au plus $p(n) + 1$ configurations successives de M
 - Tableau $R [p(n) + 1] [p(n) + 1]$ de **symboles** du ruban de M
 $R(i, j) =$ symbole dans la $j^{\text{ème}}$ case du ruban de M à la $i^{\text{ème}}$ configuration (étape)
 - Vecteur $P [p(n) + 1]$ d'**entiers** $1 \dots p(n) + 1$
 $P(i) =$ position de la tête de lecture de M à la $i^{\text{ème}}$ étape
 - Vecteur $Q [p(n) + 1]$ d'**états** de M
 $Q(i) =$ état courant de M à la $i^{\text{ème}}$ étape
 - Vecteur $C [r]$ d'**entiers** $1 \dots r$ (degré du non-déterminisme)
 $C(i) =$ choix non déterministe effectué par M à la $i^{\text{ème}}$ étape
- On considère $O(p(n)^2)$ variables
 - $r_{i,j,\alpha}$ devrait être vraie si $R(i, j) = \alpha \rightarrow (p(n) + 1)^2 \cdot |\Gamma|$ variables
 - $p_{i,j}$ devrait être vraie si $P(i) = j \rightarrow (p(n) + 1)^2$ variables
 - $q_{i,k}$ devrait être vraie si $Q(i) = k \rightarrow (p(n) + 1) \cdot |K|$ variables
 - $c_{i,j}$ devrait être vraie si $C(i) = j \rightarrow (p(n) + 1) \cdot r$ variables

} interprétation (**)

Théorème de Cook

Preuve

- On doit exprimer
 - 1) $R(i, j)$ contient exactement une valeur
 - 2) $P(i)$ contient exactement une valeur
 - 3) $Q(i)$ contient exactement une valeur
 - 4) $C(i)$ contient exactement une valeur
 - 5) La première configuration est initiale
 - 6) Chaque configuration est obtenue à partir de la précédente
 - 7) À la fin de l'exécution on arrive dans un état acceptant

Théorème de Cook

Preuve

1) $R(i, j)$ contient exactement une valeur

$$r_{i,j,\alpha_0} \vee r_{i,j,\alpha_1} \vee \dots \vee r_{i,j,\alpha_s} \quad (s = |\Gamma|)$$

$$r_{i,j,\alpha} \Rightarrow \overline{r_{i,j,\beta}}, \quad \alpha, \beta \in \Gamma, \quad \alpha \neq \beta$$

→ $O(p(n)^2)$ clauses

2) $P(i)$ contient exactement une valeur

→ $O(p(n)^2)$ clauses

3) $Q(i)$ contient exactement une valeur

→ $O(p(n))$ clauses

4) $C(i)$ contient exactement une valeur

→ $O(p(n))$ clauses

5) La première configuration est initiale

a) $R(1,1) = B, R(1,2) = \sigma_1, R(1,3) = \sigma_2 \dots, R(1, n + 1) = \sigma_n$

b) $P(1) = 1$

c) $Q(1) = s$

d) $C(1) = \text{indifférent}$

$$r_{1,1,B} \wedge r_{1,2,\sigma_1} \wedge \dots \wedge r_{1,(n+1),\sigma_n} \wedge p_{1,1} \wedge q_{1,s}$$

→ $O(n)$ clauses

Théorème de Cook

Preuve

6) Chaque configuration est obtenue à partir de la précédente

a) Les cases ne se trouvant pas sous la tête ne sont pas modifiées

$$(r_{i,j,\alpha} \wedge \overline{p_{i,j}}) \Rightarrow r_{(i+1),j,\alpha}$$

$\rightarrow O(p(n)^2)$ clauses

b) La case sous la tête est modifiée et déplacement de la tête

pour tout $i, j \in [1, p(n) + 1]$:

$$(q_{i,k} \wedge p_{i,j} \wedge r_{i,j,\alpha} \wedge c_{i,k}) \Rightarrow (q_{(i+1),k'} \wedge r_{(i+1),j,\beta} \wedge p_{(i+1),(j-1)}) \text{ si } (k, \alpha, k', \beta, \leftarrow) \in \Delta$$

$$(q_{i,k} \wedge p_{i,j} \wedge r_{i,j,\alpha} \wedge c_{i,k}) \Rightarrow (q_{(i+1),k'} \wedge r_{(i+1),j,\beta} \wedge p_{(i+1),(j+1)}) \text{ si } (k, \alpha, k', \beta, \rightarrow) \in \Delta$$

$\rightarrow O(p(n)^2)$ clauses

7) A la fin de l'exécution on arrive dans un état acceptant

$$q_{1,Y} \vee q_{2,Y} \vee \dots \vee q_{(p(n)+1),Y}$$

$\rightarrow O(p(n))$ clauses

Théorème de Cook

Preuve

- $\tau : M, w \rightarrow F$

F : conjonction des clauses décrites dans les points (1) à (7)

- Il reste à montrer M accepte w ssi F est satisfiable

– Si M accepte w

- Il existe un calcul de M qui se termine avec Y
- Il existe une suite de configurations telle que $I(F) = \top$ pour l'interprétation (**)

$\Rightarrow F$ est satisfiable

– Si F est satisfiable

- Il existe une suite de configurations partant du ruban initial avec w et telle qu'à la fin on ait $q_{i,Y}$

$\Rightarrow M$ accepte w

\Rightarrow Donc M accepte w ssi F est satisfiable

(Fin de la preuve du théorème de Cook)

Variations autour de SAT

- SAT est NP-complet (on l'a prouvé)
 - On dispose à présent d'un problème NP-complet
 - On va pouvoir s'en servir pour démontrer que d'autres problèmes sont NP-complets par réduction polynomiale.
- 3-SAT
 - Soit F une FNC comportant exactement 3 littéraux par clause
 - F est-elle satisfiable ?
- MAX-SAT
 - Soit un ensemble de clauses et un entier K
 - Existe-t-il une interprétation satisfaisant au moins K clauses ?
- *Théorèmes*

3-SAT est NP-complet

(réduction depuis SAT)

MAX-SAT est NP-complet

(réduction depuis SAT)

Variations autour de SAT

- 2-SAT
 - Soit F une FNC comportant **exactement 2 littéraux** par clause
 - F est-elle satisfiable ?
- MAX-2-SAT
 - Soit un ensemble de clauses comportant **exactement 2 littéraux** et un entier K
 - Existe-t-il une interprétation satisfaisant **au moins K clauses** ?
- Théorèmes

$2\text{-SAT} \in P$

$MAX\text{-}2\text{-SAT}$ est NP-complet (étonnamment)

21 problèmes NP-complets de Karp

Richard M. Karp, *Reducibility Among Combinatorial Problems*, 1972

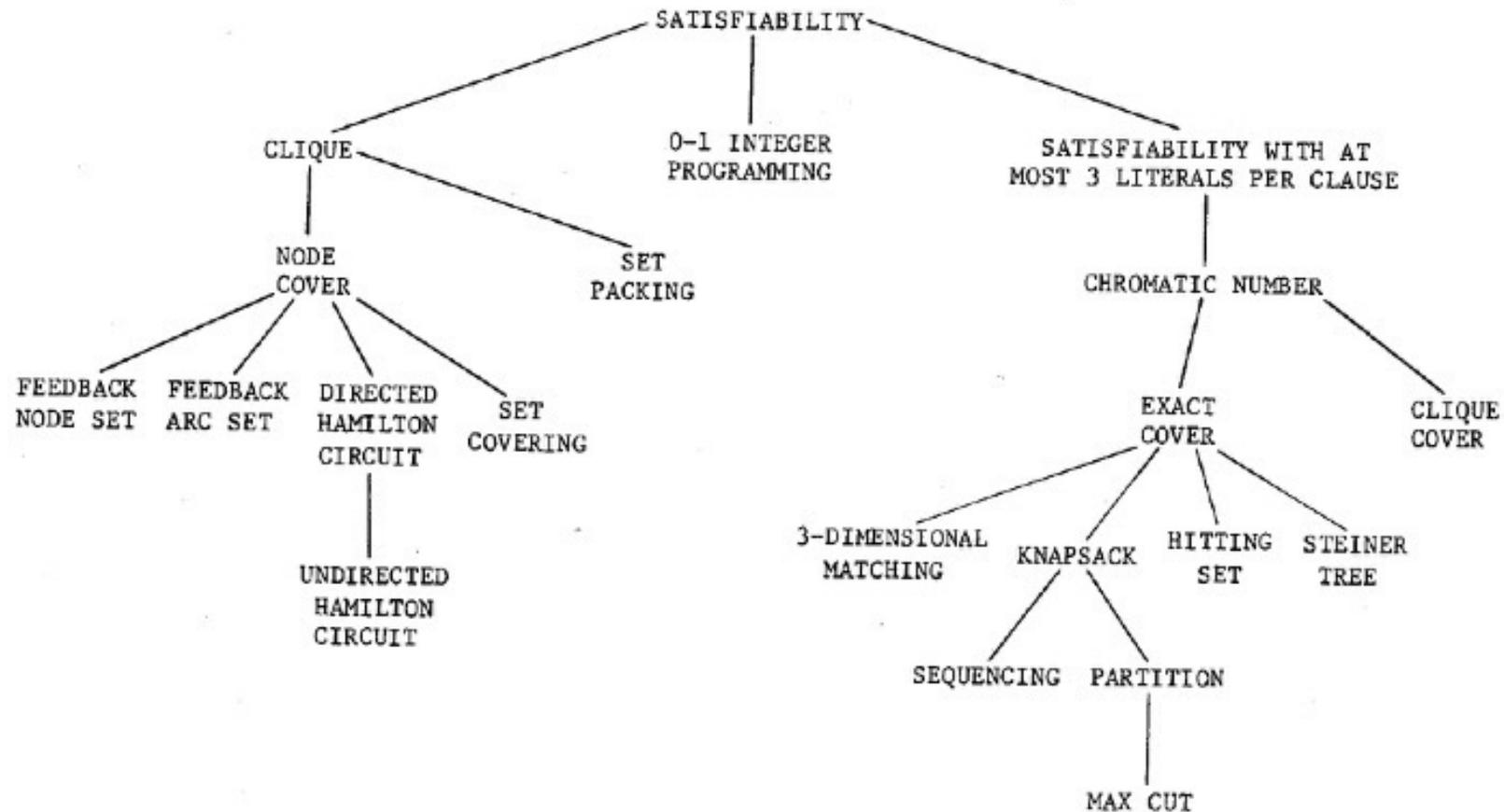


FIGURE 1 - Complete Problems

Considérations finales sur la NP-complétude

- On a un problème, on cherche un algorithme pour le résoudre, mais ce problème est établi NP-complet
 - Comme $P \neq NP$ (hypothèse), ce problème n'a pas de solution polynomiale
 - Faut-il pour autant renoncer à résoudre ce problème ? pas forcément
- La mesure de la complexité est pour le pire des cas :
 - Pas d'algo polynomial → pas d'algo pour **toutes** les instances du problème
 - Mais il peut très bien exister un algo polynomial pour certains cas, **voire presque tous**
- Il n'est obligé d'explorer tous les cas (nombre exponentiel de cas)
 - On peut utiliser des **heuristiques** pour limiter le nombre de cas à explorer
 - Critères approximatifs pour découvrir rapidement la solution recherchée (efficace des fois)
- Plutôt que de chercher **la** solution optimale, on peut chercher **une** solution s'en approchant
- On peut résoudre un ou plusieurs cas particuliers d'un problème NP-complet, en utilisant des algorithmes polynomiaux