Problèmes NP-complets : exemples et réduction.

928

I. Définitions

1. Classes P et NP

Déf 1: La classe P est l'ensemble des problèmes (vus comme des langages) décidés par une machine de Turing déterministe ou un algorithme en temps polynomial en la taille de l'entrée.

Déf 2: La classe NP est l'ensemble des problèmes décidés par une machine de Turing ou un algorithme non déterministe en temps polynomial en la taille de l'entrée.

Ex 3: Le problème CONNEX | entrée: G un graphe | sortie: oui si G est connexe, non sinon — est dans P.

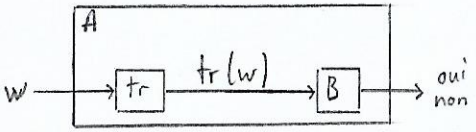
Déf 4: Un vérifieur pour un problème A est une machine de Turing M déterministe telle que w est une instance positive de A si et seulement s'il existe c tel que M accepte (w, c). Un tel c est appelé certificat.

Prop 5: $A \in NP \Leftrightarrow$ il existe un vérifieur en temps polynomial de A.

Ex 6: SAT | entrée: φ une formule du calcul propositionnel | sortie: oui si φ est satisfiable, non sinon

2. Principe de réduction

Déf 7: Une réduction polynomiale de A vers B est une fonction tr telle que:
- tr est calculable en temps polynomial
- w est une instance positive de A si et seulement si $tr(w)$ est une instance positive de B.



Prop 8: Si A se réduit à B, alors:
- si $B \in P$ alors $A \in P$
- si $A \notin P$ alors $B \notin P$
- si $B \in NP$ alors $A \in NP$
- si $A \notin NP$ alors $B \notin NP$

3. NP-Complétude

Déf 9: Un problème A est dit NP-dur si pour tout problème $B \in NP$, il existe une réduction de B à A.
Si de plus, $A \in NP$, alors A est dit NP-complet.

Rq 10: On a $P \subset NP$ et si $NP\text{-}C \cap P \neq \emptyset$, alors $P = NP$.

Thm 11 (de Cook): $SAT \in NP\text{-}C$.

II. Quelques problèmes NP-Complets

1. En logique

3SAT | entrée: φ une formule en forme normale conjonctive avec au plus 3 littéraux par clause. | sortie: oui si φ est satisfiable, non sinon

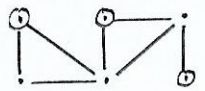
Prop 12: 3SAT est NP-Complet (réduction de SAT).

2. En théorie des graphes

INDEPENDENT SET | entrée: $G = (V, E)$ un graphe, k un entier | sortie: oui s'il existe $V' \subset V$ tel que pour tout $u, v \in V'$, $u \neq v \Rightarrow (u,v) \notin E$, et $|V'| \geq k$.

Prop 13: INDEPENDENT SET est NP-Complet (réduction de 3SAT).

Ex 14:

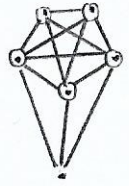


Les sommets entourés forment un ensemble de sommets indépendants maximal.

CLIQUE | entrée: $G = (V, E)$ un graphe, k un entier | sortie: oui s'il existe $V' \subset V$ une clique et telle que $|V'| \geq k$.

Prop 15: CLIQUE est NP-Complet (réduction de INDEPENDENT SET)

Ex 16:

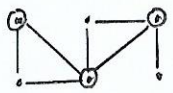


Les sommets entourés forment une 5-clique.

VERTEX COVER | entrée : $G=(V,E)$ un graphe, k un entier
sortie : oui s'il existe $V' \subset V$ tel que toute arête de G a une extrémité dans V', et $|V'| \leq k$.

Prop 17 : VERTEX COVER est NP-Complet (réduction de INDEPENDENT SET).

Ex 18 :



Les sommets entourés recouvrent le graphe G.

HAMPATH | entrée : G un graphe
sortie : oui s'il existe un chemin hamiltonien, non sinon

Prop 19 : HAMPATH est NP-Complet (réduction de 3SAT)

Ex 20 :

Un chemin hamiltonien

TSP | entrée : G un graphe pondéré, k un entier
sortie : oui s'il existe un chemin hamiltonien de poids inférieur à k

Prop 21 : TSP est NP-Complet (réduction de HAMPATH).

3COL | entrée : G un graphe
sortie : oui s'il existe une 3-coloration de G, non sinon

Thm 22 : 3COL est NP-Complet (réduction de 3SAT) DEV 1

---

3. En théorie des paquets

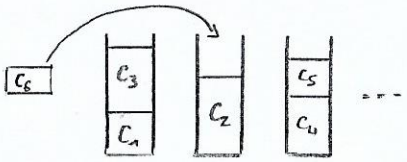
SUBSET SUM | entrée : A une partie de $\mathbb{N}$, k un entier
sortie : oui s'il existe une partie de A dont la somme des éléments vaut k, non sinon

Prop 23 : SUBSET SUM est NP-Complet (réduction de 3SAT)

BIN PACKING | entrée : $c_1, \dots, c_n \in \mathbb{N}$, k un entier, s un entier
sortie : oui s'il existe $f : \{1,\dots,n\} \to \{1,\dots,k\}$ telle que
$\forall j \in \{1,\dots,k\} \quad \sum_{i \in f^{-1}(\{j\})} c_i \leq s$

Prop 24 : BIN PACKING est NP-Complet (réduction de SUBSET SUM).

Ex 25 :



KNAPSACK | entrée : $v_1, \dots, v_n \in \mathbb{N}^n$, $w_1, \dots, w_n \in \mathbb{N}^n$, V et W des entiers
sortie : oui s'il existe $P \subset [\![1, n]\!]$ telle que
$\sum_{p \in P} v_p \geq V$ et $\sum_{p \in P} w_p \leq W$.

Prop 26 : KNAPSACK est NP-Complet (réduction de SUBSET SUM)

4. En théorie des langages et des automates

LANGUAGE SEPARATION | entrée : S, T deux langages finis, k un entier
sortie : oui s'il existe un automate complet déterministe à k états qui accepte les mots de S et rejette les mots de T.

Prop 27 : LANGUAGE SEPARATION est NP-Complet (réduction de SAT).

REG EXP NON UNIVERSALITY | entrée : R une expression rationnelle sur Σ.
sortie : oui si $\mathcal{L}(R) \neq \Sigma^*$, non sinon

Prop 28 : REG EXP NON UNIVERSALITY est NP-Complet (car PSPACE-Complet)

AUTOMATON INEQUIVALENCE | entrée : A_1, A_2 deux automates non-déterministes sur Σ
sortie : oui si $\mathcal{L}(A_1) \neq \mathcal{L}(A_2)$, non sinon

Prop 29 : AUTOMATON INEQUIVALENCE est NP-complet (réduction de REG EXP NON UNIVERSALITY).

III. Que faire en cas de NP-difficulté ?

1. Rester calme et restreindre le problème

Un problème NP-dur n'admet pas que des instances difficiles à résoudre.
Parfois, en restreignant les instances, le problème devient polynomial.

2SAT | entrée : φ une formule en forme normale conjonctive avec au plus 2 littéraux par clause.
sortie : oui si φ est satisfiable, non sinon

Prop 30 : 2SAT est dans P.

Déf 31 : Une clause est appelée clause de Horn si elle contient au plus un littéral positif.
Une formule de Horn est une conjonction de clauses de Horn.

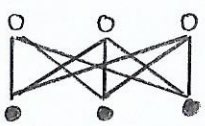
HORN-SAT | entrée : φ une formule de HORN
sortie : oui si φ est satisfiable, non sinon

Prop 32 : HORN-SAT est dans P.

2COL | entrée : G un graphe
sortie : oui s'il existe une 2-coloration de G, non sinon

Prop 33 : 2COL est dans P.

Ex 34 :



Un graphe est 2-colorable si et seulement s'il est bipartite.

2. Se contenter d'une solution approchée

Déf 35 : Soit P un problème d'optimisation. Un algorithme d'approximation de facteur ρ pour P est un algorithme tel que pour toute entrée x, l'algorithme renvoie une solution $f(x)$ telle que :
- $\frac{\text{Coût}(f(x))}{OPT(x)} \leq \rho$ si P est un problème de minimisation
- $\frac{OPT(x)}{\text{Coût}(f(x))} \leq \rho$ si P est un problème de maximisation

où $OPT(x)$ est le coût d'une solution optimale.

Prop 36 : L'algorithme suivant est une 2-approximation pour le problème VERTEX COVER, de complexité polynomiale.

```
Approx_VC(G) :
| // entrée : G = (V,E) un graphe
| // sortie : V' ⊂ V une couverture des arêtes de G.
| V' = ∅
| E' = E
| tant que E' ≠ ∅ :
|   | prendre (u,v) ∈ E'
|   | V' = V' ∪ {u,v}
|   | supprimer de E' toute arête incidente à u ou v.
| retourner V'
```

TSPE | entrée : G un graphe, w une pondération vérifiant l'inégalité triangulaire
sortie : Un chemin hamiltonien de poids minimal

Thm 37 : Il existe une 2-approximation en temps polynomial pour le problème TSPE. | DEV 2 |

Déf 38 : Un schéma d'approximation pour un problème d'optimisation P est un algorithme qui prend en entrée une instance de P et $\varepsilon > 0$ et qui retourne une $(1+\varepsilon)$-approximation de la solution en temps polynomial.

Prop 39 : Il existe un schéma d'approximation polynomial pour le problème KNAPSACK.

Références :

- Cormen, Algorithmique, 3e édition
- Garey-Johnson, Computers and intractability
- Kleinberg-Tardos, Algorithm design
- Papadimitriou, Computational complexity
- Rey, Calculabilité, complexité et approximation