TD 1 : Problèmes de décision et machines de Turing

Théorie de la complexité S5.

2022

Exercice 1 — *Décrire un problème de décision*

Décrivez les problèmes suivants formellement sous forme d'un problème de décision. Présicez l'ensemble $\mathcal{L}$ des instances et la question à laquelle il faut répondre.

1. (BIP ?) : Savoir si un graphe est biparti.

▶ **Correction**

Version en français : Soit G un graphe non orienté, G est-il un graphe biparti ?
$\mathcal{L} = \{\text{Graphes } G \text{ non orientés }\}$, $\mathcal{L}^Y = \{\text{Graphes } G \text{ non orientés bipartis}\}$

2. (SAT ?) : Connaissant une formule booléenne déterminer comment la rendre vraie.

▶ **Correction**

Version en français : Soit une formule φ avec n variables $x_1, x_2, \ldots, x_n$ existe-t-il une affectation des variables x_i qui rende φ vrai ? (autrement dit, $\exists x_1, x_2, \ldots, x_n | \varphi(x_1, x_2, \ldots, x_n)$, ou φ est satisfiable).
$\mathcal{L} = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ $\}$, $\mathcal{L}^Y = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ de sorte qu'on puisse rendre la fonction φ vraie pour au moins une affectation des variables $\}$

3. (3SAT ?) : SAT restreint aux formes normales conjonctives où les clauses ont 3 variables.

▶ **Correction**

Version en français : Soit une formule φ sous forme CNF avec 3 variables par clause, avec n variables $x_1, x_2, \ldots, x_n$, existe-t-il une affectation des variables x_i qui rende φ vrai ?
$\mathcal{L} = \{$ Formule booléenne φ sous forme CNF avec 3 variables par clause, avec n variables $x_1, x_2, \ldots, x_n$ $\}$, $\mathcal{L}^Y = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ de sorte qu'on puisse rendre la fonction φ vraie pour au moins une affectation des variables $\}$

4. (ADD ?) : Addition de deux entiers.

▶ **Correction**

Version en français : Soient trois entiers x, y et z, est-ce que $x + y = z$?
$\mathcal{L} = \{x, y, z \in \mathbb{N}\}$, $\mathcal{L}^Y = \{x, y, z \in \mathbb{N} \text{ tels que } x + y = z\}$

5. (TAU ?) : Connaissant une formule booléenne déterminer si elle est toujours vraie.

▶ **Correction**

Version en français : Soit une formule φ avec n variables $x_1, x_2, \ldots, x_n$ est-ce que, pour toute affectation des variables x_i, φ est vraie ? (autrement dit, $\forall x_1, x_2, \ldots, x_n | \varphi(x_1, x_2, \ldots, x_n)$ ou φ est une tautologie).
$\mathcal{L} = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ $\}$, $\mathcal{L}^Y = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ de sorte que φ soit toujours vraie quelle que soit l'affectation des variables $\}$

6. (3-TAU ?) : TAU restreint aux formes normales disjonctives où les clauses ont 3 variables.

▶ **Correction**

Version en français : Soit une formule φ sous forme DNF avec 3 varialbes par clause, avec n variables $x_1, x_2, \ldots, x_n$ est-ce que, pour toute affectation des variables x_i, φ est vraie ? (autrement dit, φ est une tautologie).

$\mathcal{L} = \{$ Formule booléenne φ sous forme DNF avec 3 variables par clause, avec n variables $x_1, x_2, \ldots, x_n$ $\}$, $\mathcal{L}^Y = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ de sorte que φ soit toujours vraie quelle que soit l'affectation des variables $\}$

7. (QBF ?) : Determiner si une formule booléenne quantifiée est vraie.

▶ **Correction**

Version en français : Soit une formule φ avec n variables $x_1, x_2, \ldots, x_n$ est-ce que $\exists x_1 \forall x_2 \exists \ldots \forall x_n | \varphi(x_1, x_2, \ldots, x_n)$?

$\mathcal{L} = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ $\}$, $\mathcal{L}^Y = \{$ Formule booléenne φ avec n variables $x_1, x_2, \ldots, x_n$ de sorte que $\exists x_1 \forall x_2 \exists \ldots \forall x_n | \varphi(x_1, x_2, \ldots, x_n)$ $\}$

8. (CONNECTIVITY ?) : Connexité d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté, G est-il un graphe connexe ?

$\mathcal{L} = \{$Graphes G non orientés $\}$, $\mathcal{L}^Y = \{$Graphes G non orientés connexes$\}$

9. (PLANARITY ?) : Planarité d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté, G est-il un graphe planaire ?

$\mathcal{L} = \{$Graphes G non orientés $\}$, $\mathcal{L}^Y = \{$Graphes G non orientés planaires$\}$

10. (k-COL ?) : k-colorabilité d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté, G est-il un graphe k-coloriable ? Autrement dit, peut-on attribuer à chaque nœud une couleur parmi k de sortes que deux voisins n'ont pas la même couleur ?

$\mathcal{L} = \{$Graphes G non orientés $\}$, $\mathcal{L}^Y = \{$Graphes G non orientés k-coloriable $\}$

11. (CHROMA) : Nombre chromatique d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté et un entier k, G est-il k-coloriable ?

$\mathcal{L} = \{$Graphe G non orienté et un entier k $\}$, $\mathcal{L}^Y = \{$ Graphe G non orienté et un entier k de sorte que G est k-coloriable$\}$

Notez la différence avec le problème précédent où k n'est pas donné en entrée. Clairement on résout k-COL et CHROMA de la même manière. Mais, dans le premier cas, k est considéré comme une constante. On a le problème (3-COL) (4-COL) (27-COL). Si on a un algo en n^k il serait polynomial pour (4-COL) car on aurait alors un algo en n^4. Mais il ne serait pas polynomial pour (CHROMA) où k fait partie de l'entrée et où donc n^k serait exponentiel en k.

12. (SIZE) : Nombre de nœuds d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté et un entier k, G est-il de taille k ?

$\mathcal{L} = \{$Graphe G non orienté et un entier k $\}$, $\mathcal{L}^Y = \{$ Graphe G non orienté et un entier k de sorte que G est de taille $k\}$

13. (INS) : Nombre de stabilité d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté et un entier k, G possède-t-il un stable de taille k ? (k nœuds non reliés deux à deux).
$\mathcal{L} = \{$Graphe G non orienté et un entier k $\}$, $\mathcal{L}^Y = \{$ Graphe G non orienté et un entier k de sorte que G possède un stable de taille $k\}$

14. (WINS) : Trouver un stable de poids maximum dans un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté, des poids $\omega : V \to \mathbb{N}$ et un entier k, G possède-t-il un stable dont la somme des poids des nœuds dépasse k ?
$\mathcal{L} = \{$Graphe G non orienté, des poids ω sur les nœuds de G et un entier k $\}$, $\mathcal{L}^Y = \{$ Graphe G non orienté, des poids ω sur les nœuds de G et un entier k de sorte que G possède un stable de poids supérieur à k $\}$

15. (MATCHING) : trouver un couplage maximum dans un graph.

▶ **Correction**

Version en français : Soit G un graphe non orienté et un entier k, G possède-t-il un couplage de taille supérieure à k ? (c'est à dire des arêtes sans extrémité commune)
$\mathcal{L} = \{$Graphe G non orienté et un entier k $\}$, $\mathcal{L}^Y = \{$ Graphe G non orienté et un entier k de sorte que G possède un couplage de taille supérieure à k $\}$

16. (VERTEX COVER) : trouver le nombre minimum de nœuds d'un graphe nécessaires pour couvrir toutes les arêtes.

▶ **Correction**

Version en français : Soit G un graphe non orienté et un entier k, G possède-t-il un un ensemble V' d'au plus k nœuds couvrant toutes les arêtes ? (c'est à dire que chaque arête a une extrémité dans V')
$\mathcal{L} = \{$Graphe G non orienté et un entier k $\}$, $\mathcal{L}^Y = \{$ Graphe G non orienté et un entier k de sorte qu'il existe un ensemble V' de nœuds de taille inférieur à k couvrant toutes les arêtes $\}$

17. (HAM ?) : existence d'un chemin hamiltonien.

▶ **Correction**

Version en français : Soit G un graphe non orienté, G est-il un graphe hamiltonien ? (possédant un cycle passant une et une fois par chaque nœud ; variante : une chaîne passant une et une fois par chaque nœud ; variante : G est orienté et un circuit/chemin passant une et une fois par chaque nœud)
$\mathcal{L} = \{$Graphes G non orientés $\}$, $\mathcal{L}^Y = \{$Graphes G non orientés hamiltoniens $\}$

18. (COHAM ?) : déterminer si un graphe n'est pas hamiltonien.

▶ **Correction**

Version en français : Soit G un graphe non orienté, est-ce que G n'est pas hamiltonien ?
$\mathcal{L} = \{$Graphes G non orientés $\}$, $\mathcal{L}^Y = \{$Graphes G non orientés non hamiltoniens $\}$
(Ce problème inverse les réponses de (HAM))

19. (SHP) : Trouver un plus court chemin élémentaire entre deux nœuds, poids positifs.

▶ **Correction**

Version en français : Soit G un graphe orienté, des poids $\omega : E \to \mathbb{N}$, un nœud s et un nœud t et un entier k, G possède-t-il un chemin élémentaire (ne passant qu'une fois au plus par chaque nœud) de s à t dont la somme des poids des arcs est inférieure à k ?

20. (LOP) : Trouver un plus long chemin élémentaire entre deux nœuds, poids positifs.

▶ **Correction**

Version en français : Soit G un graphe orienté, des poids $\omega : E \to \mathbb{N}$, un nœud s et un nœud t et un entier k, G possède-t-il un chemin élémentaire (ne passant qu'une fois au plus par chaque nœud) de s à t dont la somme des poids des arcs dépasse k ?

21. (TSP) : Trouver un cycle de poids minimum passant par chaque nœud d'un graphe non orienté complet pondéré.

▶ **Correction**

Version en français : Soit G un graphe non orienté complet, des poids $\omega : E \to \mathbb{N}$, et un entier k, G possède-t-il un cycle hamiltonien dont la somme des poids des arêtes dépasse k ?

22. (MSPT) : Trouver un arbre couvrant de poids minimum d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté, des poids $\omega : E \to \mathbb{N}$ et un entier k, G possède-t-il un arbre couvrant (un sous-graphe de G sans cycle passant par chaque nœud de G) dont la somme des poids des arêtes est inférieure à k ?

23. (SET COVER) : Trouver une couverture minimum d'un ensemble X à l'aide des ensemble d'une partie de $\mathcal{P}(X)$.

▶ **Correction**

Version en français : Soit un ensemble X, un ensemble S de parties de X et un entier k, existe-t-il un sous-ensemble C de S couvrant X (chaque élément de X est dans au moins un ensemble de C) et de taille inférieure à k ?

24. (UST) : Trouver un arbre de poids minimum couvrant une partie des nœuds d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe non orienté, des poids $\omega : E \to \mathbb{N}$, un ensemble X de nœuds de G et un entier k, G possède-t-il un arbre couvrant X (un sous-graphe de G sans cycle passant par chaque nœud de X et possiblement d'autres nœuds) dont la somme des poids des arêtes est inférieure à k ?

25. (DST) : Trouver une arborescence de poids minimum couvrant une partie des nœuds d'un graphe orienté et enracinée en un nœud particulier du graphe.

▶ **Correction**

Version en français : Soit G un graphe orienté, un nœud r, des poids $\omega : E \to \mathbb{N}$, un ensemble X de nœuds de G et un entier k, G possède-t-il une arborescence enracinée en r couvrant X (un sous-graphe de G sans cycle possédant un chemin de r vers chaque nœud de X et possiblement d'autres nœuds) dont la somme des poids des arcs est inférieure à k ?

26. (GRUNDY ?) : Existence fonction de Grundy d'un graphe.

▶ **Correction**

Version en français : Soit G un graphe orienté, G a-t-il une fonction de Grundy ? (c'est-à-dire que chaque nœud v a un numéro g_v de sorte que g_v est le plus petit entier positif ou nul qui n'est pas déjà attribué aux successeurs de v).

27. (SUBSET SUM ?) : Déterminer s'il est possible d'atteindre une certaine valeur en additionnant une partie des entiers d'un ensemble.

▶ **Correction**

Version en français : Soit $x_1, x_2, \ldots, x_n$ et B des entiers, est-ce qu'il existe un sous-ensemble des entiers de $x_1, x_2, \ldots, x_n$ dont la somme fait B ?

28. (PARTITION ?) : Déterminer s'il est possible de partitionner un ensemble d'entiers en 2 sous-ensembles de même somme.

▶ **Correction**

Version en français : Soit $x_1, x_2, \ldots, x_n$ des entiers, est-ce qu'il existe une partition X_1, X_2 de ces entiers telle que $\sum_{x \in X_1} x = \sum_{x \in X_2} x$?

29. (KNAPSACK) : Problème du sac à dos.

▶ **Correction**

Version en français : Soit n objets de volume $v_1, v_2, \ldots, v_n$ et de poids $p_1, p_2, \ldots, p_n$, soit deux entiers V et P, existe-t-il un sous-ensemble $I \subset [\![1; n]\!]$ de sorte que $\sum_{i \in I} v_i \leq V$ et $\sum_{i \in I} p_i \geq P$? autrement dit un sous-ensemble des objets dont le volume ne dépasse pas V mais dont le poids dépasse P ?

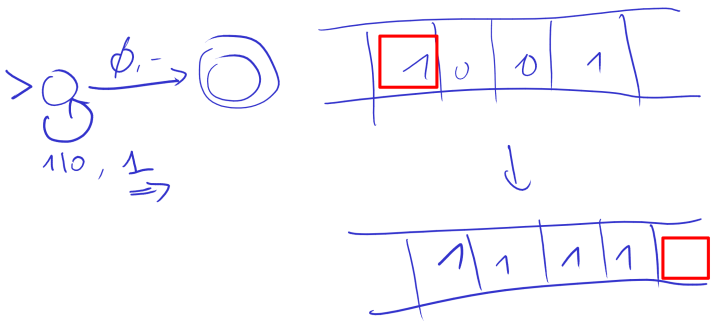
30. (LP) : Résoudre un programme linéaire en nombre réels.
31. (ILP) : Résoudre un programme linéaire en nombre entiers.
32. (FLOW) : Trouver un flot maximum dans un réseau.
33. (MREG ?) : Déterminer si une chaîne de Markov est régulière.
34. (CHESS ?) : Déterminer si une position sur un plateau d'échec est gagnante pour le joueur blanc.
35. (REVERSI ?) : Déterminer si une position sur un plateau de reverso est gagnante pour le joueur blanc.
36. (SUDOKU ?) : Résoudre un sudoku.
37. (U-SUDOKU ?) : Unicité de la solution d'un sudoku.
38. (MIN SUDOKU) : Déterminer combien d'indice minimum sont nécessaire à une configuration de sudoku.

Exercice 2 — *Machines de Turing*

Dans cet exercice, on pourra supposer vraies toutes les équivalences de l'exerice 3.

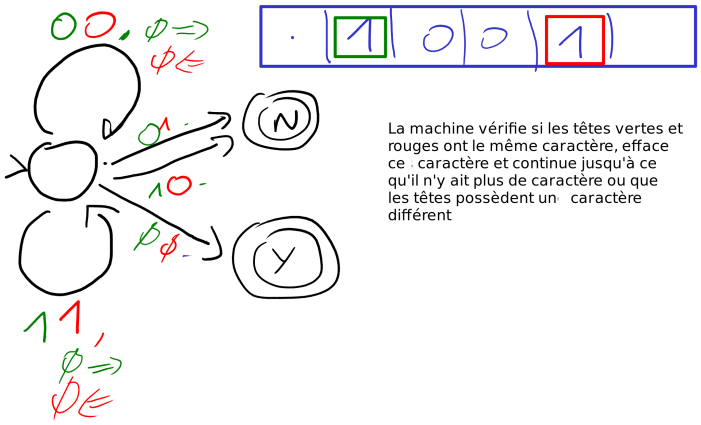
1. Écrire une machine de Turing déterministe qui remplace l'entier en entrée par des 1

▶ **Correction**



2. Écrire une machine de Turing déterministe qui vérifie si un mot est un palindrome

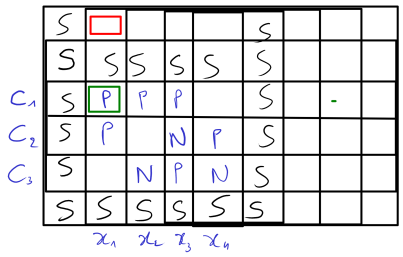
▶ **Correction**



La machine vérifie si les têtes vertes et rouges ont le même caractère, efface ce caractère et continue jusqu'à ce qu'il n'y ait plus de caractère ou que les têtes possèdent un caractère différent

3. Écrire une machine de Turing déterministe qui résoud le problème (ADD ?)
4. Écrire une machine de Turing non déterministe qui résoud le problème (SAT ?)

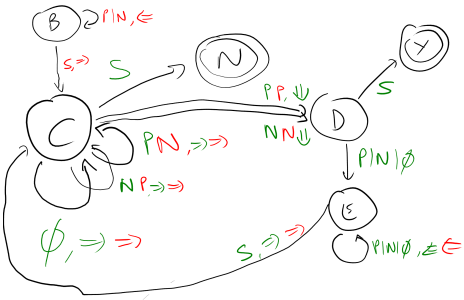
► **Correction**



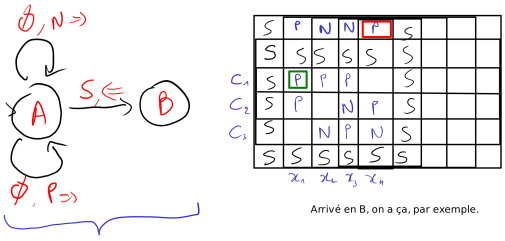
On considère ici un encodage des formules phi de la forme
phi = C1 et C2 et ... et Cm
avec Cm de la forme (a ou b ou c)
avec a, b, c de la forme x_i ; ou (non x_i)

Par exemple phi = (x1 ou x2 ou x3) et (x1 ou non x3 et x4) et (non x2 ou x3 ou non x4)

Chaque ligne correspond à une clause.
Chaque variable correspond à une colonne
(On a indiqué ici en bleu lesquels pour l'exemple)
Un P dans la ligne correspondant à C_j dans la colonne correspondant à x_i signifie que x_i apparait positivement dans C_j.
Un N signifique que (non x_i) apparait dans C_j
Un blanc signifie que ni x_i ni (non x_i) apparait dans C_j



Ici on a le vérifieur, qui vérifie que l'affectation des variables rend la formule vraie.
C parcours une clause. Si elle tombe sur une variable que le certificat a rendu VRAI et qui apparait positivement dans la clause, alors la clause est satisfaite (car c'est un OU entre des variables) et on passe à la clause suivante avec D et E.
De même si elle voit une variable que le certificat a rendu FAUX et qui apparait négativement.
D vérifie qu'on a pas atteint le bas des clauses, c'est à dire qu'on a satisfait la dernière clause.
On arrive en N si on a pas réussi à satisfaire une clause.

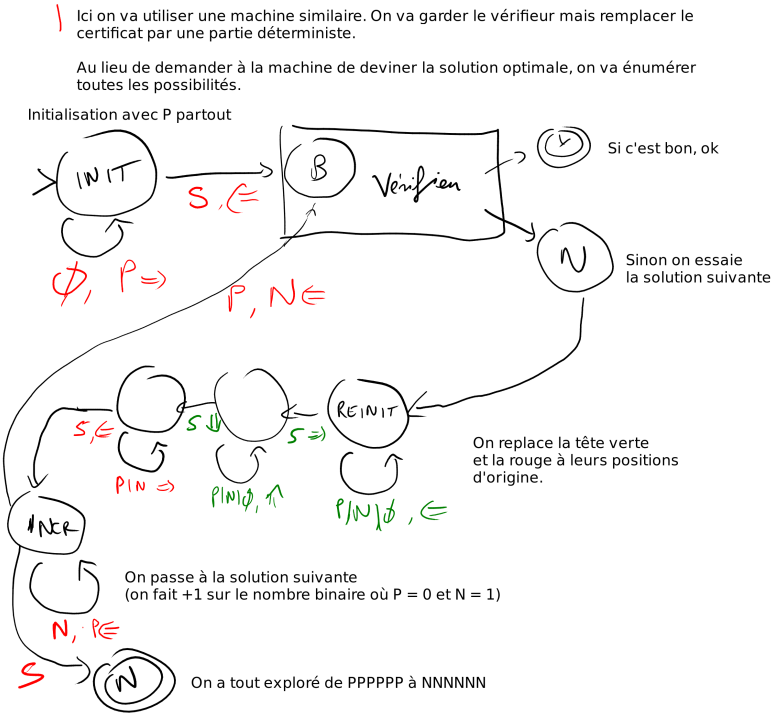


Arrivé en B, on a ça, par exemple.

Certificat, non déterministe
Indique pour chaque variable si elle est positive ou négative

5. Écrire une machine de Turing déterministe qui résoud le problème (SAT ?)

► **Correction**



6. Écrire une machine de Turing déterministe qui résoud le problème (3-COL ?)
7. Écrire une machine de Turing non déterministe qui résoud le problème (W-INS)

Exercice 3 — *Équivalences entre machines de Turing*

1. Soit une machine de Turing $\mathcal{M}$ avec un alphabet Σ, simulez $\mathcal{M}$ avec une machine $\mathcal{M}'$ classique.
2. Soit une machine de Turing $\mathcal{M}$, simulez $\mathcal{M}$ avec une machine $\mathcal{M}'$ qui ne possède que des 1 et des 0.
3. Comment encoder la réponse OUI ou NON d'un problème de décision par une machine de Turing sans état d'acceptation ou de refus ?
4. Soit une machine de Turing $\mathcal{M}$, simulez $\mathcal{M}$ avec une machine $\mathcal{M}'$ ayant une demi-bande (infinie vers la droite, finie vers la gauche).
5. Soit une machine de Turing $\mathcal{M}$ avec 2 bandes, simulez $\mathcal{M}$ avec une machine $\mathcal{M}'$ classique.
6. Soit une machine de Turing $\mathcal{M}$ avec une bande en deux dimensions, simulez $\mathcal{M}$ avec une machine $\mathcal{M}'$ classique.
7. Soit une machine de Turing $\mathcal{M}$ non déterministe, simulez $\mathcal{M}$ avec une machine $\mathcal{M}'$ déterministe.