Décibabilité : Eléments de base

Houcine Senoussi

November 7, 2018

Objectif

Question centrale :

Etant donné un problème P,

1. peut-on le résoudre sur ordinateur ?
2. et si oui, avec quel coût ?

- Résolution sur ordinateur : existence d'un algorithme.
  - Décidabilité (problème décidable : résoluble sur ordinateur).
- Coût de la résolution : en temps et en espace.
  - Complexité (temporelle et spatiale).

Objectif-2

- Objectif du cours :
  - Introduire les concepts, les méthodes et les outils permettant pour un problème donné P de répondre à la (double) question centrale donnée ci-dessus.

Rappels et Définitions :

- Un algorithme est une **suite** d'instructions servant à résoudre un problème.
- La complexité temporelle (resp. spatiale) caractérise le coût en temps (resp. en espace) d'un algorithme.
- La complexité (temporelle) d'un algorithme est le nombre d'**opérations élémentaires** qui le composent.
- Ce nombre d'opérations est calculé en fonction de la **taille du problème** notée n.
  - Taille d'un problème : taille d'une **instance** du problème.
  - Exemple : dans le cas du tri la taille du problème = nombre d'éléments du tableau à trier.

Rappels et Définitions-2 :

- On donne la complexité sous la forme $\mathcal{O}(f(n))$
  - Signification de cette notation : La nombre d'opérations a comme terme de plus haut degré $f(n)$.
  - Exemples : $\mathcal{O}(n^2)$, $\mathcal{O}(n.logn)$, $\mathcal{O}(2^n)$.

Complexité des algorithmes

- $f(n) = \mathcal{O}(g(n))$ si :
  - $\exists n_0 \in \mathbb{N}$ tels que pour $n \geq n_0$ $f(n) \leq c.g(n)$.
  - Autrement dit, f augmente au plus aussi vite que g.
- $f(n) = \Omega(g(n))$ si $g(n) = \mathcal{O}(f(n))$
  - f augmente au moins aussi vite que g.
- $f(n) = \Theta(g(n))$ si $f(n) = \mathcal{O}(g(n))$ et $f(n) = \Omega(g(n))$
  - f et g augmentent aussi vite l'une que l'autre.

Complexité des algorithmes-Exemple 1

- Donné : un tableau d'entiers tab de taille N.
- Résultat : le même tableau trié.

```
Pour taille=N, ..., 2 Faire
    Pour i=1, ..., taille-1 Faire
        Si (tab[i]>tab[i+1]) Alors
            tmp = tab[i]
            tab[i] = tab[i+1]
            tab[i+1] = tmp
```

Complexité des algorithmes-Exemple 2

- Donné : un tableau d'entiers tab de taille N et un entier e
- Résultat : un booléen trouvé qui vaut VRAI si l'élément appartient au tableau et FAUX sinon

```
Inf = 1
Sup = N
trouvé = Faux
Tant que ((inf < sup) ET (trouvé = Faux))
    Milieu = (inf + sup)/2
    Si (e = tab[milieu]) Alors Trouvé = Vrai
    Sinon
        Si (e < tab[milieu]) Alors Sup = milieu+1
        Sinon inf = milieu - 1
```

Définitions :

Définition :

Un **graphe** G(S,A) est constitué de deux ensembles :

- l'ensemble S des **sommets** $S = \{s_1, ..., s_n\}$
- l'ensemble A des **arcs** $A = \{a_1, ..., a_m\}$. Un arc a_i est un couple de sommets (s_{i1}, s_{i2}).

- La définition ci-dessus est celle d'un graphe **orienté**.
- Dans un graphe non-orienté les **couples** de sommets (**arcs**) sont remplacés par des **paires** de sommets (**arêtes**).
- Un graphe (orienté ou non orienté) peut être valué : à chaque arc (arête) est attribué une valeur (une distance, un coût, ...).

Définitions-2 :

- Chemin et cycle.
- Chemin (cycle) hamiltonien.
- Chemin (cycle) eutérien.
- Sous-graphe et graphe partiel.
- Graphe complet et clique.

Définitions

- A la base de la **logique propositionnelle** il y a :
  - Des **propositions** : symbôles ayant une valeur de vérité (Vrai ou Faux).
  - Des **connecteurs logiques** : $\neg$, $\vee$,$\wedge$,$\Longrightarrow$, $\Longleftrightarrow$.

Définitions-2

Formule bien formée :

- Une proposition est une formule bien formée.
- Si A est une fbf, alors $\neg A$ est une fbf.
- Si A et B sont des fbf, alors $A \vee B$ est une fbf.
- Si A et B sont des fbf, alors $A \wedge B$ est une fbf.
- Si A et B sont des fbf, alors $A \Longrightarrow B$ est une fbf.
- Si A et B sont des fbf, alors $A \Longleftrightarrow B$ est une fbf.

Définitions-3

Interprétation :

Etant donnée une fbf A. Soit Δ_A l'ensemble des propositions intervenant dans A. On appelle une **interprétation** de A une fonction de Δ_A dans $\{VRAI, FAUX\}$. Autrement dit une interprétaion de A est une valuation des propositions intervenant dans A.

satisfiabilité :

Etant donnée une fbf A et une interprétation I de A. On dit que A est **satisfiable** par I si A a la valeur $VRAI$ lorque les propositions intervenant dans A prennent les valeurs que leur associent I.

Définitions-4

Modèle :

Etant donnée une fbf A et une interprétation I de A. I est un **modèle** de A si A est satisfiable par I.

Formule insatisfiable :

Etant donnée une fbf A. A est dite **insatisfiable** si elle n'a pas de modèle.

Définitions-5

Littéral :

Etant donnée une proposition p, un **littéral** relatif à cette proposition est soit p soit $\neg p$.

Clause :

Une **clause** C est une disjonction de littéraux $l_1 \vee l_2 \vee ... \vee l_n$.

Forme conjonctive normale :

Une **forme conjonctive normale (fcn)** est une conjonction de clauses $C_1 \wedge C_2 \wedge ... \wedge C_n$.

Logique propositionnelle-Exemples

1. Mettre sous forme conjonctive normale la formule :
   - $y \Longleftrightarrow (x_1 \vee x_2)$.

Définitions

Alphabet :

Un **alphabet** Σ est un ensemble fini de symbôles.

Mot :

Un **mot** ω défini sur un alphabet Σ est une suite de symboles s_1, s_1, ..., s_n appartenant à l'alphabet Σ.

Langage :

Un langage L défini sur un alphabet Σ est un ensemble (fini ou infini) de mots définis surΣ.

Machine de Turing-Définition

Machine de Turing :

Une machine de Turing est définie par un quadruplet (K, Σ, δ, q_0) où

- Σ est un **alphabet**,
- K est un ensemble fini d'**états**,
- $q_0 \in K$ est l'état **initial**,
- δ est une fonction, appelée la fonction de **transition**, de $\Sigma \times K$ vers $\Sigma \times K \times \{\rightarrow, \leftarrow, -\}$.

Machine de Turing-Fonctionnement

- Elle dispose d'une bande infinie sur laquelle il est possible de lire et d'écrire.
- Elle dispose d'une tête de lecture/écriture.
- Elle reçoit en entrée une chaîne de symbôles $s_o s_1 ... s_n$ où $s_i \in \Sigma$.
- Au départ elle est dans l'état q_o.
- Elle commence la lecture par le symbôle le plus à gauche de la chaîne.
- A chaque fois qu'un symbôle s est lu :
  - elle trouve la transition $(s, q) \rightarrow (s', q', dep)$,où q est l'état en cours.
  - elle écrit s'.
  - elle effectue le déplacement dep.
  - elle passe dans l'état q'.

Machine de Turing-Exemple

- Etude de l'exemple donné dans le polycopié.

Machine de Turing non-déterministe

Définition :

Une machine de Turing non-déterministe est définie par un quadruplet (K, Σ, δ, q_0) où

- Σ, K et q_0 sont les mêmes que pour une MdT déterministe.
- δ est une 'fonction' de transition qui à chaque couple de $\Sigma \times K$ associe un ou **plusieurs triplets** de $\Sigma \times K \times \{\rightarrow, \leftarrow, -\}$.
- Autrement dit le caractère non déterministe se traduit par :
  - A partir d'une situation donnée (état + symbôle lu) la MDT a 'le choix' entre plusieurs transitions.

Machine de Turing à plusieurs bandes

Définition :

Une machine de Turing à k bandes est définie par un quadruplet (K, Σ, δ, q_0) où

- Σ, K et q_0 sont les mêmes que pour une MdT déterministe.
- δ est une fonction de $\Sigma^k \times K$ vers $\Sigma^k \times K \times \{\rightarrow, \leftarrow, -\}^k$.
- Autrement dit : à chaque étape la machine lit k symbôles (1 symbôle par bande) et en fonction de l'état et des k symbôles lus, elle écrit k symbôles, effectue k déplacements et prend un nouvel état.