COMPOSITION D'INFORMATIQUE n°1

Sujet 2 (durée : 4 heures)

L'utilisation de la calculatrice **n'est pas autorisée** pour cette épreuve.

⋆⋆⋆

Ce sujet est constitué d'un premier problème sur l'indécidabilité (à faire en 1h20 environ) et d'un deuxième problème sur les graphes (à faire en 2h40 environ) qui sont indépendants.

Problème 1 : théorème de Rice

Dans l'ensemble de ce problème, on manipule des fonctions écrites dans un langage de programmation, par exemple OCaml. On distinguera la notation f, correspondant à la fonction elle-même, ou l'algorithme, et la notation $\langle f\rangle$, désignant le **code source** de la fonction f. Ainsi, si on considère le code :

```
let rec f lst = match lst with
    | []     -> 0
    | _ :: q -> 1 + f q
```

Alors on distingue l'objet `f`, de signature `'a list -> int`, et l'objet $\langle$`f`$\rangle$, de signature `string`, correspondant à la chaîne de caractères :

```
"let rec f lst = match lst with | [] -> 0 | _ :: q -> 1 + f q"
```

Par ailleurs, si $\langle f\rangle$ est le code source d'une fonction et x est une chaîne de caractère représentant un argument pour f, on note $\langle\langle f\rangle, x\rangle$ une chaîne de caractère représentant de manière unique le couple $(\langle f\rangle, x)$. On suppose que $\langle\langle f\rangle, x\rangle$ est facilement constructible à partir de $\langle f\rangle$ et x, on peut par exemple imaginer qu'il s'agit de la concaténation de $\langle f\rangle$ et x, séparés par un double point virgule `;;`.

Pour simplifier l'étude, on suppose que l'ensemble des fonctions manipulées sont de type `string -> bool`. On suppose également disposer d'une fonction `universel : string -> bool` telle que l'appel à la fonction `universel` $\langle\langle$`f`$\rangle,$`x`$\rangle$ simule l'exécution de `f x`. On notera Σ^* l'ensemble des chaînes de caractères.

Si f est une fonction, on note $L(f)$, appelé **langage de** f, l'ensemble des arguments pour lesquels la fonction renvoie `true`, c'est-à-dire :

$$L(f) = \{x \in \Sigma^* \mid f(x) \text{ termine et renvoie } \texttt{true}\}$$

Question 1 Montrer que le problème `Appartient` suivant est indécidable et semi-décidable :

* **Instance** : un code source $\langle f\rangle \in \Sigma^*$ et un argument $x \in \Sigma^*$.
* **Question** : est-ce que $x \in L(f)$?

Question 2 Montrer que le problème `Diagonal` suivant n'est pas semi-décidable :

* **Instance** : un code source $\langle f\rangle \in \Sigma^*$.
* **Question** : est-ce que $\langle f\rangle \notin L(f)$?

Indication : on pourra supposer qu'il est semi-décidable, résolu partiellement par un algorithme $\mathcal{A}$, et s'intéresser à $\langle\mathcal{A}\rangle$ et $L(\mathcal{A})$.

Question 3 Montrer que `Diagonal` $\leqslant_m$ co`Appartient`. Que peut-on en déduire sur les problèmes complémentaires de `Diagonal` et `Appartient` ?

On appelle **propriété des langages de fonctions** tout sous-ensemble de $\mathcal{P}(\Sigma^*)$. Si $P \subseteq \mathcal{P}(\Sigma^*)$, on assimile P et le problème de décision suivant, qu'on notera également P :

* **Instance** : un code source $\langle f \rangle \in \Sigma^*$.
* **Question** : est-ce que $L(f) \in P$?

Question 4 Est-ce que `Diagonal` est une propriété des langages de fonctions ? Justifier.

On appelle **langage** toute sous-ensemble de Σ^*. Si $L \subseteq \Sigma^*$, on assimile L et le problème de décision suivant, noté également L :

* **Instance** : une chaîne $x \in \Sigma^*$.
* **Question** : est-ce que $x \in L$?

Pour la suite, on cherche à montrer le théorème de Rice suivant :

Théorème

Soit P une propriété non triviale des langages semi-décidables. Alors P n'est pas décidable.

Ici, « propriété non triviale des langages semi-décidables » signifie qu'il existe deux langages semi-décidables L_1 et L_2 tels que $L_1 \in P$ et $L_2 \notin P$ (ce n'est ni une propriété vérifiée par aucun langage semi-décidable, ni par tous). On pose P une telle propriété non triviale des langages semi-décidables. Comme P est non triviale, soit $L \in P$ semi-décidable. Comme L est semi-décidable, soit f_L un algorithme qui résout partiellement L.

Question 5 Justifier qu'on peut supposer, sans perte de généralité, que $\emptyset \notin P$. On fera cette hypothèse pour la suite.

Soient $\langle f \rangle \in \Sigma^*$ et $x \in \Sigma^*$. On définit la fonction suivante :

```
let g y =
    universel <<f>, x> && universel <<f_L>, y>
```

Question 6 En considérant la fonction `g`, montrer que `Appartient` $\leqslant_m P$.

Question 7 En déduire le théorème de Rice.

Question 8 Montrer que le problème suivant est indécidable :

* **Instance** : un code source $\langle f \rangle \in \Sigma^*$.
* **Question** : est-ce que $L(f) \neq \emptyset$?

Question 9 On considère le problème suivant :

* **Instance** : un code source $\langle f \rangle \in \Sigma^*$.
* **Question** : est-ce que le code source de f contient au moins 5 boucles `while` ?

a) Ce problème est-il décidable ?
b) Cela rentre-t-il en contradiction avec le théorème de Rice ?

Problème 2 : tas de sable

Les questions de programmation doivent être traitées en langage OCaml. On autorisera toutes les fonctions des modules `Array` et `List`, ainsi que les fonctions de la bibliothèque standard (celles qui s'écrivent sans nom de module, comme `max`, `incr` ainsi que les opérateurs comme `@`). Sauf précision de l'énoncé, l'utilisation d'autres modules sera interdite.

On identifiera une même grandeur écrite dans deux polices de caractères différentes, en italique du point de vue mathématique (par exemple n) et en Computer Modern à chasse fixe du point de vue informatique (par exemple `n`).

Sans précision supplémentaire, lorsqu'une question demande la complexité d'une fonction, il s'agira de la complexité temporelle dans le pire des cas. La complexité sera exprimée sous la forme $\mathcal{O}(f(n,m))$ où n et m sont les tailles des arguments de la fonction, et f une expression la plus simple possible. Les calculs de complexité seront justifiés succinctement.

1 Introduction aux tas de sable

1.1 Préliminaires

Dans l'ensemble du sujet, on s'intéresse à des graphes non orientés non pondérés connexes. Si $G = (S, A)$ est un tel graphe, on suppose $S = \{0, 1, \ldots, n-1\}$ avec $n = |S|$. Pour $a, a' \in A^2$ et $s \in S$, on dit que a est **incidente** à a' si et seulement si $a \neq a'$ et $a \cap a' \neq \emptyset$, c'est-à-dire si a et a' sont distinctes et partagent un sommet en commun. On dit que a est **incidente** à s si et seulement si $s \in a$. On appelle **degré** de s, noté $\deg(s)$, le nombre d'arêtes incidentes à s. On note $V(s)$ l'ensemble des voisins de s, c'est-à-dire l'ensemble des sommets t tels que $\{s, t\} \in A$, et $I(s)$ l'ensemble des arêtes incidentes à s. On remarque que $\deg(s) = |I(s)|$.

On appelle **tas de sable** un couple $G_q = (G, q)$ où $G = (S, A)$ est un graphe et $q \in S$ est un sommet appelé **puits**. Les sommets de $S \setminus \{q\}$ sont appelés sommets **réguliers**. Une **configuration** de G_q est un n-uplet $c = (c_0, \ldots, c_{n-1}) \in \mathbb{Z}^n$ tel que si s est régulier, $c_s \geqslant 0$. Informellement, une valeur c_s (ou $c(s)$ si la notation est plus pertinente) correspond au nombre de grains de sable sur le sommet s. Seul le puits peut avoir un nombre de grains de sable négatif. Le **poids** d'une configuration c est la somme $|c| = \sum\limits_{s=0}^{n-1} c_s$.

Un sommet régulier s est dit **stable** pour une configuration c si $c_s < \deg(s)$. Il est dit **instable** sinon. L'**éboulement** d'un sommet instable consiste à distribuer l'un de ses grains de sable à chacun de ses voisins. Plus formellement, si c et c' sont deux configurations, on dit que c' est obtenue par éboulement de s dans c, noté $c \to_s c'$, si et seulement si, pour tout sommet $t \in S$:

- si $t = s$, alors $c'_t = c_t - \deg(t)$;
- si $t \in V(s)$, alors $c'_t = c_t + 1$;
- sinon, $c'_t = c_t$.

On dit que c s'éboule en c', noté $c \to c'$, s'il existe $s \in S$ un sommet régulier tel que $c \to_s c'$. La figure 1 illustre un éboulement. Dans cette figure, le sommet grisé est le puits. Les valeurs des configurations sont notées à côté des sommets.

Pour $s \in S$, on pose Δ_s la configuration telle que $\Delta_s(s) = -\deg(s)$, $\Delta_s(t) = 1$ si $t \in V(s)$ et $\Delta_s(t) = 0$ sinon. On remarque que $c \to_s c + \Delta_s$.

On note $\to^*$ la clôture réflexive et transitive de $\to$, c'est-à-dire que $c \to^* c'$ si et seulement s'il existe $k \in \mathbb{N}$ et des configurations $c = c_0, c_1, \ldots, c_k = c'$ telles que pour tout $i \in [\![0, k-1]\!]$, $c_i \to c_{i+1}$.

Une configuration dont tous les sommets réguliers sont stables est dite elle-même **stable**. On appelle **avalanche** une suite d'éboulements qui termine par une configuration stable.

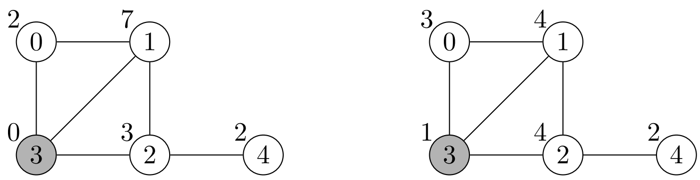


Figure 1 – Un tas de sable et deux configurations c et c' telles que $c \to_1 c'$.

Question 10 Montrer que si c s'éboule en c', alors c et c' ont le même poids.

Question 11 Montrer que les éboulements sont commutatifs, c'est-à-dire que si $s, t \in S^2$, $c \rightarrow_s c_1 \rightarrow_t c_2$ et $c \rightarrow_t c'_1 \rightarrow_s c'_2$, alors $c_2 = c'_2$.

Question 12 Représenter graphiquement, sans justification, une configuration obtenue par avalanche depuis c, la configuration de gauche dans la figure 1. Quelle est la longueur de l'avalanche (en nombre d'éboulements) ?

Pour les trois questions suivantes, on considère c une configuration d'un tas de sable G_q.

Question 13 Montrer que toute avalanche depuis c est finie.

Indication : on pourra raisonner par l'absurde et considérer l'ensemble des sommets qui s'éboulent une infinité de fois.

Question 14 Soit c une configuration d'un tas de sable G_q et soit $(s_1, s_2, \ldots, s_k)$ et (t_1, t_2, t_ℓ) deux suites de sommets telles que $c \rightarrow_{s_1} c_1 \rightarrow_{s_2} \ldots \rightarrow_{s_k} c_k$ et $c \rightarrow_{t_1} c'_1 \rightarrow_{t_2} \ldots \rightarrow_{t_\ell} c'_\ell$. On suppose que c_k et c'_ℓ sont stables. Montrer que $k = \ell$ et que les multi-ensembles $\{s_1, \ldots, s_k\}$ et $\{t_1, \ldots, t_\ell\}$ sont égaux.

Un multi-ensemble est un ensemble où on autorise les répétitions d'un même élément.

Question 15 En déduire qu'il existe une unique configuration stable obtenue par avalanche depuis c.

Par la suite, on notera $\mathrm{av}(c)$ l'unique configuration stable telle que $c \rightarrow^* \mathrm{av}(c)$.

1.2 Implémentation

On représente en OCaml un graphe par un tableau de listes d'adjacence de type :

```
type graphe = int list array
```

Dès lors, si $G = (S, A)$ est représenté par `g` de type `graphe`, alors `g` est un tableau de taille $n = |S|$, et pour $s \in S$, `g.(s)` est une liste de taille $\deg(s)$ contenant les éléments de $V(s)$ dans un ordre quelconque.

Une configuration c dans un tas de sable $G_q = ((S, A), q)$ est un tableau `c` de taille n tel que pour $s \in S$, `c.(s)` est égal à c_s.

Question 16 Écrire une fonction `eboulement (g: graphe) (c: int array) (s: int) : unit` qui calcule l'éboulement de s dans c, une configuration du graphe G. La fonction devra modifier directement le tableau `c` pour le transformer en la configuration obtenue.

Question 17 En déduire une fonction `avalanche (g: graphe) (q: int) (c: int array) : unit` qui calcule la configuration obtenue par avalanche depuis c dans le tas de sable (G, q). La fonction devra modifier directement le tableau `c` pour le transformer en la configuration obtenue.

Question 18 Déterminer la complexité temporelle et spatiale de la fonction `avalanche`, en fonction de n et de k, la longueur d'avalanche depuis c.

2 Configurations récurrentes

Question 19 Montrer qu'il existe $s \in S$ tel que le graphe induit $G[S \setminus \{s\}]$ est connexe.

Pour toute la suite, on supposera que le puits q est un tel sommet, c'est-à-dire que le sous-graphe induit par les sommets réguliers est connexe.

Une configuration c est dite **nulle** si $c_s = 0$ pour tout sommet régulier s. Elle est dite **récurrente** s'il existe une configuration c' non nulle telle que $\text{av}(c + c') = c$ (où l'addition des configurations correspond à l'addition terme à terme).

Question 20 Montrer que si $c \to^* c'$ et $d \to^* d'$, alors $c + d \to^* c' + d'$.

Question 21 Une configuration nulle peut-elle être récurrente ? Justifier.

Question 22 Montrer que pour c une configuration non nulle, il existe un entier k et une configuration c' telle que $kc \to^* c'$ et $c'_s > 0$ pour tout s sommet régulier.

On pose δ la configuration telle que $\delta_s = \deg(s)$ pour tout sommet s.

Question 23 Montrer que c est une configuration récurrente si et seulement s'il existe une configuration c' telle que $\text{av}(c' + \delta) = c$.

On pose $\beta = 2\delta - \text{av}(2\delta)$.

Question 24 Calculer β pour le tas de sable de la figure 1.

Question 25 Montrer que $\delta + \beta \to^* \delta$.

Question 26 Montrer que c est une configuration récurrente si et seulement si $\text{av}(c + \beta) = c$.

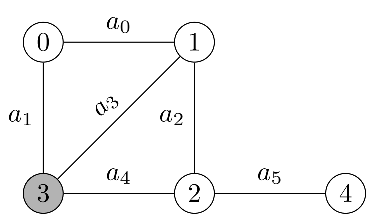
3 Parcours décroissants

Pour $(G = (S, A), q)$ un tas de sable, avec $n = |S|$ et $m = |A|$, on se donne une numérotation arbitraire des arêtes $A = \{a_0, \ldots, a_{m-1}\}$. On considère la relation $<_A$ définie sur A^2 par $a <_A a'$ si et seulement s'il existe $0 \leqslant i < j < m$ tels que $a = a_i$ et $a' = a_j$.

On appelle **parcours arêtes-sommets** une suite $\sigma = (\sigma_i)_{i \in [\![0, n+m-1]\!]}$ où chaque sommet et chaque arête de G apparaît exactement une fois. Pour σ une telle séquence et $i \in [\![0, n+m]\!]$, on note $\sigma^{<i} = \{\sigma_j \mid j < i\}$. Un parcours arêtes-sommets est dit **décroissant** si :

- si σ_i est un sommet régulier, alors σ_{i-1} est une arête incidente à σ_i ;
- si σ_i est une arête, alors cette arête est maximale selon $<_A$ parmi les arêtes qui ne sont pas dans $\sigma^{<i}$ et qui sont incidentes à un sommet de $\sigma^{<i}$.

On remarque en particulier qu'un parcours décroissant commence par le puits du tas de sable. La figure 2 donne un exemple de parcours décroissant. On note $D(G_q)$ l'ensemble des parcours décroissants de G_q.



$$\sigma = (3, a_4, a_3, 1, a_2, 2, a_5, 4, a_1, 0, a_0)$$

FIGURE 2 – Un tas de sable et un parcours décroissant.

Question 27 Soient $\sigma \neq \tau$ deux parcours décroissants d'un même tas de sable. On pose k le plus petit indice tel que $\sigma_k \neq \tau_k$. Montrer que $\{\sigma_k, \tau_k\}$ est un ensemble constitué d'un sommet et d'une arête.

Pour σ un parcours décroissant, on pose $\Phi(\sigma) = \{\sigma_{k-1} \mid \sigma_k \in S \setminus \{q\}\}$.

Question 28 Montrer que $(S, \Phi(\sigma))$ est un arbre couvrant de G.

On note $C(G)$ l'ensemble des parties de A qui forment un arbre couvrant de G.

Question 29 Montrer que Φ est une bijection de $D(G_q)$ dans $C(G)$.

Pour σ un parcours décroissant, on dit qu'une arête $\sigma_i = \{\sigma_j, \sigma_k\}$ est **forte** si elle apparaît après ses extrémités, c'est-à-dire si $j < i$ et $k < i$. Dans le parcours de la figure 2, l'arête a_0 est une arête forte.

Pour $T = (S, B)$ un arbre couvrant de G, on dit qu'une arête $a \in A \setminus B$ est **active extérieurement** si c'est l'arête minimale selon $<_A$ dans l'unique cycle de $(S, B \cup \{a\})$.

Question 30 Soit σ un parcours décroissant et $a \in A$. Montrer que a est forte pour σ si et seulement si elle est active extérieurement pour $\Phi(\sigma)$.

4 Parcours décroissants et configurations récurrentes

On rappelle que pour s un sommet, $I(s)$ désigne l'ensemble des arêtes incidentes à s.

Pour σ un parcours décroissant, on pose $\Psi(\sigma)$ la configuration c telle que $c_{\sigma_i} = |I(\sigma_i) \setminus \sigma^{<i}|$.

Question 31 Déterminer la configuration $\Psi(\sigma)$, pour σ le parcours décroissant de la figure 2.

Question 32 Montrer que si σ est un parcours décroissant, alors $\Psi(\sigma)$ est une configuration récurrente.

Indication : on pourra considérer $\Psi(\sigma) + \Delta_q$.

On note $\mathrm{Rec}(G_q)$ l'ensemble des configurations récurrentes c de G_q telles que $c_q = \deg(q)$.

Question 33 Montrer que Ψ est une bijection de $D(G_q)$ dans $\mathrm{Rec}(G_q)$.

Question 34 Montrer que si σ est un parcours décroissant, alors la somme entre le nombre d'arêtes et le nombre d'arêtes fortes pour σ est égale au poids de $\Psi(\sigma)$.

⋆⋆⋆