Option Informatique en Spé MP et MP*
Devoir surveillé du mercredi 23 mars 2005

Recommandations pour l'écriture des programmes

▶ Chaque question de programmation fixe un *objectif* : c'est le nombre maximal de lignes que vous devez écrire. Vous compterez pour une ligne l'en-tête d'une fonction, comme :

```
let toto x y = function
```

Vous compterez chaque motif pour une ligne ; notez bien qu'un motif union, comme :

```
| [] | [true] | [_;false;_] -> ...
```

compte pour une seule ligne.

▶ L'emploi de références ou de structures de données mutables est interdit.

1 La suite de Kimberling

▶ Soit $k \in \mathbb{N}^*$; nous noterons $\mu(k)$ la plus grande puissance de 2 qui divise k ; par exemple, $\mu(12) = 4$.

Question 1 • Rédigez en Caml une fonction de signature :

```
mu : int -> int
```

spécifiée comme suit : `mu k` calcule $\mu(k)$. Objectif : trois lignes.

▶ La suite de Kimberling est la suite dont le terme général est défini par la formule $2c_n = 1 + \dfrac{n}{\mu(n)}$.

Question 2 • Montrez que les termes de cette suite sont tous dans $\mathbb{N}^*$.

Question 3 • Dressez un tableau donnant la valeur de c_n pour $1 \leqslant n \leqslant 16$. Vous ne ferez apparaître que les résultats dans votre copie.

Question 4 • Rédigez en Caml une fonction de signature :

```
kim : int -> int
```

spécifiée comme suit : `kim n` calcule c_n, en faisant appel à la fonction μ. Objectif : deux lignes.

Question 5 • Quelle relation simple existe-t-il entre c_{2n} et c_n, pour $n \geqslant 1$?

Question 6 • Montrez que tout élément de $\mathbb{N}^*$ apparaît au moins une fois dans la suite de Kimberling.

Question 7 • Soit $n \geqslant 1$. Quelle est la position de la première occurrence de n dans la suite de Kimberling ?

Question 8 • Rédigez en Caml une fonction de signature :

```
kim_bis : int -> int
```

spécifiée comme suit : `kim_bis n` calcule c_n, sans faire appel à la fonction μ. Objectif : trois lignes.

Question 9 • Effacez la première occurrence de chaque élément de $\mathbb{N}^*$ dans la suite de Kimberling. Qu'obtenez-vous ?

Question 10 • Soit $p \in \mathbb{N}$. Montrez que l'ensemble $E_p = \{c_k \mid 2^p \leqslant k < 2^{p+1}\}$ est égal à l'intervalle discret $[\![1, 2^p]\!]$.

Question 11 • En déduire une expression simple de $\displaystyle\sum_{1 \leqslant k \leqslant 2^p} c_k$.

2 Codes

2.1 Quelques résultats simples sur les codes

▶ Dans ce problème, $\Sigma = \{0,1\}$. Soient u et v deux mots sur cet alphabet ; nous noterons $u \prec v$ lorsque u est préfixe de v (ce qui revient à dire qu'il existe un mot w tels que $v = uw$). Une partie X de Σ^* est *préfixe* si, quels que soient les éléments u et v de X, on n'a ni $u \prec v$, ni $v \prec u$.

Question 1 • Donnez un exemple de partie préfixe infinie de Σ^* (preuve à l'appui, bien entendu).

▶ Soit X une partie non vide de Σ^*. Nous dirons que X est un *code* si l'égalité $u_1u_2\ldots u_n = v_1v_2\ldots v_p$ (où les u_i et les v_j sont tous dans X) implique $n = p$ et $u_i = v_i$ pour tout $i \in [\![1, n]\!]$. En d'autres termes : chaque mot de X^* possède exactement une décomposition en produit d'éléments de X.

Question 2 • Le langage $M = \{0100, 01, 101, 001\}$ est-il un code ?

Question 3 • Montrez que le langage $L = \{01, 001, 01000, 101\}$ est un code.

Question 4 • Montrez que toute partie préfixe de Σ^* est un code.

Question 5 • Que pensez-vous de la réciproque ?

▶ Soit Δ un deuxième alphabet (fini, non vide). Toute fonction φ de Δ dans Σ^* induit un morphisme Φ de Δ^* dans Σ^*, défini par $\Phi(x_1x_2\ldots x_n) = \varphi(x_1)\varphi(x_2)\ldots\varphi(x_n)$ pour tout mot $x_1x_2\ldots x_n \in \Delta^*$. Φ vérifie $\Phi(uv) = \Phi(u)\Phi(v)$ quels que soient les mots u et v appartenant à Δ^* ; en particulier, $\Phi(\varepsilon) = \varepsilon$.

Question 6 • Soit X une partie de Σ^* de même cardinal que Δ. Montrez que X est un code ssi, pour toute bijection φ de Δ sur X, le morphisme Φ de Δ^* sur Σ^* induit par φ est injectif.

2.2 L'algorithme de Sardinas et Patterson

▶ Nous nous proposons de décrire un algorithme décidant si un langage X fini non vide est un code ; cet algorithme est dû à Sardinas et Patterson.

▶ Nous noterons P l'ensemble des préfixes non vides des mots de X ; remarquez que X est contenu dans P.

▶ Associons au langage X un graphe orienté $G = (P, A)$ défini comme suit : les sommets sont les éléments de P ; les arcs, qui sont des couples (u, v) de sommets, sont de l'un des deux types suivants :

- un arc (u, v) est *croisé* si $uv \in X$: un tel arc sera étiqueté par le mot uv ;
- un arc (u, v) est *direct* si $v \notin X$ et s'il existe $x \in X$ tel que $ux = v$: un tel arc sera étiqueté par le mot x.

La figure 1 montre le graphe associé au langage $X_1 = \{1, 01, 0110\}$. Les sommets qui appartiennent à X_1 ont été placés à gauche, les autres ont été placés à droite. L'unique arc direct a été tracé en trait gras, les deux arcs croisés ont été tracés en trait maigre.

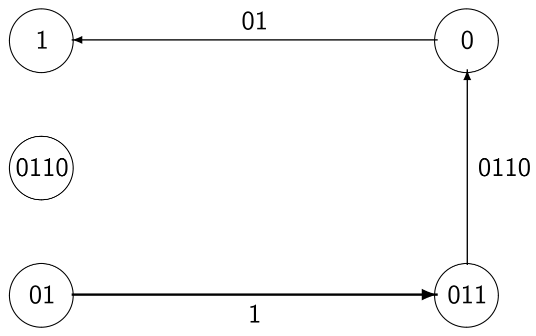

Figure 1: le graphe associé au langage X_1

Question 7 • Construisez le graphe associé au langage M étudié à la question 2. Vous suivrez les conventions adoptées dans la figure 1.

▶ Nous noterons $P - X$ l'ensemble des éléments de P qui n'appartiennent pas à X.

▶ Un *chemin* de longueur n dans le graphe G est une suite $(s_i, t_i)_{0 \leqslant i \leqslant n}$ d'arcs de G, telle que $t_i = s_{i+1}$ pour tout $i \in [\![0, n-1]\!]$. Nous dirons que ce chemin mène de s_0 à t_n.

▶ Nous allons montrer le lemme suivant : soit u appartient à $P - X$; il existe des mots y et z appartenant à X^* tels que $yu = z$ ssi il existe $x \in X$ et un chemin dans G menant de x à u. Ce lemme servira dans les questions 10 et 11.

Question 8 ⋆⋆ • Soient $x \in X$ et $u \in P - X$; supposons qu'il existe un chemin menant de x à u dans G. Montrez qu'il existe des mots y et z appartenant à X^* tels que $yu = z$. Indication : raisonnez par récurrence sur la longueur du chemin.

Question 9 ⋆⋆ • Soit $u \in P - X$; supposons qu'il existe des mots y et z appartenant à X^* tels que $yu = z$. Montrez qu'il existe $x \in X$ et un chemin menant de x à u dans G. Indication : raisonnez par récurrence sur la longueur commune des mots yu et z.

Question 10 ⋆⋆ • Supposons qu'il existe dans G un chemin de longueur non nulle, menant de $s \in X$ à $t \in X$. Montrez que X n'est pas un code.

Question 11 ⋆⋆ • Supposons que X n'est pas un code. Montrez qu'il existe des mots $s \in X$ et $t \in X$ (pas forcément distincts) et un chemin de G menant de s à t.

▶ Nous venons de montrer que X est un code ssi il n'existe dans G aucun chemin menant d'un sommet $s \in X$ à un sommet $t \in X$ (non nécessairement distinct de s).

Question 12 • En déduire un algorithme permettant de décider si un langage X fini non vide est un code.

Question 13 • Justifiez l'affirmation suivante : le coût de cet algorithme est polynomial en $\sum_{x \in X} |x|$.

FIN