

Option Informatique en Spé MP et MP*

Devoir surveillé du 24 mars 2004

Résumé

Sujet semblable à ce qui est posé aux concours communs : deux exercices et un problème.

Le premier exercice porte sur les langages rationnels et les automates finis. Vous devriez le traiter en une heure environ.

Le deuxième exercice présente un ordinateur assez particulier, puisqu'il ne comporte que deux cases mémoire ! Vous devriez le traiter en trente minutes environ.

Le problème étudie l'algorithmique des intervalles de \mathbb{Z} . Il vous donnera l'occasion de montrer votre maîtrise du langage Caml ; respectez soigneusement les consignes d'écriture des programmes, rappelées dans le texte.

Veuillez rédiger chaque partie sur une copie séparée.

Table des matières

1	Langages rationnels et automates finis	2
2	Quelques questions à propos d'un curieux ordinateur	3
3	Algorithmique des intervalles de \mathbb{Z}	4
3.1	Préliminaires	4
3.2	Arbres d'intervalles	5

1 Langages rationnels et automates finis

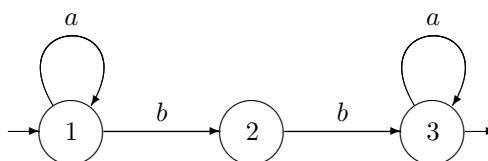
► Nous nous intéressons dans ce problème à une classe particulière d'automates finis, que nous qualifierons de *sahariens*. Un automate saharien est un quintuplet $\mathcal{A} = (Q, \delta, I, F, \Omega)$ où :

- (Q, δ, I, F) est un automate fini non déterministe ;
- Ω est un sous-ensemble de Q , dont les éléments seront appelés *oasis*.

Les notions de calcul et d'étiquette d'un calcul sont les mêmes que pour un automate classique, seule change la définition d'un calcul réussi. Soit k un naturel non nul ; nous dirons qu'un calcul passant par les états q_0 à q_n est réussi au seuil k si $q_0 \in I$, $q_n \in F$ et si toute suite de k transitions de ce calcul passe par au moins une oasis ; de façon plus formelle : si $i \geq 0$ et $i+k \leq n$, alors $\{q_i, q_{i+1}, \dots, q_{i+k-1}, q_{i+k}\} \cap \Omega \neq \emptyset$.

► Le langage reconnu par \mathcal{A} au seuil k est l'ensemble des calculs de \mathcal{A} réussis au seuil k ; vous pourrez le noter $\mathcal{L}_k(\mathcal{A})$.

► Voyons ceci sur l'exemple suivant :



Avec $\Omega = \emptyset$ et $k = 4$, le langage reconnu est $\{bb, abb, bba\}$. Avec $\Omega = \{1\}$ et $k = 2$, le langage reconnu est décrit par l'expression rationnelle a^*bb .

Question 1 • Quel est le langage reconnu par cet automate lorsque $\Omega = \{2\}$ et $k = 3$?

Question 2 • Explicitez en fonction de k le cardinal du langage reconnu par cet automate lorsque $\Omega = \emptyset$.

Question 3 • Montrez que tout langage rationnel peut être reconnu par un automate saharien.

Question 4 • Soit \mathcal{A} un automate saharien à n états. Montrez que le langage reconnu par \mathcal{A} au seuil k peut être reconnu par un automate fini classique à $(k+1)n$ états.

Question 5 • Pouvez-vous répondre à la question précédente, avec un automate ayant moins de $(k+1)n$ états ?

2 Quelques questions à propos d'un curieux ordinateur

► Nous disposons d'un *ordinateur hyperquantique* capable d'effectuer des additions et des soustractions dans \mathbb{Z} , sans limitation de la taille des nombres mis en jeu. Cet ordinateur ne dispose que de deux mémoires, notées a et b . Les actions qu'il est possible d'effectuer sont de trois types :

- recopie, par exemple $a \leftarrow b$;
- addition, par exemple $a \leftarrow a + b$;
- soustraction, par exemple $a \leftarrow b - b$.

Question 1 • L'étudiant Jean-Maurice MALHABILE souhaite échanger les contenus des deux mémoires. Pour ce faire, il propose la suite d'actions $a \leftarrow b; b \leftarrow a$. Expliquez pourquoi sa proposition est incorrecte.

Question 2 • Quel est l'effet *net* de la suite d'actions $a \leftarrow a + b; b \leftarrow a - b; a \leftarrow a - b$?

Question 3 • Proposez une suite d'actions dont l'effet net est de multiplier par -1 le contenu de la mémoire a , sans changer le contenu de la mémoire b .

Question 4 • Énoncez (et démontrez) une condition nécessaire et suffisante portant sur les contenus des mémoires a et b , pour que l'on puisse obtenir le nombre 1 dans l'une de ces mémoires après un nombre fini d'actions.

► Les capacités remarquables de notre ordinateur nous ont permis de signer un contrat de recherche et développement avec la société UMP¹. Avec les bénéfices, nous ajoutons à notre ordinateur une troisième mémoire, notée c .

Question 5 • Proposez une suite de *six* actions réalisant la permutation circulaire $(a, b, c) \leftarrow (b, c, a)$.

Question 6 • Que proposez-vous d'écrire en Caml pour réaliser la permutation circulaire des valeurs associées aux trois noms `a`, `b` et `c` ?

¹Universal Messier Prévarication

3 Algorithmique des intervalles de \mathbb{Z}

► 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 = fonction
```

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.

3.1 Préliminaires

► Le cardinal d'un ensemble fini X est noté $|X|$.

► Soient p et q deux relatifs tels que $p \leq q$. Nous noterons $\llbracket p, q \rrbracket$ l'*intervalle discret* délimité par p et q : c'est l'ensemble des relatifs k tels que $p \leq k \leq q$. Dans la suite, nous utiliserons le terme *intervalle* pour désigner un intervalle discret.

► Nous dirons que les intervalles $\llbracket p, q \rrbracket$ et $\llbracket r, s \rrbracket$ sont *consécutifs* lorsque $r = q + 1$. Dans ce cas, leur réunion est l'intervalle $\llbracket p, s \rrbracket$.

► Soient E et F deux ensembles. Nous noterons $E - F$ l'ensemble des éléments de E qui n'appartiennent pas à F . Ainsi, $\llbracket 2, 23 \rrbracket - \llbracket 15, 28 \rrbracket = \llbracket 2, 14 \rrbracket$

► Soient $\mathcal{I} = \llbracket a, b \rrbracket$ et $\mathcal{J} = \llbracket c, d \rrbracket$ deux intervalles de \mathbb{Z} . Nous noterons $\mathcal{I} \prec \mathcal{J}$ lorsque $b < c - 1$; en clair : \mathcal{I} est à gauche de \mathcal{J} , et il existe au moins un relatif entre b et c . La relation \preceq définie par

$$\mathcal{I} \preceq \mathcal{J} \iff (\mathcal{I} \prec \mathcal{J} \text{ ou } \mathcal{I} = \mathcal{J})$$

est un ordre partiel sur l'ensemble des intervalles de \mathbb{Z} .

► Soit $\mathcal{F} = (\mathcal{I}_i)_{1 \leq i \leq n}$ une famille finie d'intervalles de \mathbb{Z} . Nous noterons $\widehat{\mathcal{F}}$ l'ensemble $\bigcup_{1 \leq i \leq n} \mathcal{I}_i$, et nous

dirons que \mathcal{F} est une *représentation* de $\widehat{\mathcal{F}}$. Nous dirons qu'une partie de \mathbb{Z} est *représentable* si elle peut être représentée par une famille finie d'intervalles de \mathbb{Z} . Nous convenons que l'ensemble vide est représentable, et qu'il est représenté par la famille vide.

Question 1 • Montrez que l'ensemble des parties représentables de \mathbb{Z} est stable par union et intersection.

Question 2 • Soit A une partie représentable de \mathbb{Z} . Montrez qu'il existe une et une seule famille $\mathcal{G} = (\mathcal{J}_j)_{1 \leq j \leq p}$ d'intervalles de \mathbb{Z} vérifiant les conditions suivantes :

- pour $1 \leq j < p$, on a $\mathcal{J}_j \prec \mathcal{J}_{j+1}$;
- $\widehat{\mathcal{G}} = A$.

► Nous dirons que \mathcal{G} est la *représentation canonique* de A ; le nombre p d'intervalles qui constituent \mathcal{G} est la *complexité* de A , que nous noterons $\mathcal{C}(A)$.

Question 3 • Montrez que $\mathcal{C}(A)$ est au plus égale à $|A|$, et que cette borne peut être atteinte.

Question 4 • Soient A une partie représentable de \mathbb{Z} , et $\mathcal{F} = (\mathcal{I}_i)_{1 \leq i \leq n}$ une représentation de A . Proposez un algorithme donnant la représentation canonique de A , pour un coût $\mathcal{O}(n \ln(n))$. Notez bien que l'on ne demande pas un programme, et que le coût de votre algorithme doit être indépendant de $|A|$.

Question 5 • Rédigez en Caml une fonction de signature

```
ajoute_intervalle : int * int -> (int * int) list -> (int * int) list
```

spécifiée comme suit : si ℓ est la représentation canonique d'une partie A de \mathbb{N} et si $a \leq b$, alors `ajoute_intervalle (a,b) l` rend la représentation canonique de $A \cup \llbracket a, b \rrbracket$. Objectif : cinq lignes.

Question 6 • Soient A une partie représentable de \mathbb{Z} , et \mathcal{J} un intervalle de \mathbb{Z} . Notons $A - \mathcal{J}$ l'ensemble des éléments de A qui n'appartiennent pas à \mathcal{J} . Montrez que $A - \mathcal{J}$ est représentable.

Question 7 • Rédigez en Caml une fonction de signature

```
retranche_intervalle : int * int -> (int * int) list -> (int * int) list
```

spécifiée comme suit: si ℓ est la représentation canonique d'une partie A de \mathbb{N} et si $a \leq b$, alors `retranche_intervalle (a,b) l` rend la représentation canonique de $A - \llbracket a, b \rrbracket$. Objectif: huit lignes.

Question 8 • Soient A et B deux parties représentables non vides de \mathbb{Z} , de complexités respectives p et q . Montrez que la complexité de $A \cup B$ est comprise entre 1 et $p + q$.

Question 9 • Soient p et q deux naturels non nuls. Exhibez deux parties A et B représentables de \mathbb{Z} telles que $\mathcal{C}(A) = p$, $\mathcal{C}(B) = q$ et $\mathcal{C}(A \cup B) = 1$.

Question 10 • Soient p et q deux naturels non nuls. Exhibez deux parties A et B représentables de \mathbb{Z} telles que $\mathcal{C}(A) = p$, $\mathcal{C}(B) = q$ et $\mathcal{C}(A \cup B) = p + q$.

► Nous venons de montrer que l'encadrement établi à la question 8 est optimal.

Question 11 • Soient A et B deux parties représentables non vides de \mathbb{Z} , de complexités respectives p et q . Donnez un encadrement optimal de la complexité de $A \cap B$.

3.2 Arbres d'intervalles

► Un *arbre d'intervalles* est un arbre binaire de recherche dont les nœuds sont étiquetés par des intervalles; la figure 1 présente un arbre d'intervalles. Les feuilles ne portent pas d'information, et n'ont pas été dessinées.

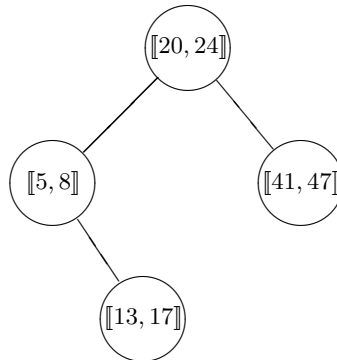


Figure 1: un arbre d'intervalles

► Rappelons que dans un arbre binaire de recherche, l'étiquette de chaque nœud majeure (resp. mineure) strictement les étiquettes des nœuds du sous-arbre gauche (resp. droit); la relation de comparaison utilisée ici est \preceq . Nous souhaitons utiliser des arbres d'intervalles pour décrire efficacement des parties représentables de \mathbb{Z} .

► Nous commençons par définir un type Caml pour décrire les arbres d'intervalles:

```
type intrv_tree = F | N of (int * int * intrv_tree * intrv_tree) ;;
```

Le constructeur constant `F` sert à décrire un arbre vide; les quatre arguments du constructeur non constant `N` sont, dans l'ordre, les bornes de l'intervalle racine de l'arbre, le sous-arbre gauche et le sous-arbre droit. Ainsi, l'arbre de la figure 1 pourrait être défini comme suit:

```
let t =
  let tgd = N(13,17,F,F)
  in let tg = N(5,8,F,tgd) and td = N(41,47,F,F)
  in N(20,24,tg,td) ;;
```

Question 12 • Soit t un arbre d'intervalles représentant une partie A représentable de \mathbb{Z} . Rédigez en Caml une fonction de signature

```
cherche : int -> intrv_tree -> bool
```

spécifiée comme suit: `cherche x t` indique si x appartient à A . Objectif: quatre lignes.

Question 13 • Soit t un arbre d'intervalles représentant une partie A représentable de \mathbb{Z} . La *taille* de t est le cardinal de A ; ainsi, la taille de l'arbre de la figure 1 est 21. Rédigez en Caml une fonction de signature

```
taille : intrv_tree -> int
```

spécifiée comme suit: `taille t` calcule la taille de l'arbre t . Objectif: trois lignes.

Question 14 • Soit t un arbre d'intervalles représentant une partie A représentable de \mathbb{Z} . Rédigez en Caml une fonction de signature

```
minimum : intrv_tree -> int
```

spécifiée comme suit: `minimum t` calcule le plus petit élément de A , ou lève une exception si cet ensemble est vide. Objectif: quatre lignes.

Question 15 • Soit t un arbre d'intervalles représentant une partie A représentable de \mathbb{Z} . Rédigez en Caml une fonction de signature

```
énumère : intrv_tree -> int list
```

spécifiée comme suit: `énumère t` dresse la liste des éléments de A . Ainsi, la fonction `énumère t` appliquée à l'arbre de notre exemple rendra la liste

```
[5; 6; 7; 8; 13; 14; 15; 16; 17; 20; 21; 22; 23; 24; 41; 42; 43; 44; 45; 46; 47]
```

Objectif: trois lignes, en utilisant la fonction `intervalle`.

Question 16 • Pour calculer le plus petit élément d'un arbre d'intervalles t , l'étudiant Jean-Maurice MALAVISÉ propose la fonction Caml suivante:

```
let minimum t = hd (énumère t) ;;
```

Ceci répond à la spécification décrite à la question 14. Quelle critique pouvez-vous formuler?

Question 17 • Pour insérer un élément x dans un arbre d'intervalles t , notre ami JMM propose la fonction Caml suivante:

```
let rec insère x = fonction
  | F -> N(x,x,F,F)
  | N(g,d,tg,td) when x < g -> let tg' = insère x tg in N(g,d,tg',td)
  | N(g,d,tg,td) when x > d -> let td' = insère x td in N(g,d,tg,td')
  | t -> t ;;
```

Expliquez ce qui ne va pas!

► Soient A une partie représentable de \mathbb{Z} et $\mathcal{F} = (\mathcal{I}_i)_{1 \leq i \leq n}$ une représentation (non nécessairement canonique) de A . Nous allons montrer comment construire un arbre d'intervalles fournissant la représentation canonique de $\widehat{\mathcal{F}}$. La méthode consiste à ajouter à un arbre initialement vide les éléments de \mathcal{F} .

► Soient t un arbre non réduit a une feuille, $\mathcal{R} = \llbracket g, d \rrbracket$ l'intervalle qui étiquette de sa racine, et $\mathcal{I} = \llbracket a, b \rrbracket$ l'intervalle que l'on veut ajouter.

Question 18 • Montrez qu'il existe huit cas de figure, selon les places respectives de \mathcal{R} et \mathcal{I} ; et que dans trois de ces cas, la racine ne sera pas modifiée.

► Supposons que l'on veuille ajouter l'intervalle $\llbracket 15, 22 \rrbracket$ à l'arbre de la figure 1. Il va falloir fusionner cet intervalle avec $\llbracket 13, 17 \rrbracket$ et $\llbracket 20, 24 \rrbracket$: au final, il ne restera plus que trois nœuds dans l'arbre. Décrivons le travail de façon plus rigoureuse: dans un premier temps, nous faisons descendre dans l'arbre binaire l'intervalle $\mathcal{I} = \llbracket a, b \rrbracket$ à ajouter; si la descente se poursuit sans encombre jusqu'à une feuille, tant mieux. Sinon, nous nous arrêtons au premier nœud $\mathcal{J} = \llbracket g, d \rrbracket$ tel que la famille $(\mathcal{I}, \mathcal{J})$ ne soit pas une représentation canonique: ceci se produit lorsque les deux intervalles sont consécutifs, ou ont une

intersection non vide. Nous remplaçons alors \mathcal{J} par l'intervalle résultant de la fusion de \mathcal{I} et \mathcal{J} . Il reste à examiner éventuellement ce qui se passe dans les sous-arbres gauche et droit. Dans notre exemple, la borne droite de la racine n'est pas modifiée, donc il n'est pas nécessaire de descendre dans le sous-arbre droit. En revanche, la borne gauche devient 15 : nous allons donc fusionner (au besoin) le nouvel intervalle racine avec une partie du sous-arbre gauche.

Question 19 • Rédigez en Caml une fonction de signature

```
étend_g : int -> intrv_tree -> int * intrv_tree
```

spécifiée comme suit : `étend_g a tg` rend un couple (a', tg') tel que a' est la nouvelle borne gauche de la racine, et tg' est le nouveau sous-arbre gauche. Dans le cas de notre exemple, `étend_g 15 tg` rend le couple $(13, N(5, 8, F, F))$. Objectif : six lignes.

Question 20 • On suppose écrite la fonction `étend_d`. En déduire une fonction de signature

```
ajoute_intervalle : int * int -> intrv_tree -> intrv_tree
```

spécifiée comme suit : `ajoute_intervalle (a,b) t` ajoute l'intervalle $\llbracket a, b \rrbracket$ à l'arbre d'intervalles t . Objectif : huit lignes.

Question 21 • Expliquez à notre ami Jean-Maurice comment rédiger sa fonction d'insertion, en utilisant `ajoute_intervalle`.

Question 22 • Rédigez en Caml une fonction de signature

```
scission : int -> intrv_tree -> intrv_tree * intrv_tree
```

spécifiée comme suit : `scission x t` découpe l'arbre d'intervalles t en deux arbres t_g et t_d ; le premier regroupe les éléments de t qui sont strictement inférieurs à x ; le deuxième regroupe les éléments de t qui sont strictement supérieurs à x . Ainsi, l'application de `scission 13` à l'arbre de notre exemple nous donne le couple formé des arbres $N(5, 8, F, F)$ et $N(20, 24, N(14, 17, F, F), N(41, 47, F, F))$. Objectif : huit lignes ; en effet, il existe sept situations différentes.

FIN