Composition d'Informatique (4 heures), Filière MP

Rapport de MM. Frédéric MAGNIEZ et Benjamin WERNER, correcteurs.

L'épreuve

Il s'agissait d'un problème de codage efficace à l'aide des codes de correction d'erreurs de Reed-Solomon.

Plus précisément, il s'agissait de réaliser des opérations rapides de polynômes dont la multiplication, la division, et l'évaluation d'un polynôme en plusieurs points.

Dans la première partie, il était demandé de programmer le codage de Reed-Solomon de manière simple mais pas le plus efficacement possible.

Dans la deuxième partie, les opérations d'addition, de soustraction, de multiplication et de division de polynômes devaient être programmées. La difficulté principale résidait dans la représentation par liste des polynômes. Par exemple, la mise au point de la division de polynômes dans cette représentation était délicate.

La troisième partie proposait une réalisation rapide de la multiplication et de la division de polynômes suivant la méthode de Karastuba. La multiplication, au programme, ne présente guère de difficulté. La division, plus délicate, était elle complètement guidée par l'énoncé.

Enfin, la quatrième partie utilisait les opérations sur les polynômes des parties précédentes, afin de réaliser une évaluation multi-points efficace à l'aide de structures arborescentes. Un premier arbre binaire de produits successifs devait être construit des feuilles vers la racine. Puis un arbre binaire d'évaluation en était déduit par calcul de restes successifs de la racine vers les feuilles.

Remarques générales

Les fonctions demandées faisaient en générale une dizaine de lignes, avec des pointes à une trentaine de lignes pour les questions les plus difficiles.

Les candidats doivent s'efforcer de fournir des réponses concises. Lorsqu'un candidat part sur une voie compliquée, en général elle est erronée. Plus une solution est compliquée plus elle doit être accompagnée d'une explication synthétique afin d'en améliorer sa lisibilité.

Parmi les candidats français, 11% ont choisi d'écrire leurs programmes en langage Pascal, soit un pourcentage à peu près stable par rapport à celui de l'an dernier. La moyenne de ces copies est similaire à celles des copies CAML.

Pour obtenir la note maximale, il fallait traiter la totalité du sujet.

Les notes des 758 candidats français se répartissent selon le tableau suivant, avec une moyenne de 11,03 et un écart type de 3,78.

$0 \leqslant \mathrm{N} < 4$	20	2,6%
$4 \leqslant \mathrm{N} < 8$	143	18,9%
$8 \leqslant \mathrm{N} < 12$	295	38,9%
$12 \leqslant \mathrm{N} < 16$	223	29,4%
$16 \leqslant \mathrm{N} \leqslant 20$	77	10,2%

Commentaire détaillé

Pour chaque question, figure entre crochets le pourcentage des candidats français ayant obtenus au moins la moitié des points à la question.

Question 1 [81%] Dans cette question, ainsi que les suivantes, on a vu de nombreuses solutions itératives qui étaient le plus souvent beaucoup plus compliquées que la solution récursive.

De nombreux candidats ne faisaient pas assez de normalisation modulo p ; cela a été peu pénalisé, même si cela témoignait d'une mauvaise compréhension des entiers informatiques.

Plus lourde était l'erreur qui consistait à introduire une opération d'exponantiation, ce qui n'était pas nécessaire. Dans ce cas la complexité devenait quadratique, même en ne considérant que le nombre d'opérations numériques, comme demandé dans l'énoncé.

Question 2 [96%] Mêmes remarques que pour la première question, mais dans ce cas, les solutions itératives nécessitaient en général de renverser la liste à la fin (ou pire, d'ajouter systématiquement des éléments en fin de liste), ce qui est peu élégant.

Question 3 [99%] Très peu de difficultés rencontrées sur cette question simple.

Question 4 [96%] Peu de difficultés. Les seul écueils éventuels étant identiques à la question 1.

Question 5 [98%] Certains ont oublié de calculer l'opposé des éléments de la seconde liste lorsque celle-ci était plus longue. D'autres candidats ont été abusé par l'énoncé et ont utilisé la fonction *inv* qui désignait en fait l'inverse multiplicatif. On a peu pénalisé cette confusion.

Question 6 [99%] Très peu de difficultés.

Question 7 [83%] Question le plus souvent bien traitée. On a trouvé ici plusieurs copies qui ont soit utilisé des tableaux (certains l'avait fait dès les premières questions). D'autres ont introduits des fonction allant chercher le i-ème élément d'une liste comme pour un tableau. Le plus souvent cela donnait du code très abscons ; souvent trop pour être vérifiable dans un temps raisonnable par les correcteurs.

Question 8 [37%] La plupart des réponses avaient compris le principe mathématique de l'algorithme. La difficulté principale était de réussir à l'implémenter en restant raisonnablement concis. Quelques erreurs courantes :

- Mauvaises prises en compte des zéros en fin de liste. C'était effectivement délicat à mettre au point sans pouvoir exécuter les programmes. Cela a été en général très peu pénalisé.
- Boucle ou fonction récursive principale qui faisait les calculs (en particulier les soustractions) après avoir renversé les listes. Cela donne des résultat faux lorsque les listes ne sont pas de la même longueur.
- Une utilisation d'un produit de polynômes à la place d'un produit scalaire dans la boucle, ce qui donne une complexité cubique et non plus quadratique.

Question 9 [93%] Pas de difficulté. On pouvait simplement utiliser la fonction définie en 8.

Question 10 [96%] Question très bien traitée.

Question 11 [68%] Les erreurs courantes ont été :

- Oubli du cas de base lorsque les arguments sont de longueur 1. De nombreuses solutions ne faisait pas apparaître la multiplication du tout.
- Les deux additions finales servant à sommer les trois composantes ne pouvaient pas être obtenues par des concaténations de listes. Les composantes pouvant avoir des facteurs communs.
- fonction scindant la liste peu élégante (ajout d'éléments en fin de liste).

Question 12 [41%] A peu près les mêmes remarques que pour la multiplication rapide (à l'exception du second point.

La question sur la complexité permettait de voir si ce point avait été vraiment compris.

Question 13 [67%] Rien de particulier.

Question 14 [41%] Parmi les copies qui proposaient un programme rendant un résultat correct, près de la moitié ne tenaient pas compte de la remarque de l'énoncé et commençaient par calculer la racine de l'arbre, ce qui ne donnait pas la bonne complexité.

Question 15 [31%] Remarque similaire que pour la question précédente.

Question 16 [33%] Question pas mal traitée, surtout pour la partie code.