Composition d'Informatique (4 heures), Filière MP

(XULC)

Rapport de MM. Charles-Edmond BICHOT, Jean-Christophe FILLIÂTRE, Paulin de NAUROIS et Mme Stéphanie DELAUNE, correcteurs.

1. L'épreuve

Il s'agissait dans ce problème de construire et d'utiliser la structure de *Zero-suppressed Binary Decision Diagrams* (appelés « arbres combinatoires » dans le sujet), une variante des BDD particulièrement adaptée à la combinatoire. Un ZDD représente un ensemble d'ensembles sous la forme d'un DAG. Comme pour les BDD, on a unicité de la représentation et partage maximal des sous-arbres, ce qui permet notamment une mémoïzation efficaces des opérations sur les ZDD.

La partie I introduisait le concept de ZDD et contenait quelques questions visant à familiariser le candidat avec cette structure de données. La partie II introduisait alors un type de données pour représenter les ZDD et consistait à écrire plusieurs opérations élémentaires sur les ZDD. Le principe de mémoïzation était introduit dans la partie III, en supposant une structure de table d'association. Puis la partie IV appliquait les ZDD au dénombrement du nombre de solutions d'un problème de pavage, à savoir le pavage d'un échiquier par des dominos 2×1. La partie V introduisait la notion de table de hachage, en écho à la partie III. Enfin la dernière partie expliquait comment garantir efficacement le partage maximal lors de la construction des ZDD.

2. Remarques générales

Il s'agissait de la deuxième édition d'une épreuve écrite commune pour les concours d'admission de l'École Polytechnique, et des Écoles Normales Supérieures.

Les notes des 732 candidats français se répartissent selon le tableau suivant, avec une moyenne de 8,13 et un écart-type de 3,38.

$0 \leqslant N < 4$	86	11,7 %
$4 \leqslant N < 8$	266	36,3 %
$8 \leqslant N < 12$	287	39,2 %
$12 \leqslant N < 16$	81	11,1 %
$16 \leqslant N \leqslant 20$	12	1,6 %
Total	732	100 %
Nombre de copies : 732		
Note moyenne : 8,13		
Écart-type : 3,38		

Concernant les questions de programmation, presque toutes les fonctions demandées pouvaient être écrites en moins de 10 lignes, indépendamment du langage de programmation choisi. Les candidats doivent être conscients du fait qu'une réponse longue doit être expliquée en détail et que très souvent un programme très long contient un grand nombre d'erreurs.

Pour obtenir la note maximale, il était nécessaire de traiter le problème en entier.

3. Commentaires détaillés

Pour chaque question, sont indiqués entre crochets le pourcentage de candidats ayant traité la question et le pourcentage de candidats ayant obtenu la totalité des points.

Partie I

Question 1 [100% - 67%]. Cette question élémentaire avait pour but d'aider les candidats à comprendre la structure d'arbre combinatoire. Elle a été incorrectement traitée par de nombreux candidats.

Question 2 [100% - 93%]. Là encore, il s'agissait d'une question élémentaire visant à aider les candidats dans leur compréhension de la structure d'arbre combinatoire. D'une façon assez surprenante, elle a été bien mieux traitée que la précédente.

Question 3 [100% - 50%]. La preuve par récurrence est facile, mais le plus souvent mal rédigée. Plusieurs candidats ont oublié de vérifier la condition de suppression avant d'appliquer l'hypothèse de récurrence. Ceux qui ont fait le raisonnement en descendant le long de la branche droite ont souvent oublié de mentionner l'argument de finitude de l'arbre.

Question 4 [82% - 14%]. La réponse correcte de 2^{2n} a été proposée par beaucoup de candidats ; en revanche, peu l'ont justifiée correctement en montrant la bijection entre les

parties de $\mathcal{P}(E)$ et les arbres combinatoires. Certains candidats pensent que $2^{2n} = 4^n$. Certains candidats, très peu nombreux, ont traité correctement la question par récurrence.

Partie II

Question 5 [97% - 64%]. Cette question est assez facile à résoudre en descendant bien le long de la branche droite. Les candidats n'ayant pas fait ce choix ont souvent fait des erreurs. De plus, quelques candidats ont été troublés par le fait qu'il fallait renvoyer « un élément arbitraire ». Ils ont confondu arbitraire et aléatoire. Quelques candidats ont proposé des réponses avec une énumération de tous les éléments puis extraction d'un élément, ce qui ne respectait pas la complexité demandée.

Question 6 [98% - 79%]. Cette question a été dans la plupart des cas très bien traitée (bien mieux que la question précédente).

Question 7 [91% - 26%]. Cette question est plus difficile. L'erreur la plus fréquente est sans doute le fait d'omettre que la liste vide peut appartenir à un arbre non réduit à une feuille. De nombreux candidats ont proposé des algorithmes explorant récursivement tous les nœuds de l'arbre, qui ne respectaient donc pas la complexité demandée.

Question 8 [87% - 73%]. Il est à noter que l'énoncé de cette question contenait une erreur : il n'est pas possible de garantir une complexité $O(card(S(A)))$. Les correcteurs en ont tenu compte dans la notation et, autant qu'ils ont pu en juger, aucun candidat n'a été gêné par cette erreur. Cette question a été correctement traitée dans la majorité des copies.

Partie III

Question 9 [98% - 74%]. Cette question élémentaire avait pour but d'aider les candidats à appréhender la notion de taille d'un arbre combinatoire. Elle a été correctement traitée dans la majorité des copies, les réponses incorrectes les plus fréquentes étant 5 ou 11.

Question 10 [67% - 42%]. Cette question constitue la première application, la plus simple, du principe de mémorisation. Dans la majorité des cas où elle a été traitée, elle l'a été correctement. L'erreur la plus fréquemment relevée est la création d'une table d'association à *chaque* appel récursif, au lieu d'utiliser une seule table commune à tous les appels récursifs.

Question 11 [42% - 1%]. Il est à noter que l'énoncé de cette question contenait une erreur : il n'est pas possible de garantir une complexité $O(T(\texttt{inter}(A_1, A_2)))$. Les correcteurs en ont tenu compte dans la notation et, autant qu'ils ont pu en juger, aucun candidat n'a été gêné par cette erreur. Cette question est plus difficile que la précédente. De nombreux candidats l'ont traitée de façon partiellement correcte, extrêmement peu de façon entièrement correcte. Parmi les erreurs les plus fréquemment commises, on note

– la non-application du principe de mémorisation,
– les erreurs sur les cas de base, et
– la non-vérification du respect de la condition de suppression dans l'appel récursif.

Question 12 [21% - 2%]. Cette question a été très peu traitée. Dans la plupart des cas, les candidats ont tenté une preuve par récurrence, sans succès. Très peu de candidats ont raisonné sur la base du code de la fonction donnée en question 11 ; ceux qui le font traitent en général correctement cette question. Aucun candidat n'a donné de preuve directe de l'inégalité proposée.

Partie IV

Question 13 [95% - 69%]. Cette question élémentaire a pour but de visualiser l'exercice proposé en partie IV. Elle est en général correctement traitée. L'erreur la plus fréquente consiste à donner une valeur numérique, au lieu d'exprimer le résultat en fonction de p. Certains candidats ont oublié de simplifier l'expression polynomiale obtenue ; il ne leur en a pas été tenu rigueur.

Question 14 [42% - 1%]. Il s'agissait probablement là de la question la plus difficile du sujet. Quelques candidats ont compris la structure de l'arbre qu'il s'agissait de construire mais très peu sont parvenus à garantir la complexité $O(\texttt{n})$. Pour ce type de question, nous recommandons aux candidats de commenter les solutions proposées, soit sous la forme de commentaires dans le programme, soit en décrivant brièvement l'algorithme, ces deux solutions n'étant pas exclusives.

Question 15 [42% - 2%]. Cette question contient deux parties, avec un code de fonction à rédiger, et une étude de complexité. Lorsqu'elle a été traitée, le code de la fonction de pavage était en général correct. L'étude de complexité a été très peu traitée, et en général incorrectement.

Partie V

Question 16 [87% - 64%]. Cette question élémentaire sur les tables de hachage a été généralement correctement traitée. Les erreurs les plus communes consistent, soit à remplacer la case du tableau par le couple (clé, valeur) plutôt que d'ajouter ce couple à la liste déjà présente dans la case, soit à renvoyer cette liste (mise à jour) sans modifier la case du tableau.

Question 17 [86% - 18%]. Cette question a été relativement bien traitée, mais très peu de candidats ont pensé à utiliser la fonction `egal` pour comparer les clés. Certains candidats ont commis l'erreur de parcourir tout le tableau.

Question 18 [84% - 19%]. Même remarque concernant la fonction `egal` que pour la question précédente. Même remarque également concernant le parcours du tableau.

Question 19 [66% - 7%] . Cette question a été assez souvent traitée mais rarement complètement correctement. Certains candidats confondent complexité en espace et complexité en temps. D'autres demandent de façon inutilement restrictive l'injectivité de la fonction `hache`.

Partie VI

Question 20 [74% - 51%]. Cette question est facile et pourtant traitée incorrectement dans de nombreuses copies. L'erreur la plus courante a été l'oubli des cas de base.

Question 21 [24% - 2%]. Cette question a été très peu traitée. Dans la plupart des cas, la solution proposée était irréaliste. De nombreux candidats confondent l'identifiant unique d'un arbre et son image par la fonction de hachage.

Question 22 [5% - 0%]. Cette question, en deux parties, a été très peu traitée. Lorsqu'elle l'a été, la première partie était en général correcte, à l'inverse de la seconde.