ÉCOLE POLYTECHNIQUE CONCOURS D'ADMISSION 2014
ENS : ULM, CACHAN, LYON, RENNES

Composition d'Informatique (4 heures), Filière MP
(XULCR)

Rapport de MM. Jean-Christophe FILLIÂTRE, Paulin de NAUROIS, Steve OUDOT et Mme Stéphanie DELAUNE, correcteurs.

1. L'épreuve

Il s'agissait dans ce problème d'étudier la structure d'arbre binaire dit *croissant*, qui permet de réaliser des files de priorité. L'épreuve alternait des questions relatives à la compréhension de cette structure de données, des questions d'algorithmique et de programmation, ainsi que des questions portant sur l'analyse de la complexité des fonctions programmées. Elle amenait le candidat à comprendre la notion de *complexité amortie*.

La partie 1 avait pour but de réaliser quelques fonctions simples sur les arbres croissants et de s'intéresser à leur dénombrement. L'objectif était de permettre au candidat de se familiariser avec cette nouvelle structure de données. La partie 2 commençait par introduire une opération essentielle sur ces arbres : la *fusion*. Après quelques questions sur cette opération, le candidat était amené à l'utiliser pour programmer quelques fonctions élémentaires (*e.g.* `ajoute`, `supprime_minimum`, `ajouts_successifs`). Enfin, les questions 9 et 10 avaient pour but de mettre en évidence des comportements « extrêmes » lors d'ajouts successifs pour des séquences particulières. La partie 3 portait sur l'analyse des performances de ces opérations. Le candidat était guidé pour effectuer une analyse fine de la complexité des fonctions proposées. La difficulté résidait dans le fait qu'une opération de fusion peut avoir un coût linéaire, mais que ce pire cas ne peut se produire plusieurs fois de suite, si bien que n opérations de fusion successives peuvent être réalisées en $O(n \log n)$. On parle alors de *complexité amortie*. Enfin, la partie 4 utilisait la structure d'arbres croissants pour résoudre un problème bien connu des candidats : le problème du tri.

2. Remarques générales

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

Les notes se répartissent selon le tableau suivant, avec une moyenne de 9,67 et un écart-type de 3. Le nombre de notes éliminatoires, c'est-à-dire inférieures à 2, est de 4.

$0 \leqslant N < 4$	7	0,97 %
$4 \leqslant N < 8$	186	25,66 %
$8 \leqslant N < 12$	370	51,03 %
$12 \leqslant N < 16$	140	19,31 %
$16 \leqslant N \leqslant 20$	22	3,03 %
Total	725	100 %
Nombre de copies : 725		
Note moyenne : 9,93		
Écart-type : 2,91		

Concernant les neuf questions de programmation, les fonctions demandées pouvaient être écrites en moins de 5 lignes par fonction en moyenne, indépendamment du langage de programmation choisi. Les candidats doivent être conscients du fait qu'une réponse longue doit 1 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. Commentaire détaillé

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.

En moyenne les neuf premières questions étaient assez faciles, les suivantes nettement plus difficiles.

Partie I

Question 1 [100% - 98%]. Question très facile et dans l'ensemble bien traitée. Quelques candidats n'ont pas exploité la structure d'arbre *croissant* et ont proposé des solutions récursives inutilement compliquées.

Question 2 [100% - 49%]. Beaucoup de candidats appellent incorrectement la fonction minimum sur un arbre vide.

Question 3 [91% - 59%]. Cette question est dans l'ensemble assez bien traitée.

Partie II

Question 4 [100% - 93%]. Cette question élémentaire avait pour but d'aider le candidat à comprendre l'opération de fusion. Elle a été bien traitée dans l'ensemble même si quelques étourderies ont mené les candidats à proposer un résultat erroné.

Question 5 [98% - 324%]. Beaucoup de récurrences mal posées, telles que « par induction structurelle » ou encore « par récurrence sur les deux arbres ». Il faut être précis, en disant par exemple « par récurrence sur $|t_1| + |t_2|$ ».

Question 6 [100% - 96%]. Question très facile et très bien traitée.

Question 7 [99% - 94%]. De même, question très facile et très bien traitée.

Question 8 [99% - 80%]. Là encore, question facile. Des solutions récursives et d'autres avec des boucles `for`. Très peu d'erreurs.

Question 9 [91% - 74%]. La plupart des candidats ont correctement identifié qu'une suite décroissante répondait à la question.

Question 10 [85% - 2%]. Bon nombre de candidats ont identifié le phénomène de remplissage des arbres. Certains en ont déduit correctement la hauteur, mais quasiment aucun candidat n'a su en fournir une démonstration rigoureuse.

Partie III

Question 11 [94% - 49%]. Certains candidats écrivent un code qui n'est pas de complexité linéaire car il appelle une fonction `taille` qui ne s'exécute pas en temps constant (ce qui n'empêche pas certains d'écrire « la complexité est bien en $|t|$ car on fait un test à chaque noeud »).

La plupart des candidats ont l'idée, correcte, d'écrire une fonction qui calcule à la fois la taille et le potentiel. Il y a cependant parfois de petites erreurs dans les détails.

Question 12 [40% - 5%]. Question peu traitée. Même remarque qu'à la question 5 concernant la façon de poser la récurrence.

Question 13 [70% - 6%]. Parmi les candidats ayant abordé cette question, une majorité a identifié le téléscopage mais les détails du calcul menant au résultat ont rarement été faits avec soin.

Question 14 [42% - 2%]. Certains candidats se contentent d'une réponse pour une valeur particulière de n (par exemple $n = 4$). Un bon nombre de candidats reprennent la suite décroissante qu'ils avaient proposée à la question 9, en lui ajoutant un dernier élément plus grand que tous les autres, ce qui ne convient pas ici. Enfin, assez peu de candidats ont donné une justification pour la notion de complexité amortie.

Question 15 [43% - 7%]. Il s'agissait d'identifier un téléscopage. Celui-ci, plus subtil qu'à la question 13, a été moins souvent identifié.

Partie IV

Question 16 [82% - 32%]. Certains écrivent un tri rapide ou un tri fusion, alors que l'énoncé disait « En utilisant la structure d'arbre croissant... ».

Question 17 [42% - 1%]. Très peu de candidats ont représenté graphiquement la construction des t_i^j. Cela permettait de comprendre le télescopage à l'œuvre dans cette question.

Question 18 [55% - 11%]. Beaucoup de candidats supposent l'existence d'une fonction log sur les entiers. Une telle fonction n'étant pas fournie par le langage, il faut la définir. On notera par ailleurs qu'il existe plusieurs solutions pour cette question n'exigeant pas le calcul de $\log(n)$. D'autres candidats supposent l'existence d'une variable `k`, représentant la valeur k de l'énoncé, alors que la fonction demandée ne prend que le tableau en argument.

Question 19 [36% - 7%]. Certains candidats proposent de compléter le tableau avec une valeur particulière pour la retirer ensuite de l'arbre obtenu. C'est une solution, mais encore faut-il choisir correctement cette valeur (par exemple, une valeur strictement plus grande que tous les éléments du tableau) et justifier qu'on peut la retirer en temps linéaire.