Rapport de jury de l'épreuve orale d'informatique fondamentale LCR, concours I/MPI 2018

Coefficients (en pourcentage du total d'admission) :
Concours I : Lyon (toutes options) 12,7% - Cachan 13,2% - Rennes 8,6%
Concours MPI : Lyon (option MI) 10,8% - Cachan et Rennes (toutes options) 11,5%

Membres du jury : S. Collange, B. Delahaye, V. Jugé

L'épreuve orale d'informatique fondamentale concernent les candidats aux ENS de Lyon, Paris-Saclay, et Rennes des concours MPI et Informatique. Cette année, 246 candidats ont participé à l'épreuve. Leurs notes sont comprises entre 4 et 19, avec une moyenne de 11,9 et un écart-type de 3,6. L'histogramme de la figure 1 présente la distribution des notes.

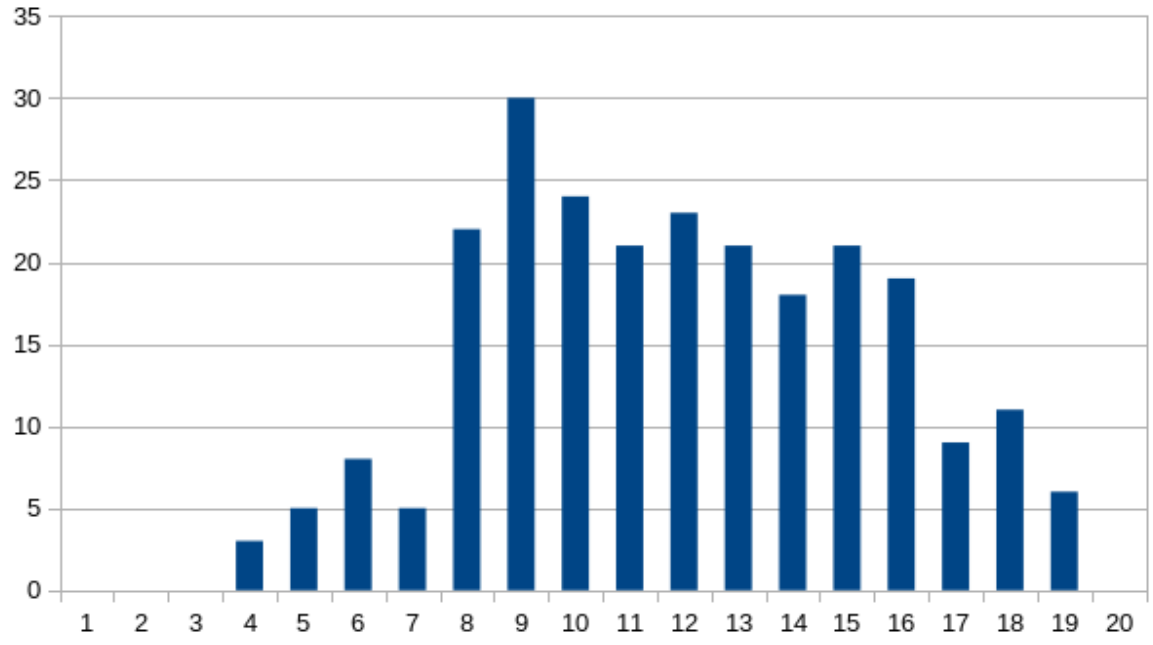

FIGURE 1 – Histogramme des notes de l'épreuve

Après avoir reçu un sujet, les candidats disposaient de 30 minutes de préparation, suivi de 30 minutes d'interrogation devant un des examinateurs. Le jury a proposé 29 sujets originaux, dont trois exemples représentatifs sont présentés en annexe.

Chaque sujet débute par un énoncé introduisant un problème d'informatique et présentant les notations, et comporte une dizaine de questions de difficulté globalement croissante. Une ou deux premières questions relativement faciles d'accès permettent aux candidats de vérifier leur compréhension de l'énoncé et de « se lancer » dans l'interrogation orale. Les questions suivantes, progressivement plus difficiles, occupent la plus majeure partie du temps de préparation et d'interrogation, et visent à départager les candidats. Enfin, la plupart des sujets terminent par une ou plusieurs questions considérées comme très difficiles. En général, il n'est pas attendu des candidats qu'ils traitent l'ensemble des questions dans le temps imparti.

L'évaluation de l'épreuve était basée sur la progression des candidats dans les questions, mais aussi sur la pédagogie de leur présentation et l'autonomie dont ils ont fait preuve pendant l'oral. À ce sujet, le jury a été favorablement impressionné par la pédagogie des exposés de nombreux candidats.

Les sujets font appel aux différentes compétences nécessaires en science informatique : comprendre des concepts nouveaux, démontrer des résultats théoriques, et construire des solutions techniques telles que des algorithmes. Si l'épreuve d'informatique *fondamentale* met l'accent sur les concepts théoriques de l'informatique, on veille néanmoins à garder a l'esprit ce que représentent concrètement les objets que l'on étudie. Un des sujets demandait par exemple de réaliser un automate au moyen d'un circuit en matériel, une question relativement simple à traiter dès lors que l'on comprend que la représentation graphique usuelle d'un automate est précisément le schéma d'une telle machine, d'où son nom d'*automate*.

Le jury a constaté avec satisfaction que la plupart des candidats sont extrêmement bien préparés aux questions techniques sur les automates et langages. Cependant, d'autres domaines du programme d'informatique ne sont pas toujours aussi bien maîtrisés. En particulier, beaucoup de candidats ne pensent pas à employer d'autres structures de données que les tableaux et les listes simplement chaînées, alors que les dictionnaires et files de priorité sont explicitement au programme. Le programme des classes préparatoires ne doit pas non plus faire oublier les résultats élémentaires sur la représentation et de manipulation des nombres, à commencer par les algorithmes d'addition, multiplication, et division étudiés à l'école primaire.

Le jury adresse les conseils suivants aux candidats.

— Si le principe même d'un concours nous impose de faire un classement et que l'épreuve orale peut sembler intimidante, nous tenons à rappeler que *tous* les candidats admissibles ont un excellent niveau. Les candidats ne doivent en aucun cas se dévaloriser ou s'auto-censurer sur la base de leur parcours, leurs notes aux classes préparatoires, ou de ce qu'ils estiment être leur position dans le classement !

— L'objectif du jury est d'amener chaque candidat à réussir à traiter au mieux le maximum de questions. Il est donc dans l'intérêt des candidats d'écouter les demandes et les indications des examinateurs et d'en tenir compte.

— Lors de la préparation, penser à lire l'ensemble du sujet pour comprendre la direction générale de l'exercice et identifier des questions éventuellement plus simples à traiter. Lors de l'interrogation, ne pas hésiter à proposer à l'examinateur de sauter ou remettre à plus tard des questions pour avoir le temps de présenter l'ensemble des résultats préparés. Plusieurs candidats cette année sont arrivés au terme des 30 minutes imparties avant d'avoir pu présenter toutes leurs solutions.

— Deux écueils à éviter sont d'une part, de traiter de manière trop superficielle les démonstrations et les algorithmes, en passant à côté de points techniques importants, et inversement, de détailler excessivement les réponses de manière trop formelle et manquer de temps pour traiter suffisamment de questions. Nous conseillons aux candidats de soigner la forme des réponses aux premières questions, plus faciles, quitte à prendre progressivement de la hauteur pour se concentrer sur le fond dans les questions difficiles. Dans tous les cas, présenter brièvement la démarche ou l'intuition suivie avant de se lancer dans une démonstration, un calcul ou un algorithme.

En annexe, nous présentons trois exemples de sujet représentatifs de cette année :

— Mots significatifs

— À la recherche du système de numération optimal

— Tests de primalité

Mots significatifs

Soit Σ un alphabet fini. Un *mot* fini w sur Σ est une suite finie de lettres : $w = a_1 \dots a_n$, $a_i \in \Sigma$. On note Σ^* l'ensemble des mots finis sur Σ. Le mot vide est le mot ne contenant pas de lettres. On note ce mot ε. Etant donné un mot fini $w = a_1 \dots a_n$, on note $|w| = n$ la longueur du mot w. Un *facteur* (resp. *préfixe*, resp. *suffixe*) de w est un mot v tel qu'on a une décomposition $w = uvu'$ (resp. $w = vu'$, resp. $w = uv$) où u et u' sont des mots finis sur Σ. Le facteur (resp. préfixe, suffixe) est *strict* si et seulement si $w \neq v$. On donne de plus le lemme de l'étoile :

Lemme (de l'étoile) :
Si L est un langage rationnel sur l'alphabet Σ, alors il existe $N \in \mathbb{N}$ tel que tout mot $w \in L$ de longueur $|w| \geq N$ possède une factorisation $w = xyz$ telle que $0 < |y| \leq N$ et

$$\forall n \in \mathbb{N}, xy^n z \in L.$$

Soit $p : \Sigma \to \mathbb{N}$ une application qui donne des *poids* aux lettres de Σ. Dans la suite, on notera Σ_k l'ensemble des lettres de poids $k \in \mathbb{N}$. On peut étendre p à Σ^* de la façon suivante : $p(\varepsilon) = 0$ et pour tout mot $w = a_1 \dots a_n, p(w) = \sum_{i=1}^{n} p(a_i)$.

Définition : *Mots significatifs*
Les mots *significatifs* sur Σ sont définis inductivement :

1. Si $a \in \Sigma_0$, alors le mot $w = a$ est significatif.
2. Si $a \in \Sigma_k$ pour $k \geq 1$ et si $w_1, \dots, w_k$ sont des mots significatifs, alors le mot $w = aw_1 \dots w_k$ est significatif.

Question 1 *Soient un alphabet $\Sigma = \{a, b, c, d\}$ et une application de poids p telle que $p(a) = 0$, $p(b) = 1$, $p(c) = 2$ et $p(d) = 3$. Donner des exemples de mots significatifs de longueurs $1, 2, 3, 4$ et calculer leurs poids.*

Question 2 *En supposant qu'il existe des mots significatifs de longueur $n \geq 0$, que peut on dire de leurs poids ? Justifier.*

Question 3 *Proposer des conditions nécessaires et suffisantes pour que le langage des mots significatifs sur Σ soit (a) non vide, et (b) infini.*

Question 4 *En supposant que $\Sigma_0 \neq \emptyset$, démontrer que le langage L des mots significatifs sur Σ est rationnel si et seulement si $\Sigma = \Sigma_0 \cup \Sigma_1$.*

Définition : *Mots équilibrés*
On dit qu'un mot $w \in \Sigma^*$ est *équilibré* si

1. $|w| = p(w) + 1$, et
2. pour tout préfixe strict v de w, on a $|v| \leq p(v)$.

Question 5 *Démontrer que tout mot significatif est équilibré.*

Question 6 *Démontrer qu'en tout point d'un mot équilibré commence un et un seul facteur équilibré.*

Question 7 *Démontrer qu'un mot est significatif si et seulement si il est équilibré.*

Question 8 *Combien y-a-t il de mots significatifs de longueur n sur l'alphabet Σ (pour une fonction de poids donnée) ?*

À la recherche du système de numération idéal

Le but de cet exercice est de trouver des systèmes de représentations de nombres entiers consécutifs ayant des propriétés avantageuses pour la mémorisation et le calcul.

Considérons une représentation d'un intervalle d'entiers positifs en numération de position en base β, où chaque chiffre est représenté en unaire. Les chiffres appartiennent à un ensemble Σ de β éléments. Nous utilisons initialement $\Sigma = \{0, 1 \dots \beta - 1\}$.

Dans ce système, un entier naturel écrit $\sum_{k=0}^{p} a_k \beta^k$ avec $a_k \in \Sigma$ est représenté par la concaténation des représentations unaires de ses chiffres

$$u(a_{p-1})u(a_{p-2}) \dots u(a_1)u(a_0),$$

où $u(k)$ est une chaîne de β bits comprenant un 1 en position k et un 0 à toutes les autres positions.

Question 1 *Comment s'écrit 17 avec $\beta = 4$? Avec $\beta = 3$?*

Question 2 *Quel est le plus grand nombre représentable avec p chiffres dans ce système ? Combien de bits sont-ils nécessaires pour représenter un nombre à p chiffres ?*

Question 3 *Quelle est la base β_{opt} qui minimise le nombre de bits nécessaire pour représenter des nombres d'une grandeur fixée ?*

.

On ne se limite plus aux chiffres positifs, et on introduit un nouvel ensemble de chiffres signés consécutifs $\Sigma' = \{h - \beta_{\text{opt}} + 1, h - \beta_{\text{opt}} + 2 \dots h\}$ avec $h = \lfloor \frac{\beta_{\text{opt}}}{2} \rfloor$. On pourra noter $\overline{1}$ le chiffre -1, $\overline{2}$ le chiffre -2, etc.

Question 4 *Décrire le système de numération à chiffres dans Σ' en base β_{opt}. Comment calculer l'opposé d'un nombre ?*
Montrer que l'arrondi au multiple le plus proche d'une puissance donnée de la base ($c \cdot \beta_{\text{opt}}^n$ pour n fixé) ne peut pas être ambigü dans ce système, et proposer une façon simple de le calculer.

Question 5 *Proposer des algorithmes d'addition et de multiplication dans le système proposé. Proposer un algorithme de division Euclidienne dont le reste est dans l'intervalle $[-\lfloor \beta/2 \rfloor, \lfloor \beta/2 \rfloor]$. Les illustrer sur les calculs respectifs de $7 + 5$, 2×2, et $5 \div 2$.*

Question 6 *On conserve la même base β_{opt} mais on s'autorise à compléter Σ' par des chiffres supplémentaires, positifs ou négatifs.*
Proposer un algorithme d'addition de deux nombres $a = \sum_{k=0}^{p} a_k \beta^k$ et $b = \sum_{k=0}^{p} b_k \beta^k$ renvoyant $r = \sum_{k=0}^{p} r_k \beta^k = a + b$ tel que chaque retenue sortante c_k au rang k puisse se calculer uniquement à partir des chiffres a_k, b_k des entrées de même rang, c'est-à-dire $\exists f \mid c_k = f(a_k, b_k)$.

Question 7 *Considérons un système en base 2 avec chiffres signés, toujours en partant de l'ensemble de chiffres Σ'. On ajoute la contrainte que le nombre zéro admet une représentation unique. Proposer un algorithme d'addition tel que chaque retenue sortante ne dépende que des chiffres du même rang et du rang immédiatement précédent, c'est-à-dire $\exists f \mid c_k = f(a_k, b_k, a_{k-1}, b_{k-1})$.*

Nous conserverons ce dernier système jusqu'à la fin de l'exercice.

Question 8 *Adapter l'algorithme de multiplication pour le nouveau système. Pour des entrées suivant une distribution uniforme codées en binaire traditionnel, quel est le nombre minimal et le nombre moyen de multiplications d'un chiffre par zéro dans l'algorithme de multiplication proposé ?*

Question 9 *Proposer un algorithme de recodage des entrées du multiplieur qui maximise le nombre de zéros dans les produits partiels.*

Tests de primalité

Le but de ce problème est d'établir des tests de primalité efficaces, c'est-à-dire de tester si un entier n est premier.

On considérera systématiquement que les quatre opérations arithmétiques de base ($+$, $-$, $\times$ et $\div$) ont été implémentées en suivant les algorithmes vus à l'école primaire. Par exemple, l'addition de deux nombres tenant respectivement sur k et ℓ bits prend un temps $\mathcal{O}(k+\ell)$.

Question 1. Parmi les entiers ci-dessous, lesquels sont des nombres premiers ?

$$2 \qquad 11 \qquad 167 \qquad 168 \qquad 561 \qquad 729 \qquad 1105 \qquad 10201.$$

Question 2. Donner un algorithme permettant de tester en temps $\mathcal{O}(\sqrt{n}\log(n)^2)$ si un nombre n est premier.

On s'intéresse maintenant à un algorithme potentiellement plus efficace, dû à Fermat. Il consiste à choisir des entiers $a_1, \ldots, a_k$ au hasard dans l'ensemble $\{0, \ldots, n-1\}$, puis à vérifier si $a_i^n \equiv a_i \pmod{n}$ pour tout $i \leqslant k$, et à déclarer que n est composé si l'égalité en question n'est pas systématiquement vérifiée. Sinon, on déclare que n est *probablement* premier.

Question 3. Quelle est, en fonction de k et de n, la complexité de cet algorithme ? Est-il possible que l'on déclare que n est composé alors que n est premier ?

Question 4. Soit $n \geqslant 1$ un entier fixé. On suppose que, quels que soient les entiers a_i choisis, l'algorithme indique que n est probablement premier. L'entier n est-il nécessairement premier ? *On pourra s'intéresser aux entiers donnés en question* 1.

L'algorithme de Fermat nous laissant insatisfait, on s'intéresse maintenant à la variante proposée en page suivante, due à Miller et Rabin, qui permet de tester si un entier n est *probablement* premier.

Question 5. Quelle est la complexité de cet algorithme ? Est-il possible que l'on déclare que n est composé alors que n est premier ?

Question 6. Soit n un entier et a et b deux entiers premiers avec n, d'ordres ω_a et ω_b dans $(\mathbb{Z}/n\mathbb{Z})^*$. Montrer que, si ω_a et ω_b sont premiers entre eux, alors ab est d'ordre $\omega_a\omega_b$.

Question 7. Soit p un nombre premier.

1. Montrer que le groupe multiplicatif $(\mathbb{Z}/p\mathbb{Z})^*$ admet un élément d'ordre $p-1$.
2. Montrer que le groupe multiplicatif $(\mathbb{Z}/p^2\mathbb{Z})^*$ admet un élément d'ordre $p(p-1)$.

Question 8. Soit n un entier impair composé. Montrer que l'algorithme de Miller et Rabin a une probabilité $p \leqslant 1/2^k$ d'indiquer que n est probablement premier.

L'algorithme de Miller et Rabin permet de tester rapidement si un entier est probablement premier, mais on souhaiterait être capable de construire des *témoins* permettant de certifier rapidement qu'un entier est premier.

Par exemple, un témoin permettant de certifier rapidement qu'un entier n n'est pas premier pourrait être une paire d'entiers (a, b) tels que $2 \leqslant a \leqslant b$ et $n = a \times b$. Si n est un entier à k bits, c'est-à-dire tel que $1 \leqslant n < 2^k$, on dispose donc d'un certificat de $2k$ bits de long et permettant de vérifier en temps $\mathcal{O}(k^2)$ que n n'est pas premier.

Question 9. Soit n un nombre premier à k bits. Montrer qu'il existe un certificat de $\mathcal{O}(k^2)$ bits permettant de vérifier en temps $\mathcal{O}(k^5)$ que n est premier.

Algorithme 1 Test probabiliste de primalité, dû à Miller et Rabin

Arguments : n, k ▷ Entier n à tester, nombre k d'échantillons choisis

$\ell \leftarrow 0$
$m \leftarrow n - 1$
tant que $m \equiv 0 \pmod 2$ **faire**
    $\ell \leftarrow \ell + 1$
    $m \leftarrow m/2$
fin tant que
pour $i = 1$ à k **faire**
    $a \leftarrow$ entier choisi au hasard entre 1 et $n - 1$ inclus
    $a \leftarrow a^m \pmod n$
    $\ell' \leftarrow \ell$
    si $a \not\equiv 1 \pmod n$ **alors**
        tant que $a \not\equiv -1 \pmod n$ et $\ell' \neq 0$ **faire**
            $\ell' \leftarrow \ell' - 1$
            $a \leftarrow a^2 \pmod n$
        fin tant que
        si $a \not\equiv -1 \pmod n$ **alors**
            déclarer que n est composé !
        fin si
    fin si
fin pour
déclarer que n est probablement premier !