Rapport de l'épreuve orale d'informatique fondamentale LCR,
concours I/MPI 2016, ÉNS de Cachan, ÉNS de Lyon et ÉNS de Rennes
Membres du jury : B. Bollig, V. Gripon, G. Naves

Ce document concerne l'épreuve orale d'informatique fondamentale LCR des écoles normales supérieures de Cachan, Lyon et Rennes, dans les concours MPI et I. Pour l'année 2016, 218 candidats ont été interrogés. L'exercice consiste en un oral de 45 minutes sans préparation. Les notes sont étalées entre 5 et 18, avec une moyenne de 11.23 et un écart-type de 2.42. Un histogramme est donné en fin de document.

Le jury a proposé plus de 40 sujets originaux cette année, typiquement conçus de la façon suivante :

— un préambule introduisant un concept nouveau pour le candidat,
— une ou plusieurs questions permettant au candidat de s'assurer qu'il a compris le concept étudié, et à l'examinateur d'évaluer les capacités du candidat à s'approprier des notions nouvelles,
— une succession de questions de difficulté croissante amenant à la démonstration d'un résultat d'informatique fondamentale lié au concept introduit.

Les sujets proposent en tout une dizaine de questions, les dernières questions étant en général considérées comme très difficiles. Il n'est pas attendu d'un candidat, même très bon, qu'il puisse atteindre la fin du sujet.

Si les concepts abordés sont nouveaux, ils reposent souvent sur des notions et résultats qui sont au programme d'informatique, en particulier en algorithmique, graphes, logique, automates et langages. Les candidats commettant des erreurs sur tout point du programme sont sévèrement sanctionnés. Nous conseillons aux candidats de lire intégralement le programme officiel.

Par ailleurs, nous avons constaté que de nombreux candidats ont aussi acquis une culture en informatique dépassant le cadre du programme officiel. Si nous apprécions ces connaissances, lorsqu'elles sont bien maitrisées, nous déplorons trop souvent leur caractère superficiel, cette superficialité s'étendant parfois aussi au contenu du programme. Nous avons eu plusieurs exemples de candidats connaissant les flots ou les colorations en théorie des graphes, hors programme, mais ne maitrisant pas les concepts liés à la connexité dans les graphes, pourtant au programme.

Nous avons aussi detecté des faiblesses typiques à de nombreux candidats :

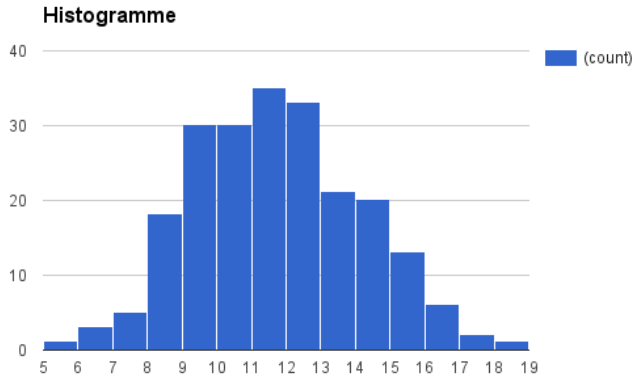
— nous proposons fréquemment des sujets faisant intervenir des probabilités, leur usage est effectivement courant en informatique. Peu de candidats reconnaissent et utilisent correctement les lois binomiales. Certains obtiennent des probabilités divergeant vers $+\infty$ ou devenant négatives, et ne sont pas choqués. Une question proposée dans plusieurs contextes revenait à effectuer k tirages de Bernoulli avec probabilité p. Nous avons obtenu beaucoup de réponses différentes pour la probabilité d'obtenir au moins un événement positif.
— Les définitions inductives sont rarement comprises par les candidats. L'exemple typique au programme est la définition des expressions rationnelles. Certains candidats proposent des preuves par récurrence sur la *longueur* de l'expression rationnelle. D'autres confondent les expressions rationnelles et les langages associés aux expressions rationnelles. Plus généralement, on attend des candidats qu'ils sachent développer des preuves par induction. Les mêmes problèmes se posent avec les expressions en logique propositionnelle, ou lorsque nous proposons une notion nouvelle définie inductivement.
— La fermeture par complémentation des langages rationnels est source de beaucoup de confusion.
— Beaucoup de candidats manquent de recul sur les notions du programme. Le lemme de l'étoile n'étant plus au programme, il serait bon que les candidats connaissent néanmoins

quelques langages qui ne sont pas rationnels. De même, connaître quelques familles classiques de graphes (complets, cycles, chemins, étoiles, bipartis, ...) aiderait les candidats à développer les bonnes intuitions.

— Ce n'est pas parce que l'énoncé contient un entier qu'il faut faire une preuve par récurrence ! En particulier, prouver une propriété de graphe par récurrence sur le nombre de sommets exige qu'il soit possible de supprimer un sommet du graphe en conservant la propriété. C'est parfois une bonne idée, mais pas sur une propriété s'exprimant de façon globale.

Le jury a l'habitude de prendre quelques minutes en fin d'oral pour discuter avec les candidats. Nous constatons régulièrement que certains candidats n'ont que bien peu d'idée de ce qu'est l'informatique fondamentale, et sont peut-être surpris par la nature des sujets de cette épreuve. Le nouveau programme d'informatique a introduit principalement des contenus de programmation. Même des sujets se prêtant à une étude plus théorique, comme les graphes, sont abordés avec un angle algorithmique/programmation très prononcé.

Les départements d'informatique des écoles normales supérieures enseignent principalement les fondamentaux théoriques de l'informatique, en particulier en logique, langage, calculabilité et algorithmique. Ils s'agit de domaines scientifiques, proche des mathématiques discrètes, reposant sur l'établissement de faits prouvés mathématiquement. Les enseignements des écoles normales supérieures ne comportent donc qu'une faible proportion de cours en "informatique appliquée", par exemple en programmation. Par ailleurs les enseignements en informatique dans les trois écoles ne différent que marginalement, l'idée selon laquelle les enseignements seraient plus appliqués à Cachan ou Rennes est fausse. Cette épreuve d'informatique fondamentale prend donc sa place dans la sélection de candidats aptes à étudier les aspects théoriques et fondamentaux de l'informatique.



Ci-après sont donnés trois exemples de sujets représentatifs :

- page 3 : Shuffles
- page 4 : Graphes Saturés
- pages 5-6 : Liste-coloration de graphes

Shuffles

Soit Σ un alphabet. Pour $w \in \Sigma^*$ et $A \subseteq \Sigma$, nous notons $\mathsf{proj}(w, A)$ la *projection* de w sur A, qu'on obtient de w en supprimant toutes les lettres qui ne sont pas dans A. Par exemple, $\mathsf{proj}(abcabc, \{a, b\}) = abab$.

Question 1. Définir la projection formellement.

Pour le reste de ce sujet, nous fixons $k \geq 1$ et $\tilde{\Sigma} = (\Sigma_1, \ldots, \Sigma_k)$, un k-uplet d'alphabets finis, pas nécessairement disjoints. Nous notons $\Sigma = \Sigma_1 \cup \ldots \cup \Sigma_k$.

Soit $L \subseteq \Sigma^*$ un langage. Le *shuffle* de L (par rapport à $\tilde{\Sigma}$) est défini par

$$\mathsf{shuffle}(L) := \{w \in \Sigma^* \mid \forall i \in \{1, \ldots, k\} : \exists u \in L : \mathsf{proj}(w, \Sigma_i) = \mathsf{proj}(u, \Sigma_i)\}\,.$$

Question 2. Soit $\tilde{\Sigma} = (\{a, b\}, \{b, c\})$ et $L = \{acb, bac\}$. Calculer $\mathsf{shuffle}(L)$.

Question 3. Montrer que le shuffle d'un langage rationnel est rationnel.

Soit $L \subseteq \Sigma^*$. Nous disons que L est un *shuffle rationnel* si $L = \mathsf{shuffle}(L)$ et L est rationnel. Nous disons que L est un *semi-shuffle rationnel* s'il existe $m \geq 1$ et des shuffles rationnels $L_1, \ldots, L_m \subseteq \Sigma^*$ tels que $L = L_1 \cup \ldots \cup L_m$.

Question 4. Soit $\tilde{\Sigma} = (\{a, b\}, \{b, c\})$. Montrer que

$$L = (a + b + c)^*(ac + ca)(a + b + c)^*$$

n'est pas un shuffle rationnel.

Question 5. Soit $\tilde{\Sigma} = (\{a\}, \{c\})$. Montrer que la classe des semi-shuffles rationnels (par rapport à $\tilde{\Sigma}$) est strictement plus grande que la classe des shuffles rationnels.

Question 6. Montrer que le langage de la Question 4 n'est pas un semi-shuffle rationnel.

Question 7. Montrer que la classe des shuffles rationnels (pour un $\tilde{\Sigma}$ approprié) n'est pas close sous complément. C'est à dire, il y a un shuffle rationnel $L \subseteq \Sigma^*$ tel que $\Sigma^* \setminus L$ n'est pas un shuffle rationnel.

Question 8. Montrer que la classe des semi-shuffles rationnels est close sous complément.

Graphes saturés

Tous les graphes considérés dans cet exercice sont symétriques (non orientés) et simples (sans boucle et arête parallèle).

Soient $G = (V, E)$ et $F = (V', E')$ deux graphes.

On appelle sous-graphe de G un graphe (V'', E'') où $V'' \subseteq V$ et $E'' \subseteq E \cap (V'' \times V'')$.

Le graphe G est dit *exempt* de F s'il n'existe pas de sous-graphe de G isomorphe à F.

Le graphe G est dit *saturé* par F s'il est exempt de F et que tout graphe $(V, E \cup \{u, v\})$, avec $\{u, v\} \in (V \times V) - E$ n'est pas exempt de F.

Question 1 *Montrer qu'il existe F tel qu'il peut exister plusieurs graphes non isomorphiques, de même ordre, et saturés par F.*

Question 2 *Montrer que pour tout graphe F contenant au moins une arête, il existe un graphe G saturé par F.*

Nous nous concentrons à présent sur le cas où F est un graphe complet à trois nœuds et nous considérons des graphes saturés par F contenant au moins tois nœuds.

Question 3 *Montrer que le nombre minimum d'arêtes dans un graphe saturé par F et contenant n nœuds est $n - 1$.*

Question 4 *Montrer que le nombre maximum d'arêtes dans un graphe saturé par F et d'ordre n est $\lfloor n^2/4 \rfloor$.*

Revenons au cas général où F est un graphe quelconque. Notons à présent $ex(n, F)$ le nombre maximum d'arêtes qu'un graphe d'ordre n saturé par F contient.

Question 5 *Que dire de la monotonie de ex par rapport à chacune de ses composantes (au sens des sous-graphes pour la seconde composante) ?*

Notons $sat(n, F)$ le nombre minimum d'arêtes qu'un graphe d'ordre n saturé par F contient.

Question 6 *Montrer que la fonction sat n'est pas monotone par rapport à sa seconde composante F (au sens des sous-graphes).*

Question 7 *Montrer que la fonction sat n'est pas monotone par rapport à sa première composante n.*

Question 8 *Soit F un graphe. On propose à présent de jouer à un jeu. On commence avec un graphe de n nœuds (au moins autant que dans F) sans arête. Deux joueurs, nommés* shortensa *et* extensa*, ajoutent à tour de rôle des arêtes dans le graphe. Le jeu s'arête lorsque le graphe est saturé. L'objectif pour* shortensa *est de faire en sorte que le graphe saturé contienne un minimum d'arêtes. L'objectif pour* extensa *est de faire en sorte que le graphe saturé contienne un maximum d'arête.*

On appelle stratégie une fonction associant à l'état courrant de la partie le choix de la prochaine arête à ajouter pour un joueur donné.

On appelle stratégie optimale pour shortensa *(resp.* extensa*) une stratégie minimisant (resp. maximisant) le nombre d'arêtes dans le graphe saturé quelques soient les choix de son adversaire.*

Déterminer le nombre d'arêtes dans le graphe saturé obtenu lorsque F est un graphe complet à 3 nœuds en fonction de n et en fonction du joueur jouant le premier, lorsque les deux joueurs utilisent une stratégie optimale.

Liste-coloration de graphes

Soit $G = (V, E)$ un graphe non-orienté simple. Une *coloration propre* de G est une fonction $\phi : V \to \mathbb{N}$ telle que pour tout arc $uv \in E$, $\phi(u) \neq \phi(v)$ (dans ce contexte, les entiers sont appelés *couleurs*). Un graphe $G = (V, E)$ est k-coloriable s'il existe une coloration propre ϕ avec $|\phi(V)| = k$.

Soit un graphe $G = (V, E)$ et pour chaque sommet $v \in V$, soit un sous-ensemble $L(v) \subset \mathbb{N}$ de couleurs (improprement appelé *liste* de couleurs). Le problème de la liste-coloration d'un graphe est de déterminer s'il existe une coloration propre ϕ de G telle que pour tout sommet $v \in V$, $\phi(v) \in L(v)$.

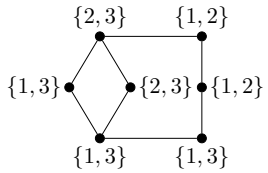


FIGURE 1 – Instance de liste-coloration.

Question 1: Déterminer si l'instance de la Figure 1 est liste-coloriable.

Un graphe G est *k-choisissable* si pour toute famille de listes $(L(v))_{v \in V}$ toutes de longueurs au moins k, (G, L) est liste-coloriable. Dans ce sujet nous nous intéressons à déterminer la choisissabilité de certains graphes.

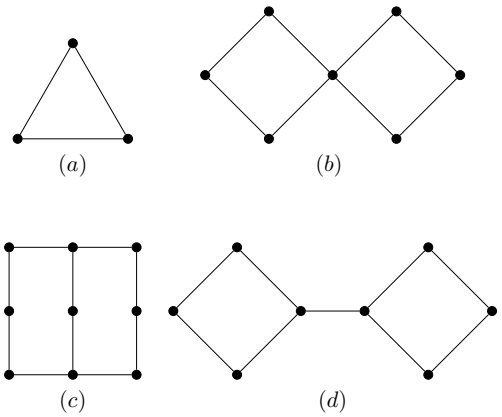


FIGURE 2 – Des graphes non-2-choisissables.

Question 2: Montrer que les graphes de la Figure 2 ne sont pas 2-choisissables.

Un graphe $G = (V, E)$ est *k-dégénéré* s'il est vide ou s'il existe un sommet $v \in V$ de degré au plus k tel que $G' = (V \setminus \{v\}, E \setminus \delta(v))$ est k-dégénéré.

Question 3: Montrer que si G est k-dégénéré, G est $k + 1$-choisissable.

Un graphe est *planaire* si on peut le dessiner dans le plan sans que deux arêtes ne se croisent sauf en une extrémité commune. Une *face* d'un graphe planaire dessiné est une composante connexe de $\mathbb{R}^2 - \text{im}(G)$.

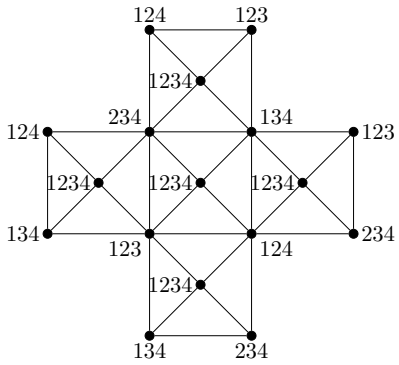


FIGURE 3 – Une instance non liste-coloriable. Pour éviter de surcharger le dessin, les listes sont données sans syntaxe.

Question 4: Le graphe de la Figure 3 est un graphe planaire. Montrer qu'il n'est pas liste-coloriable.

Question 5: Démontrer qu'il existe des graphes planaires 3-coloriables qui ne sont pas 4-choisissables.

Question 6: Montrer que pour tout $k \in \mathbb{N}$ il existe un graphe biparti non k-choisissable.

Un graphe connexe est *extérieurement planaire* s'il est planaire et admet un dessin tel que tous les sommets sont sur le bord de la face infinie.

Question 7: Montrer que les graphes extérieurement planaires sont 3-choisissables.

Question 8: On appelle $\theta_{i,j,k}$ le graphe formé à partir de deux sommets en ajoutant trois chemins sommet-disjoints entre ces deux sommets, de longueurs respectives i, j et k. Figure 2 (c) est $\theta_{4,4,2}$. Montrer que $\theta_{2,2,2k}$ est 2-choisissable pour tout $k \in \mathbb{N}^\star$.

Le *noyau* d'un graphe G est le graphe obtenu en supprimant itérativement tous les sommets de degré 1.

Question 9: Montrer qu'un graphe G connexe est 2-choisissable ssi son noyau est K_1 (le graphe à un seul sommet), C_{2k} (le cycle de longueur $2k$, pour $k \geq 2$) ou $\theta_{2,2,2k}$ (pour $k \geq 1$).

Question 10: Montrer que tout graphe planaire est 5-choisissable.