Rapport relatif à l'épreuve écrite d'Informatique Fondamentale

Banques MP/MPI inter-ENS – Session 2024

- **Écoles partageant cette épreuve** : ENS Ulm, Lyon, Paris-Saclay, et Rennes
- **Coefficients** (en pourcentage du total des points de chaque concours) :
  - Ulm : MP-Info 13,3% et MPI 13,3%
  - Lyon : MP-Info 11,3% et MPI 11,3%
  - Paris-Saclay : MP-Info 13,2% et MPI 13,2%
  - Rennes : MPI 8,3%
- **Membres du jury** : Pierre ABOULKER, Nicolad MARKEY, Marc DE VISME.

L'épreuve écrite d'informatique fondamentale concerne les candidates et candidats aux quatre Écoles Normales Supérieures sur le concours INFO, dans les deux filières MP et MPI. Le nombre de candidats ayant composé était de 165 en MP et 333 en MPI pour la session 2024, contre 225 en MP et 254 en MPI pour la session 2023, et 311 en 2022 [1]. Les notes se sont échelonnées de 0.2 à 20, avec une moyenne de 9.14 et un écart-type de 4.27.

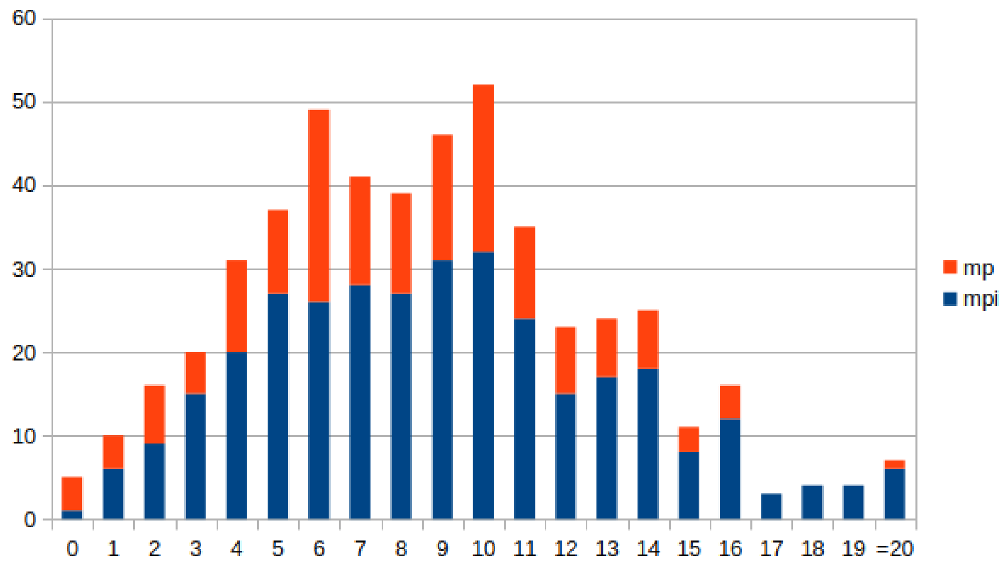


FIGURE 1 – Répartition des notes. Exemple de lecture :
20 copies de MPI et 11 copies de MP ont eu une note comprise dans l'intervalle $[4, 5[$.

1. L'année 2023 introduit la filière MPI pour la première fois, expliquant un important afflux de nouveaux candidats et candidates sur cette épreuve. Le chiffre de 2022 concerne l'épreuve « Maths-Info » du concours INFO de la filière MP.

A L'épreuve

Le sujet de 2024 est consacré à l'étude algorithmique d'un paramètre de graphe appelé couverture minimale.

- Dans la première partie on découvre progressivement certaines propriétés des couvertures minimales, leurs valeurs sur quelques familles de graphes simples et leurs liens avec d'autres paramètres classiques de la théorie des graphes.
- Dans la deuxième partie on développe deux algorithmes efficaces qui calculent un ensemble couvrant minimum d'un arbre.
- Dans la troisième partie on étudie deux heuristiques naturelles calculant un ensemble couvrant s'avérant loin d'être minimum, pour finalement découvrir un algorithme simple calculant une bonne approximation de la couverture minimale.
- Dans la quatrième partie on découvre la complexité paramétrée.
- Dans la cinquième partie on continue notre exploration de la complexité paramétrée en introduisant la notion de noyau.
- Dans la sixième partie on découvre la puissance de l'optimisation linéaire, permettant une autre approximation de la couverture minimale ainsi qu'un noyau plus petit que celui de la partie.

B Commentaires généraux

Le sujet était assez long, mais ne comportait pas de questions particulièrement difficiles. Cependant, il introduisait de nombreuses notions non couvertes par le programme. La principale difficulté résidait souvent dans la compréhension et l'assimilation rapide de ces nouvelles notions.

Notamment, plusieurs candidats et candidates ont pensé à tort reconnaitre des notions qu'ils avaient vues en cours : la notion d'ensemble couvrant du sujet a été parfois confondue avec la notion d'ensemble dominant, la notion d'arbre du sujet a été confondue avec la notion d'arbre binaire, etc.

Ainsi, le jury rappelle l'importance de faire soigneusement les questions introductives, qui visent généralement à rassurer le candidat sur sa bonne compréhension des définitions du sujet.

Beaucoup de copies contenaient, comme réponses à certaines questions, une affirmation sans preuve. Le jury tient à rappeler que chaque affirmation doit être rigoureusement justifiée, que ce soit par une hypothèse de l'énoncé ou par un raisonnement formel s'appuyant éventuellement sur l'un des théorèmes du cours.

La section 2, consacrée à la conception d'algorithmes sur les arbres, a été très mal réalisée. En particulier, il semble que la quasi-totalité des étudiants ne maîtrise pas les principes de la programmation dynamique.

De manière notable, une des questions du sujet, la IV.4, était techniquement fausse. Pour des raisons d'encodage du graphe par matrices d'adjacence, l'algorithme attendu ne pouvait pas s'exécuter en temps linéaire. Si le jury a attribué 0,4pt comme initialement prévu aux copies ayant apporté la réponse (fausse) attendue, les candidats et candidates ayant fait preuve d'un esprit critique remarquable en mettant en doute la faisabilité de la tache demandée ont obtenu entre 0,4pt et 1pt, suivant la pertinence de leur contre-argument.

C Commentaires détaillés

Afin de comparer la réussite à des questions indépendamment de leurs poids respectifs, les moyennes et écart-type par question sont donnés sur 1. Pour chaque question, on indique quatre valeurs :

— le **nombre de points** de la question ;
— le **nombre de copies** ayant obtenu des points sur cette question ;
— la **moyenne sur 1** des notes non-nulles sur cette question ;
— l'**écart-type sur 1** des notes non-nulles sur cette question.

I Découvrir les couvertures (3,2 pt)

Cette partie introductive permettait aux candidats et candidates de se familiariser avec la notion d'ensemble couvrant minimum.

L'importance des justifications : L'énoncé demandait explicitement de « justifier soigneusement » les réponses aux premières questions ; il convenait donc de ne pas se limiter à donner les valeurs demandées, mais d'expliquer synthétiquement les arguments qui conduisent à ces valeurs.

I.1 Premier exemple (0,3 pt | 497 | 0,82 ; 0,23)

Cette question et la suivante permettait de se familiariser avec les différentes notions étudiées dans la suite du sujet. Différents arguments étaient possibles ici, mais il ne suffisait pas de dire simplement qu'il n'existe pas d'ensemble couvrant de taille 3.

I.2 Quatre familles de graphes (1,3 pt | 486 | 0,75 ; 0,23)

Les graphes « chemins » et « cycles » sont ceux qui ont posé le plus de difficultés. L'argument le plus efficace est qu'un sommet couvre au plus deux arêtes, et qu'il faut donc au moins $(n-1)/2$ (et donc au moins $\lceil (n-1)/2 \rceil$) sommets pour couvrir toutes les arêtes de P_n, et au moins $n/2$ sommets pour couvrir les arêtes de C_n.

I.3 Complémentaire d'une couverture (0,6 pt | 393 | 0,85 ; 0,23)

Le complément d'un ensemble couvrant est un ensemble indépendant, et le complément d'un ensemble indépendant est un ensemble couvrant. Plusieurs copies n'ont montré qu'une seule de ces deux propriétés, et donc qu'une seule inégalité.

I.4 Couplages et couvertures (0,5 pt | 306 | 0,96 ; 0,14)

I.5 Premier algorithme exponentiel (0,5 pt | 346 | 0,87 ; 0,23)

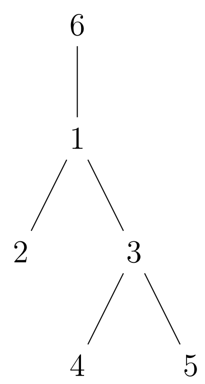
Rien de subtil dans cette question : on énumère les ensembles de sommets (éventuellement, des plus petits aux plus grands) et on teste s'ils sont couvrants.

II Couvrir les arbres (3,2 pt)

Avant de s'attaquer au cas général, cette partie s'attarde sur un cas particulier important : les arbres.

Sur la non-binarité des arbres : Bien que la définition d'un arbre ait été explicitement rappelée par le sujet, plusieurs ont supposé à tort que le sujet se restreignait aux arbres binaires. Les réponses se reposant sur les propriétés des arbres binaires (e.g. parcourir l'ensemble des fils d'un nœud en O(1)) ont été pénalisées, tandis qu'une certaine tolérance a été appliquée envers les réponses qui resteraient correctes dans le cas non-binaire.

Il ne suffisait pas d'alterner : Une idée assez présente est qu'afin de couvrir un arbre « qu'il suffit de prendre un sommet sur deux » en partant de la racine ou d'une feuille. Cela est faux, par exemple l'arbre ci-dessous admet une couverture avec deux sommets adjacents et sinon nécessite au moins trois sommets. Une variation correcte [2] de cette idée est de calculer pour chaque sommet « la plus courte distance entre ce sommet et les feuilles » et d'inclure les impairs dans la couverture, il est donc nécessaire de réaliser un parcours postfix [3] de l'arbre, et non un simple calcul de proche en proche.



1	{2, 3, 6}
2	{1}
3	{1, 4, 5}
4	{3}
5	{3}
6	{1}

II.1 Éviter les feuilles (0,5 pt | 408 | 0,88 ; 0,22)

Une subtilité importante est qu'il *peut* exister une couverture minimale contenant une feuille, et tenter d'argumenter que la couverture minimale ne va « jamais » prendre une feuille est donc faux.

II.2 Parcourir l'arbre (1,2 pt | 201 | 0,57 ; 0,28)

Cette question est surprenamment complexe. Il s'agit d'un parcours postfix [3] de l'arbre où un nœud est à inclure dans la couverture si au moins un de ses fils n'est pas dans la couverture. De manière équivalente, on peut partir des feuilles en ajoutant leur père à la couverture puis en le supprimant de l'arbre, créant ainsi potentiellement de nouvelles feuilles. Dans les deux cas, il est facile de faire une erreur de logique menant à un algorithme faux, ou de complexité quadratique.

II.3 Programmation dynamique (1,5 pt | 149 | 0,56 ; 0,28)

Les erreurs sur cette question sont de trois types :

— Mauvaises formules pour calculer les $D_\in, D_\notin, D$. En particulier, $D_\in$ utilise la valeur de D de ses fils, et non la valeur de $D_\notin$.

— Mauvais parcours de l'arbre.

2. Si l'arbre a au moins un sommet non-feuille, c'est-à-dire au moins trois sommets.

3. C'est-à-dire un parcours en profondeur qui traite les fils d'un nœud avant de traiter le nœud lui-même. L'arbre est ainsi parcouru depuis les feuilles vers la racine en temps linéaire.

— Mauvaise argumentation de la linéarité de la complexité. En effet, à chaque étape du parcours, le temps de calcul n'est pas O(1) mais dépend du nombre de fils. Il fallait argumenter pourquoi ce n'était pas un problème, par exemple en rappelant que le nombre d'arêtes d'un arbre est égal au nombre de sommets moins un.

III Approximer les couvertures (4,4 pt)

Revenant sur le cas général, cette partie s'intéresse à trois tentatives d'approximation du problème.

III.1 L'heuristique est correcte (0,2 pt | 459 | 0,95 ; 0,15)

III.2 L'heuristique n'approxime pas (0,6 pt | 445 | 0,99 ; 0,07)

La plupart des copies ont pris l'exemple du graphe « étoile » (i.e. $K_{1,n-1}$).

III.3 Le glouton sur un exemple (0,6 pt | 462 | 0,88 ; 0,19)

III.4 Le glouton n'approxime pas (1,5 pt | 352 | 0,65 ; 0,24)

Si la question précédente ne demandait pas de justification, celle-ci nécessitait de montrer (par récurrence) que l'algorithme Glouton pouvait sélectionner, dans l'ordre, les sommets de B_n^n, puis B_n^{n-1}, ..., jusque B_n^1. La minoration de Glouton(G_n)/Opt(G_n) n'a généralement pas posé de problème.

III.5 L'approximation est correcte (0,2 pt | 316 | 0,95 ; 0,14)

Il n'était pas indispensable d'utiliser la Question I.3 ici, et la plupart des copies ont donné un argument similaire à celui de la Question III.1 (ce qui a été accepté).

III.6 L'approximation approxime bien (0,8 pt | 209 | 0,94 ; 0,16)

Les arêtes sélectionnées par l'algorithme forment un couplage, donc $|S| \leq 2\mu(G)$. La question I.4 permet de conclure.

III.7 Mais pas si bien que ça (0,5 pt | 280 | 0,95 ; 0,15)

La suite de graphes bi-partis complets $K_{n,n}$ (entre autres) convient. Quelques copies ont remarqué que le graphe $K_{1,1}$ suffisait pour montrer qu'Approx n'est pas une c-approximation pour $c < 2$, certaines allant jusqu'à proposer la suite infinie constante $(K_{1,1})_{i \in \mathbb{N}}$.

IV Algorithme tractable à paramètre fixé (5 pt)

L'objectif de cette partie est d'introduire la notion de complexité paramétrée en cherchant des algorithmes simples en temps FPT.

IV.1 Deuxième algorithme exponentiel (0,3 pt | 294 | 0,93 ; 0,17)

L'algorithme est quasiment le même algorithme que pour Question I.5.

IV.2 Retirer un sommet (0,5 pt | 374 | 0,92 ; 0,19)

Cette question sert de patron pour les Questions IV.5, V.2, et V.3.

IV.3 Premier algorithme FPT (1,3 pt | 333 | 0,81 ; 0,23)

Si le cœur de la question est une récursion à deux branches, les erreurs les plus fréquentes sont sur les cas de bases :

- Le cas de base « Si le graphe n'a plus d'arêtes » est nécessaire, sans quoi l'algorithme plantera en essayant de trouver une arête dans un ensemble vide.
- Le cas de base « $k \leq 0$ » est nécessaire, sans quoi l'algorithme n'est pas FPT.

IV.4 Les graphes de degré deux (1,3* pt | 291 | 0,59 ; 0,25)

Cette question se décompose en deux moitiés. La première moitié consiste à prouver (et non simplement affirmer) que les graphes de degré deux sont des unions disjointes de chemins et cycles. La deuxième moitié consiste en un parcours de graphe linéaire, ou du moins consisterait en un tel parcours si la question était correcte. En effet, contrairement à la Partie II, il est explicitement indiqué dans le sujet que les graphes sont encodés par leur matrice d'adjacence, rendant la tâche impossible en temps linéaire. Ainsi :

- Toute copie fournissant la réponse attendue a reçu le nombre de points prévus.
- Les rares copies mettant en doute la faisabilité de la tache ont été récompensées par des points supplémentaires, compensant le manque de justification de la première moitié et/ou dépassant le barème initialement prévu. Ainsi, **le score maximal attribué à cette question est de 1,5 pt**, et le score maximal théoriquement atteignable était de 1,9 pt.

IV.5 Retirer un voisinage (0,3 pt | 221 | 0,90 ; 0,20)

Simple généralisation de la Question IV.2.

IV.6 Deuxième algorithme FPT (1,3 pt | 163 | 0,62 ; 0,26)

Similairement à la Question IV.3, les copies contenaient de nombreuses erreurs sur les cas de bases. En plus des erreurs précédentes :

- Tester « $k = 0$ » mène potentiellement à un bug, car la valeur de k peut passer de $k = 1$ à $k = -2$. Cela est suffisant pour perdre la propriété FPT.
- Pour atteindre la complexité demandée, il fallait utiliser l'algorithme de la Question IV.4 dans le cas où le graphe était de degré 2. Que cet algorithme soit de complexité linéaire (comme demandé, mais impossible) ou quadratique, la complexité totale n'est pas modifiée.

Relativement peu de copies ont pris le temps de justifier correctement la complexité de l'algorithme.

V Noyau quadratique du problème (4 pt)

Cette section introduit la notion de noyaux dans le cadre de la complexité paramétrée.

Ne pas se laisser impressionner : Si la définition de noyau pouvait sembler compliquée, la plupart des questions de cette partie pouvait être traitée sans même avoir lu cette définition. Les Questions V.2, V.3 et V.5 étaient très abordables.

V.1 Utiliser un noyau (0,7 pt | 142 | 0,75 ; 0,22)

Cette question a été peu (et souvent mal) traitée. Elle nécessitait de comprendre la notion de noyau, dont la définition est un peu complexe. S'il était assez clair qu'il fallait utiliser l'algorithme polynomial pour transformer l'instance (G, k) en une instance (G', k'), beaucoup de copies ont ensuite omis de considérer le cas où $\tau(G) > k$, utilisant directement l'algorithme de la Question IV.1 sur $(G', k;')$ en supposant $|V(G')| \leq f(k)$.

V.2 Retirer un sommet isolé (0,1 pt | 250 | 0,98 ; 0,09)

V.3 Retirer un sommet central (0,5 pt | 180 | 0,94 ; 0,16)

Similaire à la Question IV.5.

V.4 Algorithme polynomial (0,8 pt | 156 | 0,45 ; 0,23)

L'algorithme n'était pas complètement spécifié dans l'énoncé, et on attendait qu'il soit complété de façon optimale : le point crucial ici est le calcul des degrés de chaque sommet. Une implémentation naïve va réévaluer les degrés des sommets à chaque itération, ce qui donne un algorithme de complexité cubique. Quelques copies ont proposé un algorithme quadratique, obtenu en calculant initialement les degrés des sommets et en maintenant ces valeurs à jour à chaque suppression d'un sommet.

V.5 Borner le graphe résultant (1,6 pt | 161 | 0,77 ; 0,24)

Si $\tau(G') \leq k'$, et comme chaque sommet de G' a au plus k' voisins, on a facilement que G' a au plus $k'(k'+1)$ sommets et au plus k'^2 arêtes.

V.6 Noyau quadratique (0,3 pt | 129 | 0,96 ; 0,09)

Il suffit ici de reprendre ce qui a été fait aux questions précédentes : on applique l'algorithme (polynomial) de la Question V.3, qui renvoie une instance (G', k') ; les Questions V.2 et V.3 assurent la propriété (i), la Question V.5 assure la propriété (ii) pour $f(k) = k(k+1)$; enfin, la propriété (iii) vient du fait que l'algorithme ne peut que diminuer la valeur de k.

VI Couvertures fractionnaires (6,1 pt)

Cette section introduit la notion de couverture factionnaire (et d'optimisation linéaire en nombre entier) qui permet de calculer un meilleur noyaux que celui calculé dans la partie précédent, ainsi qu'une 2-approximation du problème différente de la partie III.

VI.1 Premier exemple (0,1 pt | 242 | 0,97 ; 0,12)

VI.2 Exemples de couvertures (0,2 pt | 232 | 0,95 ; 0,16)

VI.3 Fractionnaire VS Ensembliste (0,4pt | 170 | 0,81 ; 0,24)

Le contre-exemple le plus simple est le graphe complet K_n, dont la couverture minimale a été calculé à la Question I.2. Plusieurs réponses essayent de prouver l'inexistence de A en toute généralité et sans donner de contre-exemple spécifique, mais ces approches sont futiles vu qu'il existe des graphes tels que τ_f et τ soient égaux (par exemple, les chemins de taille pair P_{2n}).

VI.4 Indépendance des faibles coefficients (0,3 pt | 168 | 0,97 ; 0,12)

VI.5 Couverture des forts coefficients (1,2 pt | 127 | 0,62 ; 0,25)

La partie difficile de cette question (valant deux tiers des points) était le calcul de $|V_{1/2} \cup V_1|$.

VI.6 S reste couvrant (0,8 pt | 58 | 0,92 ; 0,19)

VI.7 Perturbation d'une couverture (1,0 pt | 43 | 0,80 ; 0,24)

VI.8 S est minimal (1,0* pt | 8 | 0,78 ; 0,16)

Le score maximal attribué à cette question est de 0,8 pt, les rares réponses à cette question ne traitant pas le cas $\epsilon = 0$ lors de la division par ϵ.

VI.9 Rejets de l'algorithme (0,3 pt | 50 | 0,90 ; 0,21)

VI.10 Noyau linéaire (0,8 pt | 19 | 0,69 ; 0,23)