Épreuve orale d'informatique, Filière MPI

Coefficient (en pourcentage du total des points du concours) **:** 15,7 %

L'épreuve orale d'informatique fondamentale concerne les candidats à l'Ecole polytechnique dans la filière MPI.

Cette année, 39 candidats français et 5 candidats étrangers ont participé à l'épreuve. On ne peut que regretter que, sur l'ensemble de ces 44 candidats, une seule soit une fille.

Les notes des candidats français sont comprises entre 5,5 et 18,5, avec une moyenne de 12 et un écart-type de 3,6. Le faible nombre de candidats étrangers ne permet pas d'exploitation statistique pertinente de leurs résultats. L'histogramme de la figure 1 présente la distribution des notes de l'ensemble des candidats.

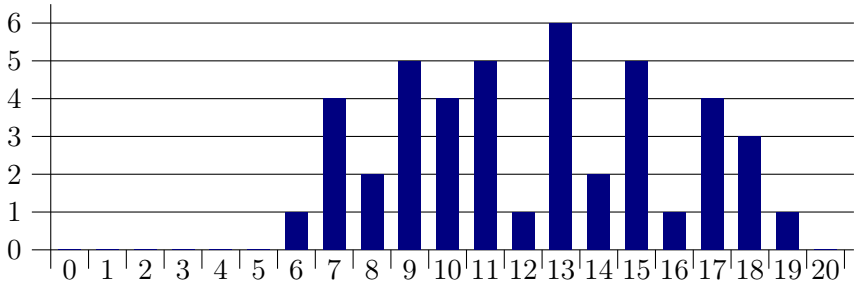


FIGURE 1 – Histogramme des notes de l'épreuve, arrondies à l'entier supérieur

Chaque interrogation durait 47 minutes ; 3 minutes étant utilisées par l'examinateur pour s'assurer de l'identité du candidat, lui permettre de signer la feuille d'émargement et lui rappeler les conditions de l'évaluation. Celle-ci se base principalement sur la progression des candidats dans les questions, mais tient également compte de la clarté de leur propos et de l'autonomie dont ils ont fait preuve.

Le jury a proposé 22 sujets originaux, dont trois exemples représentatifs sont présentés en annexe. Chaque sujet débute par un énoncé qui présente un problème d'informatique et introduit ses notations, puis comporte entre sept et onze 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 majeure partie du temps de l'interrogation, et visent à permettre de départager les candidats. 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. Par ailleurs, l'évaluation du candidat tient bien sûr compte de la difficulté des questions qui lui étaient proposées. Ainsi, tel sujet conçu pour être plus difficile d'accès que tel autre est mieux valorisé ; telle question introductive simple mais chronophage est elle aussi valorisée à hauteur du temps qu'elle coûtera aux 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 les conditions de l'épreuve, en temps limité et au tableau, ne permettent guère la mise en œuvre pratique d'éléments de programmation, il reste indispensable de garder à l'esprit le sens des objets que l'on étudie et de pouvoir discuter de la pertinence de tel ou tel choix d'implantation.

Le niveau général dont les candidats ont fait preuve pour cette première édition de l'épreuve orale d'informatique en voie MPI était excellent. Les candidats ont montré une très bonne maîtrise des algorithmes, structures de données, ainsi que d'éléments plus théoriques du programme tels que la logique et la théorie des langages. Le plus souvent, ils ont su s'appuyer sur cette maîtrise pour réfléchir de manière pertinente à des problèmes difficiles. Ainsi, les candidats ayant eu au moins 12/20, soit la moitié des candidats, ont proposé une prestation d'excellente facture ; obtenir 16 ou

plus requerrait de dépasser les attentes initiales de l'examinateur sur ce que l'on pouvait raisonnablement attendre d'un excellent candidat.

Il s'agit là d'un constat extrêmement satisfaisant, qui confirme tout le bien-fondé de la mise en place de la filière MPI pour l'apprentissage de l'informatique.

Cet excellent niveau général suggère également qu'il n'est pas raisonnable pour les candidats de faire des impasses ou d'afficher une maîtrise vacillante de tel aspect du programme, comme cela s'est vu en quelques occasions ; le candidat en ressort doublement pénalisé par le temps qu'il mettra à répondre à des questions considérées comme simples, ainsi que par l'impression négative qu'il laisse à l'examinateur, qui n'aura d'autre choix que de la sanctionner explicitement au vu du très haut niveau général. Des exemples de tels manques sont la volonté d'utiliser le lemme de l'étoile pour démontrer qu'un langage *est* rationnel, [1] ou encore l'oubli (malgré l'insistance de l'examinateur) de la compression des chemins lors des opérations *trouver* de la structure *réunir et trouver.* [2]

En annexe, nous présentons, corrigeons et commentons trois exemples de sujets représentatifs de cette année :
— Algorithme Union-Find
— Bob l'écureuil
— Langages rationnels et lemme de l'étoile

1. Le programme d'informatique en MPI fait référence à la notion de langage régulier : les deux mots *rationnel* et *régulier* peuvent être vus comme interchangeables.

2. Bien que ne la mentionnant pas explicitement, le programme fait bien référence à cette compression des chemins, puisqu'il invoque des implémentations efficaces de la structure « réunir et trouver », qu'il oppose à des implémentations naïves.

Algorithme Union-Find

L'algorithme Union-Find a pour but de gérer dynamiquement une partition d'un ensemble $\{1, 2, \ldots, n\}$ fixé. Initialement, on part de la partition maximale $\{\{1\}, \{2\}, \ldots, \{n\}\}$. En pratique, on représente les partitions de $\{1, \ldots, n\}$ comme suit :

- ▷ chaque entier k possède un (seul) *père* $\mathsf{p}_k \leqslant n$, ce que l'on notera $k \to \mathsf{p}_k$; on peut avoir $k = \mathsf{p}_k$;
- ▷ chaque entier k possède un *poids* $\mathsf{w}_k \in \mathbb{N}$;
- ▷ deux entiers distincts k et ℓ ne peuvent être ancêtres l'un de l'autre, et appartiennent au même sous-ensemble si et seulement s'ils ont un ancêtre commun ;
- ▷ si l'entier k appartient à un singleton, $w_k = 0$.

Question 1. On dit que m est le *chef* de k si m est un ancêtre de k tel que $m = \mathsf{p}_m$. Démontrer que tout entier a un unique chef, et deux entiers k et ℓ appartiennent au même sous-ensemble si et seulement s'ils ont le même chef.

L'algorithme est censé gérer trois types de requêtes, en procédant comme suit :

1. La requête auxiliaire $\mathsf{Chef}(k)$: on recherche le chef de k. Si $k = \mathsf{p}_k$, il s'agit de k lui-même. Sinon, il s'agit du chef de p_k, et ce chef devient le nouveau parent de k.
2. La requête $\mathsf{Test}(k, \ell)$: on se demande si k et ℓ appartiennent au même sous-ensemble. Cela revient à identifier les entiers $k' = \mathsf{Chef}(k)$ et $\ell' = \mathsf{Chef}(\ell)$ puis à vérifier si $k' = \ell'$.
3. La requête $\mathsf{Union}(k, \ell)$: on souhaite réunir les sous-ensembles auxquels appartiennent k et ℓ. Cela revient à identifier les entiers $k' = \mathsf{Chef}(k)$ et $\ell' = \mathsf{Chef}(\ell)$, puis :
   - ▷ si $\mathsf{w}_{k'} > \mathsf{w}_{\ell'}$, l'entier k' devient le nouveau parent de ℓ' ;
   - ▷ si $\mathsf{w}_{\ell'} > \mathsf{w}_{k'}$, l'entier ℓ' devient le nouveau parent de k' ;
   - ▷ si $k' \neq \ell'$ et $\mathsf{w}_{k'} = \mathsf{w}_{\ell'}$, l'entier k' devient le nouveau parent de ℓ' et son poids augmente de 1.

On souhaite démontrer que répondre à m de ces requêtes peut se faire en temps $\mathcal{O}(m \log^\star(m))$, où $\log^\star$ est la fonction définie par $\log^\star(m) = 1$ lorsque $m \leqslant 1$ et $\log^\star(m) = 1 + \log^\star(\log_2(m))$ lorsque $m > 1$.

Question 2. En partant de l'ensemble $\{\{1\}, \{2\}, \{3\}, \{4\}\}$, on effectue successivement les requêtes $\mathsf{Union}(1, 2)$, $\mathsf{Union}(3, 4)$, $\mathsf{Union}(2, 4)$ et $\mathsf{Test}(2, 4)$. Indiquer le père et le poids de chaque entier $k \leqslant 4$.

Question 3. On note w_k^∞ le poids d'un entier k à la fin de l'algorithme. Démontrer que $\mathsf{w}_u^\infty < \mathsf{w}_v^\infty$ pour tous les entiers $u \neq v$ tels que v a été le parent de u.

Question 4. Démontrer, pour tout entier $\ell \geqslant 1$, qu'il y a au plus $m/2^{\ell-1}$ entiers $k \leqslant n$ pour lesquels $\mathsf{w}_k^\infty = \ell$.

Question 5. Démontrer que toute requête Test ou Union effectuée au cours de l'algorithme a une complexité majorée par $\mathcal{O}(\log_2(\min\{m, n\}))$.

Question 6. Soit $\mathcal{G} = (V, E)$ le graphe dont les sommets sont les entiers de 1 à n et les arêtes sont les paires (u, v) pour lesquelles $u \neq v$ et v a été le parent de u au cours de l'algorithme, mais pas quand celui-ci se termine. Démontrer que la complexité totale de nos m requêtes est majorée par $\mathcal{O}(m + |E|)$.

Question 7. Soit $(a_\ell)_{\ell \geqslant 0}$ la suite définie par $a_0 = 0$ et $a_{\ell+1} = 2^{a_\ell}$. Pour tout entier $\ell \geqslant 0$, on note V_ℓ l'ensemble des entiers k tels que $a_\ell \leqslant \mathsf{w}_k^\infty < a_{\ell+1}$. Démontrer que $\mathcal{G}$ contient au plus $4m$ arêtes reliant deux sommets de V_ℓ, et au plus $2m$ arêtes allant d'un sommet de V_ℓ à un sommet en dehors de V_ℓ.

Question 8. Que conclure sur la complexité de l'algorithme Union-Find ?

Éléments de correction et attentes de l'examinateur

Question 1. On se place dans le graphe dont les sommets sont les entiers de 1 à n et les arêtes sont les couples (k, p_k). Par hypothèse, tout sommet y est de degré sortant 1, et les seuls cycles du graphe sont des boucles, c'est-à-dire des cycles de longueur 1. Puisque le chemin obtenu en partant d'un entier m et en remontant la liste de ses ancêtres se termine par un cycle, il contient nécessairement une boucle, centrée sur un chef de m.

En particulier, ce chef est un ancêtre de tous les ancêtres de m, donc m ne peut pas avoir deux chefs, qui seraient ancêtres l'un de l'autre. En outre, deux entiers k et ℓ ont un ancêtre commun si et seulement s'ils ont le même chef, qui sera le chef de cet ancêtre commun.

Cette question a pour but de mettre le candidat en confiance et de lui permettre de s'approprier la représentation de l'algorithme Union-Find qu'il devra manipuler au cours de l'épreuve. Elle vise également à évaluer sa capacité à introduire des graphes de manière pertinente, puis à raisonner rigoureusement sur ces graphes.

Question 2. Voici un tableau où l'on a recensé, après chaque étape de l'algorithme, le parent et le poids de chaque nœud ; la situation initiale est indiquée à l'étape « 0 ».

Étape	p_1	w_1	p_2	w_2	p_3	w_3	p_4	w_4
0	1	0	2	0	3	0	4	0
1	1	1	1	0	3	0	4	0
2	1	1	1	0	3	1	3	0
3	1	2	1	0	1	1	3	0
4	1	1	1	0	1	1	1	0

La seule difficulté est ici de ne pas oublier que la requête $\mathsf{Test}(2, 4)$ entraîne une compression du chemin liant le sommet 4 à son chef 1, qui devient désormais son parent.

Cette question vise donc à s'assurer que les élèves ont bien retenu que compresser les chemins dans l'algorithme Union-Find était indispensable ou, le cas échéant, à le leur rappeler.

Question 3. Au cours de l'algorithme, et tant qu'il est chef, un entier peut d'abord voir son poids augmenter (de 1 à chaque fois), au cours de fusions à l'occasion desquelles on lui assigne de nouveaux descendants ; puis, si un jour il cesse d'être chef, son poids ne changera plus jamais. Par conséquent, si un entier v est un jour le parent d'un entier u, soit w_v et w_u leurs poids à ce moment-là. Puisque u n'est plus chef, il ne le sera plus jamais, donc $\mathsf{w}_u^\infty = \mathsf{w}_u$; par ailleurs, le poids de v ne pourra qu'augmenter, donc $\mathsf{w}_v^\infty \geqslant \mathsf{w}_v > \mathsf{w}_u = \mathsf{w}_u^\infty$.

Cette question est la première question délicate du sujet. Elle a pour but de vérifier si le candidat maîtrise la dynamique des liens de parenté et des poids au cours de l'algorithme, et s'il est capable de jouer à la fois sur l'état du système à un moment de l'exécution de l'algorithme et une fois cette exécution achevée. Par ailleurs, cette question et chacune des questions suivantes a pour but de préparer les questions ultérieures, notamment le résultat final que l'on souhaite démontrer.

Question 4. On note Γ le graphe étiqueté dont les sommets sont les entiers de 1 à n et les arêtes sont les couples (u, v) pour lesquels $u \neq v$ et v a été parent de u au cours de l'algorithme ; chaque sommet k est étiqueté par l'entier w_k^∞. On dit alors que v est un Γ-descendant de u (qui est un Γ-ancêtre de v) si Γ contient un chemin allant de u à v.

Chaque entier k gagne ses ancêtres un par un, au cours de requêtes Union ; chaque nouvel ancêtre de k devenant un ancêtre de tous les anciens ancêtres de k. Par conséquent, Γ contient un chemin partant de k et passant par tous ses Γ-ancêtres, dont les poids ne font qu'augmenter le long du chemin. En particulier, deux sommets de même étiquette n'ont aucun Γ-descendant commun.

Or, une récurrence sur w indique que, à tout moment au cours de l'algorithme, chaque sommet de poids $\mathsf{w} \geqslant 0$ possède au moins 2^{w} descendants. Les s entiers k pour lesquels $\mathsf{w}_k^\infty = \ell$, ces s entiers ont donc, à eux tous, un total d'au moins $2^\ell s$ ou plus Γ-descendants.

En outre, tout sommet qui Γ-descend d'un entier de poids non nul a dû être un des arguments d'une requête Union. Il y a eu au plus m requêtes Union, donc au plus $2m$ entiers utilisés comme arguments de telles requêtes. On en conclut que $s \geqslant 2m/2^\ell$.

Dans la continuité de la question précédente, cette question a pour but d'affiner la compréhension de la dynamique du système étudié par le candidat. Sa difficulté principale consiste à introduire une structure de données simple mais qui contiendra des informations valides à différentes phases de l'algorithme. Elle vise également à s'assurer que le candidat est à même de naviguer entre des échelles locale et globale, ce genre d'approches étant crucial en informatique.

Question 5. En question précédente, les s entiers k pour lesquels $\mathsf{w}_k^\infty = \ell$ ont cumulé au plus $\min\{n, 2m\}$ Γ-descendants, ce qui nous fournit l'inégalité $s \leqslant \min\{n, 2m\}/2^\ell$, légèrement meilleure que l'inégalité précédente. En particulier, le plus grand des entiers w_k^∞ est un entier ℓ pour lequel $1 \leqslant s \leqslant \min\{n, 2m\}/2^\ell$, de sorte que $\ell \leqslant \log_2(\min\{m, n\}) + 1$. En outre, toute requête portant sur deux entiers dont les chefs sont de poids w et w' a une complexité $\mathcal{O}(\mathsf{w} + \mathsf{w}')$. On conclut en remarquant que w et w' sont majorés par $\mathcal{O}(\ell) \subseteq \mathcal{O}(\log_2(\min\{m, n\}))$.

Cette question a pour but de vérifier si le candidat a le réflexe d'affiner de lui-même la borne qui lui avait été proposée en question précédente, et s'il est capable d'utiliser des arguments simples pour trouver des bornes supérieures sur la complexité de telle opération.

Question 6. Considérons une requête Union ou Test d'arguments a et b, et soit ω_a et ω_b le nombre de liens de parentés que cette requête remonte en partant de a et de b. La complexité de cette requête est majorée par $\mathcal{O}(\omega_a + \omega_b + 1)$, et celle-ci résulte en l'introduction d'au moins $(\omega_1 - 1) + (\omega_b - 1)$ arêtes dans E : il s'agit des les liens de parentés détruits par notre requête, et ceux-ci ne seront jamais recréés. Ainsi, si notre requête résulte en l'ajout de e arêtes dans E, sa complexité est majorée par $\mathcal{O}(e + 1)$. Puisque l'on a m requêtes en tout et que la somme des entiers e ainsi définis est égale à $|E|$, le résultat désiré s'ensuit.

De nouveau, cette question a pour but de vérifier que le candidat est capable d'estimer la complexité d'un algorithme en fonction de paramètres auxquels il n'est pas habitué.

Question 7. Soit u un élément de V_ℓ. Les sommets accessibles par une arête de $\mathcal{G}$ partant de u sont des Γ-ancêtres de u. Ils sont donc d'étiquettes différentes, et au plus $a_{\ell+1} - a_\ell$ d'entre eux appartiennent à V_ℓ. Or, le nombre de sommets de Γ d'étiquette w est un entier $s_\mathsf{w} \leqslant m/2^{\mathsf{w}-1}$. Le nombre total d'arêtes internes de $\mathcal{G}$ internes à V_ℓ est donc majoré par

$$a_{\ell+1}|V_\ell| \leqslant a_{\ell+1} \sum_{\mathsf{w} \geqslant a_\ell} m/2^{\mathsf{w}-1} = a_{\ell+1} m/2^{a_\ell - 2} = 4m.$$

Par ailleurs, chaque requête d'arguments a et b résulte en l'ajout dans $\mathcal{G}$ d'au plus deux arêtes quittant V_ℓ : il s'agit, sur chacun des deux chemins allant de a et b à leurs chefs et que l'on explore avec cette requête, de la dernière arête partant d'un sommet de V_ℓ. Par conséquent, au plus $2m$ arêtes sortent de V_ℓ.

Il s'agit là de la question la plus difficile du sujet, censée résister même aux élèves les plus brillants.

Question 8. Puisque $\mathsf{w}_k^\infty \leqslant \log_2(m) + 1$ pour tout entier k, on sait que $V_\ell = \emptyset$ dès lors que $a_\ell > \log_2(m) + 1$. Or, la fonction $\log^\star$ est croissante, et $\log^\star(a_\ell) = \ell$ pour tout entier $\ell \geqslant 1$. Ainsi, lorsque $m \geqslant 2$ et $\ell > \log^\star(m)$, on sait que $\log^\star(a_\ell) = \ell > \log^\star(m) \geqslant \log^\star(\log_2(m) + 1)$, de sorte que $a_\ell > \log_2(m) + 1$ et $V_\ell = \emptyset$.

En particulier, $\mathcal{G}$ contient au plus $6m$ arêtes issues de chacun des $\lfloor \log^\star(m) \rfloor + 1$ ensembles V_ℓ susceptibles d'être non vides, et aucune arête issue des autres ensembles V_ℓ, nécessairement vides. Il contient donc $\mathcal{O}(m \log^\star(m))$ arêtes, ce qui conclut.

Cette question vise non seulement à obtenir explicitement le résultat espéré depuis le début du sujet, mais aussi à vérifier que le candidat est à même de prendre du recul sur les calculs de plus en plus compliqués qu'il a dû mener au cours des questions précédentes.

On pourrait s'étonner qu'elle soit substantiellement plus abordable que la question 7. Néanmoins, les candidats auraient tout à fait eu le droit de demander à traiter cette question en admettant la question 7, avant de revenir sur celle-ci ensuite. Dans ce genre de cas, il ne faut pas s'auto-censurer, et ne pas hésiter à proposer à l'examinateur pareille démarche ; ici, la réponse aurait été positive, même s'il faut bien sûr que le candidat soit prêt à renoncer à son idée si jamais l'examinateur lui répond de manière négative.

Bob l'écureuil

Bob est un écureuil qui vit le long d'une ligne de chemin de fer. Il a caché ses réserves de noisettes dans les différentes gares, et cherche maintenant à établir son nid dans l'une de ces gares. Pour des raisons pratiques, il souhaite construire son nid dans une gare où il a déjà caché des noisettes, tout en minimisant la durée moyenne entre son nid et ses réserves de noisettes. Comment choisir dans quelle gare établir son nid ?

Question 1. La ligne est composée de 9 gares, numérotées de 0 à 8 ; Bob a mis une cachette dans chacune de ces gares. On met k minutes pour aller de la gare ℓ à la gare $k + \ell$. Quelle gare Bob choisira-t-il d'établir son nid ?

Question 2. On suppose maintenant que les gares ne sont plus régulièrement espacées. Désormais, il y a n gares, toujours numérotées de 0 à $n-1$, toujours en ligne droite. Bob connaît des entiers $t_0, t_1, \ldots, t_{n-2}$ pour lesquels aller de la gare ℓ à la gare $\ell + 1$ requiert t_ℓ minutes ; ainsi, aller de la gare ℓ à la gare $\ell + 2$ requiert $t_\ell + t_{\ell+1}$ minutes.
Donner un algorithme qui permettra à Bob de choisir, en temps $\mathcal{O}(n)$, dans quelle gare installer son nid.

Question 3. Désormais, le réseau prend la forme d'un graphe connexe acyclique. Bob dispose, pour chaque gare k, d'une liste des gares ℓ auxquelles la gare k est reliée directement, ainsi que de la durée $\mathsf{d}_{k,\ell}$, en nombre de minutes, du trajet entre les gares k et ℓ.
Donner un algorithme qui permettra à Bob de choisir, en temps $\mathcal{O}(n)$, dans quelle gare installer son nid.

Bob déménage à Manhattan, là où toutes les rues sont orientées selon un axe Nord-Sud ou Est-Ouest, et espacées de 100 mètres l'une de l'autre. Il a caché ses n réserves de noisettes à n croisements de rue, chaque croisement étant identifié par des coordonnées entières. Par exemple, le croisement $(2, 3)$ se trouve 400 mètres à l'Ouest et 500 mètres au Sud du croisement $(6, 8)$, donc Bob doit marcher au minimum 900 mètres pour aller d'un croisement à l'autre.
Bob souhaite maintenant établir son nid à l'un des n croisements de rue où se trouve déjà une réserve de noisettes, mais tout en minimisant la distance moyenne entre son nid et des réserves de noisettes. À quel croisement établir son nid ?

Question 4. Bob a disposé ses 6 réserves aux points de coordonnées $(0, 0)$, $(1, 1)$, $(2, 3)$, $(3, 1)$, $(3, 2)$ et $(4, 3)$. Auquel de ces six endroits choisira-t-il d'établir son nid ?

Question 5. Bob a simplement noté les coordonnées entières $(x_0, y_0), (x_1, y_1), \ldots, (x_{n-1}, y_{n-1})$ des n réserves qu'il a choisies pour entreposer ses noisettes.
Donner un algorithme qui permettra à Bob de choisir, en temps $\mathcal{O}(n \log(n))$, à quelle intersection installer son nid.

Question 6. On suppose désormais que chacune des coordonnées x_i et y_i est comprise entre 0 et $n^2 - 1$. Comment adapter l'algorithme précédent pour s'assurer qu'il fonctionne désormais en temps $\mathcal{O}(n)$?

Question 7. Pour fluidifier le trafic, des urbanistes ont ajouté des routes orientées du Nord-Est vers le Sud-Ouest, et du Nord-Ouest vers le Sud-Est. Ainsi, le croisement $(2, 3)$ se désormais à $400\sqrt{2} + 100 \approx 666$ mètres du croisement $(6, 8)$: il suffit d'aller quatre fois vers le Nord-Est puis une fois vers le Nord. On suppose toujours que chacune des coordonnées x_i et y_i est comprise entre 0 et $n^2 - 1$, et que Bob veut installer son nid en l'une des n intersections (x_i, y_i).
Donner un algorithme qui permettra à Bob de choisir, en temps $\mathcal{O}(n)$, à quelle intersection installer son nid.

Éléments de correction et attentes de l'examinateur

Question 1. On se laisse facilement convaincre, par exemple pour des raisons de symétrie, que la meilleure gare est la gare de numéro 4. Une manière simple de démontrer proprement ce résultat consiste à s'intéresser à la somme des durées plutôt qu'à leur moyenne, puis à remarquer que, lorsque $0 \leqslant \ell \leqslant 4$, les distances de la gare k aux gares ℓ et $8-\ell$ sont de somme minimale si et seulement si $\ell \leqslant k \leqslant 8-\ell$. La gare 4 est donc la seule à être optimale vis-à-vis de l'ensemble de ces paires (dont la paire obtenue pour $\ell = 4$ est dégénérée).

Cette question a pour but de mettre le candidat en confiance et de lui permettre de s'approprier l'énoncé, tout en évaluant sa capacité à être rigoureux dès que c'est nécessaire, par exemple pour justifier que la gare 4 est optimale.

Question 2. Le raisonnement invoqué pour la question précédente fonctionne toujours : lorsque $0 \leqslant \ell \leqslant (n-1)/2$, les distances de la gare k aux gares ℓ et $(n-1)-\ell$ sont de somme minimale si et seulement si $\ell \leqslant k \leqslant (n-1)-\ell$. Par conséquent, les gares optimales sont celles de numéros $\lfloor (n-1)/2 \rfloor$ et $\lceil (n-1)/2 \rceil$; elles sont confondues lorsque n est impaire.

Cependant, obtenir un tel résultat n'est pas nécessairement pertinent au vu des questions qui suivent et de la contrainte consistant à fournir un algorithme en temps $\mathcal{O}(n)$. Une alternative est d'améliorer l'algorithme naïf consistant à calculer chaque somme de distances en temps $\Theta(n)$, ce qui requiert un temps total $\Theta(n^2)$. Pour ce faire, il suffit de remarquer que la somme, disons s_ℓ, des distances entre chaque gare ℓ et les gares $k \leqslant \ell$ est soumise à la relation $s_{\ell+1} = s_\ell + (\ell+1)t_\ell$. Celle-ci permet de calculer les nombres s_ℓ de proche en proche ; on peut ensuite procéder de même pour calculer la somme des distances entre chaque gare ℓ et les gares $k \geqslant \ell$, ce qui permet de conclure.

Cette question, nettement plus difficile, a pour but de forcer le candidat à être inventif, puis à réagir face aux indications de l'examinateur ; il n'est pas attendu du candidat qu'il résolve cette question en direct et sans indication.

Question 3. Là encore, les valeurs des durées $\mathsf{d}_{k,\ell}$ sont sans importance, et les gares optimales sont les gares « médianes » du réseau arborescent. Pour les identifier, on enracine notre arbre en un nœud arbitraire, puis on calcule récursivement, depuis les feuilles vers la racine, le nombre de descendants de chaque gare, puis la somme des distances de chaque gare à ses descendantes. Plus précisément, si une gare k possède a_k descendantes, dont ℓ enfants de numéros 0 à $\ell-1$, et si l'on note s_k la somme des durées des trajets de la gare k avec ses descendantes, alors

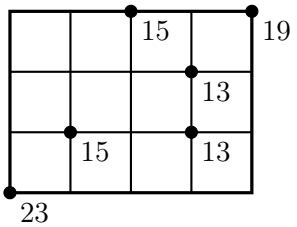
$$d_k = 1 + d_0 + d_1 + \cdots + d_{\ell-1} \text{ et } s_k = a_0 d_{0,k} + a_1 d_{1,k} + \cdots + a_{\ell-1} d_{\ell-1,k} + s_0 + s_1 + \cdots + s_{\ell-1}.$$

Dans une deuxième phase, on calcule depuis la racine vers les feuilles la somme S_k des distances de chaque gare k à l'ensemble des gares du réseau. En effet, si la gare k est le parent de la gare ℓ, une durée totale $S_k - a_\ell d_{k,\ell} - s_\ell$ est nécessaire pour aller de la gare k aux gares qui ne descendent pas de ℓ, et on en déduit que

$$S_\ell = (S_k - a_\ell d_{k,\ell} - s_\ell) + (n - a_\ell) d_{k,\ell} + s_\ell.$$

Cette question a pour but de vérifier si le candidat a bien digéré les méthodes qu'il a dû mettre en œuvre à la question précédentes et qu'il est à même de développer peu à peu des algorithmes récursifs sur les arbres.

Question 4. Dans un premier temps, on peut se contenter de calculer brutalement la somme des distances L^1 de chaque point aux autres ; ici, on a écrit cette somme à côté de chacun des points concerné. Bob élit donc domicile en l'un des points $(3,1)$ ou $(3,2)$.



Cette question a plusieurs buts : préparer le candidat aux questions qui suivent ; vérifier qu'il reconnaît des distances L^1, dites « de Manhattan » ; s'assurer que, dans le cadre d'un problème de géométrie, il aura le réflexe *indispensable* de faire un dessin au tableau.

Question 5. La distance L^1 se décompose naturellement en la somme de deux composantes horizontale et verticale. Or, la démarche utilisée en question 2 nous permet justement, dans un cadre unidimensionnel, de calculer en temps $\mathcal{O}(n)$ la somme des distances de chaque point aux autres, pourvu qu'ils soient triés. On peut donc trier nos n points par x_i croissants, en temps $\mathcal{O}(n\log(n))$, avant de calculer la somme des composantes horizontales des distances de chaque point aux $n-1$ autres ; on procède ensuite de même avec les composantes verticales.

Cette question difficile a pour but de vérifier dans quelle mesure le candidat perçoit le fait qu'une distance L^1 est une somme de deux distances unidimensionnelles, puis qu'il est capable d'utiliser un calcul de sommes pour calculer des sommes de combinaisons linéaires.

Question 6. L'enjeu est ici de trier n entiers compris entre 0 et n^2-1 en temps $\mathcal{O}(n)$. Pour ce faire, on peut s'inspirer du tri par paquets, qui permet de trier n entiers compris entre 0 et $n-1$ en s'aidant d'un tableau de taille n ; ici, on va écrire nos entiers en base n (sur deux chiffres, donc), puis les trier par unités et ensuite par n-aines.

Plus précisément, on utilise un tableau $\mathbf{T}$ de taille n, contenant n listes chaînées, et on commence par insérer chaque entier k dans la liste $\mathbf{T}[k \mod n]$. En réunissant ces listes, on a trié les entiers par chiffres des unités. On peut refaire la même opération pour les chiffres des n-aines ; la liste $\mathbf{T}[\ell]$ contient ainsi les entiers k tels que $\ell < k < (\ell+1)n$, eux-mêmes triés par chiffre des unités. En réunissant ces nouvelles listes, on a trié nos n entiers.

Cette question difficile a plusieurs buts : vérifier que le candidat saisit bien l'enjeu de la question, qui est de trier rapidement des entiers évoluant dans un intervalle entier de taille $n^{\mathcal{O}(1)}$; s'assurer qu'il connaît le tri par paquets ; le pousser à s'interroger sur la notion de tri *stable* que l'on a utilisée ici de manière sous-jacente, en re-triant des données pré-triées.

Question 7. On n'utilise plus la distance L^1, mais une nouvelle distance *ad hoc*, disons d, que l'on peut obtenir comme suit : si l'on note $\mathsf{d}x$ et $\mathsf{d}y$ les composantes horizontale et verticale de la distance entre deux points,

$$\mathsf{d} = \sqrt{2}\min(\mathsf{d}x, \mathsf{d}y) + (\max(\mathsf{d}x, \mathsf{d}y) - \min(\mathsf{d}x, \mathsf{d}y)).$$

Or, si l'on pose $u = x+y$ et $v = x-y$, on constate que

$$\mathsf{d}u + \mathsf{d}v = 2\max(\mathsf{d}x, \mathsf{d}y) \text{ et } \min(\mathsf{d}x, \mathsf{d}y) + \max(\mathsf{d}x, \mathsf{d}y) = \mathsf{d}x + \mathsf{d}y.$$

On en conclut que

$$\mathsf{d} = (s-1)\left(\mathsf{d}x + \mathsf{d}y + \frac{\mathsf{d}u + \mathsf{d}v}{\sqrt{2}}\right).$$

Calculer les sommes de chacune des composantes en x, y, u et v des distances entre points, ce que l'on peut faire en temps $\mathcal{O}(n)$ opérations, permet donc de calculer la distance de chaque point au $n-1$ autres points.

Cette question, conçue pour être à peu près infaisable dans les conditions de l'épreuve, a pour but de distinguer les tous meilleurs candidats. Par conséquent, et contrairement aux précédentes questions difficiles, toute latitude est ici laissée au candidat, que l'examinateur pourra remettre sur les rails s'il écrit quelque chose de faux, mais sera livré à lui-même pour trouver les idées permettant la résolution de cette question.

Langages rationnels et lemme de l'étoile

On recherche un langage $\mathcal{L}$ non rationnel mais pour lequel $\mathcal{L}$ et son complémentaire satisfont le lemme de l'étoile.

Question 1. Soit Σ un alphabet fini. Le langage $\mathcal{L}_1$ formé des mots $w \in \Sigma^*$ dont la longueur est paire est-il rationnel ?

Question 2. Le langage $\mathcal{L}_2$ formé des mots $w \in \Sigma^*$ dont la longueur est le carré d'un entier est-il rationnel ?

Soit $\mathcal{L}$ et $\mathcal{L}'$ deux langages sur un alphabet Σ. On dit que $\mathcal{L}$ est $\mathcal{L}'$-*étoilé* s'il existe un entier $n \geqslant 0$ tel que tout mot $w \in \mathcal{L}'$ de longueur $|w| \geqslant n$ admet une factorisation $w = s \cdot t \cdot u$ pour laquelle $|s| \leqslant n$, $1 \leqslant |t| \leqslant n$ et $s \cdot t^* \cdot u \subseteq \mathcal{L} \cup \{w\}$. Si $\mathcal{L}$ est $\mathcal{L}$-étoilé, on dit même que $\mathcal{L}$ satisfait le *lemme de l'étoile*.

Question 3. Démontrer que tout langage rationnel satisfait le lemme de l'étoile.

Question 4. Démontrer que, si $|\Sigma| = 1$, les langages qui satisfont le lemme de l'étoile sont les langages rationnels.

Soit $\mathcal{S}$ une partie finie de Σ^*. On note $\mathcal{M}(\mathcal{S})$ l'ensemble des mots $s_1 \cdot s_2 \cdots s_n$ que l'on peut factoriser en n éléments de $\mathcal{S}$ et pour lesquels il existe deux entiers i et $j = i + 1$ ou $j = i + 2$ tels que $s_i = s_j$.

Question 5. Démontrer que tout ensemble $\mathcal{M}(\mathcal{S})$ est rationnel.

Question 6. Démontrer que, si $|\mathcal{S}| \leqslant 4$, le langage $\mathcal{M}(\mathcal{S})$ est $\mathcal{S}^*$-étoilé.

Question 7. On considère l'alphabet $\Sigma = \{a, b, c, d\}$. Pour tout entier $k \leqslant 3$, on note σ_k le mot $(abc)^k \cdot (abd)^{3-k}$. Les langages $\mathcal{L}_3 = \{(\sigma_0 \cdot \sigma_1 \cdot \sigma_2)^\ell \cdot (\sigma_0 \cdot \sigma_1 \cdot \sigma_3)^\ell : \ell \geqslant 0\}$ et $\mathcal{L} = \mathcal{M}(\{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}) \cup \mathcal{L}_3$ sont-ils rationnels ?

Question 8. Démontrer que les ensembles $\mathcal{L}$ et $\Sigma^* \setminus \mathcal{L}$ satisfont le lemme de l'étoile.

Éléments de correction et attentes de l'examinateur

Question 1. L'ensemble $\mathcal{L}_1$ est défini par l'expression rationnelle $(\Sigma^2)^*$. Il est donc rationnel.

Cette question a pour but de mettre le candidat en confiance, et de vérifier qu'il est à même de reconnaître des langages rationnels simples.

Question 2. Pour tout entier $n \geqslant 1$ et tout découpage du mot $w = a^{n^2} \in \mathcal{L}_2$ en trois facteurs $p = a^k$, $q = a^\ell$ et $r = a^{n^2-k-\ell}$ tels que $w = p \cdot q \cdot r$, on sait que $p \cdot q^* \cdot r \not\subseteq \mathcal{L}_2$, par exemple parce que $n + (\ell n)^2$ est compris strictement entre $(\ell n)^2$ et $(\ell n + 1)^2$, de sorte que $p \cdot q^{(\ell n)^2+1} \cdot r \notin \mathcal{L}_2$. L'ensemble $\mathcal{L}_2$ ne satisfait donc pas le lemme de l'étoile, ce qui lui interdit d'être rationnel.

Cette question a pour but de rassurer le candidat sur sa maîtrise du lemme de l'étoile, et de s'assurer de cette maîtrise. Elle vise également à étudier l'attitude du candidat face à une question dont la réponse n'est pas évidente. Par ailleurs, le fait qu'un candidat n'aie pas l'intuition du résultat, par opposition à un autre qui serait tout de suite allé dans la bonne direction, a été volontairement peu pénalisé ; en effet, il est raisonnable d'imaginer que certains candidats, mais pas tous, auront déjà démontré pendant l'année l'irrationalité du langage $\mathcal{L}_2$, qui figure parmi les exemples les plus usuels de langages irrationnels.

Question 3. Soit n le nombre d'états d'un automate fini A reconnaissant $\mathcal{L}$, puis $w \in \mathcal{L}$ un mot de longueur $|w| = n + k \geqslant n$, et $s_0, \ldots, s_{k+n}$ un chemin acceptant de w dans A. Deux des états $s_0, \ldots, s_n$, disons s_i et s_j, coïncident, de sorte que $w_1 \cdots w_i \cdot (w_{i+1} \cdots w_j)^* \cdot w_{i+1} \cdots w_{k+n} \subseteq \mathcal{L}$.

Il s'agit là quasiment d'une question de cours. Cette question a pour buts de vérifier que le candidat est capable d'utiliser une définition précise d'une notion, susceptible de différer marginalement de l'idée qu'il en avait conçue ; qu'il a compris la démonstration du lemme de l'étoile ; qu'il est à l'aise tant avec la manipulation de quantificateurs qu'avec le fait de jongler entre états d'un automate, chemins dans cet automate, et mots associés.

L'autre finalité de cette question est d'établir explicitement le fait que, pour qu'un langage $\mathcal{L}$ soit rationnel, il est nécessaire que $\mathcal{L}$ et son complémentaire satisfassent tous deux le lemme de l'étoile ; le but du sujet étant précisément de démontrer que ce critère nécessaire n'est pas suffisant.

Question 4. Soit $\mathcal{L} \subseteq \Sigma^*$ un langage satisfaisant le lemme de l'étoile, et n l'entier mentionné par le lemme. Pour tout entier $k \geqslant n$, le mot a^k admet une factorisation $a^k = a^i a^\ell a^j$ pour laquelle $1 \leqslant \ell \leqslant j$ et chaque mot $a^{i+z\ell+j} = a^{(k-\ell)+z\ell}$ obtenu lorsque $z \geqslant 0$ appartient à $\mathcal{L}$. Puisque ℓ divise $n!$, le langage $\mathcal{L}$ est donc une réunion d'ensembles singletons ou de la forme $a^{x+n!\mathbb{N}}$. Or, une telle réunion est nécessairement finie car, lorsque $x \equiv y \pmod{n!}$, l'un des deux ensembles $a^{x+n!\mathbb{N}}$ et $a^{x+n!\mathbb{N}}$ est inclus dans l'autre. En outre, chacun des ensembles que l'on réunit est rationnel. Le langage $\mathcal{L}$ est donc lui aussi rationnel.

Cette question a pour but d'évaluer la façon dont le candidat appréhende une question qui pourra le déstabiliser, puisqu'il est peu habituel d'utiliser le lemme de l'étoile comme critère suffisant pour être rationnel, ainsi que sa réaction face aux indications de l'examinateur. L'enjeu est ici multiple : il faut comprendre que chaque mot se caractérise par sa longueur, puis se figurer le rôle crucial que jouent ici les réunions finies de progressions arithmétiques, et enfin imaginer comment se ramener à manipuler de telles réunions.

Question 5. L'ensemble $\mathcal{M}(\mathcal{S})$ est défini par l'expression rationnelle $\mathcal{S}^* \cdot \bigcup_{s \in \mathcal{S}} (s \cdot (\varepsilon + \mathcal{S}) \cdot s) \cdot \mathcal{S}^*$. Il est donc rationnel.

Cette question, beaucoup plus simple que la précédente et qui rappelle la question 1, a pour buts d'aider le candidat à s'approprier la définition des ensembles $\mathcal{M}(\mathcal{S})$, ainsi que de s'assurer qu'il est à l'aise avec la représentation d'ensembles par des expressions rationnelles.

Question 6. Soit m la longueur maximale des mots de $\mathcal{S}$, et $w = s_1 \cdot s_2 \cdots s_k$ un mot de $\mathcal{S}^*$ de longueur au moins $5m$. Puisque $k \geqslant 5$, on considère un préfixe $x = s_1 \cdot s_2 \cdots s_5$ de w, de longueur au plus $5m$, puis un mot $\sigma \in \mathcal{S}$ dont la suite s_1, s_2, s_3, s_4, s_5 contient deux occurrences, disons s_i et s_j.

Si $j = i + 1$ ou $j = i + 2$, on décide de pomper sur s_1 (c'est-à-dire de factoriser w comme $w = \varepsilon \cdot s_1 \cdot (s_2 \cdots s_k)$) si $i \geqslant 2$, et sur s_5 (c'est-à-dire de factoriser w comme $w = \varepsilon \cdot (s_1 \ldots s_4) \cdot s_5 \cdot (s_6 \cdots s_k)$) si $i = 1$ (donc $j \leqslant 3$). De la sorte, quelle que soit la puissance à laquelle on élève notre pompe (le facteur s_1 ou s_5), on reste dans $\mathcal{S}^*$, et les deux facteurs s_i et s_j restent à distance 1 ou 2 de l'autre.

Si $j \geqslant i+3$, on décide cette fois de pomper sur $s_{i+1} \cdot s_{i+2}$. En effet, si l'on élève cette pompe à la puissance 0, nos facteurs s_i et s_j se retrouvent à distance $j-i-2 \leqslant 2$ l'un de l'autre ; si on l'élève à la puissance 1, on obtient le mot w ; si on l'élève à une puissance supérieure ou égale à 2, on récupère un facteur de la forme $s_{i+1} \cdot s_{i+2} \cdot s_{i+1}$. Dans les trois cas, les mots obtenus appartiennent à $\mathcal{M}(\mathcal{S}) \cup \{w\}$.

Cette question manifestement difficile a pour but d'évaluer quelles idées intermédiaires le candidat peut mettre en œuvre, soit parce qu'il en a eu l'intuition, soit parce que l'examinateur les lui a suggérées. Ici, l'idée clé consiste à décider de ne manipuler que des facteurs s_i, de manière à faire comme si l'on travaillait sur les lettres d'un alphabet de cardinal au plus 4, puis à épuiser les uns après les autres les différents cas possibles, en s'attachant à chaque fois à obtenir deux facteurs s_i proches l'un de l'autre.

Question 7. Le langage $\mathcal{L}_3$ ne satisfait pas le lemme de l'étoile. En effet, si ℓ est assez grand, on devra choisir notre mot de pompage t au sein du préfixe $(\sigma_0 \cdot \sigma_1 \cdot \sigma_2)^\ell$, et soit sortir du langage rationnel $(\sigma_0 \cdot \sigma_1 \cdot \sigma_2)^* \cdot (\sigma_0 \cdot \sigma_1 \cdot \sigma_3)^*$, soit augmenter le nombre d'occurrences du facteur σ_2. Puisque $\mathcal{L}$ est la réunion disjointe du langage rationnel $\mathcal{M}(\mathcal{S})$ et de $\mathcal{L}_3$, où l'on a posé $\mathcal{S} = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$, il n'est alors pas rationnel non plus, ce sans quoi le langage $\mathcal{L}_3 = \mathcal{L} \setminus \mathcal{L}(\mathcal{S})$ serait rationnel lui aussi.

Cette question a pour but premier d'évaluer la familiarité du candidat avec les langages de la forme $\mathsf{L}_\ell = \{x^\ell y^\ell : \ell \geqslant 0\}$, connus pour être algébriques mais pas rationnels. Ici, $\mathcal{L}_3$ est l'image d'un tel langage par un morphisme, et il revient alors d'adapter avec rigueur à $\mathcal{L}_3$ la preuve d'irrationalité de L_ℓ. L'étude du langage $\mathcal{L}$ est ensuite censée être une simple formalité, qui vise simplement à s'assurer de la rigueur du candidat, amené à vérifier que $\mathcal{L}_3$ s'exprime de manière simple à partir de deux langages dont l'un est déjà connu pour être rationnel.

Par ailleurs, le caractère ouvert de la question est de toute évidence factice, au vu de la question 8 et de l'ambition que l'on a mise en avant en tout début de sujet. Cette question a donc aussi pour but secondaire de s'assurer que le candidat est à même de prendre du recul sur son sujet et d'avoir saisi la progression des questions qu'il est amené à traiter.

Question 8. Puisque $\mathcal{L}$ contient $\mathcal{M}(\mathcal{S})$, qui est $\mathcal{S}^*$-étoilé, $\mathcal{L}$ est également $\mathcal{S}^*$-étoilé, et il est même $\mathcal{L}'$-étoilé pour tout langage $\mathcal{L}'$ inclus dans $\mathcal{S}^*$, par exemple $\mathcal{L}' = \mathcal{L}$ lui-même. Par ailleurs, $\mathcal{S}^*$ ne contient aucun mot de $\mathcal{M}(\Sigma)$, donc $\Sigma^* \setminus \mathcal{L}$ contient $\mathcal{M}(\Sigma)$, qui est Σ^*-étoilé. Pour les mêmes raisons que précédemment, $\Sigma^* \setminus \mathcal{L}$ est donc Σ^*-étoilé, et même $\Sigma^* \setminus \mathcal{L}$-étoilé. Ainsi, $\mathcal{L}$ et $\Sigma^* \setminus \mathcal{L}$ satisfont tous deux le lemme de l'étoile.

Cette question assez difficile a pour but d'évaluer dans quelle mesure le candidat a compris les différents types de raisonnement mis en œuvre lors des questions précédentes. Les deux idées principales sont ici de comprendre les liens entre caractère $\mathcal{L}'$-étoilé et inclusion, ainsi que d'observer que $\mathcal{S}^*$ ne contient aucun mot de $\mathcal{M}(\Sigma)$. S'agissant de la dernière question du sujet, aucune indication n'était ici fournie aux candidats, qui devaient donc obtenir par eux-mêmes ces idées.