RAPPORT DE L'ÉPREUVE ÉCRITE D'INFORMATIQUE-MATHÉMATIQUES
FILIÈRE MP – CONCOURS INFO – SESSION 2020
ÉCOLES CONCERNÉES : ENS DE LYON, ENS DE PARIS, ENS DE PARIS SACLAY, ENS DE RENNES

Coefficients de l'épreuve (en pourcentage du total d'admission, modifiés pour tenir compte de l'absence d'oraux pour les ENS à la session 2020) :

LYON	**25,8%**
PARIS	**27,6%**
PARIS SACLAY	**26,3%**
RENNES	**10,0%**

MEMBRES DE JURY : F. CAPELLI & M. GLISSE & S. SCHMITZ

L'épreuve écrite d'informatique-mathématiques concerne les candidats aux quatre Écoles Normales Supérieures sur le concours INFO. Le nombre de candidats ayant composé était de 415 pour la session 2020 (pour 505 inscrits), le même nombre qu'en 2019. Les notes se sont échelonnées de 0 à 20 avec une moyenne de 8 et un écart-type de 4,19.

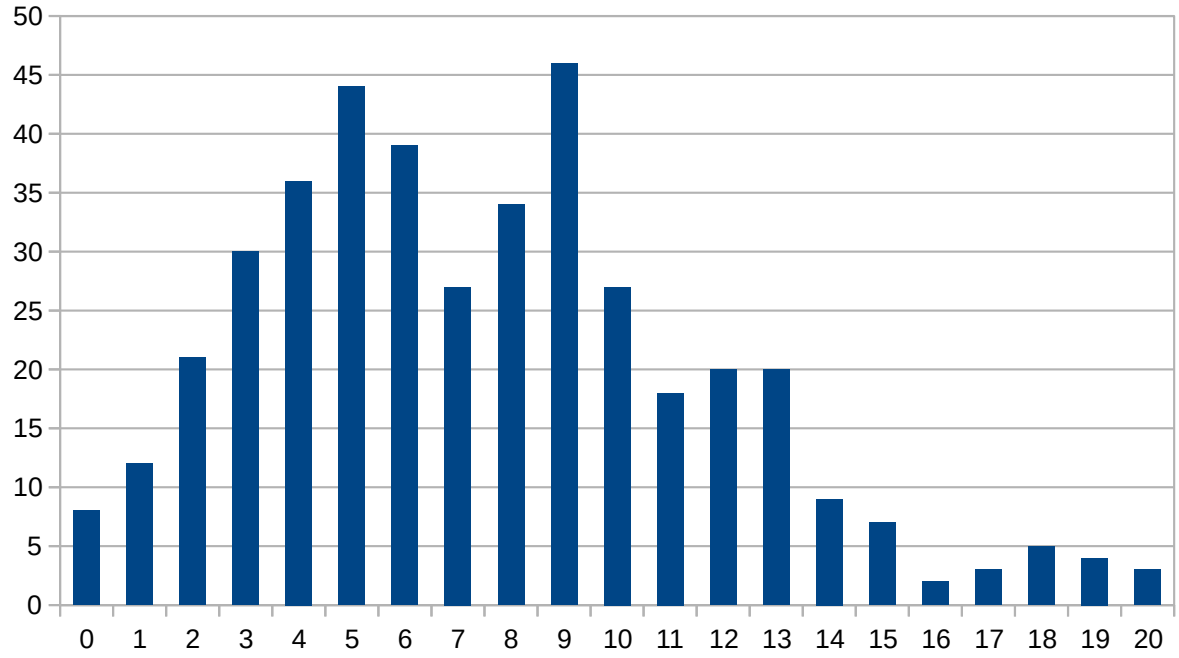

FIGURE 1 – Nombre de copies dans $[n; n+1[$.

1 L'épreuve

Afin de comparer la réussite à des questions indépendamment de leurs poids respectifs, les moyennes par question sont données ci-dessous sous la forme $n \times p$ où n est le nombre de points (poids) attribué à la question et p est donc la moyenne sur 1.

Le sujet 2020 portait sur la définition d'une sémantique pour un langage de programmation fonctionnel.

- La partie I introduit un système de type et un ordre partiel associé à ce système.
- La partie II introduit un langage de programmation et sa sémantique.
- La partie III étudiait des notions topologiques sur la sémantique de ce langage.
- La partie IV proposait de prouver un théorème de compacité sur cette sémantique.

2 Commentaires généraux

Le jury déplore le peu de soin apporté à la rédaction de nombreux candidats. Le jury tient à rappeler que cette épreuve est une épreuve de mathématiques-informatique et que des rédactions de qualité similaire à une épreuve de mathématiques sont attendues.

Le sujet étudiait la définition formelle de la sémantique d'un langage de programmation. Une certaine rigueur était attendue dans les démonstrations établissant des résultats sur le fonctionnement d'un programme. Trop de candidats se référaient à leur intuition de ce qu'un programme *devrait faire* sans forcément utiliser les définitions précises de la sémantique qui était donnée. Cela était particulièrement grave dans la définition et l'analyse de programmes récursifs, construits à partir de `rec`.

Une autre erreur couramment commise dans les copies a été l'utilisation de la valeur de l'interprétation des programmes dans les programmes eux-mêmes (avec les notations du sujet : certains candidats utilisaient des constructions de la forme $[\![p]\!]$ dans les programmes). Cette erreur, certainement une erreur d'inattention chez de nombreux candidats, est cependant grave puisqu'elle témoignait de l'incompréhension du cœur du sujet par certains candidats. En effet, cette erreur indiquait une confusion entre un objet syntaxique (le programme) et sa valeur (son interprétation), ce que ce sujet visait justement à bien définir.

Enfin, certains ignoraient les définitions données dans l'énoncé et utilisaient hors contexte des propriétés dont ils avaient l'habitude pour les réels.

3 Commentaires détaillés

Ci-dessous nous donnons pour chaque question le nombre de points associés, puis :

- le nombre de copies ayant reçu des points pour cette question (c'est à dire le nombre de copies où la question a été traitée et non nulle) ;
- la moyenne **sur 1** de toutes les copies ayant traité la question ;

— l'écart-type **sur 1** de toutes les copies ayant traité la question.

Pour chaque partie, on donne le nombre total de points qui pouvaient être obtenus, puis la moyenne et l'écart type des points obtenus.

Partie I (6,50pts | moy 0,54 stdev 0,25)

▷ **Question 1.1.a. [0,75 pt | 337 / 0,75 / 0,37]** Cette question ne présentait pas de difficultés particulières et a été globalement bien réussie. L'absence complète de justification a été légèrement pénalisée. Beaucoup de candidats ont oublié le cas de la suite constante égale à $\bot_{\mathsf{nat}}$. Le jury s'attendait à ce que tous les cas soient envisagés.

▷ **Question 1.1.b. [0,75 pt | 351 / 0,63 / 0,35]** Cette question ne présentait pas de difficultés particulières. Le jury a cependant rencontré de nombreuses justifications inexactes dans les réponses de candidats.

▷ **Question 1.2.a. [1 pt | 297 / 0,68 / 0,45]** Une question demandant une rédaction rigoureuse, sans oublier d'argument ni de cas.

▷ **Question 1.2.b. [0,50 pt | 353 / 0,69 / 0,30]** Sans grande difficulté en raisonnant par condition nécessaire et suffisante. Beaucoup de candidats oubliaient cependant de vérifier que la condition était suffisante : il faut toujours vérifier que la fonction proposée convient effectivement. Il était certes évident d'après les définitions qu'une fonction constante est croissante mais il convient au moins de l'annoncer.

Certaines copies ont traité la question pour $f \in [\![\mathsf{nat} \Rightarrow \mathsf{bool}]\!]$: attention à bien lire le sujet.

▷ **Question 1.2.c. [1,25 pt | 129 / 0,34 / 0,47]** La majorité des copies a défini une suite strictement décroissante au lieu de la suite croissante demandée. "Sans justification" n'empêche pas de vérifier au brouillon que les premiers termes sont bien dans l'ordre croissant.

▷ **Question 1.3.a. [1,50 pt | 306 / 0,52 / 0,35]** Beaucoup de copies ont manqué de rigueur dans cette question, se référant à des propriétés du sup qu'ils connaissent pour les réels mais qui n'avaient pas été clairement établis pour la définition donnée dans le sujet. Par exemple, beaucoup de candidats ont utilisé des propriétés de commutativité des sup sans le justifier. Enfin, beaucoup de copies oubliaient de prouver une partie du résultat.

Certains copies ont traité la question pour le cas particulier $\tau_1 = \mathsf{nat}$ et $\tau_2 = \mathsf{unit}$. Nous rappelons aux candidats qu'il ne faut pas se précipiter et bien lire le sujet avant de composer.

▷ **Question 1.3.b. [0,75 pt | 230 / 0,60 / 0,44]** Il était important d'utiliser les définitions exactes de l'énoncé, et de ne rien annoncer comme évident. Beaucoup de copies répondaient simplement que la fonction f' construite en 1.3.a convenait sans le justifier. D'autres montraient que f' était un sup mais point à point, sans revenir à la définition pour les fonctions.

Partie II (11pts | moy 0,37 stdev 0,22)

▷ **Question 2.1. [1 pt | 141 / 0,34 / 0,47]** La majorité des candidats a oublié au moins une des conditions. En particulier, beaucoup de copies oubliaient de dire que $t(x)(=t(y)=t(z))$ devait être un type de base. Cette condition n'est pas anodine : comment, en effet, décider de l'égalité entre deux objets de types non triviaux ?

▷ **Question 2.2.a. [1,25 pt | 403 / 0,55 / 0,33]** C'était l'occasion pour le candidat de montrer qu'il avait compris les notions d'interprétation et d'environnement en déroulant toutes les étapes. Certains candidats ont malheureusement simplement traité la question comme si p_e était un programme écrit en OCAML, sans parler explicitement d'interprétation.

▷ **Question 2.2.b. [0,50 pt | 208 / 0,51 / 0,50]** Beaucoup d'inattention, des x qui deviennent des n en milieu de programme, $x*2$ au lieu de $x/2$. La simplicité d'une question n'empêche pas de se relire, au contraire. Le jury a sévérement sanctionné ici (en mettant 0 à la question) la présence de $[\![p_e]\!]$ à l'intérieur du programme, qui indique une confusion entre syntaxe et sémantique (voir les commentaires généraux de la section précédente).

▷ **Question 2.3.a. [0,50 pt | 365 / 0,54 / 0,28]** Beaucoup de candidats ont résolu le cas le plus naturel, en oubliant toutes les possibilités d'avoir $\bot$. Il fallait en effet envisager le cas $e = \bot_{\mathsf{nat}}$ (ce qui a été relativement bien fait dans de nombreuses copies) mais aussi celui où $f(e-1) = \bot_{\mathsf{nat}}$.

À nouveau d'étranges mélanges avec des e qui deviennent n en milieu de ligne.

▷ **Question 2.3.b. [1,50 pt | 171 / 0,38 / 0,45]** Peu de bonnes réponses à cette question. Beaucoup de candidats se trompaient sur les différentes valeurs de $[\![p]\!]^{i+1}(\bot_{\mathsf{nat}\Rightarrow\mathsf{nat}})(e)$ en fonction de la valeur de e. Il fallait bien écrire une récurrence pour trouver que $[\![p]\!]^{i+1}(\bot_{\mathsf{nat}\Rightarrow\mathsf{nat}})(e) = e!$ si $e \leq i$ et $\bot$ dans tous les autres cas.

Le jury a été indulgents avec les copies présentant la bonne réponse avec une justification légère.

▷ **Question 2.3.c. [1,50 pt | 153 / 0,28 / 0,36]** Comprendre le fonctionnement de `rec` était le premier point difficile du sujet. Trop peu de copies ont proprement justifié les sup de fonctions, et certaines oublient encore le cas $\bot$.

Certaines copies utilisaient leur réponse fausse de 2.3.b pour répondre en 2.3.c avec une justification approximative. Or la réponse en 2.3.b ne leur permettait pas de démontrer correctement que $[\![\texttt{rec}p]\!](k) = k!$ pour tout $k \in \mathbb{N}$. Cela montre bien l'importance de bien rédiger ses réponses : un candidat attentif se serait rendu alors compte que sa réponse en 2.3.b était incorrecte. Le jury déplore l'attitude de certains candidats qui tentent de répondre quand même à une question alors que leur raisonnement est manifestement incorrect.

▷ **Question 2.4.a. [0,75 pt | 212 / 0,48 / 0,36]** Cette question ne présentait pas de difficultés particulières mais le jury a observé des confusions entre $\bot$ et `DIV`, entre f et p. Encore une fois, cela relève de la confusion entre syntaxe – `DIV` et p sont des **programmes** – et sémantique – $\bot$ et f sont les **valeurs** d'une interprétation.

Peu de copies ont justifié la correction de leur programme.

▷ **Question 2.4.b. [0,75 pt | 211 / 0,47 / 0,34]** Beaucoup ont bien vu qu'il suffisait d'utiliser `rec`, mais le mettent entre crochets (alors ce n'est plus un programme) ou le justifient mal (sup ponctuel vs sup de fonctions).

Certains candidats ici parlent de la conjecture de Syracuse dans leur démonstration pour justifier certains points. Cela était complètement inutile. Le jury ne l'a pas sanctionné mais il paraît important de rappeler aux candidats qu'ils doivent s'assurer de la correction de leurs arguments et ne pas simplement essayer de placer des mots-clés.

Le jury a apprécié l'humour mais n'a pas accordé de point aux candidats ayant répondu que si la conjecture de Syracuse était vraie alors `fun` $n \to$ `uu` convenait.

▷ **Question 2.5.a. [0,75 pt | 245 / 0,43 / 0,33]** Encore une fois, de nombreuses copies confondent syntaxe et sémantique en utilisant des crochets ou de $\bot$ dans un programme. Le jury attendait aussi une justification succincte expliquant pourquoi la fonction $f(\texttt{uu}) := \bot_{\mathsf{unit}}$ et $f(\bot_{\mathsf{unit}}) := \texttt{uu}$ n'était pas un élément de $[\![\mathsf{unit} \Rightarrow \mathsf{unit}]\!]$ (elle n'est pas croissante).

▷ **Question 2.5.b. [1 pt | 173 / 0,63 / 0,39]** Le plus simple ici était de montrer que $[\![\mathsf{nat} \Rightarrow \mathsf{unit}]\!]$ était indénombrable en construisant une bijection avec $\mathcal{P}(\mathbb{N})$. Un certain nombre de bijections annoncées par les candidats n'étaient en fait seulement des injections ou surjections, pas toujours dans le bon sens, et $\bot$ était parfois oublié. Si cela suffisait parfois (auquel cas le jury a très légèrement sanctionné), il convient d'être précis dans les termes employés.

Certains candidats ont essayé d'utiliser un argument diagonal mais peu ont réussi à l'écrire correctement.

▷ **Question 2.6. [0,50 pt | 104 / 0,26 / 0,38]** Encore une fois, le jury déplore les libertés prises par les candidats avec la syntaxe données par le sujet. On trouve des `if then` sans `else`, des $\bot$ et des

$[\![\cdot]\!]$ dans de nombreuses copies. La difficulté de cette question était justement d'utiliser `DIV` dans le `else` pour avoir un programme syntaxiquement correct et avec la bonne sémantique.

Le jury a légèrement sanctionné les candidats qui n'ont apporté aucune justification à la correction de leur programme. Si à ce stade du sujet, le jury peut accepter des justifications plus rapides sur certains points faciles, nous rappelons qu'il est important pour les candidats de toujours penser à apporter ces petites justifications, au moins pour montrer qu'ils sont conscients qu'il y a quelque chose à expliquer.

▷ **Question 2.7.a. [0,50 pt | 264 / 0,86 / 0,34]** Une question sans grande difficulté. Beaucoup de copies proposent des justifications un peu maladroites, mais le résultat était compris.

▷ **Question 2.7.b. [0,50 pt | 163 / 0,76 / 0,42]** Les candidats qui ont répondu ont en général bien deviné.

Partie III (15,75pts | moy 0,14 stdev 0,16)

Dans cette partie, les correcteurs ont arrêté de sanctionner les confusions entre $\bot$ et `DIV`.

▷ **Question 3.1. [0,75 pt | 285 / 0,74 / 0,37]** Prouver qu'un ensemble est ouvert nécessite l'exhibition d'un programme ! Les correcteurs attendaient donc que les candidats justifient ce qu'ils avancent avec des programmes. Dire qu'une fonction est "évidemment calculable" n'est pas suffisant car c'est justement tout l'intêret de la question. Les programmes n'étaient pas toujours bien typés, ce que les correcteurs ont sanctionné.

▷ **Question 3.2. [0,75 pt | 152 / 0,36 / 0,41]** La façon la plus simple de répondre à cette question était de simplement énumérer tous les sous-ensembles de $[\![\mathsf{unit}]\!]$ et d'indiquer ceux qui étaient ouverts ; les programmes nécessaires à cette justification ayant été définis en 2.5.a. Beaucoup de copies citent cependant la question 2.5.a mais avec des arguments vagues ou faux. Nous rappelons aux candidats que pour utiliser une question précédente du sujet, il faut avoir la même rigueur que lors de l'application d'un théorème et ne pas se contenter de citer la question en laissant le correcteur deviner le reste.

▷ **Question 3.3. [1 pt | 222 / 0,45 / 0,32]** Le jury attendait un programme explicite dont l'interprétation correspondait à $f^{-1}(U)$. Il convenait d'introduire deux programmes, p_f et p_U correspondant à f et à la fonction caractéristique de U pour écrire ces programmes. Trop de copies ont négligé ce point. Le jury attendait aussi une justification.

▷ **Question 3.4.a et 3.4.b. [0,75 pt | 211 / 0,43 / 0,33]** Les questions 3.4.a et 3.4.b étant très similaires, elles ont été notées ensemble. La plupart des candidats ont compris qu'il fallait utiliser $||_u$ et $\&_u$ mais, encore une fois, certains candidats prennent beaucoup de libertés avec la syntaxe ou oublient de justifier.

▷ **Question 3.5. [5 pt | 57 / 0,23 / 0,34]** Peu de candidats ont réussi cette question, qui était la première à présenter de réelles difficultés.

Le jury remarque qu'il était possible de simplifier en utilisant 2.7.b, mais qu'aucun candidat ne l'a fait, tous ayant essayer de construire directement le bon programme. La moitié des points de cette question était sur la correction du programme proposé par le candidat, l'autre moitié sur la justification.

Beaucoup de copies utilisaient des constructions récursives proches de la syntaxe de OCaml comme "`rec` $q = ...$".

▷ **Question 3.6.a. [0,75 pt | 124 / 0,50 / 0,47]** Cette question ne présentait pas de diffcultés particulières, il fallait juste être attentif à ce que cela signifie de quantifier universellement sur l'ensemble vide et poser $\forall_\emptyset = \mathsf{fun}\ p \to \mathsf{uu}$.

Certains candidats ont cependant répondu à tort que n'importe quel programme convenait pour $\forall_\emptyset$. Il fallait faire attention au fait que, comme $\forall e \in \emptyset, [\![p]\!](e) = \mathsf{uu}$ est toujours vrai, il faut donc avoir $[\![\forall_\emptyset\ p]\!] = \mathsf{uu}$ pour tout p et choisir alors un programme $\forall_\emptyset$ constamment égal à uu.

▷ **Question 3.6.b. [1 pt | 64 / 0,42 / 0,43]** Cette question présentait une petite difficulté car elle nécessitait une bonne compréhension des définitions du sujet, notamment sur le fait que $[\![p]\!]$ est une fonction croissante et que donc, si $[\![p]\!](\bot) = \mathsf{uu}$, alors $[\![p]\!](e) = \mathsf{uu}$ pour tout e (et qu'on a donc équivalence).

Peu de candidats ont traité cette question mais parmi ceux qui l'on traité, la plupart ont eu cette intuition, sans forcément toujours arriver à le justifier ou à construire le bon programme. Le jury a accordé des points aux candidats qui ont montré qu'ils avaient compris le principe sans forcément réussir à obtenir une démonstration complète.

▷ **Question 3.6.c. [0,75 pt | 56 / 0,48 / 0,48]** Cette question a globalement bien été traité par les candidats ayant bien répondu à 3.6.b. car l'idée générale était similaire. Quelques preuves fantaisistes mais les candidats ayant traité cette question ont en général la bonne idée.

▷ **Question 3.7. [5 pt | 54 / 0,18 / 0,26]** Cette question présentait un sens assez facile et un autre sens qui nécessitait de comprendre le comportement de $\forall_U$ sur des programmes p_i tels que $[\![p_i]\!](k) = \mathsf{uu}$ ssi $k \leq i$ puis d'inverser des sup.

Beaucoup n'ont abordé que le sens facile, ce pour quoi le jury a donné peu de points, mais quelques uns ont parfaitement répondu à la question.

Partie IV (10,50pts | moy 0,03 stdev 0,09)

▷ **Question 4.1. [1 pt | 58 / 0,30 / 0,35]** Beaucoup de candidat se sont référés à leur intuition de l'exécution habituelle des langages de programmation qu'ils utilisent, en parlant vaguement de terminaison en nombre fini d'étapes. Le jury a accordé une petite partie des points dans certains cas lorsque le candidat semblait avoir compris l'idée général. Le jury attendait cependant un argument formel de continuité.

▷ **Question 4.2. [0,50 pt | 65 / 0,71 / 0,42]** Une bonne réponse aurait dû faire intervenir l'environnement, mais comme aucune copie ou presque ne l'a fait, des arguments plus rapides ont été tolérés. Le jury observe qu'à partir de ce point, les candidats semblent commencer à manquer de temps et la qualité de la rédaction diminue.

▷ **Question 4.3.a. [1,50 pt | 29 / 0,28 / 0,34]** Il fallait utiliser ici les calculs mené en 4.2 pour réussir à mener une démonstration formellement correcte. Encore une fois, la fin de l'épreuve se faisant sentir, on observe des démonstrations brèves mais parfois convaincantes.

Certains candidats cependant semblent manipuler les symboles au hasard sans essayer de justifier ce qu'ils font.

▷ **Question 4.3.b. [0,75 pt | 18 / 0,37 / 0,43]** Cette question reposait sur le fait que $K(p) = \{k(tt :: s, p) \mid s \in B^{\mathbb{N}}\} \cup \{k(ff :: s, p) \mid s \in B^{\mathbb{N}}\}$ et que les questions précédentes établissaient des relations entre $k(b :: s, p)$ et $k(s, p/b)$. Peu de copies ont été très explicites sur les relations qu'elles utilisaient pour parvenir au résultat, certaines proposant alors des démonstrations un peu douteuses.

▷ **Question 4.3.c. [0,75 pt | 20 / 0,38 / 0,47]** Il était fondamental dans cette question d'utiliser 4.3.b pour construire une suite de booléens $s = (b_n)$ tels que $K(p/b_1/\ldots/b_n)$ était non borné pour tout n, ce qui contredisait 4.3.a pour $n = k(s, p)$. Des candidats construisaient une suite de Booléens sans mentionner explicitement 4.3.b et donc, sans mentionner que $K(p/b_1/\ldots/b_n)$ était non borné. Ils rataient donc l'argument crucial qui leur permettait d'avoir une contradiction avec 4.3.a.

▷ **Question 4.4.a. [0,50 pt | 25 / 0,48 / 0,36]** Beaucoup de copies ont négligé l'hypothèse $k(b :: s, p) > 0$ et aboutissent donc à des démonstrations fausses ou incomplètes. La fatigue des candidats se fait ressentir à ce moment

▷ **Question 4.4.b. [0,50 pt | 16 / 0,39 / 0,35]** La question 4.4.a était cruciale pour répondre ici mais il fallait d'abord s'assurer que si $|p| > 0$, alors $k(s, p) > 0$ pour tout s avant de pouvoir appliquer le résultat de 4.4.a, ce que de plusieurs candidats ont oublié de faire.

▷ **Question 4.5. [5 pt | 1 / 0,12 / 0,34]** Question difficile, une seule copie a reçu des points. Une poignée de candidats ont eu l'intuition qu'il fallait essayer de tester p sur toutes les suites de $\mathbb{B}^{\mathbb{N}}$ et qu'on pourrait s'en sortir en temps fini grâce à la finitude de $|p|$ mais seule une copie a proposé un programme à peu près correct et esquissé une démonstration.