Composition d'Informatique A, Filières MP-MPI (XULSR)

1 Le sujet

Ce sujet s'intéresse à la génération d'arbres de décision, avec pour cas d'application l'identification des plantes.

La partie I présente la modélisation probabilistes des plantes utilisées par le sujet et demande des calculs simples de probabilités conditionnelles ainsi que leurs implémentations.

La partie II propose de calculer l'arbre de décision optimal (au sens de la hauteur moyenne) en énumérant un-à-un tous les arbres et en choisissant celui de hauteur minimale. La principale difficulté était dans l'écriture d'algorithmes d'énumération récursive d'arbres.

La partie III propose de construire un algorithme glouton basé sur la notion d'entropie moyenne.

La partie IV propose une optimisation de l'énumération de la partie II en utilisant une approche *branch-and-bound*. Il est demandé ici de prouver que l'optimisation est correcte.

La partie V propose enfin un modèle logique pour décrire des questions plus complexes que l'observation d'un seul caractère. La problématique est l'évaluation d'une formule de la logique sur un élément du modèle avec une phase de pré-calcul pour mettre la formule en forme normale disjonctive afin de simplifier l'évaluation.

2 Remarques générales

Il est rappelé que la présentation de la copie est importante. Trop de candidat·e·s écrivent de manière difficile à lire. En particulier pour les questions de code, il est important de bien expliquer chaque fonction intermédiaire et de soigner sa présentation. Même si la propreté de la copie ne joue pas dans la note, une réponse ne peut rapporter des points que si les correcteur · ice · s arrivent à la déchiffrer.

Lecture de l'énoncé et invariants. Très souvent les candidat · e · s ont lu rapidement le sujet et n'ont pas fait attention à tous les invariants liés aux structures de données en jeu, conduisant à des codes faux. Les fonctions doivent toujours retourner une valeur qui satisfait les invariants énoncés pour le type correspondant.

3 Commentaires détaillés

Pour chaque question, sont indiqués entre crochets le pourcentage de copies ayant traité la question et le pourcentage de copies ayant obtenu la totalité des points, en fonction des filières.

Question 1 [MP: 98%, 84%; MPI: 100%, 85%]. La question consistait en une simple application numérique de lois classiques de probabilité et a été bien réalisée dans son ensemble.

Question 2 [MP: 93%, 79%; MPI: 95%, 66%]. Il s'agissait ici de faire le lien entre les données (σ et s_i) et la défintion de la mesure de probabilité. Il fallait donc faire le calcul et montrer qu'on trouvait bien le résultat naturel (par exemple que $P(S = s) = \sigma(s)$).

Question 3 [MP: 80%, 23%; MPI: 78%, 25%]. Il était attendu ici une application de la loi de Bayes, une simplification à l'aide de la question 2 et une explication informelle de ce que la formule signifiait. L'explication informelle a souvent manqué.

Question 4 [MP: 45%, 8%; MPI: 34%, 5%]. Il fallait ici faire des calculs similaires à ceux de la question 1. Il était tentant d'utiliser un argument d'indépendance mais les évènements $C_i = v$ et $C_j = v'$ ne sont pas indépendants, ils le sont uniquement lorsque conditionnés relativement à une espèce.

Question 5 [MP: 98%, 8%; MPI: 97%, 12%]. Simple question de mise en code des réponses de la partie précédente. Il fallait cependant bien respecter les invariants, notamment pour `proba_de` et `repondere` de ne pas avoir de coefficients nuls dans la liste.

Question 6 [MP: 89%, 70%; MPI: 89%, 72%]. Simple application de la question précédente avec les formules de la question 1.

Question 7 [MP: 72%, 47%; MPI: 71%, 40%]. Encore une simple application, il faut calculer la probabilité de la valeur, si elle est nulle lever une exception et sinon utiliser `repondere`.

Question 8 [MP: 82%, 32%; MPI: 83%, 35%]. Il fallait bien faire attention à la définition de l'arbre de décision et se rendre compte que dans l'autre ordre des questions, il y a une branche de hauteur 1 ce qui fait un arbre plus petit en moyenne que celui de l'énoncé.

Question 9 [MP: 38%, 13%; MPI: 65%, 29%]. Question d'algorithmique difficile qui consiste à calculer le produit cartésien d'un tableau de liste. La solution récursive en utilisant `List.concat_map` est la plus courte, mais d'autres solutions plus indirectes ou impératives ont aussi été acceptées.

Question 10 [MP: 10%, 1%; MPI: 29%, 9%]. Une des questions les plus difficiles du sujet qui consiste à énumérer récursivement tous les arbres de décision possibles. La principale difficulté est que la récursion nous donne une liste de possibilité pour chaque enfant lorsque l'on a choisi une question initiale et qu'il faut ensuite retourner toutes les possibilités en choisissant une possibilité par enfant (en utilisant la fonction `choix_possibles`).

Question 11 [MP: 58%, 30%; MPI: 74%, 49%]. Question simple qui nécessite le calcul de la hauteur moyenne d'un arbre.

Question 12 [MP: 77%, 12%; MPI: 85%, 12%]. Question de mathématiques sur l'entropie de Shannon. Pour conclure que la distribution uniforme est bien le maximum de H, on peut utiliser l'inégalité de Jensen sur le logarithme, mais relativement peu de copies ont trouvé cet argument.

Question 13 [MP: 64%, 2%; MPI: 77%, 14%]. Il s'agit ici d'implémenter la formule, sachant que la formulation de l'énoncé a une ambiguité: il est entendu que la somme ne porte que sur les v pour lesquels $P^\sigma(C_i = v)$ est non nul (car sinon le terme de la somme disparaît). Mais dans le code, il faut bien expliciter cela, sinon le calcul de $\sigma[i := v]$ lève une exception.

Question 14 [MP: 25%, 5%; MPI: 48%, 18%]. Le calcul de l'algorithme glouton est similaire à `enumerate` mais plus simple car on calcule un seul arbre à chaque fois, il n'y a pas de choix entre plusieurs possibilités.

Question 15 [MP: 36%, 27%; MPI: 46%, 38%]. Question relativement bien traitée, il faut dessiner les deux arbres et se rendre compte que l'un est légèrement plus petit.

Question 16 [MP: 24%, 15%; MPI: 25%, 15%]. Le but de cette question est de montrer que l'algorithme glouton ne calcule pas l'arbre optimal dans cet exemple.

Question 17 [MP: 3%, 0%; MPI: 12%, 2%]. Cette partie de preuve de programmes est restée largement intouchée. Dans cette première question, il s'agit d'énoncer les pré-conditions de `arbre_optimal_avec_oracle`, en particulier sur `oracle`.

Question 18 [MP: 1%, 1%; MPI: 14%, 2%]. Peu ont abordé cette question, centrée sur l'élaboration d'un invariant de boucle pour la ligne 11. Il s'agissait de dire qu'à chaque tour de la boucle, arbre_optimal contient bien l'arbre optimal parmi tous ceux ayant pour racine un critère déjà testé, ou rien si aucun critère n'a été testé.

Question 19 [MP: 2%, 1%; MPI: 8%, 3%]. Cette question relativement simple consiste à fermer la récursion, c'est-à-dire à déclarer une fonction récursive qui appelle `arbre_optimal_avec_oracle` avec elle-même. La preuve de correction se fait par induction sur le nombre de critère, car l'oracle est toujours appelé avec un ensemble de critères plus petit.

Question 20 [MP: 1%, 0%; MPI: 5%, 0%]. Sans doute la question la plus difficile du sujet. Il faut ajouter une approche *brand and bound* qui permet de couper le calcul d'un arbre dès que son score dépasse celui de l'arbre optimal.

Question 21 [MP: 33%, 22%; MPI: 48%, 32%]. Question assez directe en dépliant les définitions.

Question 22 [MP: 24%, 4%; MPI: 40%, 12%]. Question de complexité plutôt bien réussie dans l'ensemble même si la formule exacte est rarement trouvée.

Question 23 [MP: 12%, 4%; MPI: 18%, 8%]. Il faut ici remarquer qu'on peut évaluer une clause directement par induction sur sa structure, la conjonction se transformant en produit des probabilité grâce à la structure des clauses.

Question 24 [MP: 11%, 6%; MPI: 22%, 15%]. La conjonction de deux clauses est presque déjà une clause, sauf si les deux clauses parlent du même critère, auquel cas il faut remarquer que $c_i \in V_1 \cap c_i \in V_2$ est équivalente à $c_i \in (V_1 \cap V_2)$.

Question 25 [MP: 7%, 1%; MPI: 14%, 8%]. Simple induction sur la formule φ, le cas difficile de la conjonction étant traité à la question d'avant.

Question 26 [MP: 2%, 0%; MPI: 5%, 2%]. Il faut remarquer que la mise en clause a une complexité acceptable pour la question, et qu'on peut évaluer une disjonction de clause simplement avec la formule classique $P(A \vee B) = P(A) + P(B) - P(A \cap B)$ ce qui mène à une complexité exponentielle en la taille de la formule mais indépendante de N.