Informatique

Présentation du sujet

Le sujet porte sur la simulation de la cinétique d'un gaz parfait à l'échelle microscopique. Les molécules sont assimilées à des sphères et les interactions entre molécules et avec les parois sont modélisées par des chocs élastiques, à l'image du modèle de Boltzmann. L'ensemble des étapes de simulation est abordé, de l'initialisation à l'extraction des résultats, sur des cas à une ou plusieurs dimensions.

La première partie aborde la problématique d'initialisation aléatoire des positions et vitesses des molécules, en évitant toute interférence entre molécules et avec le récipient. Divers algorithmes sont envisagés et évalués, sur des modèles à une ou plusieurs dimensions.

La seconde partie porte sur la simulation du mouvement et des interactions, conduisant à l'écriture de plusieurs fonctions réemployées par la suite.

La troisième partie porte sur la manipulation d'une structure de données contenant les évènements (les chocs) possibles recensés et ordonnés.

La quatrième partie développe le cœur de l'algorithme de simulation en structurant les successions d'évènements et le déroulement temporel, tout en réemployant les fonctions précédemment écrites.

La cinquième partie clôt l'étude en proposant d'élaborer des requêtes SQL d'extraction de grandeurs physiques dans la base de données ayant stocké les résultats de simulation.

Analyse globale des résultats

Le sujet est relativement long pour le temps imparti mais est progressif et permet à tous les candidats de s'exprimer.

Si on a pu remarquer quelques belles copies, on ne peut que déplorer un trop grand nombre de copies avec un niveau faible. Le sujet permettait pourtant de glaner facilement des points sur les premières questions.

L'expression française en termes de contenu comme de forme est souvent problématique et peut pénaliser certains candidats qui n'arrivent pas à exprimer correctement leurs idées.

Le langage Python semble mieux maitrisé que les années précédentes à l'exception de l'utilisation des bibliothèques. Quelques rares candidats ont visiblement négligé la formation en informatique et se contentent de répondre aux questions ne relevant pas immédiatement d'informatique. Ces copies conduisent à des notes très faibles.

Beaucoup de candidats ont fait l'impasse sur le langage SQL ce qui est sans doute une erreur car les questions portant sur cette partie du programme sont relativement simples et permettent de gagner un nombre non négligeable de points.

Les petites erreurs syntaxiques n'ont pas été retenues par le jury comme un élément discriminatoire, dans la mesure où elles ne cachent pas des erreurs de fond. Les réponses pertinentes d'un point de vue algorithmique sont valorisées.

Certaines copies proposent des programmes particulièrement élégants et concis, et reflètent un vrai recul sur les différentes stratégies de programmation. Ces copies ont été valorisées.

Commentaires sur les réponses apportées et conseils aux futurs candidats

Au regard des copies évaluées, le jury conseille aux futurs candidats de prêter attention aux remarques suivantes.

L'indentation en python délimite les blocs d'instructions et doit apparaitre clairement dans la rédaction. Toute rédaction claire est bienvenue ; bien souvent, un trait vertical marquant l'alignement du bloc d'instruction est suffisant.

L'initialisation d'une variable dans une boucle ou hors de la boucle n'a pas les mêmes conséquences pour l'algorithme.

Le nombre d'itérations d'une boucle doit être bien réfléchi pour s'assurer que les indices des éléments d'une liste appelée dans la boucle sont bien définis. L'instruction `range(n)` produit `n` entiers compris entre `0` et `n-1` et permet dont d'effectuer `n` itérations indicées de `0` à `n-1`.

Les opérateurs booléens `and` et `or` fonctionnent séquentiellement en arrêtant l'évaluation dès que la valeur logique est établie, ils ne sont donc pas commutatifs. Ainsi, l'ordre des conditions dans une instruction `while` a souvent de l'importance.

Lorsque le sujet précise explicitement les paramètres des fonctions et les valeurs renvoyées, ainsi que leur type, il convient de veiller à les respecter. Certaines variables en argument peuvent être modifiées par une fonction sans nécessairement être renvoyées.

Beaucoup de questions sont indépendantes et généralement le prototype d'une fonction, donné dans une question, permet de l'utiliser dans les questions suivantes.

Les opérateurs classiques (+, *, etc.) n'ont pas toujours le même sens selon les types des opérandes (en particulier pour les listes et les tableaux numpy).

La concision et l'élégance des programmes sont appréciées dans l'évaluation. Les candidats qui réinvestissent les fonctions déjà codées sont valorisés par rapport à ceux qui recopient les lignes de code équivalentes.

Des listes de conditions en cascade nuisent à la lisibilité de l'algorithme. Une condition booléenne bien choisie distingue les candidats dont la pensée est claire.

Des noms de variables explicites aident à la compréhension du code. De trop nombreux candidats utilisent des noms de variables non significatifs (a, b, c ...) ce qui nuit à la compréhension du programme. La clarté du programme (en particulier le choix des noms de variables) ainsi que la présence de commentaires opportuns sont prises en compte dans l'évaluation.

Lors d'un calcul de complexité, une justification minimale est attendue.

L'ordre des questions importe. Il faut prendre soin de rédiger les réponses aux questions en respectant leur ordre dans le sujet.

La qualité d'expression (l'orthographe notamment) et la qualité visuelle de présentation relèvent des compétences de communication indispensables à un candidat à une école d'ingénieurs. Le correcteur n'attribue les points qu'aux éléments de réponse qu'il parvient à lire et à comprendre. Les copies obscures et difficiles à comprendre sont pénalisées.

Les variables utilisées dans une fonction doivent être définies dans cette fonction ou être explicitement définies comme variables globales (soit par le sujet, soit par le candidat). Les candidats sont invités à bien lire l'annexe contenant certaines fonctions utiles pour traiter le sujet.

I Initialisation

La première partie vise à évaluer plusieurs stratégies d'initialisation de la simulation, d'abord en une dimension, puis en trois dimensions.

Les questions 1 à 5 sont très bien réussies. Quelques rares copies ont confondu l'opérateur * pour les tableaux numpy et ce même opérateur pour les listes (question 1). Quelques candidats détaillent en français les opérations élémentaires réalisées (multiplication, tirage aléatoire, etc.) alors qu'il était attendu une explication du sens de ces lignes pour le problème posé. Un schéma était souvent plus clair qu'un long discours pour les questions 3 et 4. Étrangement, les questions 6 (tirer un nombre aléatoire entre deux bornes) et 7 (comportement d'un algorithme) sont moins bien abordées.

La question 10 demande d'écrire un algorithme décrit dans l'énoncé. Le jury a accepté tout algorithme qui permettait de répartir sans interférence les boules, même lorsque la répartition n'est pas équiprobable. Beaucoup de propositions partent du principe que la liste est triée après le tirage aléatoire. Un certain nombre d'algorithmes proposés s'éloigne notablement des consignes du sujet. La question 12 est souvent correctement traitée pour N=1, beaucoup plus rarement pour N=5 presque jamais pour N=2.

La question 13 permettait d'envisager une reformulation du code donné pour une simulation en trois dimensions. De façon inattendue, la norme euclidienne a posé beaucoup de difficultés.

II Mouvement des particules

La deuxième partie s'attache à décrire le mouvement des particules, d'abord par les lois physiques (données), puis en développant les fonctions associées à chaque phénomène (vol libre, rebond sur la paroi ou choc entre particules).

L'analyse des lois physiques a posé peu de difficultés aux candidats, bien que le jury constate que la notion de mouvement rectiligne uniforme ne soit pas toujours claire. Les fonctions à écrire étaient très simples, mais ont permis de distinguer les candidats qui proposent une formulation élégante en une ligne et d'autres dont le code est laborieux. Le sujet demandait une fonction qui mette à jour l'état de la particule par effet de bord et renvoie `None` ; certains candidats renvoient la particule.

III Inventaire des évènements

La troisième partie permet d'élaborer une structure de données mémorisant le catalogue des évènements anticipés (chocs et rebonds) et de définir les fonctions de manipulation de cette structure (initialisation, ajout d'évènements).

Beaucoup de candidats ont eu des difficultés à traduire informatiquement les conditions booléennes de collision entre particules et paroi pour les fonctions `tr` et `tc`. Certains n'utilisent qu'une seule structure conditionnelle `if` judicieusement choisie tandis que d'autres en utilisent 6 à 8, le code devenant très difficile à lire.

L'ajout d'un évènement nécessitait d'insérer un élément dans une liste triée. Certains oublient d'envisager le cas où l'élément à insérer se place au début ou à la fin de la liste.

L'ajout de tous les évènements relatifs à une particule est relativement bien abordé, mais beaucoup de candidats oublient de vérifier que les fonctions `tr` et `tc` ne renvoient pas None avant d'ajouter le résultat au catalogue. Certains ont cherché à sélectionner l'évènement le plus proche, contrairement à ce que demande le sujet.

L'initialisation du catalogue n'a pas posé de difficulté, mais le calcul de complexité nécessitait de remarquer que la taille du catalogue n'est pas en $O(N)$, ce que très peu de candidats ont vu. Un calcul de complexité ne se limite pas toujours à compter les boucles `for` imbriquées.

IV Simulation

Seuls les meilleurs candidats ont abordé correctement cette partie, qui vise à mettre en œuvre les étapes de simulation.

La fonction `etape` permet de tenir compte du déplacement rectiligne uniforme des particules entre deux évènements (vol), et de traiter l'évènement (le rebond ou le choc). Certains candidats oublient la phase de vol, soit pour toutes les particules, soit pour les particules subissant l'évènement.

La mise à jour du catalogue est souvent incomplète, en particulier pour la gestion des dates. De même, la gestion correcte des évènements invalides et du temps dans la simulation est rarement juste.

V Exploitation des résultats

La cinquième partie demande d'élaborer trois requêtes SQL d'extraction de résultats. Bien qu'elles soient en fin de sujet, ces questions sont la plupart du temps abordées, avec des résultats plutôt corrects. L'instruction GROUP BY, relative aux fonctions d'agrégation, n'est pas toujours connue. Certaines requêtes utilisent des jointures inutiles.

Conclusion

Le sujet aborde une large partie du programme d'informatique commune. Le choix d'un sujet s'appuyant la simulation d'un phénomène physique par une approche numérique, impliquant une part d'algorithmique, assure une cohérence avec la formation d'ingénieurs. Cette approche sera reconduite sur des problématiques de simulation ou d'algorithmique en informatique, à partir du programme des trois semestres d'informatique.

Les résultats à cette épreuve montrent un certain clivage entre des étudiants ayant un niveau faible et d'autres qui ont acquis des compétences en informatique bien que celles-ci doivent encore être affirmées. Le jury encourage les futurs candidats à travailler l'informatique en alliant réflexion sur feuille de papier et mise en œuvre des algorithmes sur ordinateur.