

An illustration of a desk setup. In the center is a laptop with a screen showing lines of code in blue and orange. To the left of the laptop is a clipboard with a white sheet of paper and a dark blue clip. To the right of the laptop are a pair of dark blue-rimmed glasses, a white coffee cup with orange liquid, and a dark blue pen. Below the laptop, a pair of hands is shown typing on the keyboard. In the bottom left corner, there is a white envelope. The background is a solid blue color.

Complexity Analyzer

Présentation

Benjamin Besnier



Déterminer la complexité temporel d'un algorithme en disposant de son code source



Par régression linéaire



Dans un temps convenable



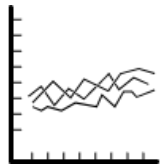
Plusieurs exécutions de l'algorithme pour des tailles d'entrées différentes

- Collecte du temps d'exécution pour chacune des entrées
- Construction du tableau permettant la régression



Réalisation d'une première régression linéaire

- Méthode des moindres carrés
- Si impossibilité de conclure, application de la seconde approche



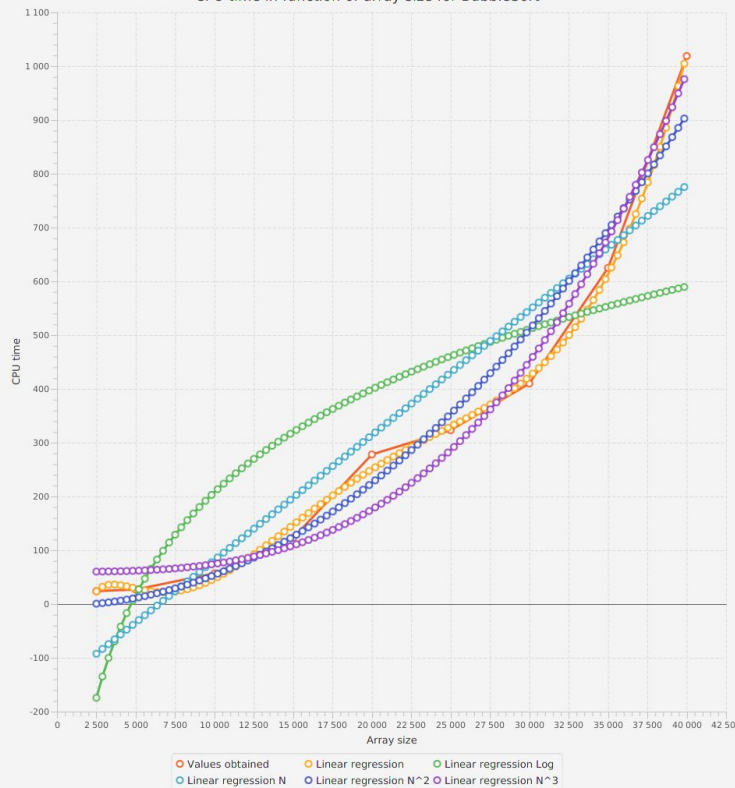
Réalisation d'une régression par classe de complexité

- Calcul de l'erreur (la somme des carrés des écarts aux points connus)
- Détermination de la classe de complexité

Résultats

3

CPU time in function of array size for BubbleSort



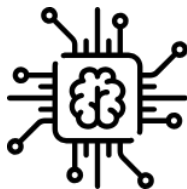
Résultats obtenus pour le tri à bulle

Régression	$O(\text{Log})$	$O(N)$	$O(N^2)$	$O(N^3)$
Erreur	29280.9	9225.85	146.96	1772.99
Indicateur	19824%	6177%	0%	1106%



Étude de codes étudiants posté sur caséine

→ En fonction des codes soumis, déterminer la complexité de nouvelles soumissions



Approche par machine learning

→ Permettre de distinguer les classes en $\log(n)$



Intégrer le code sur caséine

→ Déterminer la complexité des codes soumis par les étudiants, ce qui pourrait amener un nouveau critère d'évaluation



Fin

5

Merci de votre attention, si vous avez des questions n'hésitez pas