Université Paris Diderot, Master 1 II et ISIFAR

Projet d'algorithmique - étude et présentation d'un algorithme

Année universitaire 2007-2008

Déroulement du projet

Ce projet s'effectue par binôme. Chaque binôme doit

- Choisir un algorithme dans la liste proposée et s'inscrire auprès de M Asarin par mél jusqu'au 29/11, ou auprès de Mme Eguienta ensuite. Les premiers venus auront bien sûr un choix plus large. Jusqu'au 29/11 les algos disponibles seront affichés sur la page du cours.
- Etudier l'algorithme en utilisant la littérature et l'Internet.
- Préparer un exposé de 25' sur l'algorithme.
- S'inscrire auprès de Mme Eguienta pour une soutenance orale. Les inscriptions seront ouvertes en décembre.
- Présenter l'exposé lors d'une soutenance orale de 30' en janvier 2008. Vous aurez à votre disposition un vidéo-projecteur, et si besoin est, un PC portable. Il n'y a pas de rapport à rendre. La date sera communiquée ultérieurement.

Contenu technique

Vous devez identifier et présenter clairement les éléments suivants :

- Le problème algorithmique
- L'algorithme qui résout ce problème
- Un exemple
- La preuve de correction
- Détails d'implémentation efficace, choix de structures de données
- Analyse de complexité
- Vos sources d'informations.

Quelques conseils

- Après avoir compris et étudié l'algorithme, prenez votre temps pour préparer la présentation.
- Soyez très pédagogiques. Il est possible qu'un membre du jury ne connaît pas l'algorithme, il doit le comprendre en suivant votre exposé.
- Il **n'est pas demandé** de programmer l'algorithme, vous pouvez le faire seulement si vous en avez envie.
- Si vous souhaitez vous pouvez donner dans votre exposé autre information pertinente : histoire du problème et de l'algorithme, applications etc. C'est conseillé en cas où votre algorithme est facile.
- Si vous avez *vraiment* besoin d'un papier de recherche disponible sur un site payant d'éditeur, contactez M Asarin.

Les algorithmes

1. Composantes fortement connexes : algorithme de Tarjan. *strongly connected components*
2. Composantes fortement connexes : algorithme de Kosaraju. *strongly connected components*
3. Chemin le plus court pour tous les couples : algorithme de Johnson. *all pairs shortest path*
4. Chemin le plus court pour tous les couples : algorithme par carré de matrice en algèbre min-plus. *all pairs shortest path*
5. Algorithme de tri Radix-sort.
6. Problème 2SAT et un algorithme de solution.
7. Multiplication des entiers : algorithmes de Karatsuba-Ofman et/ou Toom-Cook.
8. Multiplication des matrices : algorithme de Strassen.
9. Arbres 2-3-4 et algorithmes associés.
10. Arbres rouges-noirs et algorithmes associés.
11. Arbres Patricia et algorithmes associés. *Patricia trees*
12. Flot maximum dans un graphe : méthode de Ford-Fulkerson. *max-flow*
13. Flot maximum dans un graphe : algorithme d'Edmonds-Karp. *max-flow*
14. Problème de mariage dans un graphe biparti. *matching in bipartite graph*
15. Recherche de sous-chaine : algorithme Shift Or. *string-matching*
16. Recherche de sous-chaine : algorithme de Galil-Seiferas. *string-matching*
17. Distance de Levenshtein entre séquences et l'algorithme de Levenshtein.
18. Recherche de deux points les plus proches : cas planaire. *closest pair of points*
19. Problème du sac à dos : algorithme pseudo-polynomial par programmation dynamique. *knapsack problem*
20. Recherche de PGCD : algorithme binaire. *binary GCD*
21. Tableaux de suffixes et algorithmes associés. *suffix array*
22. Enveloppe convexe : algorithme de Graham. *convex hull*
23. Enveloppe convexe : algorithme de Jarvis. *convex hull*
24. Arbre couvrant minimum : algorithme de Boruvka. *spanning tree*