Algorithmique 1

L3 RI

Table des matières

1 Algorithmes

1.1 Tris

- ▷ Tri par insertion : $O(n^2)$
  Considérer chaque élément un à un pour l'insérer à sa bonne place (penser à un jeu de cartes)
- ▷ Tri fusion : $O(n \log n)$
  Paradigme diviser pour régner, diviser en deux sous-problèmes
- ▷ Tri Shell : $O(n^2)$
  Suite de tris par insertion sur chaque constituant d'une partition du tableau
- ▷ Tri par tas : $O(n \log n)$
  Utiliser une structure de file de priorité, ici un tas
- ▷ Optimalité : $\Omega(n \log n)$ nécessaire, arbres de décision

1.2 Arbres binaires

- ▷ Arbre binaire : $1 + h \leq n \leq 2^{h+1} - 1$
- ▷ Arbre binaire presque complet : $2^h \leq n \leq 2^{h+1} - 1$
- ▷ Tas

- ▷ Arbre binaire de recherche (ABR)
  La recherche d'un élément ne suit qu'une branche, problème si arbre non équilibré
- ▷ Arbre AVL
  Rééquilibrage d'un arbre par des rotations : $\log_2(n+1) \leq h \leq 1.44 \log_2 n$

1.3 Graphes

- ▷ Graphes orientés, pondérés
- ▷ Implémentations par liste d'adjacence ou matrice d'adjacence
- ▷ Parcours en profondeur
  Valeurs de `Pre` et `Post` traitement, types d'arc, détection de cycles, tri topologique, composantes fortement connexes (*Algorithme de Kosaraju*), graphe quotient
- ▷ Parcours en largeur
  Recherche d'un plus court chemin (*Algorithme de Dijkstra*, *algorithme A**)
- ▷ Arbre couvrant de poids minimal (*Algorithme de Kruskal*, *Algorithme de Prim*)

1.4 Algorithmes gloutons

- ▷ Prendre un choix localement meilleur
- ▷ Algorithmes de Kruskal, de Prim
- ▷ Rendu de monnaie
- ▷ Couverture d'ensemble (exemple où le choix glouton ne donne pas forcément la solution optimale)

1.5 Programmation dynamique

- ▷ Paradigme de conception d'algorithmes
- ▷ Définir les sous-problèmes, en revoyant à la baisse l'objectif si nécessaire
- ▷ Trouver une relation de récurrence
- ▷ Écrire l'algorithme (mémoïzation)
- ▷ Exemples
  Recherche plus court chemin dans un graphe (*Algorithme de Floyd-Warshall*, *algorithme de Bellman-Ford*), recherche plus longue sous-suite croissante, problème du sac à dos

1.6 Flots

- ▷ Problème du flot maximal
- ▷ Algorithme de Ford-Fulkerson
- ▷ Réduction du problème de couplage maximal au problème de flot maximal

1.7 Programmation linéaire

- ▷ Forme canonique
- ▷ Algorithme du simplexe

2 Structures de données

2.1 Files de priorité

Implémentées par exemple avec un tas.
Méthodes :

- ▷ Enfiler
- ▷ Défiler un élément maximal
- ▷ Est vide ?
- ▷ Construire file vide

2.2 Tables de hachage

Méthodes :

- ▷ Ajout d'un élément
- ▷ Suppression d'un élément
- ▷ Contient x ?

Risques de collision, n'est pas rare (idem paradoxe des anniversaires)

2.3 Structure Union-Find

Méthodes :

- ▷ Créer partition
- ▷ Fusionner deux classes (union)
- ▷ Obtenir un représentant (find)

Implémentation par une forêt d'arbres. Complexité améliorée en utilisant la compression de chemin.

3 Autres

- ▷ Encodage de Huffman
- ▷ Formules de Horn
- ▷ FFT
- ▷ Classes P, NP, EXPTIME
- ▷ Classe NP : Réduction à SAT, Branch&Bound, Local Search