SUJETS de TD d'ALGORITHMIQUE

Compter le nombre d'arêtes de l'hypercube. Formule récursive. Conjecturer une formule. Preuve par récurrence.

Résoudre T(1)=1, T(n) = 2 T(n/2) + n + 1 . Conjecturer une formule pour T. Preuve par récurrence.

Résoudre T(1)=1, T(n) = 2 T(n/2) + n. Conjecturer une formule pour T. Preuve par récurrence.

Résoudre T(1)=1, T(n) = T(n-1) + n. Conjecturer une formule pour T. Preuve par récurrence.

Résoudre Fibonacci : F(0)=0, F(1)=1, F(n+2)=F(n)+F(n+1). Considérer f, f' (f, la plus grande, est appelée nombre d'or) les deux racines de $x^2=x+1$. Trouver a et b tels que F(n)=a f^n + b f'^n, pour deux valeurs de n, par exemple n=0 et n=1. Prouver ensuite que f(n)=F(n) pour tout n (par récurrence ou autrement). Remarque : f et f' sont les deux valeurs propres de la matrice utilisée pour le calcul rapide de F.

Comment calculer rapidement S(n), quand S(0)=s0, S(1)=s1, S(2)=s2, et S(n+3)= a S(n) + b S(n+1) + c S(n+2), où les paramètres s0, s1, s2, a, b, c sont donnés ? (indication : puissance rapide de matrice)

Tri par arbre binaire non équilibré. Programmer en ocaml l'insertion dans un arbre binaire, non équilibré, et le parcours en O(n) de l'arbre. Type 't arbre= Vide | Noeud of ('t arbre * 't * 't arbre).

Problèmes combinatoires. Programmer en ocaml le le retournement d'une liste (List.rev), le calcul des parties et des k-tuples d'un ensemble (représenté par une liste), ses permutations.

Programmer la fusion du tri par fusion en récursion terminale

Trouver la dernière occurrence d'un élément dans une liste.

Calcul du PGCD.
pgcd(0, a) = a
pgcd(a, b) = pgcd(b modulo a, a)
Généraliser l'algorithme pour calculer en plus u et v (les nombres de Bézout) tels que au + bv égale le pgcd de a et b. Programmer en ocaml.

Variante du PGCD et de l'algorithme d'Euclide étendu, calculant les nombres de Bézout
Soient a et b deux entiers naturels. Voici une méthode pour calculer leur pgcd sans utiliser de division (à part la division par 2 pour les nombres pairs).
pgcd(0, b)=b, pgcd(a, 0) = a
pgcd(2a, 2b) = 2 * pgcd(a, b)
pgcd(2a, 2b+1) = pgcd(a, 2b+1)
pgcd(a, b) = pgcd(minimum(a, b), |a-b|/2) quand a et b sont impairs (donc a-b est pair).

Faites le sur quelques exemples. Remarquez que le nombre de chiffres d'au moins un des arguments est diminué de 1 à chaque itération. En déduire la complexité dans le pire des cas. Modifiez cette méthode pour calculer aussi les nombres (des entiers relatifs) de Bézout u et v tels que a*u+b*v=pgcd(a,b). Programmer en ocaml.

Bézout, projection d'un point sur une droite
Soient a et b deux entiers naturels. Vous avez calculé g, le pgcd de a et b, ainsi que deux nombres de Bézout u et v, deux entiers relatifs tels que au+bv=g.
Proposer quelques méthodes pour calculer (U, V) tels que aU+bV=g, et (U, V) sont les plus petits possibles. Indication. Si (u, v) est solution, alors (u+tb, v-ta) aussi, pour tout entier relatif t.
Exemple : dessiner le cas a=3, b=7. Des (u,v) solutions sont (-2,1), (5, -6), etc
Indication. ax +by=g est l'équation d'une droite qui passe par les points de Bézout (u, v). Le problème est de trouver le point entier sur cette droite qui est le plus près de l'origine (0, 0). Calculer le point qui est la projection orthogonale de (0,0) sur cette droite. Vérifier que c'est (x0,y0)=(ga/N, gb/N) où $N=a^2+b^2$.

Variante du tri, le front de Pareto
Un ensemble fini de points en 2D est donné. Les coordonnées sont des entiers naturels (donc non négatives). On dit qu'un point A est meilleur qu'un point B ssi l'abcisse de A est plus petite ou égale à celle de B, et l'ordonnée de A est plus petite ou égale à celle de B. Le problème est de calculer l'ensemble des meilleurs points (il n'en existe pas de meilleurs), et de les trier par x croissant. Cet ensemble s'appelle un front de Pareto ; il survient en optimisation multicritère, ici la minimisation de x et celle de y. Un point A appartient au front de Pareto, quand il n'existe pas de point ayant en même temps une abscisse et une ordonnée plus petites que celles de A.
Dessiner quelques exemples. S'il y a n points, combien de points peut avoir le FP ?
Proposez une méthode pour générer des points au hasard au dessus de la diagonale descendante du carré [0, 512] par [0, 512]. Il existe vraisemblablement une fonction random(M) qui rend un entier pseudo-aléatoire entre 0 et M-1.
Généralisez d'abord le tri rapide. Pour cela, proposez une méthode non déterministe (recourant au hasard) pour trouver un pivot, c'est à dire un point du FP en O(n). Ensuite, éliminez les points d'abscisse et d'ordonnée plus grandes que celles du pivot. Faire deux appels récursifs, pour calculer le FP de l'ensemble des points à gauche et au dessus du pivot, et le FP de l'ensemble des points à droite et en dessous du pivot. Concaténez les résultats.
Généralisez le tri fusion. Pour cela, il vous faut généraliser la fusion de deux FP (qui sont ordonnés par x croissant). Programmer en ocaml.

Dans le problème du front de Pareto, il peut arriver que l'ensemble des points soit infini. Cependant, même quand l'ensemble des points (à coordonnées entières, non négatives) est infini, le front de Pareto a un nombre fini de points. Prouver le.
Pourquoi est-ce important ? Ce théorème, généralisé en dimension quelconque, est le lemme de Dickson : l'ensemble des éléments minimaux de tout ensemble (même infini) de vecteurs d'entiers naturels est fini. Ce lemme permet de prouver la terminaison de l'algorithme des bases de Grobner (1965, dû à Bruno Buchberger), utilisé en calcul formel (Mathematica, Mapple, etc).

Calculs de l'enveloppe convexe. Programmer l'algorithme « gift wrapping » en ocaml. Programmer une marche de Graham en ocaml.

Géométrie. Un polyèdre a 2 faces triangulaires ABC et A'B'C', et 4 quadrilatères plans ABB'A', BCC'B', CAA'C'. Que pouvez vous dire des arêtes AA', BB', CC'.

Géométrie. Trouver le principe d'une méthode pour calculer le chemin le plus court entre 2 points donnés à la surface d'un cube. Quelles sont les difficultés pour généraliser à d'autres polyèdres ?

Chemin le plus court. Comment calculeriez vous le second chemin le plus court dans un graphe ?