Metodología y tecnología de la programación

Ingeniería Técnica en Informática de Sistemas (UCM)

Hoja de ejercicios 5 | Curso 2008/2009

Ejercicios de vuelta atrás

Ejercicio 1 Determínese un algoritmo que resuelva el problema de encontrar un camino que recorra todos los escaques de un tablero de ajedrez de dimensiones $n \times n$ utilizando únicamente los movimientos del caballo. Investíguese el caso en el que la casilla inicial y la final coinciden y se tiene un camino *cerrado*. Piénsese alguna heurística que permita mejorar la eficiencia del algoritmo.

Ejercicio 2 Sea H un conjunto de n hombres y M un conjunto de n mujeres. Cada hombre ordena a las mujeres de 1 a n según su grado de preferencia y lo mismo hacen las mujeres con los hombres. Un *emparejamiento* es una correspondencia biyectiva entre los hombres y las mujeres. Un emparejamiento es *estable* si para cada dos hombres h_1 y h_2 y sus correspondientes parejas m_1 y m_2, se cumplen las siguientes dos condiciones:

1. bien h_1 prefiere a m_1 antes que a m_2, o bien m_2 prefiere a h_2 sobre h_1;
2. bien h_2 prefiere a m_2 antes que a m_1, o bien m_1 prefiere a h_1 sobre h_2.

Escribir un algoritmo que encuentre un emparejamiento estable.

Ejercicio 3 Encontrar n puntos del eje real a partir de las $n(n-1)/2$ distancias (no necesariamente diferentes) entre cada par de puntos, sabiendo que el menor de dichos puntos es el origen y que el multiconjunto de las distancias viene dado en orden creciente.

Ejemplo: Si el multiconjunto de distancias es $\{1, 2, 2, 2, 3, 3, 3, 4, 5, 5, 5, 6, 7, 8, 10\}$, una solución válida es $\{0, 3, 5, 6, 8, 10\}$.

Ejercicio 4 Una matriz booleana $M[1..n, 1..n]$ puede representar un laberinto de la siguiente forma. A partir de una casilla dada, se puede ir a las cuatro casillas adyacentes vertical u horizontalmente. Si $M[i, j] = V$ se puede pasar por la casilla (i, j) y si $M[i, j] = F$ no se puede pasar por esa casilla. Suponiendo que $M[1, 1] = M[n, n] = V$, escribir un algoritmo que encuentre, si existe, un camino de la casilla $(1, 1)$ a la casilla (n, n).

Ejercicio 5 Tenemos que asignar n tareas a n procesadores. El tiempo que el procesador i tarda en hacer la tarea j viene dado por el elemento (i, j) de una matriz $T[1..n, 1..n]$. Desarrollar un algoritmo que encuentre la asignación óptima en el sentido de que la suma total de tiempos sea mínima.

Ejercicio 6 Escribir un algoritmo para resolver el problema de las tareas con plazo, duración y coste. Es decir, cada tarea i tiene asociados un plazo p_i, una duración de t_i unidades de tiempo y un coste c_i, $i = 1, \ldots n$. Hay que seleccionar un subconjunto de tareas de forma que todas ellas se puedan realizar antes de que venza su plazo correspondiente, y además el coste que se paga por aquellas tareas no realizadas (o sea, fuera de ese subconjunto) sea mínimo.

Ejercicio 7 Vamos a organizar un festival de rock al aire libre, donde el aforo es limitado. Para ello vamos a contratar a N artistas de M disponibles ($N < M$). Cada artista, a, nos garantiza una cantidad fija de público $P[a]$ y por actuar cobra una cantidad $C[a]$. El precio de la entrada para cada concierto es constante, E. Además, no todos los artistas aceptan tocar junto a cualquier otro en el mismo festival y los "vetos" entre artistas son conocidos de antemano. Se desea saber si es posible tener beneficios en el concierto y, si es así, cuál es la asignación óptima de conciertos.