Metodología y tecnología de la programación

Ingeniería Informática (UCM)

Hoja de ejercicios 5 Curso 2006/2007

Ejercicios de coste amortizado y vuelta atrás

Ejercicio 1
- Si la multipila vista en clase incluyese una operación *multiapilar* que apilara k elementos en la pila, ¿se seguiría teniendo un coste amortizado constante para las operaciones?
- Demostrar que si en el ejemplo del contador binario se tuviera una operación *decrementar*, n operaciones podrían tener un coste en $\Theta(nk)$.

Ejercicio 2 Demostrar que si $\alpha_{i-1} \geq 1/2$ y la operación i-ésima sobre una tabla dinámica es `eliminar`, entonces el coste amortizado de la operación con respecto a la función de potencial está acotado superiormente por una constante.

Ejercicio 3 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 4 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 5 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 6 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 asignació óptima en el sentido de que la suma total de tiempos sea mínima.

Ejercicio 7 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.