CC40A - Control 2

Prof. Gonzalo Navarro

24 de Junio de 2009

El control, como puede ver, suma 8.5 puntos. Los excesos sobre 7.0 se premiarán con una felicitación, pero no son reciclables :-).

P1 (2.5 pt)

Un teorema de Lagrange dice lo siguiente: *Sea $\pi(x)$ la cantidad de números primos $\leq x$. Entonces* $\lim_{x\to\infty} \frac{\pi(x)}{x/\ln(x)} = 1$.

1. (1pt) Utilice este teorema y el algoritmo probabilístico visto en clase para *detectar* primos, para diseñar un algoritmo que *genere* un número primo mayor o igual a un n dado.
2. (1pt) Analice el tiempo promedio de su algoritmo y la probabilidad de error asociada.
3. (0.5pt) Discuta si su algoritmo es de tipo Monte Carlo, Las Vegas, u otra cosa.

P2 (2.5 pt)

Se tienen n tareas de cómputo que requieren tiempos de CPU $t_1, t_2, \ldots, t_n$, y $m \leq n$ procesadores donde distribuirlas. Se busca asignar las tareas a procesadores de modo que el procesamiento de todas ellas en paralelo demore el menor tiempo posible.

Formalmente, sea T_i la suma de los tiempos t_j de las tareas asignadas al procesador i. Se desea que la asignación minimice $T = \text{máx}_{1\leq i\leq m} T_i$. Este problema es NP-completo.

1. (0.5pt) Sea T^* el tiempo de la asignación óptima. Demuestre que $T^* \geq \frac{1}{m}\sum_{1\leq i\leq m} T_i = \frac{1}{m}\sum_{1\leq j\leq n} t_j$
2. (0.5pt) Pruebe que $T^* \geq \text{máx}_{1\leq j\leq n} t_j$.
3. (1.5pt) Se propone la siguiente heurística: Partir con los $T_1 = \ldots = T_m = 0$, y considerar las tareas t_j de a una. En cada caso, asignar t_j al mínimo actual de los T_i. Pruebe que esta heurística es en realidad una 2-aproximación.

P3 (2.5 pt)

Considere un árbol general T de n nodos donde cada nodo tiene un puntero a su padre. Se desea calcular la profundidad de todos los nodos. Sea h la altura de T.

1. (1pt) Dé un algoritmo CREW que calcule todas las profundidades en $T(n, n) = O(\log n)$. Indique qué impide que su algoritmo sea EREW.

2. (1pt) Dé un algoritmo CRCW (en la variante de que si varios escriben una celda deben escribir lo mismo) que calcule todas las profundidades en $T(n, n) = O(\log h)$. Dado que h no es conocido, preste atención al problema de cómo detenerse.

3. (0.5pt) Calcule la eficiencia de su algoritmo CRCW.

Bono (1pt): Discuta cómo mejorar la eficiencia de su algoritmo, los problemas que se presentan, soluciones parciales, etc.

Tiempo: 2.5 horas Con una hoja de apuntes