Análisis y Diseño de Algoritmos
5 de julio de 2012
Examen final (duración: 120 min.)

Instrucciones:

- No podéis consultar ningún material ni hablar con nadie.
- Poned vuestros datos en la hoja de respuestas de lectura óptica que se os entregará.
- **No olvidéis indicar la modalidad del examen en la hoja de respuestas.**
- Elegid en cada pregunta la opción que creáis correcta, y marcadla cómo se indica en la hoja de respuestas.
- La nota del examen se calcula así:

$$\text{nota} = 10 \times (\text{aciertos} - \frac{1}{2}\text{errores})/\text{preguntas}$$

- Las respuestas en blanco ni restan ni suman puntos.
- Hay que entregar tanto las hojas de preguntas como la de respuestas (aunque las hojas de preguntas las podéis marcar).
- Tenéis 120 min. para hacer el examen.
- Si tenéis alguna duda, levantad la mano y un profesor irá a atenderos.

Modalidad 2

Preguntas:

1. Los algoritmos de ordenación *Quicksort* y *Mergesort* tienen en común...
   (a) ... que ordenan el vector sin usar espacio adicional.
   (b) ... que se ejecutan en tiempo $O(n)$.
   (c) ... que aplican la estrategia de *divide y vencerás*.
2. ¿Cuál es la diferencia principal entre una solución de *vuelta atrás* y una solución de *ramificación y poda* para el problema de la mochila?
   (a) El coste asintótico en el caso peor.
   (b) El hecho que la solución de *ramificación y poda* puede empezar con una solución subóptima voraz y la de *vuelta atrás* no.
   (c) El orden de exploración de las soluciones.

3. Tenemos un conjunto de n enteros positivos y queremos encontrar el subconjunto de tamaño m de suma mínima.

   (a) Lo más adecuado sería usar una técnica de ramificación y poda, aunque en el peor caso el coste temporal asintótico (o complejidad temporal) sería exponencial.

   (b) Para encontrar la solución habría que probar con todas las combinaciones posibles de m enteros, con lo que la técnica de ramificación y poda no aporta nada con respecto a vuelta atrás.

   (c) Una técnica voraz daría una solución óptima.

4. Di cuál de estos resultados de coste temporal asintótico es falsa:

   (a) La ordenación de un vector usando el algoritmo *Quicksort* requiere en el peor caso un tiempo en $\Omega(n^2)$.

   (b) La ordenación de un vector usando el algoritmo *Mergesort* requiere en el peor caso un tiempo en $\Omega(n^2)$.

   (c) La búsqueda binaria en un vector ordenado requiere en el peor caso un tiempo en $O(\log n)$.

5. En el problema del viajante de comercio (*travelling salesman problem*) queremos listar todas las soluciones factibles.

   (a) Lo más adecuado sería usar una técnica de ramificación y poda ya que es muy importante el orden en el que se exploran las soluciones parciales.

   (b) El orden en el que se exploran las soluciones parciales no es relevante; por ello, la técnica ramificación y poda no aporta nada con respecto a vuelta atrás.

   (c) Lo más importante es conseguir una cota pesimista muy adecuada. Las diferencias entre ramificación y poda y vuelta atrás son irrelevantes en este caso.

6. La complejidad temporal (o coste temporal asintótico) en el mejor de los casos...

   (a) ... es una función de la talla, o tamaño del problema, que tiene que estar definida para todos los posibles valores de ésta.

   (b) ... es el tiempo que tarda el algoritmo en resolver la talla más pequeña que se le puede presentar.

   (c) Las dos anteriores son verdaderas.

7. Tenemos n substancias diferentes en polvo y queremos generar todas las distintas formas de mezclarlas de forma que el peso total no supere un gramo. Como la balanza que tenemos solo tiene una precisión de 0.1 gramos, no se considerarán pesos que no sean múltiplos de esta cantidad. Queremos hacer un programa que genere todas las combinaciones posibles.
   (a) No hay ningún problema en usar una técnica de vuelta atrás.
   (b) No se puede usar vuelta atrás porque las decisiones no son valores discretos.
   (c) No se puede usar vuelta atrás porque el número de combinaciones es infinito.
8. Un algoritmo recursivo basado en el esquema *divide y vencerás...*
   (a) ... alcanza su máxima eficiencia cuando el problema de tamaño n se divide en a problemas de tamaño n/a.
   (b) ... nunca tendrá un coste temporal asintótico (o complejidad temporal) exponencial.
   (c) Las dos anteriores son verdaderas.
9. Dado un problema de optimización, ¿cuándo se puede aplicar el método de vuelta atrás?
   (a) Es condición necesaria (aunque no suficiente) que el dominio de las decisiones sea discreto o discretizable.
   (b) Es condición necesaria y suficiente que el dominio de las decisiones sea discreto o discretizable.
   (c) No sólo es condición necesaria que el dominio de las decisiones sea discreto o discretizable; además, debe cumplirse que se puedan emplear mecanismos de poda basados en la mejor solución hasta el momento.
10. ¿Cuál de estas tres expresiones es cierta?
   (a) $O(2^{\log(n)}) \subset O(n^2) \subset O(2^n)$
   (b) $O(n^2) \subset O(2^{\log(n)}) \subset O(2^n)$
   (c) $O(n^2) \subset O(2^{\log(n)}) \subseteq O(2^n)$
11. Sea $f(n)$ la solución de la relación de recurrencia $f(n) = 2f(n/2) + n$; $f(1) = 1$. Indicad cuál de estas tres expresiones es cierta:
   (a) $f(n) \in \Theta(n^2)$
   (b) $f(n) \in \Theta(n \log n)$
   (c) $f(n) \in \Theta(n)$
12. Indicad cuál de estas tres expresiones es falsa:
   (a) $\Theta(n/2) = \Theta(n)$
   (b) $\Theta(n) \subseteq O(n)$
   (c) $\Theta(n) \subseteq \Theta(n^2)$

13. Indica cuál es el coste temporal asintótico (o complejidad temporal), en función de n, del programa siguiente:

```
s=0; for(i=0;i<n;i++) for(j=i;j<n;j++) s+=n*i+j;
```

(a) Es $O(n^2)$ pero no $\Omega(n^2)$.

(b) Es $\Theta(n^2)$

(c) Es $\Theta(n)$

14. Un programa con dos bucles anidados uno dentro del otro, cada uno de los cuales hace aproximadamente n iteraciones, tarda un tiempo

(a) $O(n^2)$

(b) $O(2^n)$

(c) $O(n)$

15. La eficiencia de los algoritmos voraces se basa en...

(a) ... el hecho de que, con antelación, las posibles decisiones se ordenan de mejor a peor.

(b) ... el hecho de que las decisiones tomadas no se reconsideran.

(c) En el esquema voraz no se puede hablar de eficiencia puesto que a menudo no resuelve el problema.

16. En el esquema de vuelta atrás, los mecanismos de poda basados en la mejor solución hasta el momento...

(a) ... pueden eliminar vectores que representan posibles soluciones factibles.

(b) ... garantizan que no se va a explorar todo el espacio de soluciones posibles.

(c) Las dos anteriores son verdaderas.

17. Sea $f(n)$ la solución de la relación de recurrencia $f(n) = 2f(n-1) + 1$; $f(1) = 1$. Indicad cuál de estas tres expresiones es cierta:

(a) $f(n) \in \Theta(n^2)$

(b) $f(n) \in \Theta(2^n)$

(c) $f(n) \in \Theta(n)$

18. Pertenece $3n^2 + 3$ a $O(n^3)$?

(a) No.

(b) Sólo para $c = 1$ y $n_0 = 5$

(c) Sí.

19. Las relaciones de recurrencia...

(a) ... aparecen sólo cuando la solución sea del tipo *divide y vencerás*.

(b) ... expresan recursivamente el coste temporal de un algoritmo.

(c) ... sirven para reducir el coste temporal de una solución cuando es prohibitivo.

20. El coste temporal de un algoritmo se ajusta a la siguiente ecuación de recurrencia:

$$T(n) = \begin{cases} 1 & n = 0 \\ n + \sum_{j=0}^{n-1} T(j) & n > 1 \end{cases}$$

¿qué coste temporal asintótico (o complejidad temporal) tendrá el algoritmo?

(a) $O(n \log(n))$

(b) $O(n^2)$

(c) $O(2^n)$

21. La versión de *Quicksort* que utiliza como pivote el elemento del vector que ocupa la posición central...

(a) ... no presenta caso mejor y peor para instancias del mismo tamaño.

(b) ... se comporta mejor cuando el vector ya está ordenado.

(c) ... se comporta peor cuando el vector ya está ordenado.

22. ¿Cual es el coste espacial asintótico del siguiente algoritmo?

```
int f( int n ) {
  int a = 1, r = 0;
  for( int i = 0; i < n; i++ ) {
    r = a + r;
    a = 2*r;
  }
return r;
}
```

(a) $O(1)$

(b) $O(\log(n))$

(c) $O(n)$

23. De los problemas siguientes, indicad cuál no se puede tratar eficientemente como los otros dos:

(a) El problema de cortar un tubo de forma que se obtenga el máximo beneficio posible

(b) El problema del cambio, o sea, el de encontrar la manera de entregar una cantidad de dinero usando las mínimas monedas.

(c) El problema del viajante de comercio

24. ¿Qué algoritmo es asintóticamente más rápido, el *quicksort* o el *mergesort*?

    (a) como su nombre indica, el *quicksort*.

    (b) son los dos son igual de rápidos, ya que el coste temporal asintótico de ambos es $O(n \log(n))$.

    (c) el *mergesort* es siempre más rápido o igual (salvo una constante) que el *quicksort*.

25. El coste temporal del algoritmo de ordenación por inserción es...

    (a) ... $O(n^2)$.

    (b) ... $O(n)$.

    (c) ... $O(n \log n)$.

26. Los algoritmos de programación dinámica hacen uso...

    (a) ...de que la solución óptima se puede construir añadiendo el componente óptimo de los restantes, uno a uno.

    (b) ...de que se puede ahorrar esfuerzo guardando los resultados de esfuerzos anteriores.

    (c) ...de una estrategia trivial consistente en examinar todas las soluciones posibles.

27. ¿Cuál de estos tres problemas de optimización no tiene una solución voraz (*greedy*) que sea óptima?

    (a) El problema de la mochila continua o con fraccionamiento.

    (b) El problema de la mochila discreta.

    (c) El árbol de cobertura de coste mínimo de un grafo conexo.

28. El problema de la función compuesta mínima consiste en encontrar, a partir de un conjunto de funciones dadas, la secuencia mínima de composiciones de éstas que permita transformar un número n en otro m. Se quiere resolver mediante ramificación y poda. ¿Cuál sería la forma más adecuada de representar las posibles soluciones?

    (a) Mediante un vector de booleanos.

    (b) Mediante un vector de reales.

    (c) Este problema no se puede resolver usando ramificación y poda si no se fija una cota superior al número total de aplicaciones de funciones.

29. ¿Cuál de estas tres expresiones es falsa?

    (a) $2n^2 + 3n + 1 \in O(n^3)$

    (b) $n + n\log(n) \in \Omega(n)$

    (c) $n + n\log(n) \in \Theta(n)$

30. Si el coste temporal de un algoritmo es $T(n)$, ¿cuál de las siguientes situaciones es imposible?

    (a) $T(n) \in O(n)$ y $T(n) \in \Theta(n)$

    (b) $T(n) \in \Omega(n)$ y $T(n) \in \Theta(n^2)$

    (c) $T(n) \in \Theta(n)$ y $T(n) \in \Omega(n^2)$

31. El coste temporal asintótico de insertar un elemento en un vector ordenado de forma que continue ordenado es...

    (a) ... $O(n)$.

    (b) ... $O(\log n)$.

    (c) ... $O(n^2)$.

32. ¿Qué nos proporciona la media entre el coste temporal asintótico (o complejidad temporal) en el peor caso y el coste temporal asintótico en el mejor caso?

    (a) El coste temporal promedio.

    (b) El coste temporal asintótico en el caso medio.

    (c) Nada de interés.

33. El coste temporal asintótico del programa

```
s=0;for(i=0;i<n;i++) for(j=i;j<n;j++) s+=i*j;
```

    y el del programa

```
s=0; for(i=0;i<n;i++) for(j=0;j<n;j++) s+=i*i*j;
```

    son...

    (a) ... el del primero, menor que el del segundo.

    (b) ... el del segundo, menor que el del primero.

    (c) ... iguales.

34. Se desea obtener todas las permutaciones de una lista compuesta por n elementos. ¿Qué esquema es el más adecuado?

    (a) Divide y vencerás, puesto que la división en sublistas se podría hacer en tiempo constante.

    (b) Ramificación y poda, puesto que con buenas funciones de cota es más eficiente que vuelta atrás.

    (c) Vuelta atrás, es el esquema más eficiente para este problema.

35. En ausencia de cotas optimistas y pesimistas, la estrategia de vuelta atrás...

    (a) ... no se puede usar para resolver problemas de optimización.

    (b) ... no recorre todo el árbol si hay manera de descartar subárboles que representan conjuntos de soluciones no factibles.

    (c) ... debe recorrer siempre todo el árbol.

36. La solución recursiva ingenua a un determinado problema de optimización muestra estas dos características: por un lado, se basa en obtener soluciones óptimas a problemas parciales más pequeños, y por otro, estos subproblemas se resuelven más de una vez durante el proceso recursivo. Este problema es candidato a tener una solución alternativa basada en...

    (a) ... un algoritmo del estilo de *divide y vencerás*.

    (b) ... un algoritmo de programación dinámica.

    (c) ... un algoritmo voraz.

37. La función γ de un número semientero positivo (un número es semientero si al restarle 0.5 es entero) se define como:

```
double gamma( double n ) { // Se asume n>=0.5 y n-0.5 entero
  if( n == 0.5 )
    return sqrt(PI);
  return n * gamma( n - 1);
}
```

    ¿Se puede calcular usando programación dinámica iterativa?

    (a) Sí.

    (b) No, ya que el índice del almacén sería un número real y no entero.

    (c) No, ya que no podríamos almacenar los resultados intermedios en el almacén.

38. Decid cuál de estas tres es la cota optimista más ajustada al valor óptimo de la mochila discreta:

    (a) El valor de la mochila continua correspondiente .

    (b) El valor de la mochila discreta que se obtiene usando un algoritmo voraz basado en el valor específico de los objetos.

    (c) El valor de una mochila que contiene todos los objetos aunque se pase del peso máximo permitido.

39. Un tubo de n centímetros de largo se puede cortar en segmentos de 1 centímetro, 2 centímetros, etc. Existe una lista de los precios a los que se venden los segmentos de cada longitud. Una de las maneras de cortar el tubo es la que más ingresos nos producirá. Se quiere resolver el problema mediante vuelta atrás. ¿cual sería la forma más adecuada de representar las posibles soluciones?

    (a) un vector de booleanos.

    (b) un par de enteros que indiquen los cortes realizados y el valor acumulado.

    (c) una tabla que indique, para cada posición donde se va a cortar, cada uno de los posibles valores acumulados.

40. Cuando la descomposición de un problema da lugar a subproblemas de tamaño similar al original, muchos de los cuales se repiten, ¿qué esquema es a priori más apropiado?

(a) Divide y vencerás.

(b) Programación dinámica.

(c) Ramificación y poda.