Primer Examen

Fundamentos de Análisis y Diseño de Algoritmos
Programa de Ingeniería de Sistemas
Universidad del Valle

Profesor Carlos Alberto Ramírez Restrepo

Abril 22 de 2017

DECLARACIÓN DE HONOR

Yo, ______________________________, declaro que todas mis respuestas en este parcial fueron desarrolladas por mi y no dí ni recibí copia en ningún punto. Además soy consciente de las implicaciones disciplinarias y éticas de incurrir en copia y entiendo que este tipo de acciones no contribuyen a mi formación como profesional y como persona.

Firma: ______________________________
Código:_______________
Fecha:_______________

Este es el primer examen parcial del curso *Fundamentos de Análisis y Diseño de Algoritmos*, 2017-1. El examen tiene 8 preguntas: otorga un total de 100 puntos y 10 puntos de bono. El examen es *individual* y NO es permitido el uso de libros o apuntes, ni de equipos electrónicos; no puede hablar o compartir sus soluciones con sus compañeros. Tenga en cuenta los puntos de cada pregunta y planifique adecuadamente su tiempo. El parcial tiene una duración de 3 horas.

1. **[12 pts.]** ¿Cuál es la complejidad del siguiente algoritmo? (justifique su respuesta):

```
def espectrum(n):
   r = 0
   i = n*n
   while i > 0:
      j = 1
      while j <= n:
         k = j + 1
         while k <= n:
            r = r + 1
            k = k + 1
         j = j + 1
      i = i - n/2
   return r
```

2. **[14 pts.]** Considere el siguiente algoritmo:

```
def espectrum(n):
    r = 0
    i = n*n
    while i > 0:
        j = 1
        while j <= n:
            k = j + 1
            while k <= n:
                r = r + 1
                k = k + 1
            j = j + 1
        i = i - n/2
    return r
```

1

1

$2n+1$

$2n$

$2n^2$ $\quad 2n(n+1)$

$2n^2(t_k)$

$n^2 \quad \left(n^2-\frac{n}{2}\right) \cdot \left(n^2-\frac{2n}{2}\right) \quad \left(n^2-\frac{3n}{2}\right)$

$0 \quad 1 \quad 2 \quad 3 \cdots 2n$

$i = n^2 - \frac{kn}{2}$

$\frac{kn}{2} = n^2$ $\quad \boxed{k=2n}$

Valida

$k = 2n-1$

$1 \; 2 \; 3 \ldots n \quad n+1$

$k = 2, 3, 4, 5 \ldots n$

$n \quad n \quad n \quad n \qquad n+1$

$n+1 \quad n+1 \quad n+1 \quad n+1$

$$\sum_{j=1}^{n} \sum_{p=j+1}^{n+1} 1$$

j	1	2	3	4	$\cdots$	n
$\sum_{p=j+1}^{n+1} 1$	$n-1$	$n-2$	$n-3$	$n-4$		$n-n$

$$\sum_{p=2}^{n} 1$$

$$\sum_{k=1}^{n+1} (n-k)$$

$$\sum_{p=1}^{n} c = cn$$

$$\sum_{k=1}^{n+1} n - \sum_{k=1}^{n+1} k = \overbrace{n^2 + \frac{n(n+1)}{2}}^{\text{entra}} + \overbrace{n+n+1}^{\text{salida}}$$

$$2n^2\left(n^2 + \frac{n(n+1)}{2} + n + n + 1\right)$$

$$2n^2\left(n^2 + \frac{n(n+1)}{2}\right) \longrightarrow O(n^4)$$

```
Algoritmo(n)
{
   i=1;
   j=1;
   res=0;
   while(j >= n){
      res = res + i;
      i += 2;
      j++;
   }
   return res;
}
```

(a) Qué cálcula este algoritmo?

(b) Cúal es la precondición y poscondición del algoritmo?

(c) Proponer una invariante para los ciclos del algoritmo.

(d) Demostrar que el invariante se cumple y argumentar la corrección del algoritmo.

3. **[15 pts.]** Se tienen tres algoritmos que resuelven un mismo problema:

- El algoritmo A divide el problema en 5 subproblemas de un cuarto del tamaño inicial, los resuelve recursivamente y combina las soluciones en tiempo lineal.
- El algoritmo B resuelve el problema de tamaño n planteando dos problemas de tamaño $n-1$ y combinando las soluciones en tiempo constante.
- El algoritmo C divide el problema en 3 subproblemas de un cuarto del tamaño inicial, los resuelve recursivamente y combina las soluciones en tiempo cuadrático.

  tiempo cuadrático.

Cuál es la complejidad temporal de cada uno de los tres algoritmos? Cuál es el más eficiente y por qué?

Para calcular la complejidad de cada algoritmo plantee una ecuación de recurrencia y resuelvala de ser posible mediante el método maestro, si no resuelvala con cualquiera de los otros métodos vistos en clase.

4. **[15 pts.]** Defina un algoritmo con complejidad lineal que reciba un número `N` y construya un arreglo con todos los factoriales entre 1 y `N` de forma descendente.

5. **[10 pts.]** Defina un algoritmo que utilice recursividad y permita calcular el valor de la serie de fibonacci para un número n con complejidad $\Theta(n)$. Su algoritmo no puede utilizar ningún tipo de ciclo.

6. **[18 pts.]** **Problema: Puntos máximos en el plano**

Dado un par de puntos (x, y), (x', y') en el plano, se dice que (x, y) domina a (x', y') si $x > x'$ y $(y > y')$. También se dice que (x', y') es dominado por (x, y).

Dados n puntos en el plano cartesiano, se dice que uno de ellos es máximo si no está dominado por ninguno de los otros puntos.

Considere el problema de los *puntos máximos en un plano* **PMP**:

Entrada: n puntos en el plano (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n), tales que $x_i < x_{i+1}$ ó $(x_i = x_{i+1} \wedge y_i < y_{i+1})$. Es decir, los n puntos vienen ordenados de izquierda a derecha y de abajo a arriba.

Salida: $I \subseteq \{1, 2, \ldots, n\}$ tal que $\forall_{i \in I}.(x_i, y_i)$ es un punto máximo y tal que $\forall_{i \notin I}.(x_i, y_i)$ es un punto dominado.

(a) Suponga que la entrada es $n = 8$ y el conjunto de puntos es $\{\cancel{(1,2)}, \cancel{(2,3)}, \cancel{(3,4)}, \cancel{(4,5)}, (5,4), (6,3), (7,3), (8,1)\}$. Cuál es la salida que arrojaría con esta entrada un algoritmo que resuelva este problema?

(b) Defina un algoritmo para solucionar este problema que use la técnica *divide y vencerás*. Cuál es la complejidad de su algoritmo?

7. **[18 pts.]** Considere el siguiente problema. Dados m libros enumerados $1, 2, \ldots, m$, donde $p_1, p_2, \ldots, p_m$ corresponde al número de páginas en cada libro. Se requiere hacer una copia de cada libro. Su tarea es asignar estos libros entre k escribas tales que $k \leq m$. Cada libro puede ser asignado solamente a un escriba y cada escriba debe tener una secuencia continua de libros. Esto significa que existe una sucesión creciente de números $0 = b_0 < b_1 < b_2 \ldots < b_{k-1} \leq b_k = m$ de tal manera que al i–esimo escriba le son asignados los libros con números entre b_{i-1} y b_i. El tiempo necesario para copiar todos los depende del escriba al que se le haya asignado más trabajo. De esta manera, se requiere minimizar el máximo número de páginas asignado a un solo escriba.

```
Algoritmo(n)
{
   i=1;
   j=1;
   res=0;
   while(j <= n){
      res = res + i;
      i += 2;
      j++;
   }
   return res;
}
```

$n \in \mathbb{Z}, n > 0$

(a) Qué cálcula este algoritmo?

(b) Cúal es la precondición y poscondición del algoritmo?

$res \in \mathbb{Z}, res \geq 0$

(c) Proponer una invariante para los ciclos del algoritmo.

estado

(j, res)

(d) Demostrar que el invariante se cumple y argumentar la corrección del algoritmo.

$(\overset{1}{1}, 0) \to (\overset{2}{2}, 0+1) \to (\overset{3}{3}, \underbrace{0+1+3})$

$\to (4, \underbrace{0+1+3+5}_{4})$

$(j, res) \to (j+1, res + (j)2-1)$

$(3, 4) \to (4, 4+5)$

j, res

$$\boxed{\left(j, \sum_{p=1}^{j-1} 2j-1\right)}$$

$(1, 0)$
$(2, 1)$
$(3, 4)$

$(3, \underset{1}{1} + \underset{2}{3})$

$$\left(n+1, \sum_{p=1}^{n} 2n+2-1\right)$$

$(n+1, n(2n+2-1))$

$\boxed{2n^2+n}$

3. **[15 pts.]** Se tienen tres algoritmos que resuelven un mismo problema:

- El algoritmo A divide el problema en 5 subproblemas de un cuarto del tamaño inicial, los resuelve recursivamente y combina las soluciones en tiempo lineal.
- El algoritmo B resuelve el problema de tamaño n planteando dos problemas de tamaño $n-1$ y combinando las soluciones en tiempo constante.
- El algoritmo C divide el problema en 3 subproblemas de un cuarto del tamaño inicial, los resuelve recursivamente y combina las soluciones en tiempo cuadrático.

tiempo cuadrático.

$T(n) = 5T(n/4) + O(n)$

$T(n) = 5T(n/4) + cn$

$a = 5$

$b = 4$ $\quad F(n) = O(n)$

$\log_5 4$

$\Theta(n)$

$T(n) = 2T(n-1) + c$

$O(n^2)$

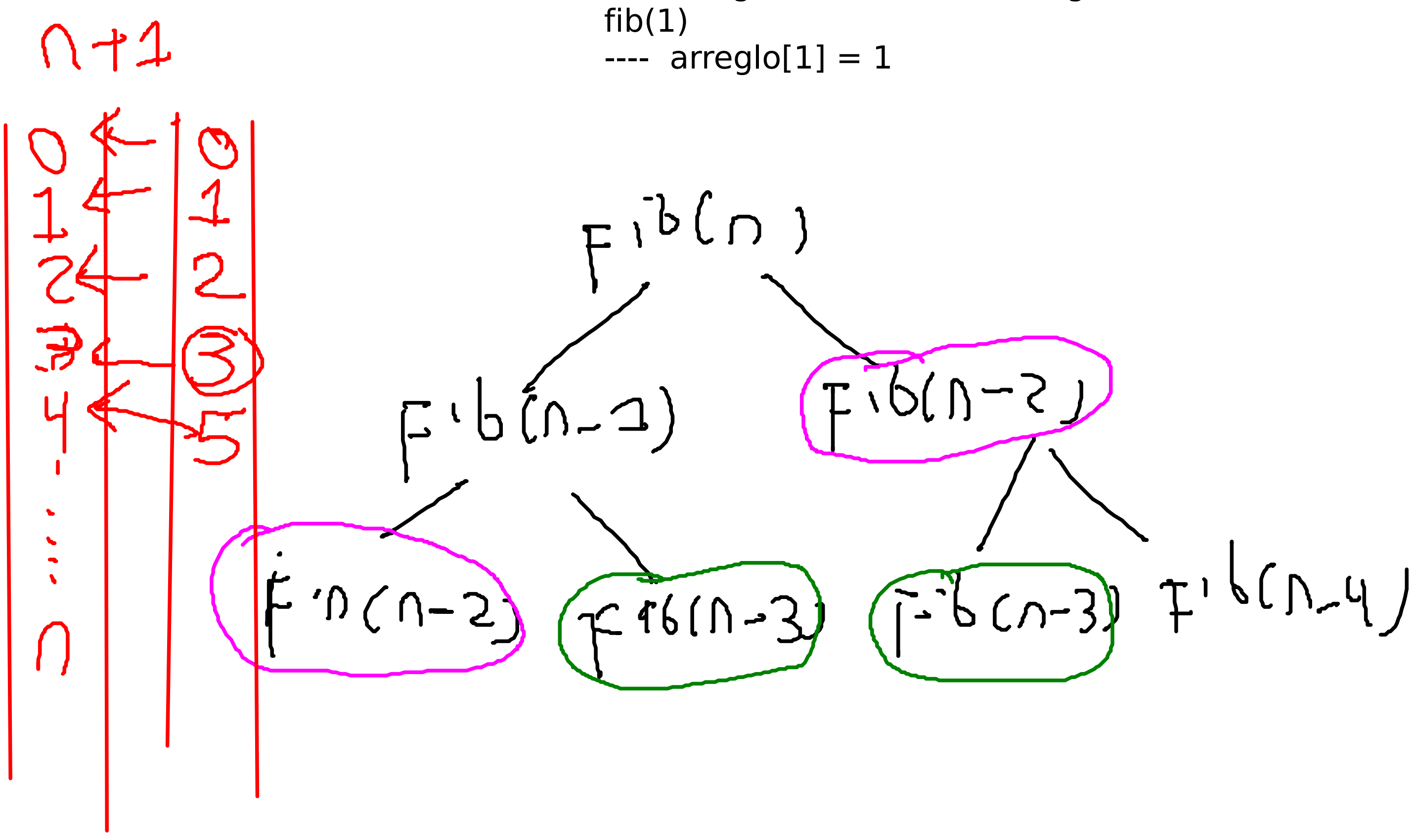
$T(n) = 3T(n/4) + n^2$

$O(n^2)$

5. **[10 pts.]** Defina un algoritmo que utilice recursividad y permita calcular el valor de la serie de fibonacci para un número n con complejidad $\Theta(n)$. Su algoritmo no puede utilizar ningún tipo de ciclo.

```
Arreglo[n+1]
Arreglo[0] = 1
fib(n)
      if n == 0 arreglo[0] = 1
      if n == 1 arreglo[1] = 1
      else  arreglo[n] = fib(n-1) + arreglo[n-2]

fib(3)
----  arreglo[3] = fib(2) + arreglo[1]
fib(2)
---   arreglo[2] = fib(1) + arreglo[0]
fib(1)
----  arreglo[1] = 1
```

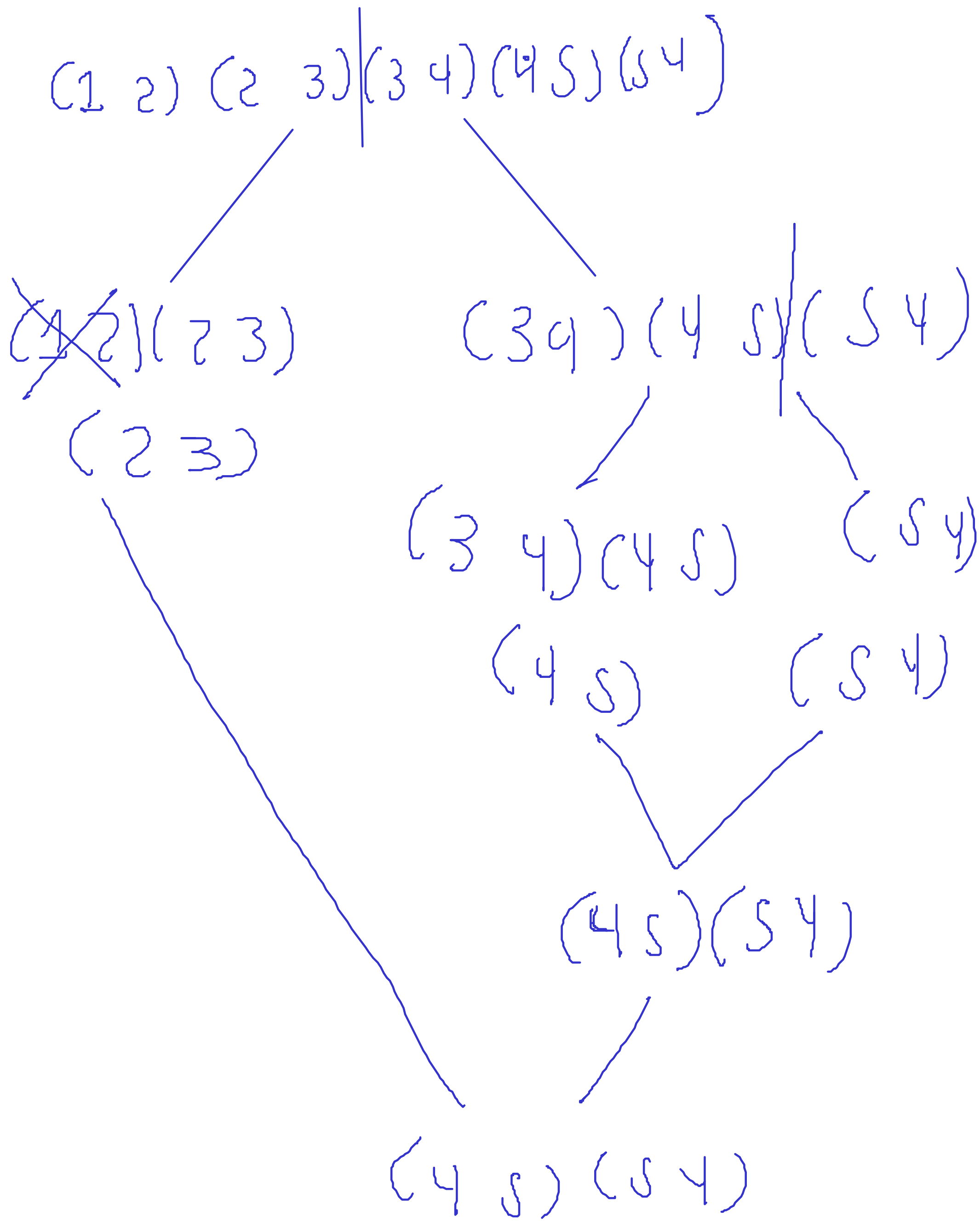


Entrada: n puntos en el plano (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n), tales que $x_i < x_{i+1}$ ó $(x_i = x_{i+1} \wedge y_i < y_{i+1})$. Es decir, los n puntos vienen ordenados de izquierda a derecha y de abajo a arriba.

Salida: $I \subseteq \{1, 2, \ldots, n\}$ tal que $\forall_{i \in I}.(x_i, y_i)$ es un punto máximo y tal que $\forall_{i \notin I}.(x_i, y_i)$ es un punto dominado.

(a) Suponga que la entrada es $n = 8$ y el conjunto de puntos es $\{\cancel{(1,2)}, \cancel{(2,3)}, \cancel{(3,4)}, \cancel{(4,5)}, (5,4), (6,3), (7,3), (8,1)\}$. Cuál es la salida que arrojaría con esta entrada un algoritmo que resuelva este problema?

(b) Defina un algoritmo para solucionar este problema que use la técnica *divide y vencerás*. Cuál es la complejidad de su algoritmo?



Por ejemplo, si se tiene que $m = 9$, $k = 3$ y

$$p_1, p_2, \ldots, p_9 = 100, 200, 300, 400, 500, 600, 700, 800, 900$$

el resultado debería ser 1700, que corresponde a asignar $\{100, 200, 300, 400, 500\}$ al escriba 1, $\{600, 700\}$ al escriba 2 y $\{800, 900\}$ al escriba 3.

(a) Construya un algoritmo que utilice la estrategia *divide y vencerás* (búsqueda binaria) que resuelva el problema.

(b) Cuál es la complejidad de su algoritmo. Explique claramente su respuesta.

8. **[8 pts.]** Resolver las siguiente recurrencia mediante el método de iteración o el método de árbol de recursión:

$$\mathbf{T(n)} = 2\mathbf{T(n-1)} + 1$$

$m = 9 \quad K = 3$

1) 4500 (sumar)

2) $\frac{4500}{3} = 1500$