ISIS 2103 Diseño de algoritmos
Semestre 2010-1 Parcial 2
Abril 15, 2010
Prof. Rodrigo Cardoso

1 **[60 puntos]**

Una compañía tiene n accionistas, con porcentajes de participación en la empresa $p_i \in$ **int**, $0<p_i\leq 100$, $1\leq i\leq n$. La compañía es dirigida por una *coalición directiva*, un grupo de accionistas que posea más de la mitad de las acciones y, por esta labor, este grupo gana una prima anual. La prima es repartida entre los accionistas que conforman la coalición directiva, proporcionalmente a su participación accionaria.

Un accionista puede llegar a dirigir la empresa, asociándose con diferentes grupos. Por ejemplo, si hay 5 accionistas, A, B, C, D y E, con participaciones respectivas de 20%, 12%, 14%, 29% y 25%, el accionista E podría asociarse con A y B, lo que le reportaría un 43.86% = (25*100/(20+12+25) de la prima anual. Sin embargo, si hiciera coalición con B y C, ganaría 49.02% = (25*100/(12+14+25) de la prima.

Se quiere resolver el problema de encontrar G, la mejor ganancia para p_n, considerando todas las posibles coaliciones directivas en las que éste participe.

1a Explique por qué la ganancia de n se maximiza en coaliciones directivas que superen minimalmente el 50% de participación colectiva.

La participación colectiva de una coalición B es `p.B=(+i| i∈B : p.i)`. La ganancia que n obtiene (si participa en ella) es `p.n/p.B` . Así, al comparar dos coaliciones directivas, la que da más ganancia a n es la que minimiza `p.B`. Claro, debe superar el 50%, para ser directiva.

[5/60]

1b Sea `const(a,j)` un predicado verdadero si y solo si el entero a se puede construir como suma de `p.n` con elementos de $\{p_1, \ldots, p_j\}$, $0\leq j<n$. Explique cómo, conociendo `const` se puede conocer la máxima ganancia posible para n.

De acuerdo con **1a**, se debe tener que

$$G = p_n/(\min a|\ a>50 \wedge const(a,n-1):\ a)$$

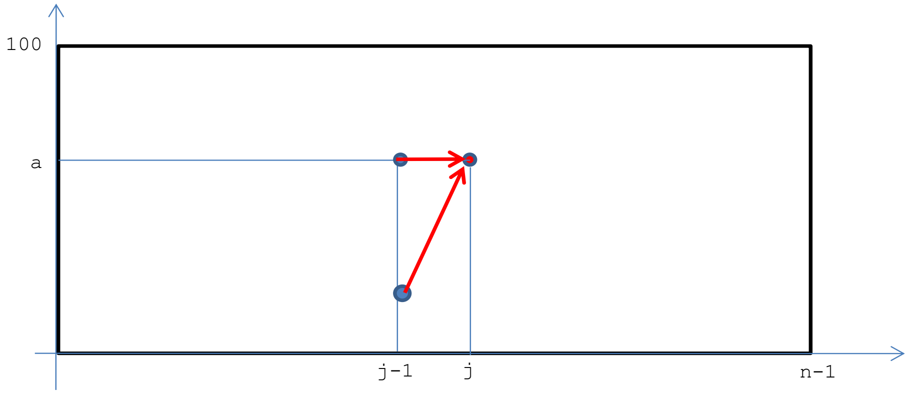
[5/60]

1c Indique cómo calcular `const(a,j)` con una recurrencia.

```
const(a,j)  ≡  a=p.n                              , si 0=j
            ≡  const(a,j-1)                       , si 0<j<n, a<p.j
            ≡  const(a,j-1) ∨ const(a-p.j,j-1)    , si 0<j≤n, a≥p.j
```

[20/60]

1d Elabore un diagrama de necesidades para el cálculo de `const`.



[10/60]

1e Estime la complejidad de un algoritmo que resuelva el problema de encontrar G.

El diagrama sugiere como estructura de datos un vector de tamaño 101 que puede llenarse de arriba abajo de manera similar al problema del morral.

Así:

$S(n) = \theta(101)$
$\quad = \theta(1)$

[10/60]

$T(n) = \theta(101n)$
$\quad = \theta(n)$

[10/60]

2 **[40 puntos]**

Considérese un tablero de ajedrez de lado $n>0$. Para denotar las casillas del tablero, sea (i,j) la casilla en la fila i, columna j, para $0\le i,j<n$. Un caballo de ajedrez puede saltar de una casilla del tablero a otra si ésta está:
- en la siguiente o en la anterior columna, a dos filas de distancia, o bien
- en la siguiente o en la anterior fila, a dos columnas de distancia.

Sea $c(p,q)$ el mínimo número de saltos que debe hacer un caballo para ir de la casilla $(0,0)$ a la casilla (p,q) del tablero, $0\le p,q<n$.

2a Describa el problema de calcular $c(i,j)$, $0\le i,j<n$, como un problema de ruta óptima sobre una grafo etiquetado.

Considérese un grafo $G(V,E,c)$, donde:

$V = \{(i,j) \mid 0\le i,j<n\}$
Nótese que: $|V| = n^2$

$E = \{\langle (i_1,j_1),(i_2,j_2)\rangle : V\times V \mid (|i_1-i_2|=1 \wedge |j_1-j_2|=2) \vee (|i_1-i_2|=2 \wedge |j_1-j_2|=1)\}$

$c:\ V\times V \rightarrow \mathbf{R^*}$

$c\langle (i_1,j_1),(i_2,j_2)\rangle = 0$, si $(i_1,j_1) = (i_2,j_2)$
$\quad = 1$, si $(|i_1-i_2|=1 \wedge |j_1-j_2|=2) \vee (|i_1-i_2|=2 \wedge |j_1-j_2|=1)$
$\quad = \infty$, en otro caso.

[15/50]

2b Utilice versiones de los algoritmos de Floyd-Warshall y de Dijkstra para calcular c(i,j) , $0 \le i,j < n$. Calcule las complejidades en cada caso, suponiendo que no se pueden dañar los datos de entrada y que se usan las mejores estructuras de datos posibles. Justifique sus respuestas.

***AYUDA:** El número de saltos de caballo desde una casilla está acotado por una constante …*

Floyd-Warshall:

$S(n) = \theta(n^4)$ // 1 matriz $|V| \times |V| = n^2 \times n^2$

$T(n) = \theta(n^6)$ // Cúbico en $|V| = (n^2)^3$

[10/50]

Dijkstra:

$S(n) = \theta(n^2)$ // 1 vector $|V| = n^2$

$T(n) = \theta(n^2)$ // Lineal en $|V| = n^2$

Para justificar el tiempo en Dijkstra, basta una representación del grafo como lista de adyacencias y observar que el número de arcos visitados desde un nodo (nodos vecinos) está acotado por una constante, dado que hay, a lo sumo, 8 saltos posibles de caballo desde una casilla. Es decir, en este caso, el costo del paso de relajación (ajuste de los valores en Dijkstra después de marcar un nodo) *tiene costo $O(1)$*. Es decir, el ciclo de Dijkstra cuesta $\theta(|V|) = \theta(n^2)$.

[15/50]

Variante en Dijkstra:

$T(n) = \theta(n^2 \log n)$ // $|V| \log|V| = n^2 \times \log n^2$

[5/50]