Programación dinámica - Noviembre 2017.

Número de decisiones: P, $<d_1, d_2, ... d_p>$ donde la decisión i-ésima es si se vende entradas a la peña i o no, representado por 1 se vende y 0 no.

Función objetivo y reestricciones: $\sum_{i=1}^{p} S[i] * d_i \mid \sum_{i=1}^{p} S[i] * d_i \leq E$

Demostración del principio de optimalidad:

Sea $<d_1, d_2, ..., d_p>$ la secuencia óptima de decisiones para el problema Partido(P, E) siendo su valor (distribuir E entradas entre P peñas) asociado:

$$\sum_{i=1}^{p} S[i] * d_i \quad \text{sujeto a} \quad \sum_{i=1}^{p} S[i] * d_i \leq E$$

Supongamos que prescindimos de la última decisión, esto es d_p. La subsecuencia que queda, esto es, $<d_1, d_2, ..., d_{p-1}>$ es la solución óptima para el subproblema:

Partido$(P-1, E - S[P] * d_p)$ que es el problema de distribuir $E - S[P] * d_p$ entradas disponibles entre $P-1$ peñas obteniendo el máximo beneficio.

Dicho valor asociado será:

$$\sum_{i=1}^{P-1} S[i] * d_i \quad \text{sujeto a} \quad \sum_{i=1}^{P-1} S[i] * d_i \leq E - S[P] * d_p$$

Ahora supongamos que $<d_1, d_2, d_3 ..., d_{p-1}>$ no es la solución óptima del problema, sino que existe otra solución $<d_1^*, d_2^*, ... d_{p-1}^*>$ que mejora su valor. Esto quiere decir que:

$$\sum_{i=1}^{P-1} S[i] * d_i < \sum_{i=1}^{P-1} S[i] * d_i^* \quad (1)$$ y se cumple además que:

$$\sum_{i=1}^{P-1} S[i] * d_i^* \leq E - S[P] * d_p$$

Sumando $S[P] * d_p$ en la ecuación (1) a ambos lados obtenemos:

$$\sum_{i=1}^{P-1} S[i] * d_i + S[P] * d_p < \sum_{i=1}^{P-1} S[i] * d_i^* + S[P] * d_p$$

Lo que significa que la secuencia $<d_1^*, ... d_p^*>$ mejoraría el resultado de la secuencia $<d_1, d_2, ..., d_p>$ para el problema Partido(P, E), lo cual contradice la hipótesis de partida pues $<d_1, d_2, ... d_p>$ era la solución óptima de dicho problema.

Ecuación recursiva y primera llamada a la función.

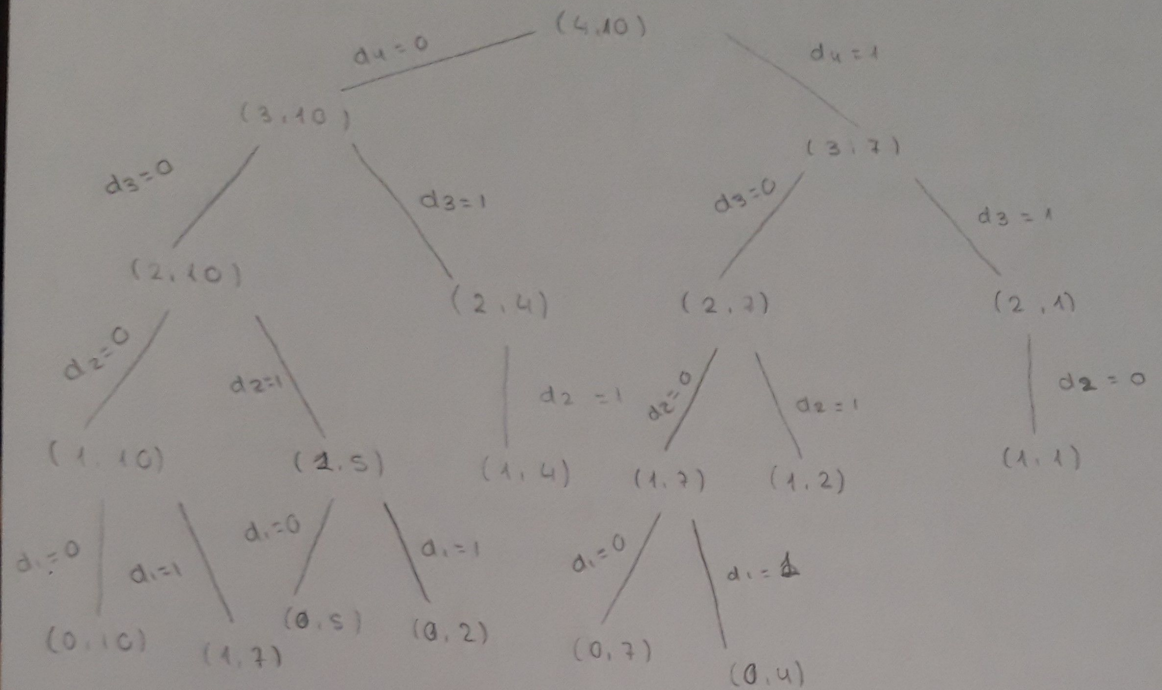
$$\text{Partido}(P, E) = \begin{cases} 0 & \text{si } p = 0 \\ \text{partido}(p-1, e) & \text{si } p > 0 \text{ y } S[p] > e \\ \max\limits_{d_p \in \{0,1\}} \{ \text{partido}(p-1, e - S[p] * d_p) + S[p] * d_p & \\ & \text{si } p > 0 \text{ y } e \geq S[p] \end{cases}$$

Donde partido(p, E) es la primera llamada a la función con P el nº total de peñas y E el número total de entradas disponibles inicialmente.

La función devuelve el máximo nº de entradas que se pueden distribuir entre las p peñas (1 a P) de las e entradas disponibles.

Árbol de llamadas para el ejemplo. Estructuras de almacenamiento.

$E = 10$, $P = 4$, $S[1..4] = \{3, 5, 6, 3\}$



Dado que nuestra función recursiva tiene dos parámetros, se precisan dos estructuras de almacenamiento, bidimensionales. Estas estructuras tendrán $P+1$ filas y $E+1$ columnas debido a $1 \le p \le P$ y $1 \le e \le E$. En Vmax, se guardará el valor asociado a cada subproble. , el máx de entradas distribuidos. En la otra, se guardará la alternativa (0 ó 1) que proporcione el máximo para cada subproblema.

Los estructuras de almacenamiento se inicializan con los resultados de los problemas triviales, que corresponde a llenar la fila 0 de cada matriz con un 0, dado que los problemas dados son del tipo Venta(0, e), Vmax[0][e] = 0 y Dec[0][e] = 0 con $0 \le x \le 10$.

Rellenado de las matrices: para solucionar el subproblema Venta(p, e) se precisa conocer la solución de los subproblemas del tipo Venta(p − 1, x), con $x \le e$, y ambas matrices se rellenan por filas en sentido creciente.

V max	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	3	3	3	3	3	3	3	3
2	0	0	0	3	3	5	5	5	8	8	8
3	0	0	0	3	3	5	6	6	8	9	9
4	0	0	0	3	3	5	6	6	8	9	9

Bmax	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1	1
2	0	0	0	0	0	1	1	1	1	1	1
3	0	0	0	0	0	0	1	1	0	1	1
4	0	0	0	0	0	0	0	0	0	0	0

$Vmax[1][1] = \max\{Vmax[0][1-3\times 0] + 3\times 0\} = 0$, $Vmax[0][1-3\times 1]+3 = \cancel{11} = 0$

$Vmax[1][2] = \max\left\{\begin{array}{l} Vmax[0][2-5\times 0] + 5\times 0 = 0 \\ Vmax[0][2-5\times 1] + 5\times 1 = 0 \rightarrow \text{no cumple reest} \end{array}\right\} = 0$

$Vmax[1][3] = \max\left\{\begin{array}{l} Vmax[0][3-3\times 0] + 3\times 0 = 0 \\ Vmax[0][3-3\times 1] + 3\times 1 = 3 \end{array}\right| = 3$ $\quad Dec[1][3] = 1$

$Vmax[1][4] = \max\left\{\begin{array}{l} Vmax[0][4-3\times 0] + 0 = 0 \\ Vmax[0][4-3\times 1] + 3\times 1 = 4 \end{array}\right\} = 3$ $\quad Dec[1][4] = 1$

$Vmax[1][5] = \max\left\{\begin{array}{l} Vmax[0][5-3\times 0] + 0 = 0 \\ Vmax[0][5-3\times 1] + 3\times 1 = 3 \end{array}\right| 3$ $\quad Dec[1][5] = 1$

$Vmax[1][6] = \max\left\{\begin{array}{l} Vmax[0][6-3\cdot 0] + 0 = 0 \\ Vmax[0][6-3] + 3 = 3 \end{array}\right| 3$ $\quad Dec[1][6]$

$Vmax[2][1] = \left\{\begin{array}{l} Vmax[1][1-5\cdot 0] + 0 = 0 \\ Vmax[1][1-5\cdot 1] + 5 = \text{no se cump} \end{array}\right|$ $\quad Dec[2][1] = 0$

$Vmax[2][4] = \left\{\begin{array}{l} Vmax[1][4-5\cdot 0] + 5\cdot 0 = 3 \\ Vmax[1][4-5] + 5 = \text{no se cump} \end{array}\right|$ $\quad Dec[2][4] = 0$

$Vmax[2][5] = \left\{\begin{array}{l} Vmax[1][5-5\cdot 0] + 5 0 = 3 \\ Vmax[1][5-5\cdot 1] + 5 = 5 \end{array}\right| 5$ $\quad Dec[2][5] = 1$

$Vmax[2][8] = \left\{\begin{array}{l} Vmax[1][8-5\cdot 0] + 0 = 3 \\ Vmax[1][8-5\cdot 1] + 5\cdot 1 = 3+5 = 8 \end{array}\right|$ $\quad Dec[2][8] = 1$