Tema 2

Fundamentos Teóricos de la Programación Dinámica

2.1. Teorema de Optimalidad de Mitten

El objetivo básico en la programación dinámica consiste en 'descomponer" un problema de optimización en k variables a una serie de problemas con menor número de variables más fáciles de resolver. Precisamente esto es lo que se ha hecho en el problema de la mochila; reducir un problema inicial de n variables a n problemas de 1 variable. En este sentido, se podría decir que la programación dinámica se basa en un método de descomposición.

¿Cualquier problema puede ser resuelto mediante la programación dinámica? NO. En el problema de la mochila se verifican diversas condiciones necesarias para que se cumpla el teorema de optimalidad, que enseguida veremos y que es la base de la programación dinámica.

Ejemplo 2.1. Supongamos que la función objetivo del problema es la siguiente:

$$f(x, y, z) = xy + xz + yz$$

Aquí, claramente, no es posible aplicar la misma idea que en el problema de la mochila, porque, a diferencia de la función objetivo de [KP], en esta no se pueden "separar"las variables, y por tanto no es posible dividir el problema en etapas.

2.1.1. Caso sin restricciones

Supongamos que estamos interesados en minimizar una cierta función de $k+1$ variables $f(x, y_1, y_2, \ldots, y_k)$. Si fuésemos capaces de *separar* f en dos funciones f_1 y f_2 de modo que $f(x, \mathbf{y}) = f_1(x, f_2(\mathbf{y}))$ verificándose que

$$Min_{(x,\mathbf{y})} f(x, \mathbf{y}) = Min_x f_1(x, Min_{\mathbf{y}} f_2(\mathbf{y})),$$

habríamos reducido el problema original de $k+1$ variables en dos problemas con menor número de variables.

Definición 2.2. *Una función f se dice que es **descomponible** en f_1 y f_2 si:*

- *f es separable en f_1 y f_2, i.e., $f(\mathbf{x}, \mathbf{y}) = f_1(\mathbf{x}, f_2(\mathbf{y}))$*
- *f_1 es monótona no-decreciente respecto de su segundo argumento.*

Teorema 2.3. *(Teorema de Optimalidad de Mitten, 1964) Sea $f(x, \mathbf{y})$ una función real con $x \in \mathbb{R}$, $\mathbf{y} \in \mathbb{R}^k$. Si f es descomponible con $f(x, \mathbf{y}) = f_1(x, f_2(\mathbf{y}))$, entonces se verifica que:*

$$Opt_{(x,\mathbf{y})} f(x, \mathbf{y}) = Opt_x f_1(x, Opt_{\mathbf{y}} f_2(\mathbf{y}))$$

Demostración.

Supondremos que Opt=Min; el teorema se demuestra de forma análoga si Opt=Max.

Sea $(x^*, \mathbf{y}^*)$ el mínimo de $f(x, \mathbf{y})$. Entonces, por definición de mínimo, se cumple que:

$$f(x^*, \mathbf{y}^*) \leq f(x, \mathbf{y}) = f_1(x, f_2(\mathbf{y})), \quad \forall (x, \mathbf{y})$$

En particular, tomando $\overline{\mathbf{y}}$ tal que $f_2(\overline{\mathbf{y}}) = Min_{\mathbf{y}} f_2(\mathbf{y})$, se tiene que

$$f(x^*, \mathbf{y}^*) \leq f_1(x, f_2(\overline{\mathbf{y}})), \quad \forall x$$

y la desigualdad anterior se sigue verificando si particularizamos en el punto $\overline{x}$, que minimiza $f_1(x, f_2(\overline{\mathbf{y}}))$. Por lo tanto,

$$Min_{(x,\mathbf{y})} f(x, \mathbf{y}) = f(x^*, \mathbf{y}^*) \leq f_1(\overline{x}, f_2(\overline{\mathbf{y}})) = Min_x f_1(x, Min_{\mathbf{y}} f_2(\mathbf{y})).$$

Comprobemos, a continuación, la desigualdad contraria.

Por la hipótesis de descomponibilidad de la función f se tiene que f_1 es monótona no decreciente respecto de su segundo argumento, luego

$$f_1(x, f_2(\overline{\mathbf{y}})) \leq f_1(x, f_2(\mathbf{y})), \quad \forall (x, \mathbf{y}),$$

con $f_2(\overline{\mathbf{y}}) = Min_{\mathbf{y}} f_2(\mathbf{y})$.

La anterior relación se sigue verificando si particularizamos en el valor $\mathbf{y}_x$ que minimiza $f_1(x, f_2(\mathbf{y}))$ para cada x. Luego,

$$f_1(x, f_2(\overline{\mathbf{y}})) \leq f_1(x, f_2(\mathbf{y}_x)), \quad \forall x$$

Tomando ahora el valor $\overline{x}$ que minimiza la expresión $f_1(x, f_2(\mathbf{y}_x))$, se tiene que

$$f_1(\overline{x}, f_2(\overline{\mathbf{y}})) \leq f_1(\overline{x}, f_2(\mathbf{y}_x)) = Min_{(x,y)} f_1(x, f_2(\mathbf{y}))$$

y por otro lado, claramente

$$Min_x f_1(x, Min_{\mathbf{y}} f_2(\mathbf{y})) = Min_x f_1(x, f_2(\overline{y})) \leq f_1(\overline{x}, f_2(\overline{\mathbf{y}}))$$

de donde se sigue que

$$Min_x f_1(x, Min_{\mathbf{y}} f_2(\mathbf{y})) \leq Min_{(x,\mathbf{y})} f_1(x, f_2(\mathbf{y})) = Min_{(x,\mathbf{y})} f(x, \mathbf{y}).$$

■

Ejemplo 2.4. Supongamos que deseamos minimizar la función $f(x, y) = x^2 - y^2$. Puedo separar esta función en $f_1(x, z) = x^2 - z$ y $f_2(y) = y^2$. De esta forma,

$$f_1(x, f_2(y)) = x^2 - f_2(y) = x^2 - y^2 = f(x, y).$$

Vamos a calcular el mínimo utilizando el teorema de optimalidad, evaluando $Min_x f_1(x, Min_y f_2(y))$.

Observemos que $Min_y f_2(y) = Min_y y^2 = 0$ que se alcanza en $\overline{y} = 0$. Por lo tanto, $f_1(x, Min_y f_2(x)) = f_1(x, 0) = x^2$, y consecuentemente, $Min_x f_1(x, Min_y f_2(y)) = Min_x x^2 = 0$, que también se alcanza en $x = 0$. En definitiva, se tendría que $Minf(x, y) = 0$ y dicho mínimo se alcanza en $(0, 0)$.

Sin embargo, esto es falso; es fácil ver que el mínimo de esta función no es cero. El fallo se produce porque con la separación que hemos hecho, la función f_1 no es monótona creciente respecto del segundo argumento, y por lo tanto f no es descomponible en f_1 y f_2.

Probemos de la siguiente manera. Consideremos $f_1(y, z) = z - y^2$ y $f_2(x) = x^2$. De nuevo, $f(x, y) = f_1(y, f_2(x))$, pero ahora la función f_1 sí es monótona no-decreciente respecto de su segundo argumento, luego podemos calcular el mínimo evaluando $Min_y f_1(y, Min_x f_2(x))$.

Observemos que $Min_x f_2(x) = Min_x x^2 = 0$ que se alcanza en $\overline{x} = 0$. Por lo tanto, $f_1(y, Min_x f_2(x)) = f_1(y, 0) = -y^2$, y consecuentemente, $Min_y f_1(y, Min_x f_2(x)) = Min_y - y^2 = -\infty$. Por lo tanto, $Min_{(x,y)} x^2 - y^2 = -\infty$.

2.1.2. Caso con restricciones

El teorema de optimalidad se puede generalizar a problemas con restricciones.

Consideremos el problema $Opt_{(x,\mathbf{y})\in\Omega} f(x,\mathbf{y})$, donde Opt=Max o Min y $\Omega \subset \mathbb{R}^{k+1}$, $\Omega \neq \emptyset$. Para cada $x \in \mathbb{R}$, sea $\Omega_x = \{\mathbf{y} \in \mathbb{R}^k; (x,\mathbf{y}) \in \Omega\}$. Puesto que es posible que $\Omega_x = \emptyset$ para algún x, convendremos que

$$Opt_{\mathbf{y}\in\Omega_x} f_2(\mathbf{y}) = \begin{cases} +\infty & \text{si } \Omega_x = \emptyset \text{ y } Opt = Min \\ -\infty & \text{si } \Omega_x = \emptyset \text{ y } Opt = Max \end{cases}$$

y también que para todo x,

$$f_1(x,+\infty) = +\infty \qquad f_1(x,-\infty) = -\infty$$

Teorema 2.5. *(Teorema de Optimalidad de Mitten. Caso con restricciones) Sea $f(x,\mathbf{y})$ una función real con $x \in \mathbb{R}$, $\mathbf{y} \in \mathbb{R}^k$. Si f es descomponible con $f(x,\mathbf{y}) = f_1(x, f_2(\mathbf{y}))$, entonces se verifica que:*

$$Opt_{(x,\mathbf{y})\in\Omega} f(x,\mathbf{y}) = Opt_x f_1(x, Opt_{\mathbf{y}\in\Omega_x} f_2(\mathbf{y}))$$

2.2. El espacio de estados y la ecuación funcional

Consideremos de nuevo un problema con restricciones del tipo $Opt_{(x,\mathbf{y})\in\Omega} f(x,\mathbf{y})$. Siguiendo el teorema de optimalidad de Mitten, se puede evaluar el óptimo de f en Ω mediante la descomposición $\mathrm{Opt}_x f_1(x, \mathrm{Opt}_{\mathbf{y}\in\Omega_x} f_2(\mathbf{y}))$.

En general, para un conjunto arbitrario Ω, el teorema de optimalidad no ofrece ninguna ventaja práctica, pues la optimización de la expresión anterior equivale a evaluar $f(x,\mathbf{y})$ listando los elementos $(x,\mathbf{y}) \in \Omega$ en un orden determinado. Como

veremos posteriormente, el interés del teorema de optimalidad depende de la estructura del dominio de soluciones Ω, más concretamente, en la posibilidad de considerar Ω como un elemento de una familia de dominios $\Omega_i(E)$ parametrizados por un vector E que denominaremos vector de estados.

Consideremos un problema de optimización con restricciones de la forma

$$\begin{aligned} &\text{Optimizar} \quad f(x_1, x_2, \ldots, x_n) \\ &\quad \text{s.a} \\ &\qquad g(x_1, x_2, \ldots, x_n) \in \mathcal{E}_t \subset \mathbb{R}^m \end{aligned}$$

Nuestra intención es descomponer el problema en muchos problemas de una variable. Para ello necesitamos que f sea descomponible en la forma:

$$f(x_1, x_2, \ldots, x_n) = f_1(x_1, \overline{f_2}(x_2, \ldots, x_n)),$$

con $\overline{f_2}$ igualmente descomponible en la forma

$$\overline{f_2}(x_2, x_3, \ldots, x_n) = f_2(x_2, \overline{f_3}(x_3, \ldots, x_n)),$$

y sucesivamente

$$\overline{f_i}(x_i, x_{i+1}, \ldots, x_n) = f_i(x_i, \overline{f_{i+1}}(x_{i+1}, \ldots, x_n)),$$

y finalmente,

$$\overline{f_{n-1}}(x_{n-1}, x_n) = f_{n-1}(x_{n-1}, f_n(x_n)),$$

La idea es empezar desde el final optimizando x_n y en cada paso resolver un problema de una variable, pero ¿cómo se interpreta la restricción $g(x_1, x_2, \ldots, x_n) \in \mathcal{E}_t$ en cada una de las etapas? Necesito poder descomponer g de una forma análoga a la que he descompuesto f.

Supongamos que existe un espacio de estados $\mathcal{E} \subset \mathbb{R}^m$ de forma que podemos

descomponer g en funciones $h_1, h_2, \ldots, h_n$ de modo que:

$$\begin{aligned} h_1(x_1) &= E_1 \in \mathcal{E} \\ h_2(x_2, E_1) &= E_2 \in \mathcal{E} \\ h_3(x_3, E_2) &= E_3 \in \mathcal{E} \\ &\cdots\cdots \\ h_{n-1}(x_{n-1}, E_{n-2}) &= E_{n-1} \in \mathcal{E} \\ h_n(x_n, E_{n-1}) &= E_n \in \mathcal{E} \end{aligned}$$

y $g(x_1, \ldots, x_n) = E_n \in \mathcal{E}_t$, para que se alcance una solución factible. Las funciones h_i se denominan funciones de transición. De esta forma, para cada $(x_1, \ldots, x_n)$, f y g se pueden calcular recursivamente como:

$$\begin{aligned} f(x_1, \ldots, x_n) &= f_1(x_1, f_2(x_2, f_3(x_3, \ldots, f_{n-1}(x_{n-1}, f_n(x_n)) \ldots) \\ g(x_1, \ldots, x_n) &= h_n(x_n, h_{n-1}(x_{n-1}, h_{n-2}(x_{n-2}, \ldots, h_2(x_2, h_1(x_1)) \ldots) \end{aligned}$$

y por lo tanto, el problema original quedaría como

$$\begin{aligned} \text{(P) Optimizar} \quad & f_1(x_1, f_2(x_2, f_3(x_3, \ldots, f_{n-1}(x_{n-1}, f_n(x_n)) \ldots) \\ \text{s.a} \quad & \\ & g(x_1, \ldots, x_n) = h_n(x_n, h_{n-1}(x_{n-1}, h_{n-2}(x_{n-2}, \ldots, h_2(x_2, h_1(x_1)) \ldots) \in \mathcal{E}_t \end{aligned}$$

Para cada $i = 1, 2, \ldots, n$, definamos

$$\overline{h_i}(x_i, \ldots, x_n, E) = h_n(x_n, h_{n-1}(x_{n-1}, h_{n-2}(x_{n-2}, \ldots, h_{i+1}(x_{i+1}, h_i(x_i, E)) \ldots)$$

obsérvese que se verifica que

$$\overline{h_i}(x_i, \ldots, x_n, E) = \overline{h_{i+1}}(x_{i+1}, \ldots, x_n, h_i(x_i, E)).$$

Entonces, el problema (P) quedaría como

$$\begin{aligned} \text{(P) Optimizar} \quad & f_1(x_1, \overline{f_2}(x_2, x_3, \ldots, x_n)) \\ \text{s.a} \quad & \\ & \overline{h_2}(x_2, \ldots, x_n, h_1(x_1)) \in \mathcal{E}_t \end{aligned}$$

Aplicando el teorema de Mitten, podríamos resolver en primer lugar el problema

$$\begin{aligned} (P_2(E_1))\ \text{Optimizar} \quad & \overline{f_2}(x_2, x_3, \ldots, x_n) \\ \text{s.a} \quad & \\ & \overline{h_2}(x_2, \ldots, x_n, E_1) \in \mathcal{E}_t \end{aligned}$$

cuyo valor óptimo denotamos por $f_2^*(E_1)$ y después obtener f^* optimizando $f_1(x_1, f_2^*(h_1(x_1))$.

Igualmente, para resolver $(P_2(E_1))$, puesto que $\overline{f_2}(x_2, x_3, \ldots, x_n) = f_2(x_2, \overline{f_3}(x_3, \ldots, x_n))$, aplicando el teorema de optimalidad, debemos resolver primero el problema

$$\begin{aligned} (P_3(E_2))\ \text{Optimizar} \quad & \overline{f_3}(x_3, \ldots, x_n) \\ \text{s.a} \quad & \\ & \overline{h_3}(x_3, \ldots, x_n, E_2) \in \mathcal{E}_t \end{aligned}$$

cuyo valor óptimo denotamos por $f_3^*(E_2)$, y posteriormente obtener $f_2^*(E_1)$ optimizando $f_2(x_2, f_3^*(h_2(x_2, E_1))$.

De esta manera, los valores $f_i^*(E_{i-1})$, para $2 \leq i \leq n-1$, se pueden obtener recursivamente, para todo $E_{i-1} \in \mathcal{E}$, mediante

$$f_i^*(E_{i-1}) = Opt_{x_i} f_i(x_i, f_{i+1}^*(h_i(x_i, E_{i-1}))) \tag{2.1}$$

y para $i = n$, obtener, para cada $E_{n-1} \in \mathcal{E}$, $f_n^*(E_{n-1})$ como el valor óptimo del problema

$$\begin{aligned} (P_n(E_{n-1}))\ \text{Optimizar} \quad & f_n(x_n) \\ \text{s.a} \quad & \\ & h_n(x_n, E_{n-1}) \in \mathcal{E}_t \end{aligned}$$

De este modo, es posible resolver el problema (P) por recurrencia, encontrando en cada paso los valores $f_i^*(E_{i-1})$, para $i = n, n-1, \ldots, 2$ finalmente obteniendo f^* a partir de f_2^*. La expresión 2.1 se denomina **ecuación fundamental de la programación dinámica**.

Algoritmo de retroceso (Algoritmo "backward")

Etapa n

$\forall E \in \mathcal{E}$, calcular

$$(P_n(E)) \text{ Optimizar} \quad f_n(x_n)$$
$$\text{s.a}$$
$$h_n(x_n, E) \in \mathcal{E}_t$$

Si para algún $\overline{E} \in \mathcal{E}$ el problema anterior es infactible, tomar

$$f_n^*(\overline{E}) = \begin{cases} +\infty & \text{si } Opt = Min \\ -\infty & \text{si } Opt = Max \end{cases}$$

Etapa $i = n-1, n-2, \ldots, 2$

$\forall E \in \mathcal{E}$, calcular

$$f_i^*(E) = Opt_{x_i} f_i(x_i, f_{i+1}^*(h_i(x_i, E)))$$

Etapa 1

Obtener f^* mediante

$$f_1^* = Opt_{x_1} f_1(x_1, f_2^*(h_i(x_1)))$$

Como se observa a partir del anterior algoritmo, al igual que ocurría en la resolución de los problemas tipo mochila, la metodología general de la programación dinámica

consiste en:

- considerar el problema dado dentro de una familia de problemas de la misma naturaleza
- vincular, mediante una relación de recurrencia, los valores óptimos de los problemas de esta familia
- resolver tal relación de recurrencia (la ecuación funcional) para encontrar el valor óptimo del problema dado

El siguiente ejemplo pone de manifiesto la necesidad de la condición de monotonía en la descomponibilidad de la función para asegurar la validez de la ecuación fundamental de la Programación Dinámica.

Ejemplo 2.6. Consideramos el siguiente problema:

$$\text{Maximizar } x \cdot y^3$$

$$\text{s.a}$$

$$x + y \leq 3, x \geq -2, -2 \leq y \leq 0$$

Ahora la función es separable, $f(x,y) = f_1(x, f_2(y))$ con $f_1(x,w) = x \cdot w$, $f_2(w) = w^3$. $\xi = [-2, +\infty)$

Etapa 2

$$F_2^*(E) = \text{Max} \quad y^3$$

$$\text{s.a}$$

$$E + y \leq 3$$

$$-2 \leq y \leq 0$$

- Si $E > 5 \quad E + y > 3$ para todo $y; -2 \leq y \leq 0 \Rightarrow F_2^*(E) = -\infty$

- Si $3 \leq E \leq 5, y^* = 3 - E$ (para que se verifique la restricción) $\Rightarrow F_2^*(E) = (3-E)^3$
- Si $-2 \leq E < 3$, entonces para todo valor de $y; -2 \leq y \leq 0$, se verifica la restricción, y por tanto $y^* = 0 \Rightarrow F_2^*(E) = 0$

Etapa 1

$$F_1^*(0) = \text{Max} \quad x \cdot F_2^*(x)$$
$$\text{s.a}$$
$$x \geq -2$$

- Si $x > 5 \Rightarrow F_2^*(x) = -\infty$
- Si $3 \leq x \leq 5 \Rightarrow x \cdot F_2^*(x) = x \cdot (3-x)^3$. Como esta función es decreciente en el intervalo $[3, 5]$, el mejor valor sería $x = 3; F_1(0) = 0$
- Si $-2 \leq x < 3 \Rightarrow F_2^*(x) = 0 \Rightarrow F_1(0) = 0$

Por tanto se obtiene que el valor óptimo del problema es $F_1^*(0) = 0$. Sin embargo, se ve claramente que el valor óptimo del problema es en realidad 16 correspondiente a la solución $(-2, -2)$. Esto es consecuencia de que la función objetivo a pesar de ser separable no es descomponible.

2.3. El principio de optimalidad

El algoritmo "backward" cuya validez se deriva del teorema de optimalidad, se puede interpretar como la construcción de una solución óptima a partir de soluciones óptimas parciales. Cuando llegamos a la etapa i con estado E, de todas las posibles soluciones $(x_i, x_{i+1}, \ldots, x_n)$ que conducen a estamos terminales $(\overline{h_i}(x_i, x_{i+1}, \ldots, x_n, E) \in \mathcal{E}_t)$ sólo guardamos la correspondiente a la solución óptima (la correspondiente a $f_i^*(E)$).

La solución óptima obtenida se ajusta al **principio de optimalidad** establecido por R. Bellman en 1957.

> Una política óptima tiene la propiedad de que, cualquiera que sea el estado inicial y las decisiones iniciales, las restantes decisiones deben constituir una política óptima con respecto al estado resultante de la primera decisión

2.4. Ejemplos de funciones descomponibles

Las funciones descomponibles más usuales son:

- $f(x_1, ..., x_n) = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n)$
- $f(x_1, ..., x_n) = f_1(x_1) \cdot f_2(x_2) \cdot \cdots \cdot f_n(x_n) \quad f_i(x_i) \geq 0, \forall i$
- $f(x_1, ..., x_n) = Opt\{f_1(x_1), ..., f_n(x_n)\}$

2.5. Conjunto de estados finito y la búsqueda del camino más corto en un grafo

Supongamos que el conjunto de estados ξ es finito, $\|\xi\| = p$ y que la función f es descomponible en la forma $f(x_1, ..., x_n) = f_1(x_1) + \cdots + f_n(x_n)$
Se considera el grafo $G = [X, A]$ definido del siguiente modo:

- El conjunto de nodos X consta de los siguientes elementos:
  - Un nodo inicial que denotaremos por I

- np nodos de la forma (E, i), con $E \in \xi$ y $1 \leq i \leq n$
- Un conjunto de nodos terminales $X_t \subset X$ que son aquellos de la forma (E, n) con $E \in \mathcal{E}_t$ (conjunto de estados terminales).

- El conjunto de arcos A consiste en:
  - Los arcos de la forma $[I, (E, 1)]$, donde $E = h_1(x_1)$ para todos los posibles valores de x_1 tales que $h_1(x_1) \in \xi$. A cada arco de este tipo se le considera longitud $f_1(x_1)$.
  - Para cada $i = 2, ..., n$, y para todo $E \in \xi$, se consideran los arcos de la forma $[(E, i-1), (E', i)]$, donde $E' = h_i(x_i, E)$ para todos los valores posibles de x_i tales que $h_i(x_i, E) \in \xi$. A cada arco de este tipo se le considera longitud $f_i(x_i)$.

De esta forma, cada camino entre I y X_t es una solución factible del problema (por cuanto satisface la restricción $g(x_1, ..., x_n) \in \mathcal{E}_t$ y la longitud de tal camino es $f_1(x_1) + ... + f_n(x_n)$. Por lo tanto, la resolución del problema por programación dinámica es equivalente a la búsqueda del camino más corto (o más largo, en caso de maximizar) entre I y X_t en el grafo G.

A partir de esta equivalencia entre la programación dinámica y la búsqueda del camino más corto se puede definir un segundo algoritmo de programación dinámica, en el cual, a diferencia del anterior, los cálculos se llevan a cabo en orden creciente de los subíndices de las variables inmersas en el problema.

Para llevar a cabo el algoritmo se asocia a cada vértice (E, i) un número $P(E, i)$. Al final de la etapa $i-1$, $P(E, i-1)$ representa la longitud de un camino óptimo entre el vérice inicial I y el vértice $(E, i-1)$. En la etapa i, se toma, para todo E, $P(E, i) = -(+)\infty$ (según la el sentido de optimización del problema). La etapa i consiste en, a partir de los valores óptimo $P(E, i-1)$, actualizar los valores (E, i) de forma que al final de

la etapa i, $P(E,i)$ represente efectivamente la longitud del camino más corto (o más largo) entre I y (E,i).

Algoritmo de avance (Algoritmo Forward)

Etapa 1

Para todo $E \in \xi$, se establece:

$$P(E,1) = +\infty \text{ si Opt=Min}$$

$$P(E,1) = -\infty \text{ si Opt=Max}$$

Para todos los posibles valores de x_1 tales que $h_1(x_1) \in \xi$ calcular $E = h_1(x_1)$. Hacer $P(E,1) = f_1(x_1)$.

Etapa i $\quad \forall i = 2, ..., n$

Para todo $E \in \xi$, se establece:

$$P(E,i) = +\infty \text{ si Opt=Min}$$

$$P(E,i) = -\infty \text{ si Opt=Max}$$

Para todo E tal que $\|P(E,i-1)\| < \infty$ y para todos los posibles valores de x_i tales que $h_i(x_i,E) \in \xi$ calcular:

$$E' = h_i(x_i,E)$$

$$P(E',i) = Opt\{P(E',i), P(E,i-1) + f_i(x_i)\}$$

Etapa n

$$F^* = Opt_{E \in \mathcal{E}_t} P(E,n)$$

De esta manera se obtiene el valor óptimo del problema, pero no la solución óptima $\mathbf{x}^*$ que lo alcanza. Para ello se define, para cada $i = 1, ..., n$ y para cada $E \in \xi$ la cantidad $x_i'(E)$ que contiene el valor de la variable x_i para la que se ha obtenido el valor óptimo $P(E, i)$.

En la etapa 1, $x_1'(E) = x_1$ (siendo $E = h_1(x_1)$) y en cada etapa general i $(1 \leq i \leq n)$ se actualiza $x_i'(E) = x_i$, cada vez que una etiqueta $P(E, i)$ sea mejorada y reemplazada por $P(E', i-1) + f_i(x_i)$. Con esta notación, se obtiene una solución óptima mediante el siguiente algoritmo:

a) $E = E^*$, donde E^* es tal que

$$F^* = Opt_{E \in \mathcal{E}_t} P(E, n) = P(E^*, n)$$

b) Para cada $i = n, n-1, ..., 1$ sucesivamente:

$$x_i^* = x_i'(E)$$

Hacer $E = E'$, donde E' es tal que $E = h_i(x_i, E')$.

Ejemplo 2.7. Considérese el siguiente problema de mochila para resolverlo por el algoritmo Forward.

$$\text{Maximizar } f(x_1, x_2, x_3, x_4) = 4x_1 + 5x_2 + 3x_3 + 2x_4$$

s.a

$$4x_1 + 3x_2 + 2x_3 + x_4 \leq 8$$

$$x_i \in \{0, 1\}$$

La función f se descompone en las siguientes funciones:

$$f_1(x_1) = 4x_1 \quad f_2(x_2) = 5x_2 \quad f_3(x_3) = 3x_3 \quad f_4(x_4) = 2x_4$$

Las funciones de transición son:

$$h_1(x_1) = 4x_1 \quad h_2(x_2, E) = 3x_2 + E \quad h_3(x_3, E) = 2x_3 + E \quad h_4(x_4, E) = x_4 + E$$

El espacio de estados es $\xi = \{0, 1, 2, ..., 8\}$

Etapa 1

$$P(E, 1) = -\infty \qquad \forall E \in \xi$$

Los posibles valores de x_1 tales que $h_1(x_1) = 4x_1 \in \xi$ son $x_1 = 0$ y $x_1 = 1$

$$P(0, 1) = f_1(0) = 0 \qquad x_1(0) = 0$$

$$P(4, 1) = f_1(1) = 4 \qquad x_1(4) = 1$$

Etapa 2

$$P(E, 2) = -\infty \quad \forall E \in \xi$$

$\|P(0, 1)\| = 0 < \infty$. Tanto para $x_2 = 0$ como para $x_2 = 1$, $h_2(x_2, 0) \in \xi$

$$0 = h_2(0, 0) \quad P(0, 2) = Max\{-\infty, P(0, 1) + f_2(0) = 0\} = 0 \qquad x_2(0) = 0$$

$$3 = h_2(1, 0) \quad P(3, 2) = Max\{-\infty, P(0, 1) + f_2(1) = 5\} = 5 \qquad x_2(3) = 1$$

De la misma forma, cogiendo $P(4, 1)$:

$$P(4, 2) = 4 \qquad x_2(4) = 0$$

$$P(7, 2) = 9 \qquad x_2(7) = 1$$

Etapa 3

Repitiendo el proceso anterior se obtienen los siguientes resultados:

$$P(0, 3) = 0 \qquad x_3(0) = 0$$

$$P(2,3) = 3 \qquad x_3(2) = 1$$

$$P(3,3) = 5 \qquad x_3(3) = 0$$

$$P(5,3) = 8 \qquad x_3(5) = 1$$

$$P(4,3) = 4 \qquad x_3(4) = 0$$

$$P(6,3) = 7 \qquad x_3(6) = 1$$

$$P(7,3) = 9 \qquad x_3(7) = 0$$

Nótese que al partir de $E = 7$ el único valor posible para x_3 es cero, pues $h_3(1,7) = 9 \notin \xi$.

Etapa 4

En la última etapa se obtienen los siguientes resultados:

$$P(0,4) = 0 \qquad x_4(0) = 0$$

$$P(1,4) = 2 \qquad x_4(1) = 1$$

$$P(2,4) = 3 \qquad x_4(2) = 0$$

$P(3,4) = 5$ $\qquad x_4(3) = 1$ a través de P(2,3) o $x_4(3) = 0$ a través de $P(3,3)$

$$P(4,4) = 7 \qquad x_4(4) = 1$$

$$P(5,4) = 8 \qquad x_4(5) = 0$$

$$P(6,3) = 10 \qquad x_4(6) = 1$$

$P(7,4) = 9$ $\qquad x_4(7) = 1$ a través de P(6,3) o $x_4(7) = 0$ a través de $P(7,3)$

$$P(8,4) = 11 \qquad x_4(8) = 1$$

Por tanto, el valor óptimo del problema es $F^* = 11$

Etapa Solución óptima

El óptimo se alcanza en $P(8,4)$. Por tanto se parte de $E = 8$.

$x_4(8) = 1$, por tanto $x_4^* = 1$. Encontrar E' tal que $h_4(1, E') = 8$. $E' = 7$. Hacer $E = E'$

$E = 7$, entonces, $x_3^* = x_3(7) = 0$. Encontrar E' tal que $h_3(0, E') = 7$. $E' = 7$. Hacer $E = E'$.

$E = 7$, entonces, $x_2^* = x_2(7) = 1$. Encontrar E' tal que $h_2(1, E') = 7$. $E' = 4$. Hacer $E = E'$.

$E = 4$, entonces, $x_1^* = x_1(4) = 1$.

Por tanto la solución óptima es:

$$(x_1, x_2, x_3, x_4) = (1, 1, 0, 1)$$

2.6. Ejercicios

1. Resolver, mediante programación dinámica, el siguiente problema no lineal entero:

$$\text{Max } x_1 \cdot x_2^2 \cdot x_3^3$$

s.a

$$x_1 + 2x_2 + 3x_3 \leq 10$$

$$x_1, x_2, x_3 \geq 1$$

$$x_1, x_2, x_3 \in \mathbb{Z}_+$$

2. Resolver el siguiente problema mediante programación dinámica

$$\text{Max } 3x_1^2 - x_1^3 + 5x_2^2 - x_2^3$$

s.a

$x_1 + 2x_2 \leq 4$

$x_1, x_2 \geq 0$

3. Resolver, mediante programación dinámica, el siguiente problema no lineal entero:

Max $18x_1 - x_1^2 + 20x_2 + 10x_3$

s.a

$2x_1 + 4x_2 + 3x_3 \leq 11$

$x_1, x_2, x_3 \in \mathbb{Z}_+$

4. Resolver el siguiente problema no lineal mediante programación dinámica

Max $36x_1 + 9x_1^2 - 6x_1^3 + 36x_2 - 3x_2^3$

s.a

$x_1 + x_2 \leq 3$

$x_1, x_2 \geq 0$

5. Utilizar programación dinámica para resolver el siguiente problema:

Max $x_1^2 \cdot x_2$

s.a

$x_1^2 + x_2 \leq 2$

6. Resolver el siguiente problema mediante programación dinámica

$$\text{Max } 2x_1^2 + 2x_2 + 4x_3 - x_3^2$$

s.a

$$2x_1 + x_2 + x_3 \leq 4$$

$$x_1, x_2, x_3 \geq 0$$

7. Utilizar programación dinámica para resolver el siguiente problema:

$$\text{Max } 2x_1 + x_2^2$$

s.a

$$x_1^2 + x_2^2 \leq 4$$

$$x_1, x_2 \geq 0$$

8. Resolver el siguiente problema utilizando programación dinámica:

$$\text{Max } x_1^3 + 4x_2^2 + 16x_3$$

s.a

$$x_1 \cdot x_2 \cdot x_3 = 4$$

$$x_i \geq 1 \quad i = 1, 2, 3$$

Resolverlo también añadiendo la restricción $x_j \in \mathbb{N}$.

9. Resolver el siguiente problema mediante programación dinámica

$$\text{Max } (x_1+2)^2 + x_2x_3 + (x_4-5)^2$$

s.a

$$x_1 + x_2 + x_3 + x_4 \leq 5$$

$$x_i \in \mathbb{Z}_+ \quad i = 1,2,3,4$$

10. Resolver con programación dinámica el siguiente problema de programación lineal:

$$\text{Max } 2x_1 + 5x_2$$

s.a

$$2x_1 + x_2 \leq 430$$

$$2x_2 \leq 460$$

$$x_1, x_2 \geq 0$$

11. Resolver el siguiente problema mediante programación dinámica

$$\text{Max } 5x_1 + x_2$$

s.a

$$2x_1^2 + x_2 \leq 13$$

$$x_1^2 + x_2 \leq 9$$

$$x_1, x_2 \geq 0$$

12. Resolver el siguiente problema de la mochila mediante el algoritmo de avance:

$$\text{Max } 5x_1 + 3x_2 + x_3 + x_4$$

$$\text{s.a}$$

$$4x_1 + 2x_2 + 3x_3 + 2x_4 \leq 5$$

$$x_i \in \{0, 1, 2\} \quad i = 1, 2, 3, 4$$

Representar gráficamente el grafo sobre el cual el cálculo del camino más largo es equivalente a la resolución por programación dinámica de este problema.

13. Resolver el siguiente problema de la mochila mediante el algoritmo de avance:

$$\text{Max } 3x_1 + 5x_2 + x_3 + x_4 - x_5$$

$$\text{s.a}$$

$$2x_1 + 4x_2 + 3x_3 + 2x_4 + x_5 = 5$$

$$x_i \in \{0, 1\} \quad i = 1, 2, 3, 4, 5$$

Representar gráficamente el grafo sobre el cual el cálculo del camino más largo es equivalente a la resolución por programación dinámica de este problema.