UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA

PROGRAMACIÓN DINÁMICA

Idalia Flores de la Mota

DIVISIÓN DE INGENIERÍA MECÁNICA E INDUSTRIAL

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA

PROGRAMACIÓN DINÁMICA

IDALIA FLORES DE LA MOTA

División de Ingeniería Mecánica e Industrial
Departamento de Ingeniería de Sistemas

FLORES de la Mota, Idalia.
Programación dinámica.
México, Universidad Nacional Autónoma de México.
Facultad de Ingeniería, 2015, 310 pp.

Programación dinámica

Primera edición: 23 de septiembre de 2015

Facultad de Ingeniería
Avenida Universidad núm. 3000, Ciudad Universitaria,
Delegación Coyoacán, C. P. 04510, México, D. F.
http://www.ingenieria.unam.mx/

ISBN: 978-607-02-7160-1

Cuidado de la edición: María Alicia Medina, Unidad de Apoyo Editorial.
Diseño de portada: Nismet Díaz Ferro, Unidad de Apoyo Editorial.
Imagen de la portada: Kris Northem.

PRÓLOGO

Como parte de las actividades del Departamento de Ingeniería de Sistemas de la División de Ingeniería Mecánica e Industrial de la Facultad de Ingeniería de la UNAM (Universidad Nacional Autónoma de México), nos hemos propuesto el desarrollo de una serie de apuntes que sirvan de apoyo a los diferentes cursos que se imparten y, desde luego, como material de referencia o de lectura para quienes, así, lo requieran.

En dichos apuntes se busca mantener un alto nivel, de manera, que contribuyan a una sólida formación teórica del alumno; pero, al mismo tiempo, se intenta acercarlo a las aplicaciones de tales conocimientos como es en síntesis el objetivo central del Posgrado de Ingeniería.

Estos apuntes aportan conocimientos básicos a los alumnos de la Maestría en Ingeniería de Sistemas, así como, a los alumnos de licenciatura interesados en el tema de la "programación dinámica".

La redacción de este material fue una tarea agradable y la autora espera que los estudiantes de ingeniería lo encuentren grato e informativo cuando traten de aprender, cómo debe aplicarse la estadística en sus campos de interés.

Idalia Flores

AGRADECIMIENTOS

A la DGAPA (Dirección General de Asuntos del Personal Académico) de la UNAM por el apoyo económico proporcionado para la elaboración de este material, a través del proyecto PAPIIT IN-116012. Así como, a la División de Ingeniería Mecánica e Industrial por las gestiones realizadas para la consecución de dicho proyecto.

Agradezco a mis alumnos de la Maestría en Investigación de Operaciones que aportaron sus conocimientos en el desarrollo de los mismos: Oroselfia Sánchez, Julio César Guevara, Mauricio Martínez, Javier Lara de Paz, Blanca García, J. J. Arellano, Israel Gil, Carlos González, Miguel Aguilar y Gabriela Ramírez.

La autora

Finalmente, quiero agradecer, de manera especial, a la Maestra María Cuairán Ruidíaz, jefa de la Unidad de Apoyo Editorial de la Facultad de Ingeniería de la UNAM, por el respaldo brindado y por la colaboración para la realización de la edición de esta obra.

A la Lic. Elvia Angélica Torres Rojas porque hizo la revisión de la primera versión de la obra.

A la LDG Nismet Díaz Ferro por la dedicación y disposición para el diseño de la portada que integra esta obra.

Y a la Lic. María Alicia Medina por su dedicación, revisión acertada y minuciosa del estilo y la redacción de la obra.

CONTENIDO

CAPÍTULO 1

LA IDEA DE RECURSIVIDAD

1.1. INTRODUCCIÓN

La parte fundamental de la metodología de la PD (Programación Dinámica) es la recursividad, por lo que, se construyen ecuaciones recursivas para dar solución a los problemas que se buscan resolver, por esto, en este primer capítulo, el objetivo es dar una idea general de recursividad, ya que, si bien, no es un concepto complicado, sí es necesario aprender a pensar de esta manera al plantear los modelos. Así como, en PLE (Programación Lineal Entera) se tiene una idea iterativa al resolver un modelo, en PD la idea es recursiva.

Los algoritmos recursivos son de gran importancia en la ciencia de la computación para la solución de problemas de alta complejidad operacional, ya que, son sorprendentemente concisos, fáciles de entender y algorítmicamente eficientes. Para quienes estudian la recursividad por primera vez, la recursión puede parecer oscura y difícil. A diferencia de otras técnicas de solución que se observan en la vida diaria, la recursión es una idea que no es familiar y que a menudo requiere pensar en los problemas de una manera nueva y diferente.

En términos generales, la recursión es un proceso para resolver un problema complejo reduciéndolo a uno o más subproblemas con las siguientes características:

1) Tienen una estructura idéntica al problema original.

2) Son más sencillos de resolver.

Una vez que se ha hecho la subdivisión original, se usa la misma técnica de descomposición para dividir cada uno de los subproblemas en nuevos, que sean menos complejos. Eventualmente, los subproblemas son tan simples que a menudo se resuelven sin tener que subdividir más y la solución completa se obtiene reensamblando los componentes resueltos.

La idea subyacente es la de una organización tipo pirámide, en la cual una persona ingresa como vendedor con una inversión inicial y el objetivo es que consiga que más personas entren como vendedores, lo cual tiene como consecuencia más ventas y, por lo tanto, más ganancias. Estas personas, a su vez, tienen que reclutar a más vendedores y así, sucesivamente. Dichas ganancias se reparten entre los que ingresan y los que están más arriba de la misma, como se puede ver en la figura 1.1.

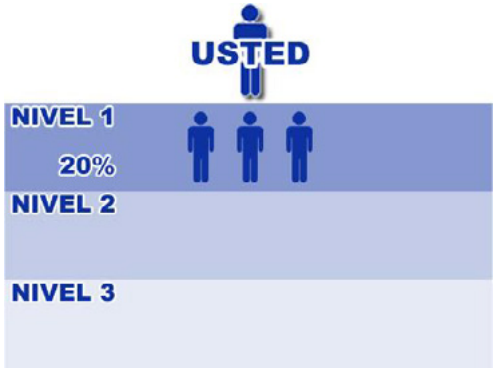


FIGURA 1.1. Estructura piramidal

1.2. ALGUNOS EJEMPLOS DE RECURSIVIDAD

Entre los años 1907 y 1914, floreció una nueva fase del arte moderno en París. Sus detractores le dieron el nombre de cubismo, este movimiento estaba basado en la teoría de que la naturaleza debe representarse en términos de sus componentes geométricos, tales como, conos, cilindros y esferas. Aunque, la comunidad cubista se disolvió con el inicio de la Primera Guerra Mundial, las ideas del movimiento permanecieron como una poderosa fuerza que dio forma a las ideas posteriores en el arte abstracto. En particular, estas ideas influyeron en la obra del pintor holandés Piet Mondrian, cuyo trabajo está caracterizado por la rigidez de patrones geométricos de líneas verticales y horizontales.

La tendencia del trabajo de Mondrian, a través de una estructura geométrica simple, lo hace particularmente apropiado para la simulación en computadora. Muchos de los primeros esfuerzos por generar arte por computadora se basaron en este estilo. En la figura siguiente, podemos ver que el diseño consiste en un gran rectángulo que se corta en rectángulos menores por una secuencia de líneas verticales y horizontales. Aquí, el punto es encontrar una estrategia general para generar un diseño, así que, como se puede observar, se parte del hecho de dividir dicho rectángulo.

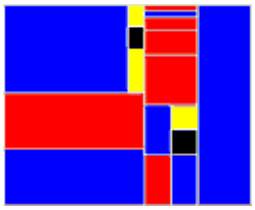


FIGURA 1.2. Recursividad en el arte[1]

Esto nos lleva a la idea de fractales (ver anexo A) donde se observa. También, se aprecia que hay un patrón el cual se repite, como podemos verlo en la figura siguiente:

FIGURA 1.3. Imagen de fractal[2]

Y, posteriormente, lo veremos en su expresión matemática.

Aquí, podemos dar muchos ejemplos, pero solo nos limitaremos a uno más, ya que, posteriormente en el curso volveremos a retomarlos.

La sucesión de Fibonacci

Para iniciar, consideremos la siguiente sucesión de números:

1, 1, 2, 3, 5, 8, 13, 21, 34,...

[1] "Mondrimat". Disponible en: http://www.stephen.com/mondrimat/
[2] "Fractales", 2010. Disponible en: http://dubrieldice.blogspot.mx/2010/10/fractales.html

En donde, cada número a partir del tercero, se obtiene sumando los dos que le preceden.

Por ejemplo: 21 = 13 + 8, el siguiente a 34 será 34 + 21 = 55.

Esta sucesión es la llamada *Sucesión de Fibonacci* (Leonardo de Pisa 1170-1240).

Los cocientes (razones) entre dos números de la sucesión, se aproximan más y más al número áureo (1'61803...).

Esta sucesión de números aparece en la naturaleza en formas curiosas. Las escamas de una piña aparecen en espiral alrededor del vértice, si contamos el número de espirales de una piña, encontraremos que siempre es igual a uno de los números de la sucesión de Fibonacci.

FIGURA 1.4. Espirales en una piña y Fibonacci[3]

También, esta sucesión aparece en el estudio de las leyes mendelianas de la herencia, en la divergencia foliar, en la formación de la concha de algunos moluscos.

[3] "La espiral en el mundo vegetal". Disponible en: http://www.ite.educacion.es/formacion/enred/web_espiral/naturaleza/vegetal/vegetal.htm

FIGURA 1.5. Espiral en la concha de Nautilus[4]

Una manera práctica de dibujar una espiral es mediante la construcción rectangular en las espirales de los cuadrados o sea, se trata de dibujar el cuadrante de un círculo en cada nuevo cuadrado que se añada.

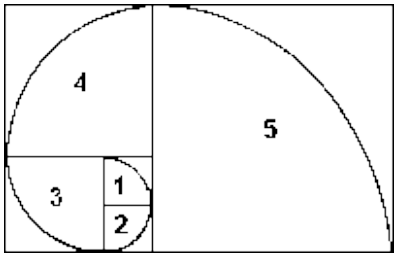


FIGURA 1.6. Dibujo de una espiral

En la construcción anterior, se empieza con un cuadrado de 1 unidad de lado (el número 1), se añade uno igual para formar un rectángulo de 2 × 1, a continuación, añadimos un cuadrado de 2 × 2 (el número 3) para formar un rectángulo de 3 × 2; después, un cuadrado de 3 × 3 (el número 4), de manera que, el siguiente rectángulo es 5 × 3, el siguiente cuadrado es 5 × 5 (el número 5) y así, sucesivamente.

Asimismo, en literatura se encuentra en cuentos hindúes, así como, en una célebre novela de Lewis Carroll, entre otros.

[4] "Nautilus", 2015. Disponible en: http://moirajohnson.blog.com/2013/06/24/nautilus/

1.3. CARACTERÍSTICAS DE LOS ALGORITMOS RECURSIVOS

En cada uno de los ejemplos mencionados anteriormente, se observa que el encontrar subproblemas más sencillos a partir de un problema más complejo es una tarea razonablemente sencilla. Estos problemas se pueden resolver bajo la estrategia de divide y vencerás, haciendo que las soluciones recursivas sean particularmente apropiadas.

En la mayoría de los casos, la decisión de usar la recursividad está sugerida por la naturaleza misma del problema. Para que un problema sea un candidato apropiado para resolverse con recursión debe tener las siguientes características:

1) El problema original debe ser posible descomponerlo en partes más simples que el del mismo problema.

2) Una vez que cada uno de esos subproblemas más simples ha sido resuelto, debe ser posible combinar esas soluciones para producir una solución del problema original.

3) Como el problema original se ha dividido en problemas, sucesivamente, de menor complejidad, debe llegar un momento en que sean tan simples de resolver, que ya no sea necesario subdividirlos más.

1.3.1. ALGORITMOS RECURSIVOS

Se dice que un objeto es *recursivo,* si en parte está formado por sí mismo o se define en función de sí mismo. La recursión se encuentra no solo en matemáticas, sino también en la vida diaria. Una imagen pictórica de esta idea es la figura 1.7.

FIGURA 1.7. Recursividad en imagen de Escher[5]

[5] "Recursive paintings and prints", 2008. Disponible en: http://wordaligned.org/articles/recursive-pictures

El concepto de recursividad va ligado al de repetición. Son recursivos aquellos algoritmos que, estando encapsulados dentro de una función, son llamados desde ella misma una y otra vez, en contraposición a los algoritmos iterativos, que hacen uso de bucles while, do-while, for, etc. Para que una definición recursiva sea válida, la referencia a sí misma debe ser *relativamente más sencilla* que el problema original considerado.

A continuación, se presentan unos ejemplos de algoritmos recursivos:

Ejemplo 1.1. Definición de número natural.

§ El 0 es un número natural.

§ Si $n-1$ es natural, entonces n lo es.

Ejemplo 1.2. La función factorial $n!$ para enteros no negativos.

§ $0!=1$

§ $n>0:\ \ n!=n\,(n-1)!$

Ejemplo 1.3. Estructuras de un árbol en una red.

Cualquier red está formada por: (1) nodos, (2) arcos que unen a los nodos y (3) flujos en los arcos. Una red se dice que es un *árbol,* si está conectada y sin circuitos. Por ejemplo: la red de la figura 1.8 muestra un árbol.

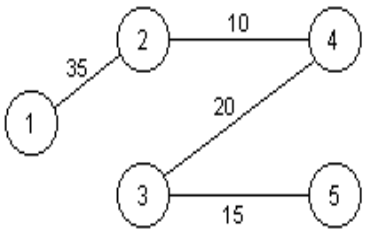


FIGURA 1.8. Estructura de un árbol

A continuación, se presenta la construcción de dicho árbol. Se parte de la red sin conectar, es decir, se consideran los nodos solamente y los arcos con el menor costo asociados a cada nodo. Por la definición de árbol, se sabe que un nodo con un arco asociado, también, es un árbol, como se puede ver en la figura 1.9.

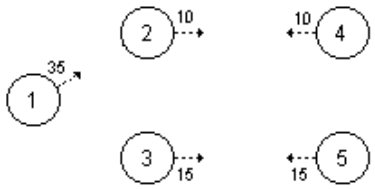


FIGURA 1.9. Formación de un árbol, inicio

Una propiedad importante de los árboles que servirá para la construcción de un árbol es la siguiente:

Si t_1 y t_2 son árboles, entonces, las estructuras formadas por un nodo con dos árboles descendientes, también, son un árbol.

Nuevamente, en la figura 1.10, vemos el paso siguiente para la construcción de un árbol. Se muestran dos árboles construidos a partir de los cinco originales, el último paso da como resultado el árbol de la figura 1.8.

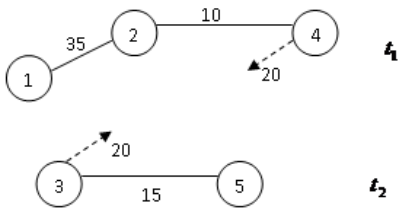


FIGURA 1.10. Formación de un árbol, primer paso

1.4. ALGUNAS CONSIDERACIONES SOBRE LA RECURSIVIDAD

El poder de la recursión radica en la posibilidad de definir un conjunto infinito de objetos mediante una proposición finita. De la misma manera, un número infinito de cálculos pueden describirse con un programa recursivo finito, pese a que no contenga repeticiones explícitas. Sin embargo, los *algoritmos recursivos* son idóneos, principalmente, cuando el problema por

resolver, la función por calcular o la estructura de datos por procesar ya están definidos en términos recursivos. En general, un programa recursivo *P* puede expresarse como una composición *P* de un conjunto de proposiciones *S* (que contienen a *P*) y *P*.

$$P \equiv P[S, P]$$

La herramienta necesaria y suficiente para expresar los programas, recursivamente, es el procedimiento o subrutina, ya que permite dar a una proposición un nombre con el cual puede ser llamada. Si un procedimiento *P* contiene una referencia explícita a sí mismo, se le denomina *directamente recursivo*; por otro lado, si *P* contiene una referencia a otro procedimiento llamado *Q,* que a su vez incluye una referencia de forma directa o indirecta a *P,* entonces, se dice que es *indirectamente recursivo.* Esto indica que el uso de la recursión puede ser evidente o no en el texto de un programa.

En general, se asocia un conjunto de objetos locales a un procedimiento, es decir, un conjunto de variables, constantes, tipos y procedimientos que definen localmente a este procedimiento, pero que carecen de significado fuera de él. Cada vez que el procedimiento se activa de forma recursiva, se crea un nuevo conjunto de variables locales acotadas. Aunque, llevan el mismo nombre que sus elementos correspondientes en el conjunto local del caso anterior del procedimiento, sus valores son específicos y se evita cualquier conflicto en la imposición del nombre por medio de las reglas de cobertura de los identificadores, ya que estos siempre se refieren al conjunto de variables de más reciente creación. La misma regla es aplicada para los parámetros de procedimiento.

Al igual que las proposiciones repetitivas, los procedimientos recursivos tienen la posibilidad de llevar a cabo cálculos con ciclos infinitos, es por eso que hay que tener presente el proceso de la finalización. Para dichos procedimientos, un requisito fundamental es que las llamadas recursivas de *P* estén sujetas a una condición *B* que en un momento determinado resultará falsa. El *esquema general de los algoritmos recursivos* se expresa en cualquiera de las dos siguientes alternativas:

$$P \equiv IF\ B\ THEN\ P[S, P]\ END$$

$$P \equiv\ P[S,\ IF\ B\ THEN\ P\ END]$$

En las repeticiones, la técnica básica para demostrar la terminación consiste en:

1) Definir una condición $f(x)$, donde x representa al conjunto de las variables, tal que, $f(x) \leq 0$ implica la condición de terminación.

2) Probar que $f(x)$ disminuye en cada paso de la repetición.

De la misma manera, la finalización de una recursión se puede probar demostrando que cada ejecución de *P* disminuye un poco $f(x)$ y que $f(x) \leq 0$ implica que tiende a la condición *B*. Una manera de garantizar dicha terminación es asociando un valor o parámetro n con P y llamar recursivamente a P con $n-1$ como valor del parámetro. Sustituir $n > 0$ para B entonces, asegura la terminación.

En la práctica es necesario demostrar que la finalidad de la recursión no es solamente finita, sino que en realidad es muy pequeña, ya que después de cada recursión de un procedimiento P es necesaria cierta cantidad de memoria para alojar las variables. Además, de esas variables locales, el estado actual de la computación debe registrarse con el fin de que se recupere, cuando finalice la recursión de P y se reanude la anterior.

En un algoritmo recursivo se distinguen como mínimo dos partes:

Caso trivial, base o de fin de recursión
En este caso, el problema puede resolverse sin tener que hacer uso de una nueva llamada a sí misma. Evita la continuación indefinida de las partes recursivas.

Parte puramente recursiva
Esta parte relaciona el resultado del algoritmo con resultados de casos más simples. Se hacen nuevas llamadas a la función, pero están más próximas al caso base. El siguiente ejemplo muestra en un pseudocódigo lo que aquí se menciona.

Ejemplo 1.4. Función factorial.

Para el ejemplo de los números factoriales, los algoritmos iterativo y recursivo correspondientes son los siguientes:

- Iterativo

```
int Factorial( int n )
{ int i, res=1;
for(i=1; i<=n; i++ )
res = res*i;
return(res);
```

- Recursivo

```
int Factorial( int n ) {
if(n==0) return(1);
return(n*Factorial(n-1));
```

Ejemplo 1.5. Las Torres de Hanói.

Este problema clásico de matemáticas consiste en pasar una cierta cantidad de discos de diferentes tamaños de un poste a otro en un juego de tres postes.

La conocida Torre de Hanói fue inventada por el matemático francés Édouard Lucas y se vendió como juguete en 1883. Originalmente, llevaba el nombre de "Profesor Claus" del Colegio de "Li-Sou-Stian", pero pronto se descubrió que eran anagramas del "Profesor Lucas" del Colegio de "Saint Louis". El problema consiste en transferir la torre de ocho discos a cualquiera de las dos clavijas vacías en el menor número de movimientos, moviendo un disco cada vez y sin colocar un disco encima de otro más pequeño. Considere los tres postes A, B y C como se muestran en la figura 1.11.

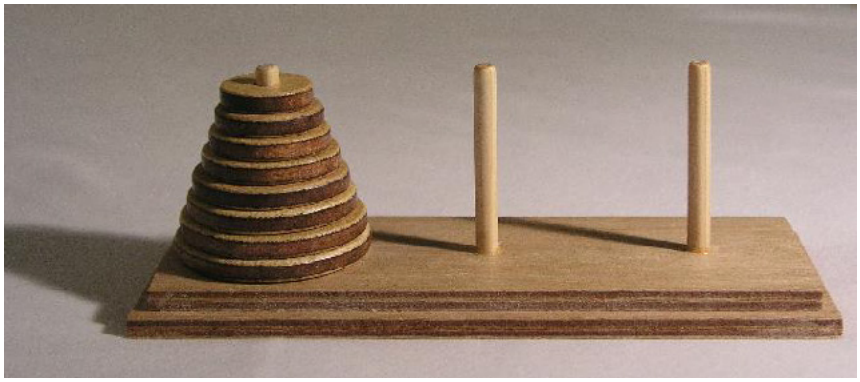


FIGURA 1.11. Las torres de Hanói[6]

Inicialmente, el poste A tiene cierta cantidad de discos de distintos tamaños, el más grande en la parte inferior y otros sucesivamente más pequeños encima. El problema se puede resolver sencillamente con el siguiente algoritmo: se imaginan los postes dispuestos en un triángulo; con un número de movimientos impar, se mueve el disco más pequeño hacia un poste en el

[6] "Tower of Hanoi", 2015. Disponible en: http://en.wikipedia.org/wiki/Tower_of_Hanoi

sentido de las manecillas del reloj; con un número de movimientos pares, se hace un único movimiento válido que no implique al disco más pequeño.

No es difícil demostrar que hay una solución, sin importar cuántos discos hay en la torre y que el número mínimo de movimientos necesarios está expresado por la fórmula $2^n - 1$, donde n es el número de discos. De este modo, se pueden transferir tres discos en siete movimientos, cuatro en 15, cinco en 31 y así, sucesivamente. Para los ocho discos se requieren 255 movimientos.

La descripción original del juguete decía que este era una versión simplificada de una mítica *Torre de Brahma* de un templo de la ciudad de Benarés, India.

La descripción dice que esta torre está formada por 64 discos de oro, que están en el proceso de ser transferidos por los sacerdotes del templo. Antes de que ellos completen su tarea, se decía que el templo se desmoronaría hecho polvo y el mundo se desvanecería al estampido de un trueno. Puede ponerse en duda la desaparición del mundo, pero no así, el desmoronamiento del templo. La fórmula $2^{64} - 1$ da por resultado un número de 20 dígitos.

$$18, 446, 744, 073, 709, 551, 615$$

Y, suponiendo que los sacerdotes trabajaran día y noche moviendo un disco por segundo, terminar el trabajo les llevaría varios miles de millones de años.

Una característica del 64 es que no es número primo, pero si se aumenta el número de discos por ejemplo a 89, 107 o 127, el número de movimientos necesarios para transferirlos es primo, en cada caso. Estos son ejemplos de los llamados números de Mersenne: primos que tienen la forma de $2^n - 1$. El propio Lucas fue el primer hombre que verificó, que $2^{127} - 1$ era primo. Este monstruoso número de 39 dígitos era el primo más grande que se conocía hasta 1952, cuando se utilizó una gran computadora electrónica para encontrar cinco números primos de Mersenne mayores que ese. "Actualmente (febrero de 2013), solo se conocen 48 números primos de Mersenne, siendo el mayor de ellos $M_{57.885.161} = 2^{57.885.161} - 1$, un número de más de diecisiete millones de cifras".[7]

Este problema se resuelve recursivamente de la siguiente manera: al mover un solo disco a la vez, se puede entender como un problema nuevo de $n-1$ discos en lugar del problema original de n discos. Así, en cada paso, al mover un nuevo disco, el problema original ya no es de tamaño n sino de tamaño $n-1$, $n-2$ y así, sucesivamente, esto se puede ver en forma esquemática en el siguiente pseudocódigo:

[7] "Número primo de Mersenne", 2014. Disponible en: http://es.wikipedia.org/wiki/N%C3%BAmero_primo_de_Mersenne

```
/* Solucion:
    1- Mover n-1 discos de A a B
    2- Mover 1 disco de A a C
    3- Mover n-1 discos de B a C
*/
void Hanoi( n, inicial, aux, final )
{
  if( n>0 )
  {
     Hanoi(n-1, inicial, final, aux );
     printf("Mover %d de %c a %c", n, inicial, final );
     Hanoi(n-1, aux, inicial, final );
  }
}
```

1.5. TIPOS DE RECURSIÓN

Los algoritmos recursivos pueden ser simples o múltiples, así como anidados o cruzados. A continuación, se dan las definiciones y algunos ejemplos.

- Recursividad simple: aquella en cuya definición solo aparece una llamada recursiva. Se puede transformar con facilidad en algoritmos iterativos.

- Recursividad múltiple**:** se da cuando hay más de una llamada a sí misma dentro del cuerpo de la función, resultando un algoritmo iterativo más complejo.

  El siguiente algoritmo genera la serie de Fibonacci a través de la recursividad múltiple:

```
int Fib( int n ) /* ej:
Fibonacci */
if(n<=1) return(1);
return(Fib(n-1) + Fib(n-2));
```

- Recursividad anidada: en algunos de los argumentos de la llamada recursiva hay una nueva llamada a sí misma. Un ejemplo clásico se encuentra en la función de Ackermann.[8]
  En teoría de la computación, la *función de Ackermann* es una función matemática recursiva encontrada en 1926 por Wilhelm Ackermann. Esta función tiene un

[8] "Función de Ackermann", 2014. Disponible en: http://es.wikipedia.org/wiki/Funci%C3%B3n_de_Ackermann

crecimiento extremadamente rápido de interés para la ciencia computacional teórica y la teoría de la computación. Hoy en día, hay una serie de funciones que son llamadas funciones Ackermann; todas ellas tienen una forma similar a la ley original, la función de Ackermann y, también, tienen un comportamiento de crecimiento similar. Esta función toma dos números naturales como argumentos y devuelve un único número natural. Como norma general se define así:

$$A(m,n) = \begin{cases} n+1, & \text{si } m = 0; \\ A(m-1,1), & \text{si } m > 0 \text{ y } n = 0; \\ A(m-1, A(m,n-1)), & \text{si } m > 0 \text{ y } n > 0 \end{cases}$$

El siguiente algoritmo ejecuta la función de Ackermann: siendo *n* y *m* números naturales, devuelve un único número natural.

```
int Ack( int n, int m )
/* ej: Ackerman */
{ if(n==0 ) return(m+1);
else if(m==0) return(Ack(n-1,1));
return(Ack(n-1, Ack(n,m-1)));
```

- Recursividad cruzada o indirecta: son algoritmos donde una función provoca una llamada a sí misma de forma indirecta, a través de otras funciones.

  El siguiente es un programa para saber si un número entero dado es par o impar:

  Ejemplo. Par o impar:

```
int par( int nump )
{
if(nump==0) return(1); return( impar(nump-1));
}
int impar( int numi )
{
if(numi==0) return(0); return( par(numi-1));
}
```

Finalmente, se presenta un ejemplo específico de un algoritmo recursivo que ordena una lista de números dividiendo la lista de entrada, recursivamente, en listas más pequeñas.

Ejemplo 1.6. Algoritmo MEZCLAS.

Descripción:

Entrada: la lista $L=\{a_i+a_{i+1}+\cdots+a_j\}$ de n números naturales.

PASO 0: si L tiene un solo elemento, la lista está ordenada. Terminar.

PASO 1: dividir la lista L en dos listas L_1 y L_2. Asignar $k \leftarrow [i+j/2]$ donde $[i+j/2]$ representa al mayor entero menor que $(i+j)/2$. Las listas L_1 y L_2 se construyen considerando *k* como el último elemento de la primera lista. Si la lista original *L* tiene 11 elementos, entonces L_1 tendrá 6 elementos y L_2 tendrá 5.

PASO 2: ordenar por separado las listas L_1 y L_2 utilizando el algoritmo MEZCLAS, cada vez que se llama a sí mismo, el algoritmo *k* disminuye.

PASO 3: mezclar las listas ordenadas en el paso 2. Hallar la lista MEZCLA (L_1, L_2).

Este algoritmo divide la lista de entrada, siempre que sea posible, en dos listas de aproximadamente igual tamaño, L_1 y L_2, y resuelve a continuación el mismo problema de partida por separado para las listas L_1 y L_2 y mezcla las soluciones parciales obtenidas.

Observación 1: este algoritmo se *llama a sí mismo*; es decir, en alguno de sus pasos se da la instrucción de aplicar el mismo algoritmo a un conjunto más pequeño contenido en el conjunto de entrada.

Observación 2: este algoritmo recursivo divide sucesivas veces el conjunto de entrada en partes iguales facilitando la solución del problema en dichas partes por ser más pequeñas; y mezclando, así, las soluciones parciales de manera sucesiva, se obtiene la solución general del problema.

A esta técnica algorítmica se le conoce como *divide y vencerás*.

Aplicación

Se aplica el algoritmo MEZCLAS a la lista de números {2, 5, 1, 9, 4, 6, 3, 8}, donde se observa que se divide la lista a la mitad sucesivamente hasta que en el paso 3, una vez ordenados los números, se inicia la mezcla de las listas.

Entrada: $L = \{2, 5, 1, 9, 4, 6, 3, 8\}$

PASO 1: $L_1 = \{2, 5, 1, 9\}, L_2 = \{4, 6, 3, 8\}$ k = [(1+8)/2] = [4.5] = 4

PASO 2: L_1 : PASO 1: $L_1 = \{2, 5\},\ L_2 = \{1, 9\}$

PASO 2: L_1 : PASO 1: $L_1 = \{2\},\ L_2 = \{5\}$

PASO 2: $L_1 = \{2\},\ L_2 = \{5\}$

PASO 3: $L = \{2, 5\}$

L_2 : PASO 1: $L_1 = \{1\},\ L_2 = \{9\}$

PASO 2: $L_1 = \{1\},\ L_2 = \{9\}$

PASO 3: $L = \{1,\ 9\}$

PASO 3: $L = \{1, 2, 5, 9\}$

L_2 : PASO 1: $L_1 = \{4, 6\},\ L_2 = \{3, 8\}$

PASO 2: L_1 : PASO 1: $L_1 = \{4\},\ L_2 = \{6\}$

PASO 2: $L_1 = \{4\},\ L_2 = \{6\}$

PASO 3: $L = \{4, 6\}$

L_2 : PASO 1: $L_1 = \{3\},\ L_2 = \{8\}$

PASO 2: $L_1 = \{3\},\ L_2 = \{8\}$

PASO 3: $L = \{3, 8\}$

PASO 3: $L = \{3, 4, 6, 8\}$

PASO 3: $L = \{1, 2, 3, 4, 5, 6, 8, 9\}$

Salida: la lista L obtenida en el paso 3 es la deseada.

1.6. CONCLUSIONES

Como podemos observar a lo largo de este capítulo, la recursividad es algo que se encuentra en la naturaleza, en el arte, en las matemáticas y en una amplia variedad de las ciencias, por lo que, tener un conocimiento básico de recursividad y cómo funciona es fundamental para la construcción de los modelos de PD.

Al final del capítulo, vimos algunos ejemplos de cómo se maneja algorítmicamente la recursividad, en el próximo capítulo se tratarán ejemplos con un cierto grado de sencillez para ir introduciendo los conceptos necesarios para modelar problemas que pueden resolverse a través de la PD.

1.7. NOTAS HISTÓRICAS

Zenón de Elea

Nació y murió en Elea (hoy Italia) (alrededor del año 490 a. C. - 425 a. C.). Muy poco se sabe de la vida de Zenón de Elea. Fue amigo y discípulo de Parménides. Parménides fundó la escuela filosófica eleática, su filosofía se basaba en el principio 'todo es uno'. Zenón estuvo muy influenciado por los argumentos de Parménides.

Zenón ha pasado a la historia por las paradojas. Las paradojas de Zenón se basan en las dificultades derivadas del análisis de las magnitudes continuas. Zenón también supone que si algo no tiene magnitud no puede existir.

La primera paradoja de Zenón es la *dicotomía*. En ella se niega el movimiento: No hay movimiento porque para que algo recorra un espacio, debe primero llegar a la mitad (1/2), después a los 3/4, después a los 7/8, después a los 15/16, después a los 31/32 y así indefinidamente. Según esta paradoja nunca alcanzaríamos el final.

Si razonamos de otra forma: para llegar al final debemos llegar a la mitad, pero para llegar a la mitad debemos llegar a la mitad de la mitad, pero antes debemos llegar a la mitad de la mitad de la mitad y, antes, a la mitad de la mitad de la mitad de la mitad y así, indefinidamente. Conclusión: Nunca comenzamos el movimiento.

Algo parecido ocurre con las sumas infinitas:
1/2 + 1/4 + 1/8 + 1/16 +... tiende a 1 pero nunca lo alcanza.

La paradoja más famosa es sin duda la de *Aquiles y la tortuga*:

Aquiles, héroe griego y una tortuga, participan en una carrera. La tortuga parte con ventaja. ¿Adelantará Aquiles a la tortuga?

Zenón argumenta así: en el momento inicial, Aquiles estará en la posición a_0 y la tortuga en la posición t_0. Cuando Aquiles llegue al punto t_0, la tortuga estará en el punto t_1 y cuando Aquiles llegue al punto t_1, la tortuga estará en el punto t_2. Aunque la distancia entre Aquiles y la tortuga disminuye continuamente, la tortuga siempre estará por delante.

Evidentemente, hay un error en el razonamiento, pero ¿dónde está?

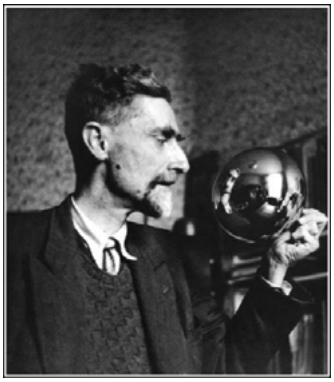
El error es suponer que se necesita un tiempo infinito para recorrer una distancia finita, dividida en un número infinito de trozos.

La paradoja de la *flecha* dice que una flecha en vuelo está realmente parada. Zenón parte de que un objeto que ocupa un espacio igual a su tamaño está en reposo. En cada instante, la flecha en vuelo ocupa un espacio exactamente igual a su longitud, luego está en reposo. Esta paradoja presenta la dificultad de calcular la velocidad instantánea cuando el espacio y el tiempo son cero (v = s/t. cuando s = 0 y t = 0, v = 0/0).

Las paradojas de Zenón influyeron negativamente en el desarrollo del concepto de infinitesimales, pero son los primeros antecedentes del razonamiento infinitesimal.[9]

Pieter Cornelis Mondriaan

(Amersfoort, 7 de marzo de 1872 - Nueva York, 1 de febrero de 1944), conocido como Piet Mondrian, fue un pintor vanguardista neerlandés, miembro de De Stijl y fundador del neoplasticismo, junto con Theo van Doesburg. Evolucionó desde el naturalismo y el simbolismo hasta la abstracción, de la cual es el principal representante inaugural junto a los rusos Wassily Kandinski y Kazimir Malévich.[10]



Maurits Cornelis Escher

(Leeuwarden, Países Bajos, 17 de junio de 1898 - Hilversum, Países Bajos, 27 de marzo de 1972), más conocido como M. C. Escher, fue un artista neerlandés conocido por sus grabados xilográficos y litográficos que tratan sobre figuras imposibles, teselados y mundos imaginarios. Su obra experimenta con diversos métodos de representar (en dibujos de 2 o 3 dimensiones) espacios paradójicos que desafían a los modos habituales de representación.[11]

[9] "Zenón de Elea y Meliso de Samos", 2011. Disponible en: http://www.buenastareas.com/ensayos/Zen%C3%B3n-De-Elea-y-Meliso-De/3219782.html
[10] "Piet Mondrian", 2015. Disponible en: http://es.wikipedia.org/wiki/Piet_Mondrian
[11] "M. C. Escher", 2015. Disponible en: http://es.wikipedia.org/wiki/M._C._Escher

Édouard Lucas

François Édouard Anatole Lucas (Amiens, 4 de abril de 1842 - París, 3 de octubre de 1891) fue un reconocido matemático francés. Trabajó en el observatorio de París y más tarde fue profesor de matemáticas en la capital del Sena. Se le conoce sobre todo por sus trabajos sobre la serie de Fibonacci y por el test de primalidad que lleva su nombre, pero también fue el inventor de algunos juegos recreativos matemáticos muy conocidos como el de las *Torres de Hanói*.[12]

Marin Mersenne

Marin Mersennus o le Père Mersenne (Oizé, 8 de septiembre de 1588 – París, 1 de septiembre de 1648) fue un filósofo francés del siglo XVII que estudió diversos campos de la teología, matemáticas y la teoría musical.

Nacido en una familia de campesinos cerca de Oizé (hoy Sarthe), en la provincia francesa de Maine, fue educado en Le Mans y en la universidad jesuita de La Flèche, donde conoció y frecuentó la amistad de René Descartes. Aunque en ocasiones se afirma que fue jesuita, lo cierto es que nunca llegó a ingresar en la Sociedad de Jesús. El 17 de julio de 1611 se hizo miembro de los Mínimos, dedicándose al estudio de la teología y el hebreo. Después de este período recibió la orden sacerdotal en París en 1613.

Tras su consagración estuvo un tiempo enseñando filosofía y teología en Nevers, pero en 1620 regresó a París. Allí entró en el convento de L'Annonciade y, en compañía de personajes como Descartes, Étienne Pascal, Gilles de Roberval y Nicholas-Claude Fabri de Peiresc, estudió matemáticas y música. Tuvo una nutrida correspondencia con diversos eruditos de Francia, Italia, Inglaterra y Holanda, tales como Pierre de Fermat, Galileo Galilei, Giovanni Doni y Christiaan Huygens. Desde 1620 hasta 1623 se dedicó exclusivamente a escribir en materia de filosofía y teología, y en 1623 publicó *Quaestiones celeberrimae in Genesim*, a la que rápidamente siguieron otras obras como *L'Impieté des déistes* (1624) en la que refuta las ideas de Giordano Bruno y *La Vérité des sciences* (1624).

Murió después de una serie de complicaciones que se derivaron de una intervención quirúrgica. En su testamento vital, pidió que su cuerpo fuera sometido a autopsia como último servicio al interés de la ciencia.[13]

[12] "Édouard Lucas", 2014. Disponible en: http://es.wikipedia.org/wiki/%C3%89douard_Lucas
[13] "Marin Mersenne", 2015. Disponible en: http://es.wikipedia.org/wiki/Marin_Mersenne

Fibonacci

Leonardo de Pisa, Leonardo Pisano o Leonardo Bigollo

(c. 1170 -1250), también llamado Fibonacci, fue un matemático italiano, famoso por haber difundido en Europa el sistema de numeración indo-arábigo actualmente utilizado, el que emplea notación posicional (de base 10, o decimal) y un dígito de valor nulo: el cero; y por idear la sucesión de Fibonacci.

El apodo de Guglielmo (Guillermo), padre de Leonardo, era *Bonacci* (simple o bien intencionado). Leonardo recibió póstumamente el apodo de *Fibonacci* (por *filius Bonacci*, hijo de Bonacci). Guglielmo dirigía un puesto de comercio en Bugía (según algunas versiones era el cónsul de Pisa), en el norte de África (hoy Bejaia, Argelia), y de niño Leonardo viajó allí para ayudarlo. Allí aprendió el sistema de numeración árabe.

Consciente de la superioridad de los numerales árabes, Fibonacci viajó a través de los países del Mediterráneo para estudiar con los matemáticos árabes más destacados de ese tiempo, regresando cerca de 1200. En 1202, a los 32 años de edad, publicó lo que había aprendido en el *Liber abaci* (abaci en el sentido de aritmética y no del ábaco instrumento). Este libro mostró la importancia del nuevo sistema de numeración aplicándolo a la contabilidad comercial, conversión de pesos y medidas, cálculo, intereses, cambio de moneda, y otras numerosas aplicaciones. En estas páginas describe el cero, la notación posicional, la descomposición en factores primos, los criterios de divisibilidad. El libro fue recibido con entusiasmo en la Europa ilustrada, y tuvo un impacto profundo en el pensamiento matemático europeo.

Leonardo fue huésped del Emperador Federico II, que se interesaba en las matemáticas y la ciencia en general. En 1240, la República de Pisa lo honra concediéndole un salario permanente (bajo su nombre alternativo de Leonardo Bigollo).[14]

Wilhelm Ackermann

(29 de marzo 1896 - 24 de diciembre 1962) fue un matemático alemán. Es conocido, sobre todo, por la función de Ackermann nombrada en su honor, un ejemplo importante en la teoría de la computación.

Ackermann nació el 29 de marzo de 1896 en Schönebecke (que pertenecía al distrito de Altena y ahora forma parte del municipio de Herscheid) (Alemania). Se doctoró en 1925 con su

[14] "Leonardo de Pisa", 2015. Disponible en: http://es.wikipedia.org/wiki/Leonardo_de_Pisa

tesis *Begründung des "tertium non datur" mittels der Hilbertschen Theorie der Widerspruchsfreiheit*, que fue una prueba de consistencia de la aritmética sin inducción. Desde 1928 hasta 1948 fue profesor en el instituto Arnoldinum en Burgsteinfurt, y desde entonces hasta 1961 enseñó en Lüdenscheid. Además, fue miembro de la Akademie der Wissenschaften (*Academia de las Ciencias*) en Göttingen, así como profesor honorífico de la Universidad de Münster en Westfalia.

Escribió *Grundzüge der Theoretischen Logik* (*Fundamentos de la lógica teórica*) junto con David Hilbert, enfrentándose al Entscheidungsproblem (problema de decisión) también construyó las pruebas de la consistencia para la teoría determinada (1937), la aritmética completa (1940), la lógica tipo-libre (1952) y una nueva axiomatización de la teoría determinada (1956). Escribió el libro *los casos solubles del problema de la decisión* (Holanda del norte, 1954). El fue un matemático destacado de ese siglo. Wilhelm Ackermann murió en Lüdenscheid (Alemania) el 24 de diciembre de 1962.[15]

1.8. EJERCICIOS PROPUESTOS

1) Busque al menos tres ejemplos de recursividad en la naturaleza, el arte, las matemáticas y la ingeniería.

2) Establezca en sus propias palabras, ¿cuál es la relación entre recursividad y fractales?

3) Desarrolle el siguiente algoritmo con un ejemplo de 10 elementos.

Algoritmo máximo binario

Entrada: la lista $L = \{a_1,...,a_i, a_{i+1},...,a_j\}$ con n elementos, $i = 1$
PASO 0: si $i = j$, entonces, $m = a_1$ e ir al paso 5
PASO 1: $k = [i+j/2]$ (donde [ ] indica parte entera de $i+j/2$)
PASO 2: $L_1 = \{ a_i,...,a_k\}$, $L_2 = \{a_{k+1},...,a_j\}$
PASO 3: m_1 = MÁXIMO BINARIO (L_1), m_2 = MÁXIMO BINARIO (L_2)
PASO 4: si $m_1 > m_2$, entonces, $m = m_1$. En caso contrario $m = m_2$
PASO 5: FIN
Salida: el máximo es m

[15] "Wilhelm Ackermann", 2013. Disponible en: http://es.wikipedia.org/wiki/Wilhelm_Ackermann

CAPÍTULO 2

CONCEPTOS BÁSICOS DE PROGRAMACIÓN DINÁMICA

2.1. INTRODUCCIÓN

En este capítulo se presentan los conceptos básicos de la PD (Programación Dinámica), cómo se elaboran modelos y cómo se resuelven paso a paso. El objetivo del capítulo es que el alumno pueda modelar, de manera sencilla, los problemas cuya solución pueda apoyarse en forma visual y, también, se hará una comparación con los modelos de otras áreas de la investigación de operaciones, siempre y cuando, el problema se ajuste para que se pueda modelar de distintas maneras. Al final del capítulo viene un ejemplo desarrollado donde se plantean tres modelos: uno de PD, otro de teoría de redes y, finalmente, uno de PLE (Programación Lineal Entera) y se compara la complejidad computacional de los algoritmos utilizados en la solución del problema.

La PD fue fundada por Richard Bellman. El primer tratamiento sistemático del tema lo dio en su libro *Dynamic Programming* en 1957; aunque, mucho de su trabajo ya se había publicado con anterioridad en numerosos reportes de la RAND Corporation (Corporación RAND), así como, en otras publicaciones de Bellman. Posteriormente, se publicaron más libros de PD y aplicaciones a la teoría del control por parte de Bellman y sus colaboradores. También, hay aplicaciones de la PD en procesos de Markov, teoría de inventarios, ingeniería química, cálculo de variaciones y economía. Varias de ellas las discutiremos, posteriormente.

La PD es un método particular para optimizar. Usamos la palabra método, porque la PD no es un algoritmo en particular, en el sentido euclidiano del término algoritmo o del método simplex para la PLE. La PD es un método para resolver cierto tipo de problemas de optimización, algunos de los cuales pueden, en principio, resolverse por otros procedimientos.

La PD es una manera de ver un problema que puede contener un gran número de variables de decisión, interrelacionadas, de tal manera, que el problema se puede ver como una sucesión de problemas, cada uno con una o unas cuantas variables. Lo que se busca idealmente es sustituir un problema de *n* variables con problemas de una variable, siempre que esto sea posible, lo que requiere mucho menor esfuerzo computacional. Al resolver *n* problemas más pequeños, el esfuerzo computacional es proporcional a *n* que es el número de problemas de una variable, si cada uno de ellos consta de una sola variable.

El principio o punto de vista que nos permite hacer esta transformación se conoce como el *principio de optimalidad* establecido por Bellman que dice:

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

Un enunciado más simple de este principio dice: "Cada política óptima consiste solamente de sub-políticas óptimas".

A propósito, veamos un ejemplo que muestra cómo funciona este principio. Un inversionista tiene una suma fija de dinero *D* que puede invertir en cinco oportunidades de inversión diferentes (bonos, acciones, bienes raíces, etc.). El problema es que él quiere invertir su dinero en el presente; suponiendo que él tiene alguna preferencia con base en la tasa de retorno que espera de cada inversión. Ya que cada inversión tiene ciertas características, como tasas de interés, ganancias diferidas, cantidades máximas requeridas, él quiere saber cuánto del total de *D* debe invertir en cada una de las cinco opciones, con el objeto de maximizar el retorno total.

Un método consiste en resolver el problema generando todas las combinaciones posibles de las cinco inversiones y ver, cuál de ellas proporciona el mayor retorno. Este método se llama de *enumeración total* y generalmente no es práctico, aún con una computadora en casos donde el número de variables es más realista. Además, si suponemos que la cantidad a invertir en cada una de las oportunidades es fija, la decisión a tomar es invertir o no.

Aún, en el caso simplificado, solo tenemos una manera de invertir en todas las oportunidades, cinco maneras de invertir en cuatro de ellas, diez maneras de invertir en tres, diez maneras de invertir en dos y cinco para invertir en una, es decir, el número total de maneras de invertir en las cinco oportunidades (para una cantidad fija de inversión en cada una) es:

$$\binom{5}{5}+\binom{5}{4}+\binom{5}{3}+\binom{5}{2}+\binom{5}{1}=31$$

Para 20 inversiones tendríamos que evaluar 1 048 575 combinaciones. El número crece rápidamente con el número de inversiones, por lo que, la enumeración total es impráctica.

Lo que hacemos en PD es examinar un número pequeño de esas combinaciones, con la garantía de que en este conjunto se encuentra la solución óptima. En este caso, aunque las inversiones se tienen que hacer en un mismo tiempo, supondremos que se hacen con un cierto orden. Lo que significa que invertimos una cierta cantidad I_1 en la inversión 1, una cantidad I_2 en la inversión 2, etc. La única restricción que tenemos es la suma de $I_1 + I_2 + I_3 + I_4 + I_5 = D$ y que las inversiones son cantidades no negativas. Aplicando el principio de optimalidad que dice: "No importa cuánto se gaste del presupuesto D en las primeras k inversiones, el dinero restante se debe distribuir entre las $5 - k$ inversiones remanentes".

Ya que hemos numerado las inversiones como se mencionó con anterioridad, si ya hemos gastado una cierta cantidad de dinero en la primera de las cuatro inversiones, lo que quede se invertirá en la quinta inversión, hasta aquí, no hemos tomado alguna decisión. Ahora, consideremos lo que se debería invertir en I_4, aquí, sí se toma una decisión que involucra cuánto invertir en la opción 4, de tal manera que, se tenga un ahorro para la inversión 5. Esta es una decisión que solo involucra una variable. Lo mismo sucede con la inversión I_3, se toma una decisión de una variable, cuánto invertir en la opción 3, de tal forma que, quede dinero para invertir en la 4 y en la 5, así, se continúa para las opciones 1 y 2, de esta manera, se tienen que hacer cinco decisiones simples de una sola variable, en lugar de una sola decisión más compleja de cinco variables.

Lo que estamos diciendo con todo esto es lo siguiente: cuando decidamos cuánto invertir en I_3, no importa cuánto hayamos decidido invertir en I_1 y en I_2, solamente, debe ser óptima la cantidad de inversión para I_3 con respecto al capital que quede remanente. Ya que, aún no decidimos cuánto se invertirá en I_4 y en I_5, la inversión óptima y el retorno desde I_3 se debe determinar para todas las cantidades factibles remanentes para invertir.

2.2. ETAPAS Y ESTADOS

La PD es una técnica matemática que se utiliza para la solución de problemas matemáticos seleccionados, en los cuales se toma una serie de decisiones en forma secuencial.

La PD proporciona un procedimiento sistemático para encontrar la combinación de decisiones que maximicen la efectividad total al descomponer el problema en *etapas*, las cuales pueden

ser completadas por una o más formas, *estados*, enlazando cada etapa a través de cálculos recursivos.

- Etapa: es la parte del problema que posee un conjunto de alternativas mutuamente excluyentes, de las cuales se seleccionará la mejor alternativa.

- Estado: es el que refleja la condición o estado de las restricciones que enlazan las etapas. Representa la *liga* entre las etapas, de tal manera que, cuando cada etapa se optimiza por separado, la decisión resultante es automáticamente factible para el problema completo. La definición de estado varía dependiendo de la situación que se va a modelar.

Aunque, se revisen en cada aplicación estos conceptos es útil considerar las siguientes preguntas:

1) ¿Por medio de qué relaciones se ligan las etapas?

2) ¿Qué información es necesaria para tomar decisiones factibles en la etapa actual, sin reexaminar las decisiones hechas en las etapas previas?

Para tener una manera correcta de interpretar y resolver el problema es fundamental que se cuente con una definición clara y entendible de lo que son los estados.

- Esquema de una etapa

  q_i = variable de estado en la etapa i
  X_{ij} = uno de los valores que puede adoptar la variable de decisión "X_i" en la etapa i
  X_i^* = decisión óptima de la etapa i

2.3. FORMULACIÓN Y SOLUCIÓN DE PROBLEMAS

La PD no cuenta con una formulación matemática estándar, sino que se trata de un enfoque de tipo general para la solución de problemas y las ecuaciones específicas que se usan se desarrollan para que representen cada situación individual.

Comúnmente, la PD resuelve el problema por etapas y en cada etapa interviene, exactamente, una variable de optimización (u optimizadora).

La teoría unificadora fundamental de la PD es el *principio de optimalidad* que nos indica, en forma básica, cómo se puede resolver un problema adecuadamente descompuesto en etapas, utilizando cálculos recursivos.

"Una política óptima tiene la propiedad de que, independientemente de las decisiones tomadas para llegar a un estado particular, en una etapa particular, las decisiones restantes deben constituir una política óptima para abandonar ese estado", en otras palabras, la trayectoria óptima, desde el punto de partida al punto final, tiene la propiedad de que para cualquier punto intermedio, la trayectoria debe ser aquella óptima desde el punto de partida hasta aquel punto intermedio.

Para resolver problemas con PD se necesita:

- Un grado de creatividad.
- Un buen conocimiento de la estructura general de los problemas de PD para reconocer, cuándo un problema se puede resolver por medio de estos procedimientos y cómo esto se puede llevar a cabo.

Características de los problemas de PD:

- El problema se puede dividir en etapas que requieren una política de decisión en cada una.
- Cada etapa tiene cierto número de estados asociados a ella.
- El efecto de la política de decisión, en cada etapa, es transformar el estado actual en un estado asociado con la siguiente etapa.
- El procedimiento de solución está diseñado para encontrar una política óptima para el problema completo.
- Dado un estado actual, una política óptima para las etapas restantes es independiente de la política adoptada en las etapas anteriores (principio de optimalidad).
- El procedimiento de solución se inicia al encontrar la política óptima para la última etapa.
- Se dispone de una relación recursiva que identifica la política óptima para la etapa n, dada la política óptima para la etapa $(n+1)$.

Recursividad:

En los problemas de PD existen dos formas de plantear la fórmula de recursividad:

- Recursividad de retroceso: el problema se resuelve partiendo de la última etapa hacia la primera.
- Recursividad de avance: el problema se resuelve partiendo de la primera etapa hacia la última.

Las formulaciones de avance y retroceso son en realidad equivalentes en términos de cálculo. Sin embargo, hay situaciones donde habría alguna diferencia en la eficiencia del cálculo, según la formulación que se utilice. Esto sucede en particular en problemas donde interviene la toma de decisiones conforme transcurre el tiempo. En este caso, las etapas se designan con base en el estricto orden cronológico de los periodos que ellas representan y la eficiencia de los cálculos dependerá, si se utiliza la formulación de avance o retroceso.

2.4. EJEMPLOS RESUELTOS

*Ejemplo 2.1***.** Ruta más corta (Dreyfus y Law, 1977).

La PD es un procedimiento de optimización que es aplicable particularmente a problemas que requieren una sucesión de decisiones interrelacionadas. Cada decisión transforma la situación actual en una situación nueva. Una sucesión de decisiones, que a su vez producen una sucesión de situaciones, buscan maximizar (o minimizar) alguna medida de valor. El valor de la sucesión de decisiones generalmente es igual a la suma de los valores de las decisiones individuales y situaciones en la sucesión. De esta forma, se le puede plantear al problema, el encontrar la mejor trayectoria de un lugar a otro.

Problema de trayectoria simple:

Para iniciar, suponga que vive en una ciudad cuyas calles se encuentran como en la figura siguiente.

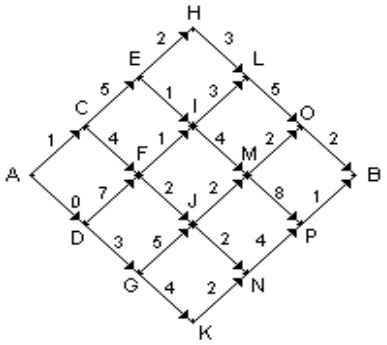


FIGURA 2.1. Red que representa las calles con distancias

Todas las calles son de un solo sentido y los números representan el esfuerzo (que usualmente es tiempo, pero algunas veces costo o distancia) requerido para atravesar cada bloque o cuadra individual. Usted vive en *A* y quiere llegar a *B* con el mínimo esfuerzo total. Se puede resolver este problema enumerando todas las trayectorias posibles de *A* a *B*. Sumando los esfuerzos bloque por bloque para cada trayectoria y positivamente escogiendo la menor suma.

Sin embargo, hay 20 trayectorias posibles de *A* a *B,* ¿por qué? Si enumeramos las calles de la siguiente manera:

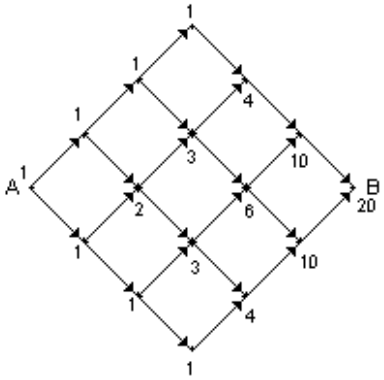


FIGURA 2.2. Enumeración de calles

Se pone un 1 en *A,* pues para llegar ahí solo hay una posibilidad (de hecho se va a salir de ahí). Después, se pone un 1 en cada esquina inmediata, porque solo hay una forma de ir a ellas desde *A*; 2 en la siguiente esquina, pues hay dos formas de llegar a ella y, así, en cada esquina se pone

la suma de los dos números más cercanos situados sobre él o los caminos que conduzcan hasta ella. Si cortamos a la mitad esta figura y la rotamos observamos:

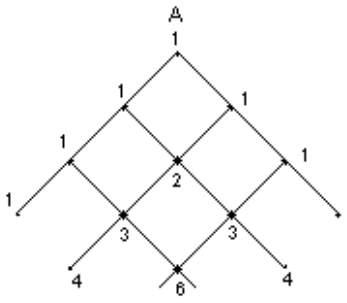


FIGURA 2.3. La red cortada y rotada

Los números de este triángulo son los números del triángulo de Pascal; entonces, el algoritmo para contar el número de trayectorias mínimas que bajan desde la cúspide no es otro que, el método usual para construir el triángulo de Pascal.

Asimismo, para una red de *nxm* el número de trayectorias posibles de *A* a *B* es igual al número de combinaciones de $n + m$ elementos tomados de n en n.

$$\frac{n+m}{n}$$

si $n = m$ se tiene:

$$\frac{(2n)!}{n!n!} = \frac{(2(3))!}{3!\ 3!} = \frac{6!}{3!\ 3!} = \frac{6\times5\times4}{3\times2} = \frac{120}{6} = 20$$

si se toma una red de 30 × 30 son 2 000 años.

Entonces, se tienen 20 caminos de *A* a *B* y 5 sumas que producen la suma de 6 números a lo largo de la trayectoria, así, 100 sumas podrían producir las 20 sumas de las trayectorias para ser comparadas. Ya que, una comparación produce el menor de 2 números, una comparación adicional (de ese número con un tercero) produce el menor de 3 y, así, sucesivamente. Las enumeraciones completan esta solución del problema.

Sin embargo, se puede resolver este problema usando PD.

La solución con PD (Programación Dinámica):

Para desarrollar el problema en términos de PD, razonamos de la siguiente forma: "No sé si voy a avanzar diagonalmente hacia arriba o diagonalmente hacia abajo desde *A*, pero si sólo conozco 2 números adicionales -es decir el esfuerzo total requerido para ir de *C* a *B* por la mejor trayectoria (el mínimo esfuerzo) y el esfuerzo requerido de ir de *D* a *B* por la mejor trayectoria- puedo entonces hacer la mejor selección desde *A*".

El mínimo esfuerzo se denota desde *C* a *B* por S_C y el mínimo esfuerzo desde *D* a *B* por S_D, entonces, se puede aumentar el esfuerzo de ir de *A* a *C* a S_C, obteniendo, así, el esfuerzo requerido de la mejor trayectoria de ir de *A* a *B* usando *C*. Lo mismo se hace con S_D y finalmente se comparan estas dos sumas para encontrar el mínimo esfuerzo y la mejor decisión.

Por supuesto, todo esto presupone el conocimiento de los números S_C y S_D. Sin embargo, una de las ideas centrales de la PD se ha establecido, ya que, solo el esfuerzo a través de las mejores trayectorias desde *C* y desde *D* hacia *B* son relevantes para los cálculos y el esfuerzo a lo largo de las otras 9 trayectorias respectivas desde *C* y *D* a *B* nunca necesitan calcularse. Esta observación a menudo se conoce como el *principio de optimalidad* y se establece como sigue:

Principio de optimalidad

La mejor trayectoria de *A* a *B* tiene la propiedad de que cualquiera que sea la decisión inicial en *A*, la trayectoria remanente a *B*, comenzando desde el siguiente punto después de *A*, debe ser la mejor trayectoria desde ese punto a *B*.

Ya sean S_C y S_D como ya se había mencionado, podemos citar el principio de optimalidad para justificar la fórmula:

$$S_A = \text{mín}\begin{cases} 1 + S_C \\ 0 + S_D \end{cases}$$

Donde S_A es el mínimo esfuerzo de ir de *A* a *B* y el símbolo "$\text{mín}[\ ]$" significa "el menor entre *x* y *y*".

La segunda idea importante es: mientras los dos números S_C y S_D son desconocidos para nosotros, inicialmente podemos calcular S_C, si conocemos S_E y S_F (los esfuerzos mínimos desde *E* y *F* para llegar a *B* respectivamente) y remitiéndonos al principio de optimalidad escribimos:

$$S_C = \text{mín}\begin{cases} 5 + S_E \\ 4 + S_F \end{cases}$$

y, también:

$$S_D = \text{mín}\begin{cases} 7 + S_E \\ 3 + S_G \end{cases}$$

Así, continuamos con S_E, S_F y S_G, pero, si no los conocemos los podemos calcular si S_H, S_I, S_J y S_K se conocen y continuamos con este razonamiento hasta llegar a O y P y trabajando hacia atrás desde O y P hasta A, tenemos:

$$S_E = \text{mín}\begin{cases} 2 + S_H = 2 + 10 \\ 1 + S_I = 1 + 8 \end{cases} = 9$$

$$S_F = \text{mín}\begin{cases} 1 + S_I = 1 + 8 \\ 2 + S_J = 2 + 6 \end{cases} = 8$$

$$S_G = \text{mín}\begin{cases} 5 + S_J = 5 + 6 \\ 4 + S_K = 4 + 7 \end{cases} = 11$$

$$S_C = \text{mín}\begin{cases} 5 + S_E = 5 + 9 \\ 4 + S_F = 4 + 8 \end{cases} = 12$$

$$S_D = \text{mín}\begin{cases} 7 + S_F = 7 + 8 \\ 3 + S_G = 3 + 11 \end{cases} = 14$$

$$S_A = \text{mín}\begin{cases} 1 + S_C = 1 + 12 \\ 0 + S_D = 0 + 14 \end{cases} = 13$$

$$S_L = 5 + S_O = 7$$

$$S_H = 3 + S_L = 10$$

$$S_M = \text{mín}\begin{cases} 2 + S_O = 2 + 2 \\ 8 + S_P = 8 + 1 \end{cases} = 4$$

$$S_N = 4 + S_P = 5$$

$$S_I = \text{mín}\begin{cases} 3 + S_L = 3 + 7 \\ 4 + S_M = 4 + 4 \end{cases} = 8$$

$$S_J = \text{mín}\begin{cases} 2 + S_M = 2 + 4 \\ 2 + S_N = 2 + 5 \end{cases} = 6$$

$$S_K = 2 + S_N = 2 + 5 = 7$$

Como la anterior trayectoria de $S_A = 13$, desarrollamos 1 suma en cada punto H, L, O, K, N y P, (que son 6 puntos) donde solo fue posible una decisión y desarrollamos dos sumas y una comparación por cada uno de los nueve puntos remanentes donde las dos decisiones iniciales fueron posibles. Esto da 24 sumas y nueve comparaciones, lo cual contrasta con 100 sumas y 19 comparaciones usando enumeración exhaustiva (o fuerza bruta).

Una vez que conocemos el esfuerzo total, queremos saber cuál es la trayectoria, esta se obtiene fácilmente fijándonos en las dos primeras decisiones y buscando el mínimo en nuestro cálculo previo.

Si denotamos x un nodo inicial cualquiera y P_x el nodo posterior al nodo x en la trayectoria óptima desde x a B, entonces, se tiene:

$$P_M = O, \text{ ya que, } 2 + S_O \text{ fue menor que } 8 + S_P$$
$$P_I = M, \text{ ya que, } 4 + S_M < 3 + S_L, \text{ etc.}$$

Así, se tiene que:

$P_O = B$ $P_P = B$

$P_L = 0$ $P_M = 0$ $P_N = P$

$P_H = L$ $P_I = M$ $P_J = M$ $P_K = N$

$P_R = I$ $P_F = J$ $P_G = J \; o \; K$

$P_C = F$ $P_D = G$

$P_A = C$

La mejor trayectoria es:

$$A \rightarrow C \rightarrow F \rightarrow J \rightarrow M \rightarrow O \rightarrow B$$

Sumando las distancias a lo largo de la trayectoria se tiene:

$$1 + 4 + 2 + 2 + 2 + 2 = 13$$

que es igual a S_A.

Terminología:

- La regla que asigna los valores a varios subproblemas es la función de valor óptimo (en este caso *S*).
- El subíndice de *S* -por ejemplo S_A- es el argumento de la función *S* y cada argumento se refiere a un subproblema en particular (por la definición de *S*, en este caso, *A* significa la mejor trayectoria de *A* a *B* que es deseada).
- La regla que asocia la primera mejor decisión con cada subproblema -la función *P*- es la función de política óptima.
- El principio de optimalidad produce una fórmula o un conjunto de fórmulas relativas a varios valores de *S*. Esta fórmula es la relación de recurrencia. Finalmente, el valor de la función de valor óptimo *S* para ciertos argumentos se supone obvia por el tipo de problema y la definición de *S*, que no requieren cálculos. Estos valores obvios se conocen como condiciones de frontera a *S*.

En este caso, sea:

S_x = trayectoria más corta de x a B
X = A, C, D,
S_B = 0 (condición de frontera)

Ecuaciones recursivas:
C_{xy} = longitud del nodo x al y
C_{xz} = longitud del nodo x al z

Además, se tiene la longitud mínima de todo nodo al nodo B.

Ejemplo 2.2. El problema de la diligencia.

Un caza fortunas de Missouri decidió irse al oeste para unirse a la fiebre del oro en California. Tenía que hacer el viaje en diligencia a través de territorios sin ley donde existían serios peligros de ser atacados por merodeadores. Aún, cuando su punto de partida y destino eran fijos, tenía muchas opciones en cuanto a qué estados debía elegir como puntos intermedios. Se desea estimar la ruta más segura. Como el costo de la póliza para cualquier jornada de la diligencia está basado en una evaluación de seguridad del recorrido, la ruta más segura debe ser aquella que tenga el costo total más barato.

¿Cuál es la ruta que minimiza el costo total de la póliza?

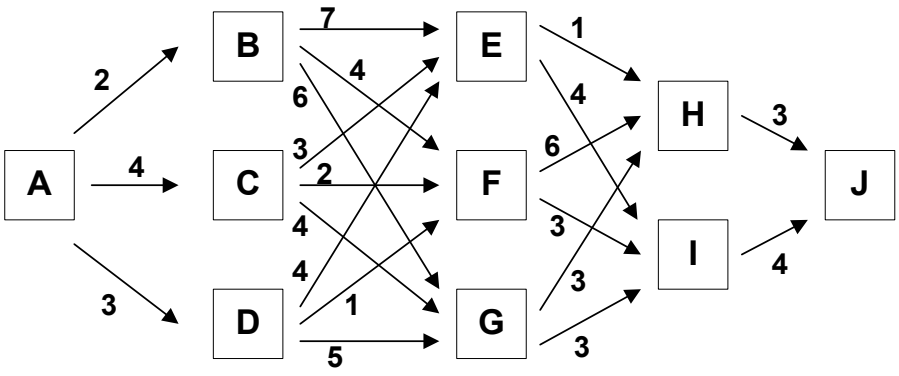


FIGURA 2.4. Sistema de caminos y los costos del problema de la diligencia

Costos de transición

	B	C	D
A	2	4	3

	E	F	G
B	7	4	6
C	3	2	4
D	4	1	5

	H	I
E	1	4
F	6	3
G	3	3

	J
H	3
I	4

Solución:

Como se vio en el ejemplo 2.1, el procedimiento de este problema de ruta más corta es semejante en:

- Los cálculos se realizan en etapas dividiendo el problema en subproblemas.
- Después, se considera por separado cada subproblema con el fin de reducir el número de operaciones de cálculo.
- Se comienza con una pequeña porción del problema original y se encuentra la solución óptima.
- Luego, se agranda el problema gradualmente y se encuentra la solución óptima actual a partir de la que le precede, hasta resolver el problema original completo.
- En cada problema aumentado, se puede encontrar la solución óptima tomando en cuenta los resultados obtenidos en la interacción anterior.

Para este caso, se empleará el desarrollo del problema con un recorrido hacia atrás. Cuando el caza fortunas tiene una sola etapa por recorrer ($n = 4$) su ruta, de ahí en adelante, está perfectamente determinada por su estado actual (ya sea H o I) y su destino final $x_4 = J$, de manera que, la ruta para esta última jornada en diligencias es $s \rightarrow J$.

La solución al problema es:

$$f^*_4(H) = 3$$
$$f^*_4(I) = 4$$

Con un costo de $C_{F,H} = 6$ o $C_{F,I} = 3$. Si se elige el estado H, el costo adicional mínimo al llegar ahí es 3, por tanto, el costo de decisión es 6 + 3 = 9, de igual forma, si se elige el estado I, el costo total es 3 + 4 = 7 que es menor, por lo tanto, se escogerá el estado I.

Cuando se tienen dos etapas por recorrer ($n = 3$), se analiza de la siguiente manera: suponga que se encuentra en el estado *F*, entonces, como se ve en la figura 2.4 se debe ir al estado *H* o al estado *I*.

Con un costo de $C_{F,\,H} = 6$ o $C_{F,\,I} = 3$, si se elige el estado *H*, el costo adicional mínimo al llegar ahí es 3, por tanto, el costo de decisión es 6 + 3 = 9, de igual manera, si se elige el estado *I*, el costo total es 3 + 4 = 7 que es menor, por lo tanto, se escogerá el estado *I*.

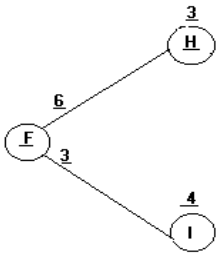


FIGURA 2.5. Estado *F* en la red

Se trabaja de manera similar con los otros dos estados posibles $s = E$ y $s = G$, cuando quedan dos jornadas por viajar, los resultados son:

$$f^*_3\,(E) = 4$$
$$f^*_3\,(F) = 7$$
$$f^*_3\,(G) = 6$$

La solución para el problema de tres etapas ($n = 2$) se obtiene en forma parecida. Por ejemplo, supóngase que el agente se encuentra en el estado *C*, como se muestra en el diagrama. Ahora, deberá ir al estado *E, F* o *G* con un costo inmediato de $C_{C,E} = 3$ o $C_{C,F} = 2$ o $C_{C,G} = 4$, respectivamente.

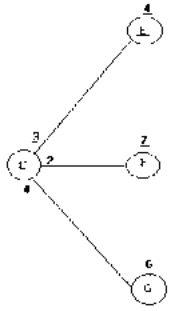


FIGURA 2.6. Estado *C* en la red

Si se pasa al problema de cuatro etapas ($n = 1$), los cálculos son parecidos a los que se acaban de mostrar para el problema de tres etapas ($n = 2$), excepto que, ahora, hay solo un inicio posible, $s = A$, como se muestra en el diagrama.

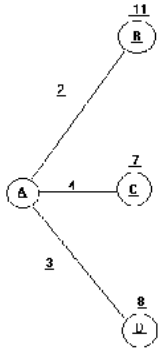


FIGURA 2.7. Estado A en la red

Los resultados se resumen de la siguiente manera:

$$
\begin{array}{ll}
x_1 = B & f_1(A,B) = c_{A,B} + f^*_2(B) = 2 + 11 = 13 \\
x_1 = C & f_1(A,C) = c_{A,C} + f^*_2(C) = 4 + 7 = 11 \\
x_1 = D & f_1(A,D) = c_{A,D} + f^*_2(D) = 3 + 8 = 11
\end{array}
$$

Como el mínimo costo es 11, por tanto, los caminos pueden ser C o D. En este punto, se puede identificar la solución óptima. Los resultados indican los caminos óptimos a seguir:

$$A \rightarrow C \rightarrow E \rightarrow H \rightarrow J \text{ o } A \rightarrow D \rightarrow F \rightarrow I \rightarrow J$$

Las dos tienen un costo total de 11

Ejemplo 2.3. Ruta más corta en una red de carreteras.

Bien, suponga que se desea ir de A hasta J en la carretera mostrada en la red de la figura 2.8.

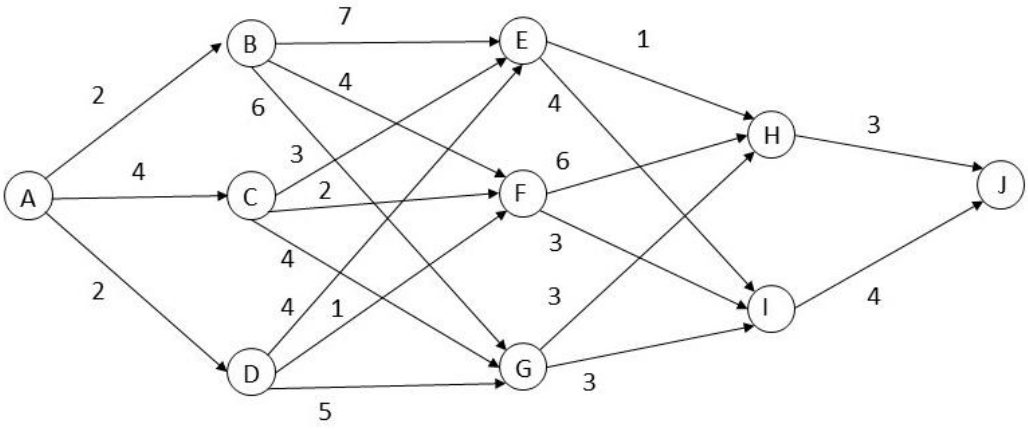


FIGURA 2.8. Red de carreteras

Los números en los arcos representan las distancias. Debido a la estructura especial del problema, este se puede dividir en etapas. La etapa 1 contiene el nodo *A*, la etapa 2 los nodos *B, C* y *D*, la etapa 3 contiene los nodos *E, F* y *G*, la etapa 4 contiene *H* e *I* y la etapa 5 contiene el nodo *J*. Los estados en cada etapa corresponden a los nombres de los nodos. Así, la etapa 3 contiene los estados *E, F* y *G*.

Ya sea *S* un nodo en la etapa *J* y sea $f_j(S)$ la distancia más corta desde el nodo *S* al destino *J*, se puede escribir:

$$f_j(S) = \min_{\text{nodos } z \text{ en etapa } j+1} \{c_{sz} + f_{j+1}(Z)\}$$

Donde c_{sz} denota la longitud del arco *SZ*. Esto nos da la recursividad para resolver el problema, entonces, se parte con $f_5(J) = 0$.

A continuación, se muestra el resto de los cálculos:

ETAPA 4

En la etapa 4, realmente, no hay decisiones que tomar, hay que ir hasta el destino *J*.

- $f_4(H) = 3$ para ir a *J*
- $f_4(I) = 4$ para ir a *J*

ETAPA 3

Aquí, hay más decisiones. Calcular $f_3(F)$. Desde F se puede ir a H o a I. El costo inmediato de ir a H es 6. El siguiente costo es $f_4(H) = 3$. El costo total es 9.

El costo inmediato de ir a I es 3. El siguiente costo es $f_4(I) = 4$ con un total de 7.
Luego, encontrándose en F, la mejor decisión a tomar es ir a I. Con un costo total de 7, así, $f_3(F) = 7$.

La siguiente tabla resume los otros cálculos:

TABLA 2.1. Cálculos para la etapa 3

S_3	$Cs_3z_3 + f_4(Z_3)$		$f_3(S_3)$	Decisión de ir a
	H	***I***		
E	4	8	4	*H*
F	9	7	7	*I*
G	6	7	6	*H*

Se calcula hacia atrás etapa por etapa, cada vez completando el cálculo por etapa antes de continuar a la que le precede. Los resultados son:

ETAPA 2

TABLA 2.2. Cálculos para la etapa 2

S_2	$Cs_2z_2 + f_3(Z_2)$			$f_2(S_2)$	Decisión de ir a
	E	***F***	***G***		
B	11	11	12	11	*E* o *F*
C	7	9	10	7	*E*
D	8	8	11	8	*E* o *F*

ETAPA 1

TABLA 2.3. Cálculos para la etapa 3

S_1	$Cs_1z_1 + f_2(Z_1)$			$f_1(S_1)$	Decisión de ir a
	B	***C***	***D***		
A	13	11	10	10	*D*

Ejemplo 2.4. Problema de inversión de capital (Taha, 1995).

Una corporación tiene \$5 millones para invertir en sus tres plantas para una posible expansión. Cada planta ha presentado un número de propuestas sobre cómo pretende gastar el dinero. Cada propuesta entrega el costo de la expansión (c) y la ganancia esperada (r). La siguiente tabla resume las propuestas:

TABLA 2.4. Posibilidades de inversión

Propuesta	**Planta 1**		**Planta 2**		**Planta 3**	
	c_1	r_1	c_2	r_2	c_3	r_3
1	0	0	0	0	0	0
2	1	5	2	8	1	3
3	2	6	3	9	-	-
4	-	-	4	12	-	-

Cada planta solo podrá realizar una de sus propuestas. El objetivo es maximizar el retorno de la firma en su inversión de \$5 millones. Es de suponer que, si no se gastan los \$5 millones, completamente, ese dinero se perderá.

Una forma de resolver este problema es intentar todas las posibilidades y elegir la mejor. En ese caso, hay solo $3 \times 4 \times 2 = 24$ formas de invertir el dinero. Muchas de estas no son factibles (por ejemplo, las propuestas 3, 4 y 1 para las tres plantas cuestan \$6 millones). Otras propuestas son factibles, pero son muy pobres con el retorno (como las propuestas 1, 1 y 2 con un retorno de solo \$3 millones).

Desventajas de una enumeración completa:

1) Para problemas de gran tamaño, la enumeración de todas las posibles soluciones pueden no ser factibles, computacionalmente.

2) Las combinaciones no factibles no pueden ser detectadas *a priori*, llevando a una ineficiencia.

3) Información sobre combinaciones, previamente, investigadas no se usan para eliminar otras combinaciones menos buenas o no factibles.

Un método para calcular la solución es:

Se divide el problema en 3 etapas: cada etapa representa el dinero asignado a una única planta. Así, la etapa 1 representa el dinero asignado a la planta 1. Artificialmente, se dará un orden a las etapas, asumiendo que primero se asignará a la planta 1, luego a la planta 2 y, finalmente, a la planta 3.

Cada etapa está dividida en estados. Un estado guarda la información requerida para ir desde una etapa a la siguiente. En este caso, los estados por etapa 1, 2 y 3 son:

- $\{0,1,2,3,4,5\}$: cantidad de dinero gastado en la planta 1, representado como x_1
- $\{0,1,2,3,4,5\}$: cantidad de dinero gastado en las plantas 1 y 2 (x_2)
- $\{5\}$: cantidad de dinero gastado en las plantas 1, 2 y 3 (x_3)

Diferentemente, es necesario notar, en relación con PLE, las x_i no representan variables de decisión, ellas son simplemente representaciones de un estado genérico en la etapa.

Un retorno se asocia a cada estado. Se debe notar que para tomar una decisión en el estado 3, solo es necesario conocer cuánto se gastó en las plantas 1 y 2 y no cómo esto fue gastado. También, note que se desea que x_3 sea 5.

Entonces, determinando los retornos asociados a cada estado, lo más fácil es en la etapa 1, los estados x_1. La tabla 2.5 muestra el retorno asociado con x_1.

TABLA 2.5. Cálculos en la etapa 1

Si el capital disponible x_1 es:	**Entonces, la propuesta óptima es:**	**Y el retorno para la etapa 1 es:**
0	1	0
1	2	5
2	3	6
3	3	6
4	3	6
5	3	6

Ahora, se pueden calcular para la etapa 2. En este caso, deseamos encontrar la mejor solución tanto para la planta 1 como para la planta 2. Si se desea calcular el mejor retorno para un x_2 dado, simplemente se analizarán todas las propuestas de la planta 2, asignando la cantidad

requerida a la planta 2 y se debe usar la tabla anterior para ver cómo la planta 1 gastará el excedente.

Por ejemplo, supongamos que deseamos determinar la mejor asignación para el estado $x_2 = 4$. En la etapa 2 se puede hacer una de las siguientes propuestas:

1) Propuesta 1 da un retorno de 0, deja 4 para la etapa 1, el cual retorna 6. Total: 6

2) Propuesta 2 da un retorno de 8, deja 2 para la etapa 1, el cual retorna 6. Total: 14

3) Propuesta 3 da un retorno de 9, deja 1 para la etapa 1, el cual retorna 5. Total: 14

4) Propuesta 4 da un retorno de 12, deja 0 para la etapa 1, el cual retorna 0. Total: 12

Lo mejor que se puede hacer con 4 unidades es la propuesta 1 para la planta 2 y la propuesta 2 para la planta 1, con un retorno de 14 o la propuesta 2 para la planta 2 y la propuesta 1 para la planta 1, también, con un retorno de 14. En cualquier caso, el retorno para esta etapa con $x_2 = 4$ es 14. El resto de la tabla 2.6 se puede interpretar análogamente.

TABLA 2.6. Cálculos en la etapa 2

Si el capital disponible x_2 es:	**Entonces, la propuesta óptima es:**	**Y el retorno para las etapas 1 y 2 es:**
0	1	0
1	1	5
2	2	8
3	2	13
4	2 o 3	14
5	4	17

Ahora, se puede analizar la etapa 3. El único valor en el que estamos interesados es $x_3 = 5$.

Nuevamente, se analizarán todas las propuestas para esta etapa, determinando la cantidad de dinero remanente y usando la tabla 2.3 para decidir el valor de las etapas previas. Así, se puede realizar lo siguiente en la planta 3:

- Propuesta 1 da un retorno de 0, deja 5. Las etapas previas dan 17. Total: 17
- Propuesta 2 da un retorno de 3, deja 4. Las etapas previas dan 14. Total: 17

De esta forma, la solución óptima es implementar la propuesta 2 de la planta 3, la propuesta 2 o 3 en la planta 2 y la propuesta 3 o 2 (respectivamente) en la planta 1. Esto da un retorno de 17.

Si se estudia este procedimiento, podemos observar que los cálculos son efectuados *recursivamente.* Los cálculos de la etapa 2 están basados en la etapa 1, la etapa 3 solo está basada en la etapa 2. A su vez, estando en un estado, todas las futuras decisiones se toman independientemente de la manera en que se llegó a ese estado. Este es el *principio de optimalidad*, en el cual se sustenta la PD.

Se pueden resumir estos cálculos en las siguientes fórmulas:

Sean $R_j(k_j)$ el retorno para la propuesta k_j en el estado j y $c_j(k_j)$ el costo correspondiente. Sea $f_j(x_j)$ el retorno o rendimiento óptimo del estado x_j en el etapa j. Luego, se harán los siguientes cálculos:

$$f_1(x_1) = \max_{\text{propuestas factibles } k_1} \{R_1(k_1)\}$$

$$f_j(x_j) = \max_{\text{propuestas factibles } k_j} \{R_j(k_j) + f_{j-1}(x_{j-1})\}$$

Algunas observaciones son necesarias hacer a estas fórmulas, la función $f_j(x_j)$ depende solamente de x_j, entonces, es necesario que el segundo miembro de la ecuación recursiva se exprese en términos de x_j y no en términos de x_{j-1}, esto es posible a través de la siguiente sustitución:

$$x_{j-1} = x_j - c_j(k_j)$$

Donde $c_j(k_j)$ es el costo de la alternativa k_j en la etapa j. También, esto es importante ya que una propuesta k_j es factible, si su costo $c_j(k_j)$ no excede el estado del sistema x_j en la etapa j. Entonces, considerando esto se reescribe la ecuación recursiva como sigue:

$$f_1(x_1) = \max_{c_1(k_1) \le x_1} \{R_1(k_1)\}$$

$$f_j(x_j) = \max_{c_j(k_j) \le x_j} \{R_j(k_j) + f_{j-1}(x_j - c_j(k_j))\}$$

Los cálculos de cada etapa son como sigue:

ETAPA 1

$$f_1(x_1) = \max_{\substack{c_1(k_1) \le x_1 \\ k_1 = 1,\ 2,\ 3}} \{R_1(k_1)\}$$

TABLA 2.7. Cálculos para la etapa 1

	$R_1(k_1)$			Solución óptima	
x_1	$k_1 = 1$	$k_1 = 2$	$k_1 = 3$	$f_1(x_1)$	k_1^*
0	0	-	-	0	1
1	0	5	-	5	2
2	0	5	6	6	3
3	0	5	6	6	3
4	0	5	6	6	3
5	0	5	6	6	

ETAPA 2

$$f_2(x_2) = \max_{\substack{c_2(k_2) \le x_2 \\ k_2 = 1, 2, 3, 4}} \left\{ R_2(k_2) + f_1\left[x_2 - c_2(k_2)\right] \right\}$$

TABLA 2.8. Cálculos para la etapa 2

	$R_2(k_2) + f_1(x_2 - c_2(k_2))$				Solución óptima	
x_2	$k_2 = 1$	$k_2 = 2$	$k_2 = 3$	$k_2 = 4$	$f_2(x_2)$	k_2^*
0	0 + 0 = 0	-	-	-	0	1
1	0 + 5 = 5	-	-	-	5	1
2	0 + 6 = 6	8 + 0 = 8	-	-	8	2
3	0 + 6 = 6	8 + 5 = 13	9 + 0 = 9	-	13	2
4	0 + 6 = 6	8 + 6 = 14	9 + 5 = 14	12 + 0 = 12	14	2 o 3
5	0 + 6 = 6	8 + 6 = 14	9 + 6 = 15	12 + 5 = 17	17	4

ETAPA 3

$$f_3(x_3) = \max_{\substack{c_3(k_3) \le x_3 \\ k_3 = 1, 2}} \left\{ R_3(k_3) + f_2\left[x_3 - c_3(k_3)\right] \right\}$$

TABLA 2.9. Cálculos para la etapa 3

	$R_2(k_2) + f_1(x_2 - c_2(k_2))$		Solución óptima	
x_3	$k_3 = 1$	$k_3 = 2$	$f_3(x_3)$	k_3^*
5	0 + 17 = 17	3 + 14 = 17	17	1 o 2

La solución óptima se puede leer directamente de las tablas, comenzando en la etapa 3 para $x_3 = 5$ la propuesta óptima es $k_3^* = 1$ o bien, $k_3^* = 2$ para el primer caso como $c_3(1) = 0$, entonces :

$$x_2 = x_3 - c_3(1) = 5$$

Para las etapas 2 y 1:

La etapa 2 muestra que $x_2 = 5$ produce $k_2^* = 4$, como $c_2(4) = 4$, esto lleva a $x_1 = 5 - 4 = 1$. Entonces, de la etapa 1 $x_1 = 1$ con $k_2^* = 4$. Por lo tanto, una combinación óptima de propuestas para las etapas 1, 2 y 3 es (2, 4, 1). La siguiente figura 2.8 muestra cómo se determinan los óptimos alternativos de manera sistemática.

En este caso, los cálculos se efectuaron a través de un procedimiento recursivo de avance. Sin embargo, en la mayoría de los casos el procedimiento recursivo es de retroceso, aunque, es necesario notar que, también, se calcularon algunos elementos más desde la última etapa hacia la primera. La diferencia en los cálculos de avance o retroceso es la forma en que se define el estado del sistema, en este caso para los cálculos de retroceso se define:

- y_1 = cantidad asignada a las etapas 1, 2, y 3
- y_2 = cantidad asignada a las etapas 2 y 3
- y_3 = cantidad asignada a la etapa 3

Esto define una recursividad *hacia atrás*. Gráficamente, esto se ilustra en la figura 2.9.

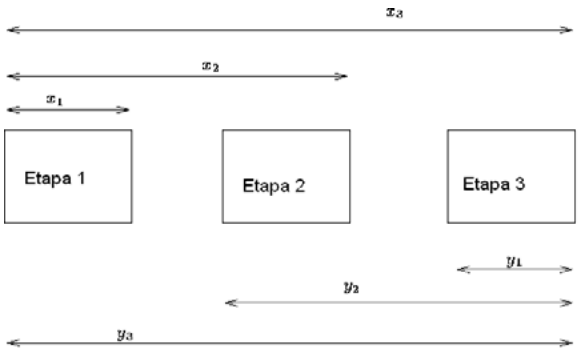


FIGURA 2.9. Recursividad de avance y de retroceso

Las fórmulas correspondientes son:

- $f_3(y_3)$ es el retorno óptimo para la etapa 3, dado por y_3
- $f_2(y_2)$ es el retorno óptimo para las etapas 2 y 3, dado por y_2
- $f_1(y_1)$ es el retorno óptimo para las etapas 1, 2 y 3, dado por y_1

Las fórmulas de recursividad son:

$$f_3(y_3) = \max_{\substack{c_3(k_3) \le y_3 \\ k_3 = 1, 2}} \{R_3(k_3)\}$$

y

$$f_j(y_j) = \max_{kj:cj(kj) \le yj} \left\{R_j(k_j) + f_{j+1}\left[y_j - c_j(k_j)\right]\right\}$$

Si se llevan a cabo todos estos cálculos, se llegará al mismo resultado como veremos en las tablas siguientes. Aunque, la recursividad hacia adelante parezca más natural, se introdujo una recursividad hacia atrás. En este caso particular, el orden de las etapas no es relevante. En otros casos pueden haber ventajas computacionales en la elección uno *versu*s otro orden. En general, la recursividad hacia atrás es más efectiva.

ETAPA 3

$$f_3(y_3) = \max_{\substack{c_3(k_3) \le y_3 \\ k_3 = 1, 2}} \{R_3(k_3)\}$$

TABLA 2.10. Cálculos para la etapa 3

	$R_3(k_3)$		Solución óptima	
y_3	$k_3 = 1$	$k_3 = 2$	$f_3(y_3)$	k_3^*
0	0	-	0	1
1	0	3	3	2
2	0	3	3	2
3	0	3	3	2
4	0	3	3	2
5	0	3	3	2

ETAPA 2

$$f_2(y_2) = \max_{\substack{c_2(k_2) \le y_2 \\ k_2 = 1, 2, 3, 4}} \left\{ R_2(k_2) + f_3\left[y_2 - c_2(k_2)\right] \right\}$$

TABLA 2.11. Cálculos para la etapa 2

	$R_2(k_2) + f_3(y_2 - c_2(k_2))$				Solución óptima	
y_2	$k_2 = 1$	$k_2 = 2$	$k_2 = 3$	$k_2 = 4$	$f_2(y_2)$	k_2^*
0	0 + 0 = 0	-	-	-	0	1
1	0 + 3 = 3	-	-	-	3	1
2	0 + 3 = 3	8 + 0 = 8	-	-	8	2
3	0 + 3 = 3	8 + 3 = 11	9 + 0 = 9	-	11	2
4	0 + 3 = 3	8 + 3 = 11	9 + 3 = 12	12 + 0 = 12	12	3 o 4
5	0 + 3 = 3	8 + 3 = 11	9 + 3 = 12	12 + 3 = 15	15	4

ETAPA 1

$$f_1(y_1) = \max_{\substack{c_1(k_1) \le y_1 \\ k_1 = 1, 2, 3}} \left\{ R_1(k_1) + f_2\left[y_1 - c_1(k_1)\right] \right\}$$

TABLA 2.12. Cálculos para la etapa 1

	$R_1(k_1) + f_2(y_1 - c_1(k_1))$			Solución óptima	
y_1	$k_1 = 1$	$k_1 = 2$	$k_1 = 3$	$f_1(y_1)$	k_1^*
5	0 + 15 = 15	5 + 12 = 17	6 + 11 = 17	17	2 o 3

2.5. CARACTERÍSTICAS COMUNES

Hay un conjunto de características que son comunes a estos ejemplos y a todos los problemas de PD.

1) El problema se puede dividir en *etapas* y se requiere una *decisión* en cada etapa. En el problema de inversión de capital, las etapas fueron las asignaciones para cada planta, la decisión fue cuánto gastar. En el problema de la ruta más corta, las etapas se definieron siguiendo la estructura del grafo, la decisión fue ir al siguiente.

2) Cada etapa tiene un conjunto de *estados* asociados a ella. Los estados para el problema de asignación de capital corresponden a la cantidad gastada en ese punto, en ese instante de tiempo. Los estados de la ruta más corta fueron los nodos visitados.

3) La decisión en una etapa transforma un estado de ella y en un estado en la siguiente etapa. En la asignación de capital: la decisión de cuánto gastar, dada una cantidad total gastada en la siguiente etapa. En la ruta más corta: la decisión de dónde seguir, dada la llegada en la siguiente etapa.

4) Dado el estado actual, la decisión óptima para cada uno de los estados que faltan no depende de los estados o de las decisiones previas. En la asignación de capital: no es necesario conocer cómo se gastó el dinero en las etapas previas, solo cuánto se gastó. En la ruta más corta: no fue necesario conocer cómo se llegó al nodo, solo se necesitaba saber que se llegó a ese nodo.

5) Hay una relación de recursividad que identifica la decisión óptima para la etapa j, dado que la etapa $j+1$ ya fue resuelta.

6) La etapa final debe resolverse por sí misma.

Las dos últimas propiedades obedecen a las relaciones de recursividad dadas, anteriormente. La potencialidad de la PD, con el arte involucrado, consiste en tomar un problema y determinar sus etapas y estados, tal que, las características mencionadas se cumplan. Si esto es posible, entonces, la relación de recursividad permite encontrar los valores en una forma relativamente fácil. La dificultad está en la determinación de las etapas y de los estados. Para visualizar este aspecto se recomienda estudiar y analizar los siguientes ejemplos.

2.6. VENTAJAS Y LIMITACIONES DE LA PD (PROGRAMACIÓN DINÁMICA)

Ya hemos mencionado algunas de las ventajas al usar la PD, la primera de ellas es sobre la transformación de un problema de varias variables en subproblemas de una sola variable, sin embargo, podemos enumerarlas como sigue:

- Solución de n problemas de una variable, en lugar de un problema de n variables.
- La PD determina un máximo o un mínimo global en lugar de uno local.
- Manejo de problemas con restricciones de variable entera.
- Tener soluciones parciales de un problema como en el caso de inversión, visto al inicio de este capítulo, se puede tener la solución para 3 o 4 inversiones y no solo para las 5.

Las limitaciones de la PD las podemos ver como sigue:

- La dimensionalidad del espacio de los estados tiene como consecuencia el que se tengan problemas de almacenamiento en la computadora.
- Si el número de variables crece mucho, la complejidad del planteamiento con la PD crece y no es eficiente.

2.7. El problema de PLG (Programación Lineal General) visto como un problema de PD (Programación Dinámica)

Se plantea un problema de PLE de la siguiente manera:

$$\text{maximizar } z = c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$$

sujeta a

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &\leq b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &\leq b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mm}x_n &\leq b_m \\ x_1, x_2, \ldots, x_n &\geq 0 \end{aligned}$$

El problema de PLG puede formularse como un modelo de PD. Cada actividad j (j = 1,2, ...,n) puede considerarse como una etapa. El nivel de actividad x_j ($\geq$ 0) representa las alternativas en la etapa j. Ya que x_j es continua, cada etapa posee un número infinito de alternativas dentro del espacio factible. Por razones que se van a establecer enseguida, se supone que todas las $a_{ij} \geq 0$.

El problema de PLE es un problema de asignación. Por consiguiente, similar a los ejemplos vistos con anterioridad, los estados pueden definirse como las cantidades de recursos que se van a asignar a la etapa actual y a las etapas subsecuentes. (Esto dará lugar a una ecuación recursiva de retroceso.) Ya que existen m recursos, los estados deben representarse con un vector de m dimensiones.

Ya sean (B_{1j}, B_{2j} ..., B_{mj}) los estados del sistema en la etapa j, esto es, las cantidades de recursos 1,2, …, m, asignadas a la etapa j, j + 1, …, n. Usando la ecuación recursiva, sea f_j

(B_{1j}, B_{2j}, ..., B_{mj}) el valor óptimo de la función objetivo para las etapas (actividades) j, $j + 1$, …, n dados los estados B_{ij}, ..., B_{mj}, por consiguiente, se tiene:

$$f\left(B_{1n}, B_{2n}, ..., B_{mn}\right) = \max_{\substack{0 \le a_{in} x_n \le B_{in} \\ i=1,2,...,m}} \left\{c_n x_n\right\}$$

$$f\left(B_{1n}, B_{2n}, ..., B_{mj}\right) = \max_{\substack{0 \le a_{ij} x_j \le B_{ij} \\ i=1,2,...,m \\ j=1,2..,n-1}} \left\{c_j x_j + f_{j+1}\left(B_{1j} - a_{1j} x_j, ..., B_{mj} - a_{mj} x_j\right)\right\}$$

donde $0 \le B_{ij} \le b_i$ para todas i y j.

Ejemplo 2.5. Considere el problema de PLE siguiente:

$$\text{maximizar } z = 2x_1 + 5x_2$$

$$\begin{aligned} &\text{sujeta a} \\ 2x_1 + x_2 &\le 430 \\ 2x_2 &\le 460 \\ x_1, x_2 &\ge 0 \end{aligned}$$

Como existen dos recursos, los estados del modelo de PD equivalentes se describen solo con dos variables. Sean (v_j, w_j) los estados en la etapa j (j = 1,2). Por lo tanto:

$$f_2(v_2, w_2) = \max_{\substack{0 \le x_2 \le v_2 \\ 0 \le 2x_2 \le w_2}} \left\{5x_2\right\}$$

ya que, $x_2 \le$ mín $\{v_2, w_2/2\}$ y $f_2(x_2 | v_2, w_2) = 5x_2$, entonces,

$$f_2(v_2, w_2) = \max_{x_2} f_2(x_2 | v_2, w_2) = 5 \text{ mín}\left(v_2, \frac{w_2}{2}\right)$$

$$\text{y} \quad x_2^* = \text{mín}\ (v_2, w_2/2)$$

ahora,

$$f_1(v_1, w_1) = \max_{0 \le 2x_1 \le v_1} \left\{2x_1 + f_2\left(v_1 - 2x_1, w_1\right)\right\} = \max_{0 \le 2x_1 \le v_1} \left\{2x_1 + 5\ \text{mín}(v_1 - 2x_1, \frac{w_1}{2})\right\}$$

Ya que esta es la última etapa $v_1 = 430$, $w_1 = 460$. Por consiguiente, $x_1 \le v_1/2 = 215$ y

$$f_1(x_1 \mid v_1, w_1) = f_1(x_1 \mid 430, 460) = 2x_1 + 5 \quad \text{mín}\left(430 - 2x_1, \frac{460}{2}\right)$$

$$= 2x_1 + \begin{cases} 5(230) & 0 \le x_1 \le 100 \\ 5(430 - 2x_1), & 100 \le x_1 \le 215 \end{cases}$$

$$= \begin{cases} 2x_1 + 1\,150 & 0 \le x_1 \le 100 \\ -8x_1 + 2150 & 100 \le x_1 \le 215 \end{cases}$$

Por lo tanto, para los intervalos o *rangos* dados de x_1:

$$f_1(v_1, w_1) = f_1(430, 460) = \max_{x_1} (2x_1 + 1150, -8x1 + 2\,150) =$$

$$= 2(100) + 1150 = -8(100) + 2\,150 = 1\,350$$

lo cual se logra en $x_1^* = 100$.

Para obtener x_2^*, observe que:

$$v_2 = v_1 - 2x_1 = 430 - 200 = 230$$
$$w_2 = w_1 - 0 = 460$$

en consecuencia,

$$x_2^* = \text{mín}\left(v_2 \frac{w_2}{2}\right) = \text{mín}\ (230,\ 460/2) = 230$$

por consiguiente, la solución óptima es $z = 1\,350$, $x_1 = 100$, $x_2 = 230$.

En este ejemplo todos los coeficientes de restricción son no negativos. Si algunos de los coeficientes son negativos, entonces, para una restricción del tipo (≤), ya no es cierto que el segundo miembro dé el valor máximo de la variable de estado. Este problema se agudizará más, si sucede que la solución es no acotada. Entonces, la conclusión general es que la PD no es adecuada para resolver el problema de PLG. Quizá, esto destacó el punto de que la PD se basa en un poderoso principio de optimización que es no factible, en cuanto al cálculo se refiere, para algunos problemas.

2.8. PROBLEMAS DE REEMPLAZO DE EQUIPO

Un tipo de problemas que no solo se limita al reemplazo de equipo es este que, en general, se refiere a reemplazo y que se aplica, también, a cuestiones financieras, industriales y de diseño, comúnmente. A continuación, se desarrolla un ejemplo sencillo y posteriormente otro en donde se hará uso de otros dos métodos de solución, que servirán para comparación y, también, como es importante en cualquier modelado, como validación del modelo de PD.

Ejemplo 2.6. Reemplazo de equipo (Taha, 2004).

En los próximos cinco años, supóngase que un negocio necesita tener una máquina. Cada máquina nueva tiene un costo \$1 000.00. El costo por mantener la máquina durante el año i-ésimo de operación es: m_1 = \$60.00, m_2 = \$80.00 y m_3 = \$120.00. Una máquina se puede usar por tres años y luego ser rematada. El valor de remate de la máquina después de i años es s_1 = \$800.00, s_2 = \$600.00 y s_3 = \$500.00. ¿Cómo podría el dueño del negocio minimizar los costos sobre un periodo de cinco años?

Las etapas están asociadas a cada año. El estado será la edad de la máquina en ese año. Las decisiones son: ya sea mantener la máquina o rematarla y reemplazarla por una nueva.

Sea $f_t(x)$ el mínimo costo desde el instante t al 5, ya que, la máquina tiene x años de antigüedad en el instante t.

Y como se debe rematar en el instante 5:

$$f_5(x) = -s_x$$

Ahora, se consideran los otros periodos de tiempo. Si se tiene en el instante t una máquina con 3 años de antigüedad, esta se debe rematar en:

$$f_t(3) = -500 + 1000 + 60 + f_{t+1}(1) \qquad \text{(venta)}$$

Si tiene dos años de antigüedad, se puede rematar o mantenerla.

Costo de remate:

$$-600 + 1000 + 60 + f_{t+1}(1)$$

Costo por mantenerla:

$$120 + f_{t+}1 \ (3)$$

Así, la mejor decisión con una máquina que tiene dos años de antigüedad es el mínimo de los dos costos.

Análogamente:

$$f_t(1) = \text{mín}\left\{-800 + 1\,000 + 60 + f_{t+1}(1), 80 + f_{t+1}(2)\right\}$$

finalmente, en el instante inicial se debe comprar:

$$f_0(0) = 1\,000 + 60 + f_1(1)$$

Cuando se usa una recursividad de retroceso se tiene:

ETAPA 5

TABLA 2.13. Cálculos para la etapa 5

Edad *x*	$f_5(x)$
1	– 800
2	– 600
3	– 500

ETAPA 4

TABLA 2.14. Cálculos para la etapa 4

Edad *x*	**Venta**	**Mantener**	$f_4(x)$	**Decisión**
1	– 540	– 520	– 540	vender
2	– 340	– 380	– 380	mantener
3	– 240		– 240	vender

ETAPA 3

TABLA 2.15. Cálculos para la etapa 3

Edad *x*	**Venta**	**Mantener**	$f_3(x)$	**Decisión**
1	– 280	– 300	– 300	mantener
2	– 80	– 120	– 120	mantener
3	20		20	vender

ETAPA 2

TABLA 2.16. Cálculos para la etapa 2

Edad *x*	**Venta**	**Mantener**	$f_2(x)$	**Decisión**
1	– 40	– 40	– 40	vender o mantener
2	160	140	140	mantener

ETAPA 1

TABLA 2.17. Cálculos para la etapa 1

Edad *x*	**Venta**	**Mantener**	$f_1(x)$	**Decisión**
1	220	220	220	vender o mantener

ETAPA 0

TABLA 2.18. Cálculos para la etapa 0

Edad *x*	**Venta**	**Mantener**	$f_0(x)$	**Decisión**
0		1 280	1 280	mantener

El costo es de $1 280.00 y una solución es rematarla en los años 1 y 2. Existen otras soluciones óptimas, se recomienda determinarlas.

Ejemplo 2.7. Compra de auto.

El problema siguiente se resolverá utilizando PLE, teoría de redes y PD con el fin de comparar los tres procedimientos.

Ahora, considere que acaba de comprar un auto nuevo en $12 000.00, el costo de mantenimiento cambiará según la edad del mismo. Para evitar estos costos, supóngase que puede dar como adelanto el valor de su antiguo auto, el cual depende de la edad que tenga. También, suponga que por facilidad, el nuevo automóvil costará $12 000.00, independientemente, del momento de la compra. Su meta es minimizar el costo neto incurrido durante los próximos cinco años.

TABLA 2.19. Datos del problema

Edad del auto	Costo de mantenimiento	Valor del auto viejo
0	2 000	------
1	4 000	7 000
2	5 000	6 000
3	9 000	2 000
4	17 000	1 000
5	-------	0

1) Solución usando PLE.

Para mayor claridad en el planteamiento del problema, primero se expondrá la red con las distintas decisiones y costos por los que podrían optarse.

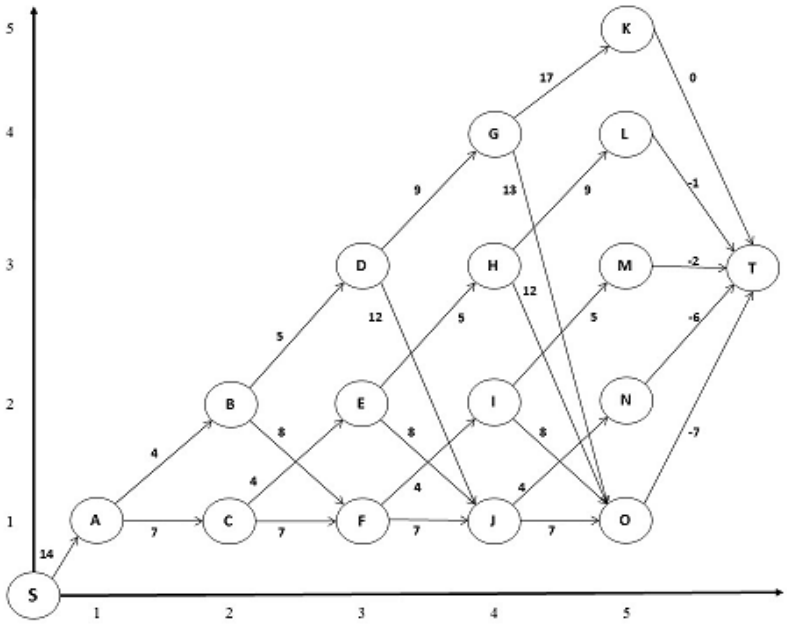


FIGURA 2.10. Representación gráfica o en red de las opciones y solución con PLE

Interpretación:

- S→A En el momento en que se decide comprar el auto, se realiza un gasto por la cantidad de $12 000.00 + $2 000.00 de mantenimiento en el año 0, es decir, se realiza un desembolso total de $14 000.00.
  Cada año, después de haber realizado la adquisición, se tiene la posibilidad de quedarse con el auto o comprar uno nuevo. Por lo tanto, en cada nodo deben existir dos arcos salientes que representen cada una de estas opciones.

- A→B Representa la decisión de quedarse el auto durante el primer año, por lo que, se realizará un gasto de $4 000.00, únicamente, por mantenimiento correspondiente a dicho año.

- A→C Representa la decisión de vender el auto el primer año. En cualquier año, se deben pagar $12 000.00 por la compra del vehículo nuevo, pero pueden tomar a cuenta el auto viejo, que en este año debe valer $7 000.00. De esta forma estarían faltando $5 000.00 por cubrir. No se debe olvidar que en el año 0 se tiene un costo de mantenimiento de $2 000.00, por lo que, en total faltan cubrir $5 000.00 + $2 000.00 = $7 000.00.

- B→D Representa que el año anterior se conservó el auto y que este año, también, se conserva, así que se deben pagar $5 000.00 de mantenimiento correspondientes al segundo año de posesión.

- C→E Representa la decisión de mantener el auto un año más, pero el año pasado se había comprado un auto nuevo, por lo que, en esta ocasión debe pagarse el mantenimiento correspondiente a la posesión de un año, es decir, $4 000.00.

- B→F Representa el haber decidido vender el auto en el segundo año. El auto viejo será tomado en cuenta por $6 000.00, faltando $6 000.00 para cubrir el costo del auto nuevo. Si agregamos el costo del mantenimiento del año 0, en total se deben cubrir $8 000.00.

- G→K Representa la decisión de mantener el auto un año más, ya que se había mantenido durante los cuatro años anteriores, o sea que, se debe cubrir el costo de mantenimiento de cinco años.

Si consideramos que se busca minimizar los costos, el modelo queda planteado de la siguiente manera:

sean:

$$X_{ij} = \left\{ \begin{matrix} 1, \text{si se elige ir del nodo } i \text{ al nodo } j \\ 0 \text{ en caso contrario} \end{matrix} \right\}$$
$$i = A, B, C, \dots, O, S, T$$
$$j = A, B, C, \dots, O, S, T$$

Entonces, se tiene un modelo de la siguiente forma:

$$\begin{aligned} \text{minimizar } & 14X_{SA} + 4X_{AB} + 7X_{AC} + 5X_{BD} + 8X_{BF} + 4X_{CE} + 7X_{CF} + 9X_{DG} + 12X_{DJ} + 5X_{EH} \\ & + 8X_{EJ} + 4X_{FI} + 7X_{FJ} + 17X_{GK} + 13X_{GO} + 9X_{HL} + 12X_{HO} + 5X_{IM} + 8X_{IO} + 4X_{JN} \\ & + 7X_{JO} - 7X_{OT} - 6X_{NT} - 2X_{MT} - 1X_{LT} \end{aligned}$$

sujeta a

$$\begin{aligned} & X_{SA} = 1 \\ & X_{SA} - X_{AB} - X_{AC} = 0 \\ & X_{AB} - X_{BD} - X_{BF} = 0 \\ & X_{AC} - X_{CE} - X_{CF} = 0 \\ & X_{BD} - X_{BD} - X_{BF} = 0 \\ & X_{CE} - X_{EH} - X_{EJ} = 0 \\ & X_{CF} + X_{BF} - X_{FI} - X_{FJ} = 0 \\ & X_{DG} - X_{GK} - X_{GO} = 0 \\ & X_{EH} - X_{HL} - X_{HO} = 0 \\ & X_{FI} - X_{IM} - X_{IO} = 0 \\ & X_{DJ} + X_{EJ} + X_{FJ} - X_{JN} - X_{JO} = 0 \\ & X_{GK} - X_{KT} = 0 \\ & X_{HL} - X_{LT} = 0 \\ & X_{IM} - X_{MT} = 0 \\ & X_{JN} - X_{NT} = 0 \\ & X_{GO} + X_{HO} + X_{IO} + X_{JO} - X_{OT} = 0 \\ & X_{KT} + X_{LT} + X_{MT} + X_{NT} + X_{OT} = 1 \end{aligned}$$

Después de resolver el modelo usando LINDO[16] en cinco iteraciones, se obtiene el siguiente resultado:

$$Z = 31$$

$$X_{SA} = X_{AC} = X_{CE} = X_{EJ} = X_{JN} = X_{NT} = 1$$

Por lo tanto, la solución es:

Costo total = comprar el auto (14) + vender el primer año (7) + mantener el primer año (4) + vender el segundo año (8) + mantenerlo el primer año (4) + venderlo (– 6) = 31.

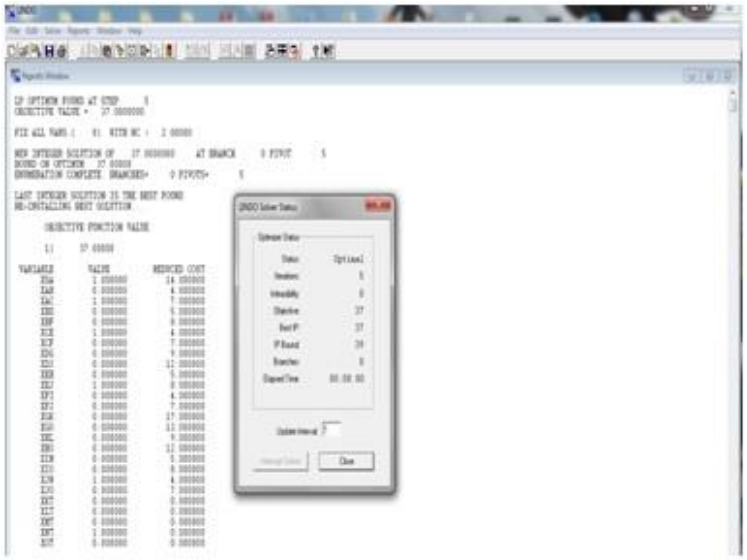


FIGURA 2.11. Resultados obtenidos con LINDO

2) Solución usando teoría de redes.

Para el planteamiento de la red, se utilizó el siguiente esquema de la figura 2.12, a través de los nodos X_{ij} donde i representa el año de decisión y j la edad del automóvil.

[16] LINDO (Linear INteractive Discrete Optimization-Optiminización Lineal Discreta e Interactiva). Software que permite realizar la optimización de problemas de programación lineal y cuadrática, definidos sobre variables reales y/o binarias.

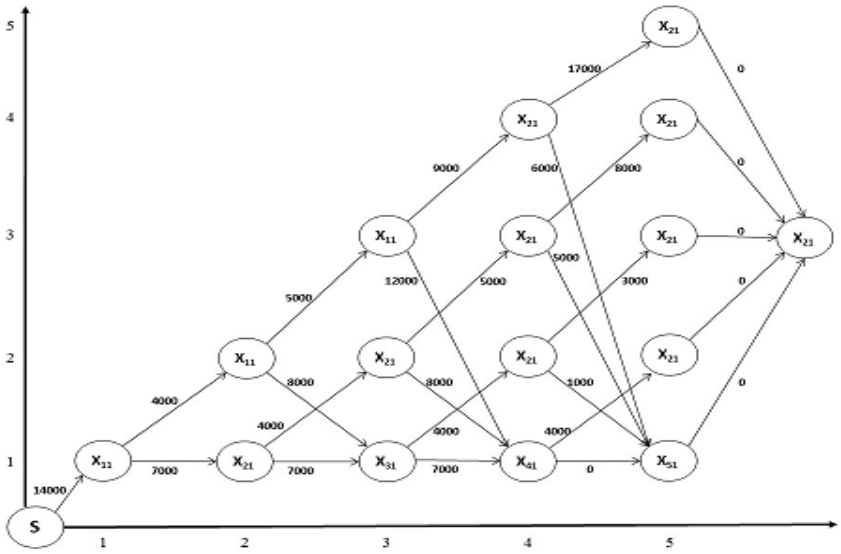


FIGURA 2.12. Red del problema

La solución se plantea como un problema de ruta más corta y se resuelve a través del algoritmo de Dijkstra. Este algoritmo que es propio de la teoría de redes, a pesar de tener varios años de haberse formulado, sigue siendo muy eficiente debido a la naturaleza de sus cálculos, como se verá en la sección siguiente.

Algoritmo de Dijkstra

Propósito: obtener la arborescencia de las rutas más cortas de la raíz s en una red $G = [N, A, d]$ con costos no negativos en los arcos.

Descripción:

PASO 1: (Iniciación de etiquetas). Sea $d(s) = 0$ y márquese esta etiqueta como permanente. Sea $d(x) = \infty$, para todo $x \neq s$ y considérense estas etiquetas como temporales. Sean $a(x) = x$ (estas etiquetas indicarán el predecesor de x en la arborescencia). Sea $p = s$.

PASO 2: (Actualización de etiquetas). Para todo $x \in \Gamma^+(p)$ que tenga etiqueta temporal, actualizar etiquetas de acuerdo a:

$$d(x) = \text{mín } \{d(x), d(p) + d(p, x)\}$$

Si $d(x)$ se modificó, hacer $a(x)=p$. Sea x^*, tal que, $d(x^*)$=mín $\{d(x) \mid d(x)$ es temporal$\}$. Si $d(x^*)= \infty$, terminar. En este caso no existe arborescencia alguna de raíz s. En otro caso, marcar la etiqueta $d(x^*)$ como permanente. Sea $p=x^*$.

PASO 3: (*i*) (Si solo se desea la ruta de s a t). Si $p=t$, terminar: $d(p)$ es la longitud del camino más corto. Si $p \neq t$, ir al paso 2.

(*ii*) (Si se desea la arborescencia). Si todos los nodos tienen etiquetas permanentes, terminar; esta es la longitud deseada del camino y el conjunto de arcos $\{a(x), x\}$ forman la arborescencia de caminos más cortos. En otro caso, ir al paso 2.

A continuación, se exponen las primeras tres iteraciones:

Iteración 1:

$$s = 0, d(Xij) = \infty \; con \; i = 1, \dots, 5 \; j = 1, \dots, 5$$
$$\Gamma^+(p) = \{X_{11}\}$$
$$d(X_{11}) = \text{mín } \{\infty, 14\,000\} = 14\,000$$
$$\text{mín}\left(d(X_{ij})\right) = d(X_{11}) = 14\,000, d(p) = 14\,000 \text{ y marcamos } x_{11} \; como \; permanente$$

Iteración 2:

$$\Gamma^+(p) = \{X_{21}, X_{22}\}$$
$$d(X_{21}) = \text{mín } \{\infty, 14\,000 + 7\,000\} = 21\,000$$
$$d(X_{22}) = \text{mín } \{\infty, 14\,000 + 4\,000\} = 18\,000$$
$$\text{mín}\left(d(X_{ij})\right) = d(X_{22}) = 18\,000 \text{ y marcamos } x_{22} \; como \; permanente$$

Iteración 3:

$$\Gamma^+(p) = \{X_{33}, X_{31}\}$$
$$d(X_{33}) = \text{mín } \{\infty, 18\,000 + 5\,000\} = 23\,000$$
$$d(X_{31}) = \text{mín } \{\infty, 18\,000 + 8\,000\} = 26\,000$$
$$\text{mín}\left(d(X_{ij})\right) = d(X_{21}) = 21\,000 \text{ y marcamos } x_{21} \; como \; permanente$$

Ahora, siguiendo el algoritmo para encontrar la ruta más corta, es decir, $p = T$, después de ocho iteraciones, obtenemos la siguiente red:

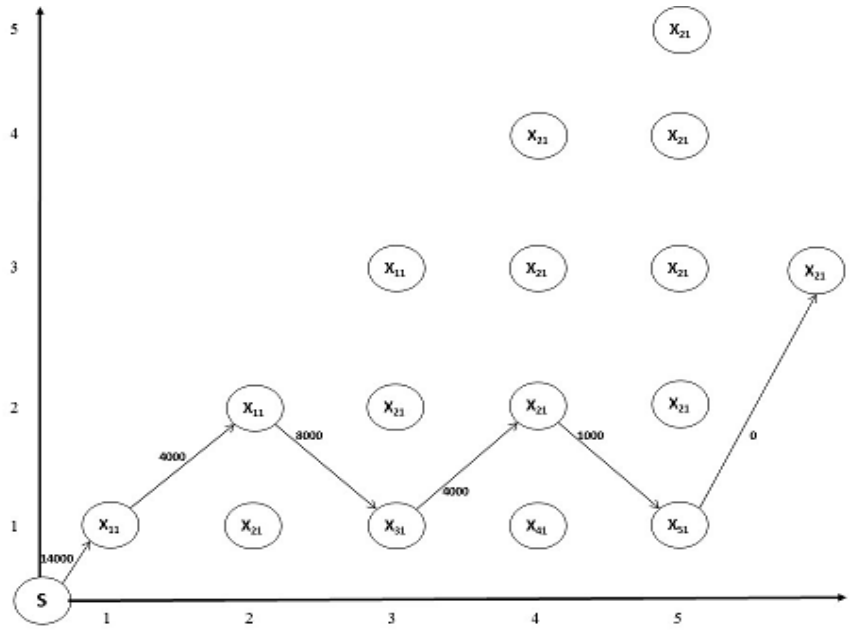


FIGURA 2.13. Diagrama de la solución del algoritmo

La solución se puede interpretar de la siguiente manera: compramos el automóvil en el año 0 y consideramos el año de mantenimiento para llegar al año 1 (con un costo de \$12 000.00 + \$2 000.00 = \$14 000.00); mantenemos el automóvil 1 año (+ \$4 000.00) para llegar al año 2; vendemos el automóvil en el año 2 y pagamos el mantenimiento para llegar al año 3 (\$12 000.00 + \$2 000.00 – \$6 000.00 = \$8 000.00); pagamos el mantenimiento del año 4 (+ \$4 000.00) y finalmente compramos un auto el año 5 y lo vendemos al final del año (\$8 000.00 – \$7 000.00 = \$1 000.00).

Costo total = 14 000.00 + 4 000.00 + 8 000.00 + 4 000.00 + 1 000.00 = \$31 000.00

3) Solución usando PD.

Sean:
K = la decisión de mantener el auto
R = la decisión de reemplazarlo
I = costo del auto nuevo
$c(t)$ = costo de mantenimiento
$s(t)$ = valor de recuperación

f_i = costo para los años i, i = 1,2,3,4

$$\Rightarrow f_i = \text{mín}\begin{cases} c(t) + f_{i+1}(t+1) \\ c(0) + I - s(t) + f_{i+1}(1) \end{cases}$$

Para la primera etapa, se tiene que $f_5 = 0$, además, si se reemplaza el auto en el año 4, los costos incluirán el valor de recuperación del auto reemplazado, así como, el del auto nuevo (cifras en miles).

ETAPA 4

t	$K = c(t) - s(t+1)$	$R = c(0) + I - s(1) - s(t)$	$f_4(t)$	Decisión
1	$4 - 6 = -2$	$2 + 12 - 7 - 7 = 0$	–2	*K*
2	$5 - 2 = 3$	$2 + 12 - 7 - 6 = 1$	1	*R*
3	$9 - 1 = 8$	$2 + 12 - 7 - 2 = 5$	5	*R*
4	$17 - 0 = 17$	$2 + 12 - 7 - 1 = 6$	6	*R*

ETAPA 3

t	$K = c(t) + f_4(t+1)$	$R = c(0) + I - s(t) + f_4(1)$	$f_3(t)$	Decisión
1	$4 + 1 = 5$	$2 + 12 - 7 - 2 = 5$	5	*K* o *R*
2	$5 + 5 = 10$	$2 + 12 - 6 - 2 = 6$	6	*R*
3	$9 + 6 = 15$	$2 + 12 - 2 - 2 = 10$	10	*R*

ETAPA 2

t	$K = c(t) + f_3(t+1)$	$R = c(0) + I - s(t) + f_3(1)$	$f_2(t)$	Decisión
1	$4 + 6 = 10$	$2 + 12 - 7 + 5 = 12$	10	*K*
2	$5 + 10 = 15$	$2 + 12 - 6 + 5 = 13$	13	*R*

ETAPA 1

t	$K = c(t) + f_2(t+1)$	$R = c(0) + I - s(t) + f_2(1)$	$f_1(t)$	Decisión
1	$4 + 13 = 17$	$2 + 12 - 7 + 10 = 17$	17	*K* o *R*

Por lo tanto, la solución gráfica es:

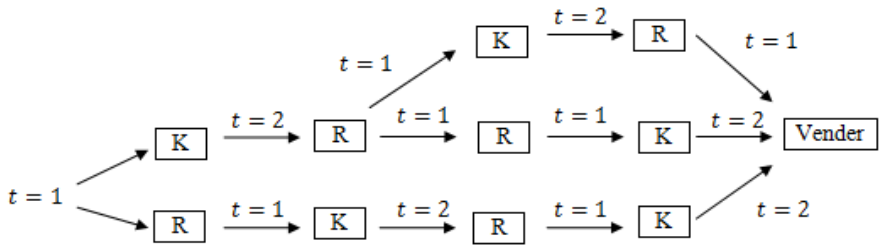


FIGURA 2.14. Representación gráfica de la solución, utilizando la PD

O sea, se tienen tres soluciones posibles:

La primera solución es comprar el auto, mantenerlo el primer año, venderlo el segundo año y adquirir otro; este mantenerlo el tercer año, venderlo el cuarto año y adquirir otro; y finalmente, este último venderlo el quinto año.

La segunda solución es: mantenerlo el primer año, venderlo el segundo año y adquirir otro; este venderlo el tercer año y adquirir otro; este último mantenerlo el cuarto año y finalmente venderlo el quinto año.

Por último, la tercera solución es: venderlo el primer año y adquirir otro; este mantenerlo el segundo año, venderlo el tercer año y adquirir otro; este último mantenerlo el cuarto año y finalmente venderlo el quinto año.

Todas estas soluciones con un costo total de $17 000.00 + $12 000.00 de la compra inicial + $2 000.00 de mantenimiento inicial = $31 000.00.

En este caso y en problemas de tamaño pequeño o mediano, se puede ver la gran ventaja al usar la PD que, por un lado, nos provee de soluciones óptimas múltiples cuando existen y, además, nos arroja óptimos para cada etapa del algoritmo. En términos de complejidad computacional, brevemente, se presenta para cada algoritmo su complejidad.

2.9. BREVE INTRODUCCIÓN A LA COMPLEJIDAD COMPUTACIONAL

Para entender las comparaciones que se hacen a continuación, es necesario definir la complejidad computacional, pero como no es el objetivo central de este capítulo, se verá un

ejemplo sencillo que da una idea de las comparaciones y los detalles se verán en el anexo B al final de estos apuntes.

Para que una computadora lleve a cabo una tarea es preciso decirle qué operaciones debe realizar, es decir, debemos describir cómo debe realizar la tarea. Dicha descripción se llama algoritmo.

Un algoritmo describe el método mediante el cual se realiza una tarea. Un algoritmo consiste en una secuencia de instrucciones, las cuales cuando son realizadas adecuadamente dan lugar al resultado deseado.

La noción de algoritmo no es exclusiva de la computación o de las matemáticas. Existen algoritmos que describen toda clase de procesos de la vida real, por ejemplo, las recetas de cocina, las partituras musicales, etc.

En todos los casos anteriores, el ejecutor o procesador de las instrucciones que realiza la tarea correspondiente es el hombre. Sin embargo, el agente que interpreta y realiza las instrucciones de un algoritmo se llama procesador.

Un procesador puede ser una persona, una computadora o cualquier otro sistema electrónico o mecánico. Un procesador realiza un proceso siguiendo o ejecutando el algoritmo correspondiente. La ejecución de un algoritmo requiere la ejecución de cada uno de los pasos o instrucciones que lo constituyen. De aquí, se desprende que una computadora no es más que un tipo particular de procesador.

Un aspecto importante es el diseño de algoritmos. ¿Cómo diseñar buenos algoritmos? El diseño de buenos algoritmos requiere creatividad e ingenio y no existen, en general, reglas para diseñar algoritmos. En otras palabras, no existe un algoritmo para diseñar algoritmos.

Si existen varios algoritmos para resolver un problema ¿Cuál de ellos es el *mejor*, en el sentido de que necesita menos recursos informáticos? ¿Cuáles son los mínimos recursos informáticos necesarios para llevar a cabo una tarea determinada? es decir, ¿Qué recursos utilizará el mejor algoritmo posible para realizar dicha tarea? ¿Se puede averiguar cuál es el mejor algoritmo? ¿Existen problemas para los cuales el mejor algoritmo posible requerirá tantos recursos que hará inviable su ejecución, incluso con la computadora más grande y rápida existente?

Todas estas cuestiones se engloban para su estudio bajo el título: *complejidad de algoritmos*.

El tiempo de ejecución requerido por un algoritmo para resolver un problema es uno de los parámetros importantes en la práctica para medir la bondad de un algoritmo pues, entre otros factores, el tiempo de ejecución equivale al tiempo de utilización de la computadora y, en consecuencia, al costo económico, sin embargo, el *tiempo de ejecución* se define como: *el número de operaciones que requiere un algoritmo para encontrar una solución*, es decir, se refiere a cuántas operaciones se tienen que realizar y no al tiempo real, ya que, se pueden tener diferentes procesadores con distintas velocidades. La complejidad computacional de un algoritmo es, entonces, una función que acota superiormente el tiempo de ejecución y se usa la notación $O(.)$, esto se verá con más claridad en el siguiente ejemplo.

Ejemplo 2.8. Se busca resolver un sistema de tres ecuaciones lineales con tres incógnitas a través del método de Gauss con sustitución regresiva, se demostrará que el tiempo de ejecución de este algoritmo es de $O(n^3)$.

Así, sea el siguiente sistema de ecuaciones:

$$\begin{aligned} x_1 - 3x_2 + x_3 &= 5 \\ 3x_1 - 8x_2 + 2x_3 &= -1 \\ x_1 - x_2 - 2x_3 &= 7 \end{aligned}$$

El cual se puede representar, matricialmente, bajo la forma $Ax=b$ donde A es una matriz de 3×3, x es un vector columna de 3×1 y b es un vector columna de 3×1, de la siguiente manera:

$$A = \begin{pmatrix} 1 & -3 & 1 \\ 3 & -8 & 2 \\ 1 & -1 & -2 \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \qquad b = \begin{pmatrix} 5 \\ -1 \\ 7 \end{pmatrix}$$

En el método de Gauss con sustitución regresiva se busca a través de operaciones elementales (como son el intercambio de renglones, multiplicación de un renglón por un escalar y multiplicación un renglón por un escalar y sumarlo a otro) crear una matriz triangular superior que contiene ceros debajo de la diagonal principal. Esto se puede hacer tanto con matrices cuadradas como rectangulares, en cuyo caso se busca obtener una matriz escalonada.

Se utilizan los elementos de la diagonal principal como pivote, es decir, como elemento al que se le aplique alguna de las operaciones elementales y al contar la cantidad de operaciones que se realicen se estarán contando las operaciones a realizar en el algoritmo y, por lo tanto, su complejidad computacional.

Se usa el elemento $a_{11} = 1$ como pivote para comenzar a usar el método de Gauss con sustitución regresiva, de esta manera, se procede a hacer ceros debajo de la primera columna de la matriz A. Las operaciones elementales se pueden escribir como $E_{ij}^{(k)}$ donde i es el renglón que se va a multiplicar por el escalar k y se suma al renglón j. Por lo tanto, para la primer columna son $E_{12}^{(-3)}$ y $E_{13}^{(-1)}$, quedando la segunda matriz. Posteriormente, se usa el pivote ubicado en la posición $a_{22} = 1$ y se hace $E_{23}^{(-2)}$, quedando la tercera matriz como sigue:

$$A=\begin{pmatrix}1 & -3 & 1\\ 3 & -8 & 2\\ 1 & -1 & -2\end{pmatrix}\approx\begin{pmatrix}1 & -3 & 1\\ 0 & 1 & -1\\ 0 & 2 & -3\end{pmatrix}\approx\begin{pmatrix}1 & -3 & 1\\ 0 & 1 & -1\\ 0 & 0 & -1\end{pmatrix}$$

Las mismas operaciones se realizan en el vector b, sin embargo, ahora no se contabilizan, porque son mínimas. Se presenta el desarrollo:

$$b=\begin{pmatrix}5\\ -1\\ 7\end{pmatrix}\approx\begin{pmatrix}5\\ -16\\ 34\end{pmatrix}$$

Y, sustituyendo se tiene $x_1 = -111$, $x_2 = -50$ y $x_3 = -34$. El conteo de operaciones es en general. Si A es una matriz de nxn, entonces, para la primer columna se desarrollan $(n-1)$ $(n-1)$ operaciones, es decir, $(n-1)^2$, esto es, porque se considera que para hacer ceros debajo de la primera columna, ya no se hace ninguna operación, solo en las $(n-1)$ entradas restantes.

Para la segunda columna son $(n-2)$ $(n-2) = (n-2)^2$ y, así, sucesivamente al sumarlas tenemos, en general, para n:

$$(n-1)^2 + (n-2)^2 + \ldots + 1 = \frac{n(n+1)(2n+1)}{6} = \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6}$$

Para la sustitución regresiva, se puede ver que de la última ecuación se usa una operación de despeje, para la penúltima dos operaciones y, así, sucesivamente, (se pide al lector comprobar esto). Sumando estas operaciones se tiene:

$$1 + 2 + 3 + \ldots + n = \frac{n^2}{2} + \frac{n}{2}$$

Si se comparan las dos series para una n muy grande, la que crece más rápido es la primera, ya que, tiene un término de n^3 y, en general, decimos que está acotada por esta función, por lo que, se concluye que la complejidad computacional del algoritmo de Gauss con sustitución regresiva es $O(n^3)$.

***Complejidad computacional para el algoritmo de PLE* (*Programación Lineal Entera*)**

El problema se resolvió usando el *software* LINDO que usa un algoritmo de ramificación y acotamiento, cuya complejidad computacional es como sigue: ya que en el esquema de este algoritmo se requiere del uso del método simplex para su ejecución tantas veces como se ramifica en el árbol de decisiones, que es parte central del mismo y como la búsqueda en dicho árbol está acotada por $O(n)$, mientras que el método simplex tiene una complejidad de $O(n^3)$, entonces, la mayor cota en el caso de medir la complejidad es la que determina la misma, por lo cual, se puede afirmar sin pérdida de generalidad que la complejidad de este algoritmo es $O(n^3)$, para algoritmos polinomiales se dice que son eficientes, aunque, puede haber casos, sobre todo en problemas grandes, donde se tiene que recurrir a otros métodos. Como se puede ver en este caso, se tiene $n = 25$, pero el número de iteraciones no crece polinomialmente dada la estructura misma del problema.

***Complejidad computacional para el algoritmo de Dijkstra*[17]**

Para considerar la complejidad computacional de este algoritmo se consideran dos operaciones básicas:

1) Selección de nodos. En este caso, el algoritmo desarrolla esta operación n veces y cada operación requiere que se revise cada nodo etiquetado temporalmente, por lo que, el tiempo total para la selección de nodos es:

$$n+(n-1)+(n-2)+\ldots+1=O(n^2)$$

2) Actualización de distancias. El algoritmo desarrolla esta operación $\left|A(i)\right|$ veces para cada nodo i. En total, el algoritmo realiza esta operación:

$$\sum_{i\in N}\left|A(i)\right|=m$$

Ya que, cada operación de actualización de distancias requiere $O(1)$ vez, el algoritmo en total requiere $O(m)$ veces para actualizar las etiquetas de distancias.

Cuando se calcula la complejidad computacional de un algoritmo, se considera el paso o la instrucción que mayor tiempo requiera, que en este caso es $O(n^2)$, por lo que, se considera esta la complejidad computacional del algoritmo.

[17] Ahuja, Ravindra K., *et al*., *Network Flows*: *Theory, Algorithms, and Applications*, Englewood Cliffs, Prentice Hall, 1993.

***Complejidad computacional del algoritmo de PD* (*Programación Dinámica*)**

En problemas de ruta más corta, los algoritmos de PD tienen una complejidad computacional que es igual a los de ramificación y acotamiento por su naturaleza recursiva, es decir, $O(n^3)$.

2.10. CONCLUSIONES

El objetivo de este capítulo ha sido mostrar los conceptos básicos y los fundamentos de la PD a través de ejemplos que conllevan una interpretación gráfica como es el caso de la ruta más corta y algunos otros, en los cuales no es tan obvia la solución gráfica como el ejemplo de la compra del auto, donde se presenta una solución usando teoría de redes y PLE, con la finalidad de comparar los métodos de solución, pero, también, y esto es muy importante, que el estudiante cuente con métodos alternos que permitan validar los modelos de la PD.

2.11. NOTAS HISTÓRICAS

Richard Ernest Bellman
(1920–1984) fue un matemático aplicado, cuya mayor contribución fue la metodología denominada programación dinámica.

Bellman estudió matemáticas en la Universidad de Brooklyn, donde obtuvo una diplomatura, y luego en la Universidad de Wisconsin, su licenciatura. Posteriormente comenzó a trabajar en el Laboratorio Nacional Los Álamos en el campo de la física teórica. En 1946, obtuvo su doctorado en la Universidad de Princeton. También, ejerció la docencia en la Universidad del Sur de California (EEUU), fue socio de la Academia Americana de las Artes y las Ciencias (1975) y de la Academia Nacional Americana de Ingeniería (1977). En 1979, el IEEE le otorgó la medalla de honor por su contribución a la teoría de los sistemas de control y de los procesos de decisión, en especial por su contribución con la programación dinámica y por la ecuación de Bellman.

Su primer estudiante de doctorado fue Austin Esogbue, que es actualmente profesor en el Instituto Tecnológico de Georgia, en el Departamento de Ingeniería Industrial y de Sistemas.
En algún lugar, entre 400 y 300 A. C., el gran matemático griego **Euclides** inventó un algoritmo para encontrar el Máximo Común Divisor (MCD) entre dos números naturales. El MCD de X y Y es el mayor entero que divide a ambos. Por ejemplo, el MCD de 80 y 32 es 16.

Los detalles del algoritmo no nos interesan por el momento, pero el algoritmo de Euclides se considera el primer algoritmo no trivial desarrollado.

La palabra **algoritmo** se deriva del nombre del matemático persa Mohammed Al-Khowârizmî, quien vivió durante el siglo noveno y quien tiene el mérito de haber elaborado paso a paso reglas para sumar, restar, multiplicar y dividir números decimales ordinarios. El nombre de este matemático en latín se convirtió en Algorismus, de donde declinó en algoritmo. Claramente, Euclides y Al-Khowârizmî fueron algorítmicos por excelencia.

El problema de la ruta más corta y sus generalizaciones tienen una voluminosa literatura. Como una guía al respecto hasta antes de 1984, remitimos al lector a la extensa bibliografía compilada por Deo y Pang (1984). En este escrito presentamos algunas referencias seleccionadas; referencias adicionales pueden encontrarse en los artículos de Ahuja, Magnanti y Orlin (1989, 1991).

El primer algoritmo de etiquetado fue sugerido por **Dijkstra** (1959) e, independientemente, por Dantzig (1960) y Whiting y Hillier (1960). La instrumentación original del algoritmo de Dijkstra corre en un tiempo de $O(n^2)$, el cual es el tiempo óptimo de corrida para redes completamente densas [aquellas con $m = \Omega\ (n^2)$] debido a que cualquier algoritmo debe examinar todos los arcos.

Además, el uso de apilamientos nos permite obtener mejores tiempos de corrida para redes esparcidas. El uso del apilamiento-d del algoritmo de Dijkstra con $d = \text{máx}\ \{2, [m/n]\}$ corre en un tiempo de $O(m \log_d n)$ y fue hecho por Johnson (1977). El uso del apilamiento de Fibonacci hecho por Fredman y Tarjan (1984) corre en un tiempo de $O(m + n \log n)$. Johnson (1982) sugirió el uso del algoritmo de Dijkstra $O(m \log \log C)$ basado en un trabajo anterior de Boas, Kaas y Sijlstra (1977). El algoritmo escalable de Gabow (1985) es otro eficiente algoritmo de ruta más corta. [18]

Edsger Wybe Dijkstra
(Róterdam, Países Bajos, 11 de mayo de 1930 - Nuenen, Países Bajos, 6 de agosto de 2002) fue un científico de la computación de los Países Bajos.

Poco después de su muerte en el 2002, recibió la distinción ACM *PODC Influential Paper Award* en computación distribuida por su trabajo en la auto-estabilización en programas computacionales. Este premio fue renombrado a *Premio Dijkstra* el siguiente año en su honor.

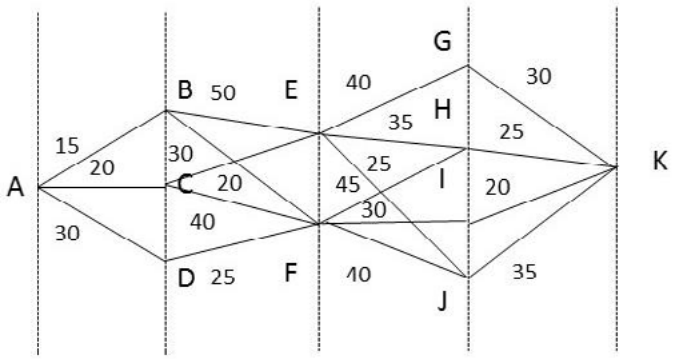
[18] "Richard Bellman", 2015. Recuperado de: http://es.wikipedia.org/wiki/Richard_Bellman

Dijkstra estudió física teórica en la Universidad de Leiden. Trabajó como investigador para *Burroughs Corporation* a principios de los años 1970. En la Universidad de Texas en Austin, Estados Unidos, ocupó el *Schlumberger Centennial Chair in Computer Sciences*. Se retiró en 2000.

Entre sus contribuciones a las ciencias de la computación está la solución del problema del camino más corto, también conocido como el algoritmo de Dijkstra, la notación polaca inversa y el relacionado algoritmo *shunting yard*, *THE multiprogramming system*, el algoritmo del banquero y la construcción del semáforo para coordinar múltiples procesadores y programas. Otro concepto debido a Dijkstra, en el campo de la computación distribuida, es el de la auto-estabilización, una vía alternativa para garantizar la confiabilidad del sistema. El algoritmo de Dijkstra es usado en la ruta más corta primero (SPF) que es usado en el protocolo de enrutamiento *Open Shortest Path First* (OSPF). También se le debe la autoría de la expresión «Crisis del software», aparecida en su libro The Humble Programmer y usada ampliamente en la famosa reunión de la OTAN de 1968 sobre desarrollo del software. Recibió el Premio Turing en 1972. [19]

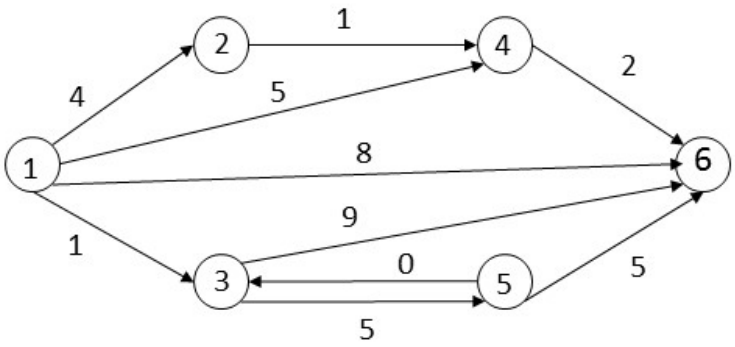
2.12. EJERCICIOS PROPUESTOS

1) Encuentre la ruta más corta en la siguiente red.

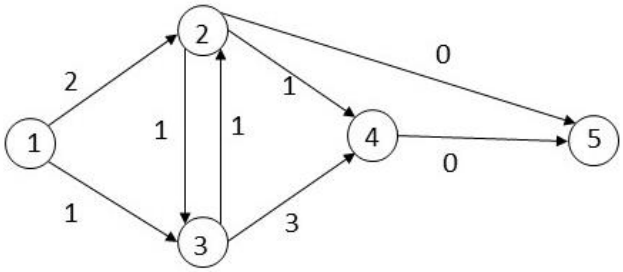


2) Localice la ruta más corta del nodo 1 al nodo 6 usando PD y verifique la solución con algún otro método de redes o PLE.

[19] "Edsger Dijkstra", 2014. Recuperado de: http://es.wikipedia.org/wiki/Edsger_Dijkstra



3) Halle la ruta más corta del nodo 1 al nodo 5 usando PD y verifique la solución como en el ejercicio 1.



CAPÍTULO 3

PROBLEMAS CLÁSICOS DE OPTIMIZACIÓN

3.1. INTRODUCCIÓN

En los capítulos anteriores, se introdujeron conceptos y algunos modelos sencillos de PD (Programación Dinámica), también, se mostraron otros métodos para resolverlos a modo de comparación y validación de los modelos de PD.

El objetivo de este capítulo es resolver algunos de los modelos clásicos de la optimización, usando modelos de PD; por supuesto, se harán algunas comparaciones cuando esto sea posible y se mostrarán algunos casos donde no se tiene linealidad en las ecuaciones recursivas, el caso de las cadenas de Márkov, aunque se consideren modelos estocásticos, también, se exponen aquí a través de un ejemplo sencillo.

3.2. EL PROBLEMA DE LA MOCHILA

Muchos problemas industriales se pueden formular como problemas de tipo mochila, por ejemplo, problemas de cargo fijo, selección de proyectos, corte en inventarios, control de presupuestos, etc. La versión más popular del problema contiene solo una restricción lineal, pero, casi cualquier problema lineal entero y muchos otros problemas combinatorios se pueden reducir a él.

También, el problema de la mochila se presenta como un subproblema en varios algoritmos de la PLEP (Programación Lineal Entera Pura).

Del problema de la mochila hay muchas versiones distintas, en nuestro caso, consideraremos el problema de la mochila 0 – 1 que se expresa de la siguiente manera.

$$\text{maximizar } z = \sum_{i=1}^{n} p_i x_{ij}$$

sujeta a

$$\sum_{i=1}^{n} w_i x_{ij} \le W_j \qquad j = 1,...,m$$

$$\sum_{i=1}^{n} c_i x_i \le 1 \qquad i = 1,...,n$$

$$x_{ij} = 0,1 \qquad i = 1,...,n$$

$$j = 1,...,m$$

Donde p_i, w_i (i = 1, 2,..., n) y W son números enteros. En otros términos, suponga que se tiene que llenar una mochila con diferentes objetos con un beneficio p_i y peso w_i sin exceder un peso total dado W. El problema consiste en encontrar una asignación factible de objetos para que el valor total de los objetos en la mochila sea el máximo. Cada artículo que puede ir en la mochila tiene un tamaño y un beneficio asociado. La mochila tiene una capacidad máxima. ¿Qué se debe llevar en la mochila para maximizar el beneficio total?

El problema de la mochila 0 – 1 es un caso especial del problema de la mochila acotado que se define igual que el anterior y solo difiere en la restricción binaria, porque, en este caso se tiene $0 \le x_i \le b_i$, donde x_i es entero, i = 1,2,..., n.

En el problema de la mochila acotado, la mochila se puede llenar con, a lo más, b_i objetos del tipo i. En el problema general de la mochila, que a veces se denomina no acotado, la restricción se relaja a $x_i \ge 0$, x_i entero, i = 1,2,..., n.

Sin pérdida de generalidad, podemos suponer que los parámetros p_i, w_i y W en los problemas anteriores satisfacen las condiciones:

p_i y w_i son enteros positivos con $w_i \le W$ i = 1, 2, ..., n y

$$\sum_{i=1}^{n} w_i x_i \le W$$

Los problemas de la mochila 0 – 1 acotado y generalizado, a veces, se conocen como problemas unidimensionales, donde el 1 se refiere al número de restricciones lineales en el problema. Los más populares son los de valor independiente (cuando w_i=p_i).

Los problemas tipo mochila unidimensionales se pueden generalizar de muchas maneras, la generalización más natural es aquella en la cual, los objetos que tenemos que guardar pueden ponerse en m mochilas, cada una con capacidad W_j (j = 1,2,..., m). Sea x_{ij} una variable 0 – 1, tal que, x_{ij} = 1, si el i-ésimo objeto se asigna a la j-ésima mochila. El problema 0 – 1 multi-mochila, se expresa como:

$$\text{maximizar } z = \sum_{i=1}^{n}\sum_{j=1}^{m} p_i x_{ij}$$

sujeta a

$$\sum_{i=1}^{n} w_i x_{ij} \le W_j \qquad j = 1,...,m$$

$$\sum_{j=1}^{m} x_{ij} \le 1 \qquad i = 1,...,n$$

$$x_{ij} = 0,1 \qquad i = 1,...,n$$

$$j = 1,...,m$$

La primera restricción significa que en una restricción factible de objetos no se sobrecarga ninguna mochila y la segunda que cada objeto puede asignarse, a lo más, a una mochila. De esta forma, pueden formularse aquí las versiones acotada y no acotada de este problema.

Si antes de poner los objetos en una mochila se tienen que comprar a un costo c_i para el i-ésimo objeto y se tiene una cantidad limitada de dinero C, entonces, se tiene el problema de asignar a la mochila objetos que no pesen más de W y no cuesten más de C, así, el problema se convierte en:

$$\text{maximizar } z = \sum_{i=1}^{n} p_i x_{ij}$$

sujeta a

$$\sum_{i=1}^{n} w_i x_{ij} \le W_j \qquad j = 1,...,m$$

$$\sum_{i=1}^{n} c_i x_i \le 1 \qquad i = 1,...,n$$

$$x_{ij} = 0,1 \qquad i = 1,...,n$$

$$j = 1,...,m$$

En general, se pueden introducir muchas restricciones al asignar objetos a una mochila, entonces, el problema se convierte en un problema de mochila multidimensional, donde evidentemente se pueden considerar los casos acotado y no acotado.

Los problemas de tipo mochila a menudo se refieren como problemas de cargo, pero de hecho, el problema de cargo estándar consiste en asignar objetos dados con volúmenes conocidos a cajas que tienen restricciones de capacidad, con el objeto de minimizar el número de cajas usadas. Sea k_j la capacidad de la j-ésima caja y w_i el volumen del i-ésimo objeto. El problema de cargo se define como sigue:

$$\text{minimizar } z = \sum_{j=1}^{m} y_j$$

$$\text{sujeta a}$$

$$\sum_{i=1}^{n} w_i x_{ij} \leq k_j y_j \quad j = 1,...,m$$

$$\sum_{j=1}^{m} x_{ij} \leq 1 \qquad i = 1,...,n$$

$$x_{ij}, y_j = 0,1 \qquad i = 1,...,n$$

$$j = 1,...,m$$

En un cargo factible, tenemos $x_{ij} = 1$, si el i-ésimo objeto se pone en la j-ésima caja y $y_j = 1$, sí se usa la j-ésima caja.

El problema de la mochila acotado o no acotado, también, puede representar el problema de cortar objetos unidimensionales (por ejemplo, la longitud de un papel, vidrio y acero) en piezas pequeñas de valores y tamaños dados para maximizar el valor total de las piezas o minimizar el material que sobra.

Este pequeño panorama de las posibles generalizaciones y modificaciones del problema de la mochila 0 – 1 indica la variedad de los problemas del mundo real que se pueden modelar por variantes, que provienen de la familia de problemas de tipo mochila.

El problema de la mochila es un problema clásico de la PLE (Programación Lineal Entera) para la cual existen algoritmos de solución propios de la PLE, ya sean exactos o heurísticos.

Ejemplo 3.1. (Taha, 2004).

A modo de ejemplo, supongamos que hay tres artículos como se muestra en la tabla 3.1 y presuma que la capacidad de la mochila es 5.

TABLA 3.1. Artículos para la mochila

Artículo (*ij*)	**Peso (w_i)**	**Beneficio (v_i)**
1	2	65
2	3	80
3	1	30

En este ejemplo pequeño, la solución óptima se puede obtener por inspección, sin embargo, es importante hacer notar que en un problema real se tienen muchos más artículos y, por lo tanto, la solución ya no es tan obvia.

Primero, consideramos el problema general de N artículos. Si k_j es el número de unidades del artículo *i,* el problema queda como:

$$\text{maximizar } v_1k_1 + v_2k_2 + \ldots + N_kN_k$$

sujeta a

$$w_1k_1 + w_2k_2 + \ldots + w_Nk_N \leq W$$
$$k_i \text{ entero no negativo}$$

El modelo de PD se construye considerando los siguientes tres elementos básicos:

1) La *etapa j* está representada por el artículo j, $j = 1,2,\ldots N$.

2) El *estado* y_j, en la etapa j, es el peso total asignado a las etapas $j, j + 1,\ldots,N$; $y_1 = W$ y $y_j = 0, 1,\ldots,W$ para $j = 2, 3,\ldots,N$.

3) La *alternativa* k_j, en la etapa j, es el número de unidades del artículo j. El valor de k_j puede ser tan chico como cero o tan grande como $[W/w_j]$, donde este cociente es el máximo entero menor que se obtiene de (W/w_j).

Entre este problema y el visto en el capítulo 2 de inversión de capital (2.4) existe una gran similitud, ya que, ambos son de asignación de recursos.

Las etapas representan los artículos: luego se tienen tres etapas $j = 1, 2, 3$. El estado y_i, en la etapa j, representa el peso total de los artículos j, más todos los artículos que se agregarán posteriormente a la mochila. La decisión en la etapa j es: ¿Cuántos artículos j se deben poner en la mochila? Sea ese valor k_j.

Luego, se tienen las siguientes fórmulas recursivas:

sea

$f_j(y_j) =$ el valor óptimo de las etapas $j, j + 1, ..., N$ dado el estado y_j.

Las ecuaciones recursivas de retroceso son:

$$f_N(y_N) = \max_{\substack{kn=0,1,...,\lfloor yN/wN \rfloor \\ yN=0,1...W}} \{v_N k_N\}$$

$$f_j(y_j) = \max_{\substack{kj=0,1,...,\lfloor yj/wj \rfloor \\ yj=0,1,...,W}} \{v_j k_j + f_{j+1}(y_j - w_j k_j)\} \qquad j = 1, 2, ..., N-1$$

En particular se tiene:

ETAPA 3

$$f_3(y_3) = \max_{k_3} \{30k_3\}, \qquad \max\ k_3 = \left\lfloor \frac{5}{1} \right\rfloor = 5$$

TABLA 3.2. Cálculos para la etapa 3

	30 k_3						**Solución óptima**	
	$k_3 = 0$	**1**	**2**	**3**	**4**	**5**	**$f_3(y_3)$**	**k_3***
y_3	**$v_3k_3 = 0$**	**30**	**60**	**90**	**120**	**150**		
0	0	--	--	--	--	--	0	0
1	0	30	--	--	--	--	30	1
2	0	30	60	--	--	--	60	2
3	0	30	60	90	--	--	90	3
4	0	30	60	90	120	--	120	4
5	0	30	60	90	120	150	150	5

ETAPA 2

$$f_2(y_2) = \max_{k_2}\{80k_2 + f_3(y_2 - 3k_2)\}, \qquad \text{máx } k_2 = \left\lfloor \frac{5}{3} \right\rfloor = 1$$

TABLA 3.3. Cálculos para la etapa 2

	80 k_2 +$f_3(y_2 - 3k_2)$		**Solución óptima**	
	$k_2 = 0$	**1**	**$f_2(y_2)$**	**k_2***
y_2	**$v_2k_2 = 0$**	**80**		
0	0 + 0 = 0	--	0	0
1	0 + 30 = 30	--	30	0
2	0 + 60 = 60	--	60	0
3	0 + 90 = 90	80 + 0 = 80	90	0
4	0 + 120 = 120	80 + 30 = 110	120	0
5	0 + 150 = 150	80 + 60 = 140	150	0

ETAPA 1

$$f_1(y_1) = \max_{k_1}\{65k_1 + f_2(y_1 - 2k_1)\}, \qquad \text{máx } k_1 = \left\lfloor \frac{5}{2} \right\rfloor = 2$$

TABLA 3.4. Cálculos para la etapa 1

	65 k_1 +$f_2(y_1 - 2k_1)$			**Solución óptima**	
	$k_1 = 0$	**1**	**2**	**$f_1(y_1)$**	**k_1***
y_1	**$v_1k_1 = 0$**	**65**	**130**		
0	0 + 0 = 0	--		0	0
1	0 + 30 = 30	--		30	0
2	0 + 60 = 60	65 + 0 = 65		65	1
3	0 + 90 = 90	65 + 30 = 90		90	0 y 1
4	0 + 120 = 120	65 + 60 = 125	130 + 0 = 130	130	2
5	0 + 150 = 150	65 + 90 = 155	130 + 30 = 160	160	2

En las tablas, se puede ver que dada $y_1 = W = 5$, la solución óptima es $(k_1^*, k_2^*, k_3^*) = (2, 0, 1)$ con un valor total de 160. Se puede observar que en la etapa 1 es suficiente construir la tabla

para $y_1 = 5$, sin embargo, al construir toda la tabla, se pueden contemplar cambios para W lo cual es una especie de análisis de sensibilidad dada por la PD.

Planteamiento alternativo

Para formular el problema de la mochila existe otra forma. Esto ilustrará cuán arbitraria puede ser la definición de etapa, estado y decisiones. Además, da una visión de la flexibilidad de las reglas en PD.

Se va a trabajar con una recursividad de avance. Para el problema de la mochila, se indexarán las etapas por w el peso que se lleva. La decisión es determinar el último artículo agregado para llegar al peso w. Luego, hay solo un estado por etapa. Sea $g(w)$ el máximo beneficio que se puede ganar de una mochila con un peso w. Continuamos usando b_j y w_j como el peso y beneficio, respectivamente, para el artículo j. Lo siguiente relaciona $g(w)$ con valores de g calculados, previamente:

$$g(w) = \max_j \{ b_j + g(w - wj) \}$$

Intuitivamente, para tener una mochila con un peso de w, se debe terminar agregando algún artículo. Si se agrega el artículo j, terminaremos con una mochila de tamaño $\varpi - \varpi_j$ disponible para ser llenada. Para ilustrar esto, usamos el ejemplo anterior:

$g(0) = 0$
$g(1) = 30$
Agrega artículo 3
$g(2) =$ máx $\{65 + g(0) = 65, 30 + g(1) = 60\} = 65$
Agrega artículo 1
$g(3) =$ máx $\{65 + g(1) = 95, 80 + g(0) = 80, 30 + g(2) = 95\} = 95$
Agrega artículo 1 o 3
$g(4) =$ máx $\{65 + g(2) = 130, 80 + g(1) = 110, 30 + g(3) = 125\} = 130$
Agrega artículo 1
$g(5) =$ máx $\{65 + g(3) = 160, 80 + g(2) = 145, 30 + g(4) = 160\} = 160$
Agrega artículo 1

Esto da un máximo de 160 que se gana al agregar 2 unidades del artículo 1 y 1 del artículo 3.

Si resolvemos el problema de la mochila usando PLE, entonces, el planteamiento es el siguiente:

Si v_i = al número de artículos del tipo i por cargar en la mochila, entonces, buscamos maximizar este beneficio sujeta a la restricción de capacidad de la mochila, las variables deben ser enteras, pues son artículos por llevar.

$$\text{máximizar } z = 65v_1 + 80\ v_2 + 30v_3$$

$$\text{sujeta a}$$
$$2v_1 + 3v_2 + v_3 \leq 5$$
$$v_i \geq 0 \text{ y enteros}$$

Entonces, resolviendo con LINGO se tiene el siguiente resultado:

$$Z = 160 \qquad v_1 = 2 \qquad v_2 = 0 \qquad v_3 = 1$$

Por lo que, se puede afirmar que el modelo de PD queda validado.

3.3. El PAV (Problema del Agente Viajero) o TSP (Traveling Salesman Problem)

El PAV es un problema clásico en optimización combinatoria y uno de los más viejos en la optimización de redes que ha atraído mucho la atención en los últimos cuarenta años. Por ejemplo, cuenta con numerosas aplicaciones en problemas de distribución donde se tiene que el almacén central de una compañía desea distribuir materia prima a cada una de sus sucursales a un costo mínimo o con problemas de carteros, ¿Cómo debe el cartero atravesar la ciudad para repartir la correspondencia minimizando el tiempo de viaje? O considere otra situación: una fábrica produce n artículos y se debe cambiar el montaje siempre que cambie la producción de artículos, conociendo el costo del montaje entre los artículos, se desea encontrar una secuencia de la producción óptima.

El problema es muy sencillo plantearlo y lo haremos a continuación:

Un agente viajero debe visitar cada ciudad de su territorio exactamente una vez y regresar a su punto de partida. Dado el costo del viaje entre cada par de ciudades, ¿Cómo debe planear su itinerario, de tal forma que, visite cada ciudad una sola vez y el costo total del circuito sea el mínimo?

En términos de la teoría de redes, el problema consiste en encontrar en una gráfica completa de n nodos, un circuito de peso mínimo de longitud n. Recuérdese que una gráfica completa es

aquella en la cual, entre cualquier par de nodos, existe un arco que los une. Por ejemplo, considere un problema con cinco ciudades como se muestra en la figura 3.1.

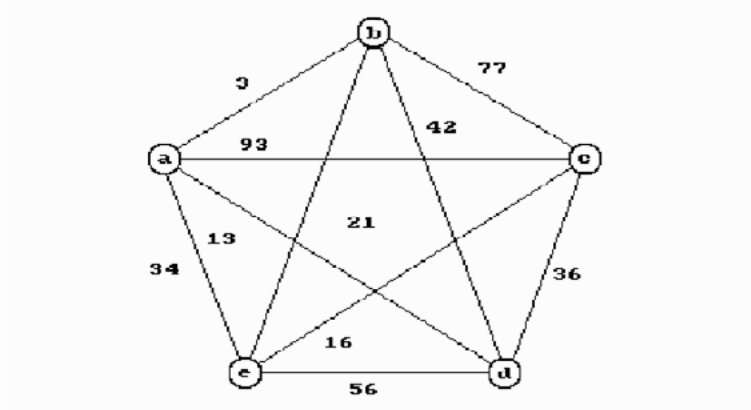


FIGURA 3.1. El PAV para 5 ciudades

El peso del circuito o costo de la ruta (a, b, c, d, e, a) es 206, mientras que el circuito de peso mínimo es (a, b, e, c, d, a) con un peso de 89.

En este problema hay distintas variantes, pero primero, comenzaremos por precisar los conceptos:

Sea $W = [w_{ij}]$ el peso de la matriz de la gráfica, entonces, se puede tener que los pesos en los arcos puedan no ser simétricos $w_{ij} = w_{ji}$, en cuyo caso, estamos tratando con una gráfica dirigida y resolviendo un PAV asimétrico. O algunas veces, la gráfica dada no es completa, lo que significa que existen parejas de nodos que no están directamente conectadas por arcos. No obstante, toda gráfica completa siempre tiene circuitos de longitud n, una que no lo es puede no tener circuitos de longitud n. Por ejemplo, la gráfica en la figura 3.2 no tiene circuitos de longitud 5, en tal caso, el PAV no tiene solución.

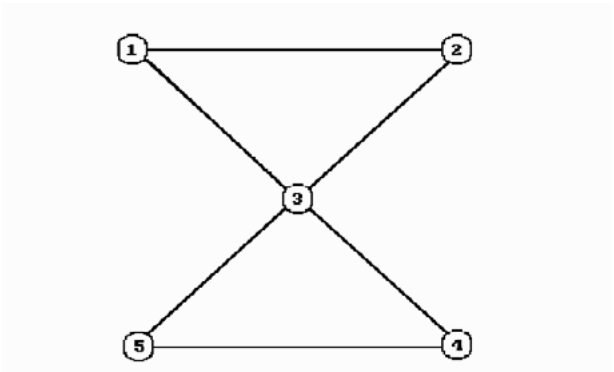


FIGURA 3.2. Gráfica no completa

El problema de determinar, si una gráfica dada con n nodos tiene un circuito de longitud n se conoce como un problema de circuito hamiltoniano originado en un juego inventado por el matemático irlandés Sir William Rowan Hamilton en 1859. Trataba sobre un viaje alrededor del mundo, el cual se representaba en forma simplificada por un dodecaedro (poliedro de 12 caras pentagonales con 20 vértices) y se requiere que se pase una sola vez por cada vértice o ciudad, usando solamente las caras del dodecaedro y se regrese al punto inicial. Este juego es equivalente a buscar un circuito hamiltoniano en la figura 3.3.

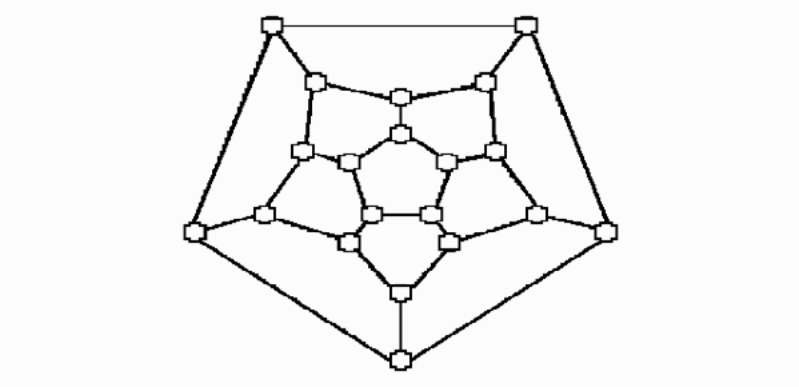


FIGURA 3.3. El juego del icosiano de Hamilton

Un circuito que pasa por cada nodo en la gráfica es llamado un circuito hamiltoniano. Si el agente viajero no quiere regresar al nodo inicial, después de visitar todos los nodos en la gráfica exactamente una vez, el problema se convierte, entonces, en encontrar la trayectoria de peso mínimo de longitud $(n - 1)$ y no es un circuito de longitud n.

Se podrá notar un cambio muy drástico, si al resolver este problema se permite que el agente viajero visite un nodo más de una vez (si esto es más barato). Por ejemplo, en la figura 3.4 se puede ver que, si se permite que visite un nodo más de una vez la ruta óptima (a, b, c, d, e, c, a), que, además, no es un circuito, tiene un peso de 60, pero, si no se permite que visite un nodo más de una vez, la ruta (a, b, c, d, e, a) es la única solución posible con un peso de 140.

La condición bajo la cual, al menos uno de los circuitos hamiltonianos tenga un circuito de peso mínimo, es que los pesos satisfagan la desigualdad del triángulo $w_{ij} \leq w_{ik} + w_{kj}$.

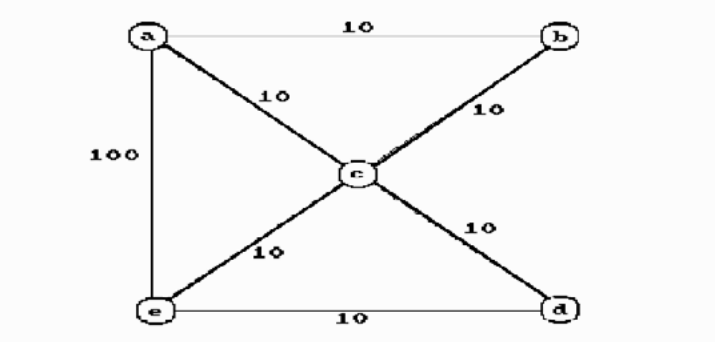


FIGURA 3.4. Trayectoria de peso mínimo en una gráfica no completa

Una forma de resolver el PAV es haciendo una enumeración exhaustiva de todas las soluciones posibles y, entre ellas, escoger la mejor. Hay $n!$ permutaciones de los n nodos, pero, solo $(n-1)!$ de ellas son circuitos hamiltonianos distintos (en una gráfica dirigida completa), porque, podemos fijar alguno de los n nodos y, posteriormente, hay $(n-1)!$ formas de acomodar los $(n-1)$ nodos restantes en el circuito. En caso de que la red dada no sea una red dirigida, entonces, hay $(n-1)!/2$ circuitos distintos.

Teóricamente, el problema se puede resolver generando los $(n-1)!$ circuitos y comparando sus pesos, sin embargo, como método este es muy ineficiente. Por ejemplo, dada una gráfica de 20 nodos se tienen $19! > 10^{17}$ circuitos, por lo que, se requieren años de cálculos continuos para resolver el problema.

Como se podrá ver, el PAV es uno de los problemas de optimización combinatoria para los cuales no se conoce un algoritmo eficiente de tiempo polinomial. Todos los algoritmos conocidos requieren de un tiempo de computadora exponencial en el número n de ciudades.

Para salvar esta dificultad computacional hay dos formas de resolverla:

1) Usando técnicas refinadas, tales como, ramificación y acotamiento o PD que reducen drásticamente el efecto que se da en la enumeración exhaustiva. Tales técnicas refinadas enumerativas son buenas para encontrar una solución óptima, pero, en el peor de los casos, si se tiene un problema muy grande se puede requerir un número exponencial de cálculos que se hacen prohibitivamente grandes.

2) Empleando métodos de solución aproximados, pero, rápidos (en tiempo polinomial), que no producen una solución óptima, pero, sí soluciones subóptimas que estén, aceptablemente, cercanas a la óptima.

*Ejemplo 3.2***.** El PAV consiste en visitar un conjunto de ciudades con una mínima distancia. Por ejemplo, un político comienza en New York y tiene que visitar Miami, Dallas y Chicago antes de volver a New York. ¿Cómo podría minimizar la distancia recorrida? Las distancias se muestran en la tabla 3.5.

TABLA 3.5. Datos para el PAV

	Nueva York	**Miami**	**Dallas**	**Chicago**
Nueva York	--	1 334	1 559	809
Miami	1 334	--	1 343	1 397
Dallas	1 559	1 343	--	921
Chicago	809	1 397	921	--

La dificultad está una vez más en definir las etapas, estados y decisiones. Una forma natural es que la etapa *t* represente las ciudades *t* visitadas y la decisión es dónde ir a continuación. ¿Cuáles son los estados? Supongamos que elegimos la ciudad en la que estamos como un estado. No se podría tomar la decisión de dónde seguir a continuación, ya que no sabemos lo que se ha realizado anteriormente. En consecuencia, el estado debe incluir información acerca de todas las ciudades visitadas, más la ciudad a donde queremos terminar. Luego, un estado estará representado por el par (*i*, *S*) donde *S* es el conjunto de *t* ciudades ya visitadas e *i* es la última ciudad visitada (así, *i* debe estar en *S*). Asignamos números a cada ciudad: Nueva York = 1, Miami = 2, Dallas = 3, Chicago = 4. Usando recursividad se tiene:

ETAPA 3

Los cálculos en la etapa 3 representan: *i* la última ciudad visitada y *t* las ciudades ya visitadas.

$$f_3\{2,\{2,3,4\}\} = 1\ 334$$
$$f_3\{3,\{2,3,4\}\} = 1\,559$$
$$f_3\{4,\{2,3,4\}\} = \ \ 809$$

Para las otras etapas, la recursividad es:

$$f_t(i,S) = \min_{j\neq 1,\, j\notin S}\{c_{ij} + f_{t+1}(j, S\cup j)\}$$

ETAPA 2

$$f_2(2,\{3,4\}) = \min_{j\neq 1,\, j\notin S}\{c_{23} + f_3(3), c_{24} + f_3(4)\} = \{2\,902,\ 2\,206\ \} = 2\,206$$
$$f_2(3,\{2,4\}) = \min_{j\neq 1,\, j\notin S}\{c_{32} + f_3(2), c_{34} + f_3(4)\} = \{2\,677,\ 1\,730\} = 1\,730$$
$$f_2(4,\{2,3\}) = \min_{j\neq 1,\, j\notin S}\{c_{42} + f_3(2), c_{43} + f_3(3)\} = \{2\,731,\ 2\,480\} = 2\,480$$

ETAPA 1

$$f_1(2,\{3,4\}) = \min_{j\neq 1,\, j\notin S}\{c_{23} + f_2(3), c_{24} + f_2(4)\} = \{3\,073,\ 3\,877\} = 3\,073$$

No se efectúan los cálculos para los nodos 3 y 4, porque, se repiten los costos y se corre el riesgo de formación de subcircuitos.

Finalmente, se cierra el circuito de 2 a 1 y siguiendo los cálculos recursivos de regreso se tiene:

$$1 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$$

con un costo de $4 407.00.

Un aspecto importante de este problema es la NP-completitud (puede revisar el anexo B sobre complejidad computacional para documentarse más sobre este concepto). El espacio de búsqueda es tan grande que llega a ser imposible encontrar una solución, aún para problemas de tamaño pequeño. Por ejemplo, suponga un problema con 20 ciudades, el número de estados en la etapa 10 es más de un millón. Para 30 ciudades, el número de estados en el etapa 15 es más de un billón. Y para 100 ciudades, el número de estados en el etapa 50 es más de 5 000 000 000 000 000 000 000 000 000 000. Este no es un problema fácil de resolver, aún, con una configuración computacional ideal.

El ejemplo anterior, se conoce como un problema de *agente viajero simétrico,* ya que, las distancias de cualquier ciudad o nodo *i* a otro nodo *j* es igual que la distancia de *j* a *i*. Esto es, $d_{ij}=d_{ji.}$

En los dos ejemplos siguientes, tenemos el caso de un agente viajero no simétrico.

Ejemplo 3.3. Considere el problema de cuatro ciudades con las distancias de la tabla 3.6[20].

TABLA 3.6. Distancias del PAV

Ciudades distancias	**1**	**2**	**3**	**4**
1	-	2	9	10
2	1	-	6	4
3	15	7	-	8
4	6	3	12	-

[20] "Example of Dynamic Programming Algorithm for the TSP". Recuperado de: http://searches.uninstallmaster.com/search/web?channel=unknown&type=other&q=Example+of+Dynamic+Programming+Algorithm+for+the+TSP http://www.mafy.lut.fi/study/DiscreteOpt/tspdp.pdf

La solución con PD consiste en definir las etapas por el número de ciudades visitadas *t,* comenzando con un conjunto de cero ciudades visitadas ($S = Ø$) 4 se tienen las ecuaciones recursivas:

$$g_t(i, S) = \text{mín}\ \{c_{ij} + g_{t-1}(i, S)\} \text{ donde } k \text{ es la cardinalidad de } S$$

El recorrido lo comenzamos desde el nodo 1 y obtenemos sus distancias a las ciudades 2, 3 y 4 en la etapa 0.

ETAPA 0

$g(2, Ø) = c_{21} = 1$
$g(3, Ø) = c_{31} = 15$
$g(4, Ø) = c_{41} = 6$

ETAPA 1

Para $k = 1$ se consideran conjuntos con 1 elemento:

Conjunto {2}:
$g_1\{3,\{2\}\} = c_{32} + g(2, Ø) = c_{32} + c_{21} = 7 + 1 = 8$ $\quad p(3, \{2\}) = 2$
$g_1\{4,\{2\}\} = c_{42} + g(2, Ø) = c_{42} + c_{21} = 3 + 1 = 4$ $\quad p(4, \{2\}) = 2$

Conjunto {3}:
$g_1\{2,\{3\}\} = c_{23} + g(3, Ø) = c_{23} + c_{31} = 6 + 15 = 21$ $\quad p(2, \{3\}) = 3$
$g_1\{4,\{3\}\} = c_{43} + g(3, Ø) = c_{43} + c_{31} = 12 + 15 = 27$ $\quad p(4, \{3\}) = 3$

Conjunto {4}:
$g_1\{2,\{4\}\} = c_{24} + g(4, Ø) = c_{24} + c_{41} = 4 + 6 = 10$ $\quad p(2, \{4\}) = 4$
$g_1\{3,\{4\}\} = c_{34} + g(4, Ø) = c_{34} + c_{41} = 8 + 6 = 14$ $\quad p(3, \{4\}) = 4$

ETAPA 2

$K = 2$, considere conjuntos de 2 elementos:

Conjunto {2, 3}:
$g_2(4, \{2, 3\}) = \text{mín}\{c_{42} + g(2, \{3\}), c_{43} + g(3, \{2\})\} = \text{mín}\{3 + 21, 12 + 8\} = \text{mín}\{24, 20\} = 20$
$p(4, \{2, 3\}) = 3$

Conjunto {2, 4}

$g_2(3, \{2, 4\}) = \text{mín}\{c_{32} + g(2, \{4\}), c_{34} + g(4, \{2\})\} = \text{mín}\{7 + 10, 8 + 4\} = \text{mín}\{17, 12\} = 12$

$p(4, \{2, 3\}) = 4$

Conjunto {3, 4}

$g_2(2, \{3, 4\}) = \text{mín}\{c_{23} + g(3, \{4\}), c_{24} + g(4, \{3\})\} = \text{mín}\{6 + 14, 4+27\} = \text{mín}\{20, 31\} = 20$

$p(2, \{3, 4\}) = 3$

ETAPA 3

Longitud del recorrido óptimo:

$F = g_3(1, \{2, 3, 4\} = \text{mín}\{c_{12} + g(2, \{3, 4\}), c_{13} + g(3, \{2, 4\}), c_{14} + g(4, \{2, 3\})\} =$

$= \text{mín}\{2 + 20, 9 + 12, 10 + 20\} = \text{mín}\{22, 21, 30\} = 21$

Sucesor del nodo 1 : $p(1, \{2, 3, 4\}) = 3$

Sucesor del nodo 3 : $p(3, \{2, 4\}) = 4$

Sucesor del nodo 4 : $p(4, \{2\}) = 2$

Recorrido óptimo: 1→ 3 → 4 →2 → 1

Ejemplo 3.4. (Dreyfus y Law, 1977).

Para el siguiente PAV, considere la matriz de distancias de la red que consiste de 5 ciudades:

TABLA 3.7. Matriz de distancias para el PAV

	1	**2**	**3**	**4**	**5**
1	-	3	1	5	4
2	1	-	5	4	3
3	5	4	-	2	1
4	3	1	3	-	3
5	5	2	4	1	-

Se consideran:

$$N_j = \{2, 3, \ldots, j - 1, j + 1, \ldots, N\}$$

S es un subconjunto de N_j que contiene i miembros, entonces, se define el óptimo de la función $f_i(j, S)$ como:

$f_i(j, S)$ = la longitud de la ruta más corta de la ciudad 1 a la ciudad j a través del conjunto i de ciudades intermedias S (la variable etapa i indica el número de ciudades en S).

Las ecuaciones recursivas se definen como:

$$f_i(j,S) = \min_{k \in S} \left\{ f_{i-1}\left(k, S - \{k\}\right) + d_{kj} \right\} \qquad i = 1,2,...,N-2; j \neq 1;\ S \subseteq N_j$$

La condición de frontera es:

$$f_0(j,-) = d_{1j}$$

Además, la longitud del recorrido más corto está dada por:

$$\min_{j=2,,3,...,N} \left\{ f_{N-2}(j, N_j) + d_{j1} \right\}$$

Intuitivamente, podemos explicar estas ecuaciones considerando que queremos calcular $f_i(j, S)$, entonces, considere la ruta más corta de la ciudad 1 a la ciudad j, vía el conjunto S que contiene la ciudad k como la ciudad inmediatamente precedente a j. Entonces, como las ciudades en $S - \{k\}$ se deben visitar en un orden óptimo, la longitud de esta trayectoria es:

$$f_{i-1}(k, S - \{k\}) + d_{kj}$$

Más aún, ya que somos libres de escoger la ciudad k de manera óptima, es claro que la ecuación recursiva es correcta. Entonces, para calcular la longitud del recorrido más corto comenzamos calculando $f_1(j, S)$ para todo par (j, S) de los valores de f_0. Así, se calcula $f_2(j, S)$ para todo par (j, S) de los valores de f_1. Continuamos de la misma manera hasta que $f_{N-2}(j, S)$ se ha calculado para toda ciudad j. El recorrido mínimo se puede obtener a través de la información para la ciudad k en cada etapa y cada estado.

En este caso, tenemos que los cálculos son los siguientes (la política óptima se da entre paréntesis):

Condiciones de frontera

$f_0(2, \_) = d_{12} = 3$ (1)
$f_0(3, \_) = d_{13} = 1$ (1)
$f_0(4, \_) = d_{14} = 5$ (1)
$f_0(5, \_) = d_{15} = 4$ (1)

ETAPA 1

$f_1(2, \{3\}) = f_0(3, \_) + d_{32} = 1 + 4 = 5$ (3)
$f_1(2, \{4\}) = f_0(4, \_) + d_{42} = 5 + 1 = 6$ (4)
$f_1(2, \{5\}) = f_0(5, \_) + d_{52} = 4 + 2 = 6$ (5)
$f_1(3, \{2\}) = f_0(2, \_) + d_{23} = 3 + 5 = 8$ (2)
$f_1(3, \{4\}) = f_0(4, \_) + d_{43} = 5 + 3 = 8$ (4)
$f_1(3, \{5\}) = f_0(5, \_) + d_{53} = 4 + 4 = 8$ (5)
$f_1(4, \{2\}) = f_0(2, \_) + d_{24} = 3 + 4 = 7$ (2)
$f_1(4, \{3\}) = f_0(3, \_) + d_{34} = 1 + 2 = 3$ (3)
$f_1(4, \{5\}) = f_0(5, \_) + d_{54} = 4 + 1 = 5$ (5)
$f_1(5, \{2\}) = f_0(2, \_) + d_{25} = 3 + 3 = 6$ (2)
$f_1(5, \{3\}) = f_0(3, \_) + d_{35} = 1 + 1 = 2$ (3)
$f_1(5, \{4\}) = f_0(4, \_) + d_{45} = 5 + 3 = 8$ (4)

ETAPA 2

$f_2(2, \{3, 4\}) = \text{mín}\ \{f_1(3, \{4\}) + d_{32}, f_1(4,\{3\}) + d_{42}\} = \text{mín}\ \{8 + 4, 3 + 1\} = 4$ (4)
$f_2(2, \{3, 5\}) = \text{mín}\ \{f_1(3, \{5\}) + d_{32}, f_1(5,\{3\}) + d_{52}\} = \text{mín}\ \{8 + 4, 2 + 2\} = 4$ (5)
$f_2(2, \{4, 5\}) = \text{mín}\ \{f_1(4, \{5\}) + d_{42}, f_1(5,\{4\}) + d_{52}\} = \text{mín}\ \{5 + 1, 8 + 2\} = 6$ (4)
$f_2(3, \{2, 4\}) = \text{mín}\ \{f_1(2, \{4\}) + d_{23}, f_1(4,\{2\}) + d_{43}\} = \text{mín}\ \{6 + 5, 7 + 3\} = 10$ (4)
$f_2(3, \{2, 5\}) = \text{mín}\ \{f_1(2, \{5\}) + d_{23}, f_1(5,\{2\}) + d_{53}\} = \text{mín}\ \{6 + 5, 6 + 4\} = 10$ (5)
$f_2(3, \{4, 5\}) = \text{mín}\ \{f_1(4, \{5\}) + d_{43}, f_1(5,\{4\}) + d_{53}\} = \text{mín}\ \{5 + 3, 8 + 4\} = 8$ (4)
$f_2(4, \{2, 3\}) = \text{mín}\ \{f_1(2, \{3\}) + d_{24}, f_1(3,\{2\}) + d_{34}\} = \text{mín}\ \{5 + 4, 8 + 2\} = 9$ (2)
$f_2(4, \{2, 5\}) = \text{mín}\ \{f_1(2, \{5\}) + d_{24}, f_1(5,\{2\}) + d_{54}\} = \text{mín}\ \{6 + 4, 6 + 1\} = 7$ (5)
$f_2(4, \{3, 5\}) = \text{mín}\ \{f_1(3, \{5\}) + d_{34}, f_1(5,\{3\}) + d_{54}\} = \text{mín}\ \{8 + 2, 2 + 1\} = 3$ (5)
$f_2(5, \{2, 3\}) = \text{mín}\ \{f_1(2, \{3\}) + d_{25}, f_1(3,\{2\}) + d_{35}\} = \text{mín}\ \{5 + 3, 8 + 1\} = 8$ (2)
$f_2(5, \{2, 4\}) = \text{mín}\ \{f_1(2, \{4\}) + d_{25}, f_1(4,\{2\}) + d_{45}\} = \text{mín}\ \{6 + 3, 7 + 3\} = 9$ (2)
$f_2(5, \{3, 4\}) = \text{mín}\ \{f_1(3, \{4\}) + d_{35}, f_1(4,\{3\}) + d_{45}\} = \text{mín}\ \{8 + 1, 3 + 3\} = 6$ (4)

ETAPA 3

$f_3(2, \{3, 4, 5\}) = \text{mín } \{f_2(3, \{4, 5\}) + d_{32}, f_2(4,\{3,5\}) + d_{42}, f_2(5,\{3,4\}) + d_{52}\} =$
$= \text{mín } \{8 + 4, 3 + 1, 6 + 2\} = 4 (4)$
$f_3(3, \{2, 4, 5\}) = \text{mín } \{6 + 5, 7 + 3, 9 + 4\} = 10 (4)$
$f_3(4, \{2, 3, 5\}) = \text{mín } \{4 + 4, 10 + 2, 8 + 1\} = 8 (2)$
$f_3(5, \{2, 3, 4\}) = \text{mín } \{4 + 3, 10 + 1, 9 + 3\} = 7 (2)$

Entonces, se tiene que:

$$\text{mín } \{f_3(j, (2, 3, 4, 5) - \{j\} + d_{j1}\} = \text{mín } \{4 + 1, 10 + 5, 8 + 3, 7 + 5\} = 5$$

El circuito hamiltoniano de menor longitud es 5 con las ciudades:

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 4 \rightarrow 2 \rightarrow 1$$

El planteamiento de PLE para el PAV tiene algunas variantes, una de ellas es modelarlo como un problema de asignación y después romper los subcircuitos considerando los que tengan menos ciudades.

El modelo de asignación es como sigue:

$$\text{minimizar } z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

sujeta a

$$\sum_{j=1}^{n} xij = 1 \quad i = 1,\ldots,n$$

$$\sum_{i=1}^{n} xij = 1 \quad j = 1,\ldots,n$$

$$xij = 0 \text{ o } 1$$

donde $c_{ij} = \infty$ para $i = j$ y $x_{ij} = 1$, si el agente viajero va de la ciudad i a la ciudad j.

Excepto, por el requerimiento de que la solución sea un circuito hamiltoniano, la formulación es un modelo de asignación y es resuelto con ramificación y acotamiento. Desafortunadamente, no hay una garantía de que la solución óptima del modelo de asignación será un circuito

hamiltoniano. Más bien, la solución nos presentará una serie de subcircuitos a partir de los cuales podremos construir dicho circuito:

1) Se determinan cotas superiores e inferiores en la función objetivo óptima del PAV, es decir, dado z^* es el valor óptimo asociado al circuito, entonces, z^L y z^U se determinan, tales que, $z^L \leq z^* \leq z^U$.

2) Se especifica el procedimiento exacto para ramificar en cada nodo. Inicialmente, se hace $z^U = \infty$. Sin embargo, si el circuito es factible con $c_{12} + c_{23} + ... + c_{n1} < \infty$, entonces, z^U es igual a este valor. El valor inicial de z^U se determina resolviendo el problema de asignación del problema original. Si z^0 es el valor óptimo de la función objetivo, entonces, $z^L = z^0$ es una cota inferior inicial.

Algoritmo asignación

En la iteración r-ésima, se tiene que z^r es la función objetivo óptima, como la cota inferior z^* cambia con el nodo, z^r se definirá automáticamente como z^L.

Descripción:

PASO 0: determine z^0. Si la solución asociada es el circuito completo, pare. De otra manera, guarde $z^U = c_{12} + c_{23} + ... + c_{n1}$ y (1, 2,..., n, 1) como el circuito asociado. Realice $r = 0$ y vaya al paso 1.

PASO 1: seleccione un subcircuito asociado a z^r con el menor número de ciudades e inicialice tantas ramas como número de variables x_{ij}, que son igual a uno en el subcircuito. Para la rama (i,j), defina una nueva matriz de costos que difiere de la primera en que $c_{ij} = \infty$. Haga $r = r + 1$ y vaya al paso 2.

PASO 2: seleccione uno de los nodos que no se han ramificado. Si no queda ninguno, deténgase, el recorrido asociado con z^U es el óptimo. De otra manera, vaya al paso 3.

PASO 3: resuelva el problema de asignación asociado con el nodo seleccionado. De aquí, pueden resultar tres casos:

i) Si $z^r \geq z^U$, entonces, el nodo actual se dice que se ha examinado, ya que, no puede dar un mejor circuito que el asociado con z^U. Haga $r = r + 1$ y vaya al paso 2.

ii) Si $z^r < z^U$ y la solución asociada es un circuito, entonces, haga $z^U = z^r$ y guarde el recorrido o circuito asociado como el mejor disponible hasta este momento.

iii) Si $z^r < z^U$, pero, la solución asociada no es un recorrido, entonces, haga $r = r + 1$ y vaya al paso 1.

Usando este algoritmo escribimos el PAV como un problema de asignación usando LINGO:

```
min 3x12 + x13 + 5x14 + 4x15 + x21 + 5x23 + 4x24 + 3x25 + 5x31 + 4x32 + 2x34 + x35 + 3x41 + x42 + 3x43 +
3x45 + 5x51 + 2x52 + 4x53 + x54
st
x12 + x13 + x14 + x15 =1
x21 + x23 + x24 + x25 =1
x31 + x32 + x34 + x35 =1
x41 + x42 + x43 + x45=1
x21 + x31 + x41 + x51=1
x12 + x32 + x42 + x52=1
x13 + x23 + x43 + x53=1
x14 + x24 + x34 + x54 =1
x15 + x25 + x35 + x45 =1
end
int 20
```

El comando *int* indica que se tienen 20 variables enteras binarias y la solución está dada por:

```
Global optimal solution found.
  Objective value:     5.000000
Variable      Value
   X12       0.000000
   X13       1.000000
   X14       0.000000
   X15       0.000000
   X21       1.000000
   X23       0.000000
   X24       0.000000
   X25       0.000000
   X31       0.000000
   X32       0.000000
   X34       0.000000
   X35       1.000000
   X41       0.000000
   X42       1.000000
   X43       0.000000
   X45       0.000000
   X51       0.000000
   X52       0.000000
   X53       0.000000
   X54       1.000000
```

Las variables $x_{ij} = 1$ nos indican la solución óptima que, en este caso, no formaron subcircuitos y no se tuvo que hacer igual a cero ninguna variable, como sí lo es en el ejemplo que se desarrolla en el anexo C de estos apuntes. Igual que en el caso de la mochila, con esta solución queda validado del modelo de PD.

3.3.1. COMPLEJIDAD COMPUTACIONAL DEL ALGORITMO DE PD (PROGRAMACIÓN DINÁMICA)

En la etapa i, $f_i(j,S)$ se debe evaluar para $(N-1)\binom{N-2}{i}$ diferentes parejas (j, S).

Cada evaluación requiere i sumas e $i - 1$ comparaciones. Entonces, para todas las etapas tenemos el siguiente número de sumas y comparaciones requeridas:

Número de sumas:

$$= (N-1)\sum_{i=1}^{N-2} i\binom{N-2}{i} = (N-1)(N-2)\sum_{i=1}^{N-2}\frac{(N-3)!}{(i-1)!(N-2-i)!} =$$

$$= (N-1)(N-2)\sum_{i=1}^{N-2}\binom{N-3}{i-1} = (N-1)(N-2)\sum_{j=0}^{N-3}\binom{N-3}{j} = (N-1)(N-2)2^{N-3}$$

Número de comparaciones:

$$= (N-1)\sum_{i=1}^{N-2}(i-1)\binom{N-2}{i} = \text{número de sumas} - (N-1)\sum_{i=1}^{N-2}\binom{N-2}{i} =$$

$$= (N-1)(N-2)2^{N-3} - (N-1)\left(2^{N-2}-1\right) = (N-1)2^{N-3}\left[(N-2)-2\right] + (N-1) \approx (N-1)(N-4)2^{N-3}$$

No se consideró el número de sumas $(N - 1)$ y comparaciones $(N - 2)$ necesarias para calcular la respuesta con el circuito óptimo, pero, no es necesario, pues es una cantidad pequeña que no afecta el resultado final. Si consideramos un problema de 20 ciudades, se requiere aproximadamente 85 millones de operaciones.

En realidad, el límite en el tamaño del problema que se quiera resolver con PD tiene que ver más con la capacidad de almacenamiento, que con la cantidad de operaciones. En la etapa i, se requieren $(N-1)\binom{N-2}{i}$ lugares de almacenamiento para almacenar f_i. Para N par, el número de

lugares requeridos es máximo cuando $i = (N - 2)/2$. En el caso de $N = 20$ se necesitan, aproximadamente, 925 mil lugares de almacenamiento para guardar f_9.

Para calcular f_i, debemos tener todos los valores de f_{i-1} disponibles en el almacén central. Esto significa que dicho almacén central debe ser lo suficientemente grande para dos etapas consecutivas. Sin embargo, este requerimiento se puede evitar al usar un almacén auxiliar. Los valores de f_i se pueden poner en este almacén conforme se van calculando y después regresarlos al almacén central cuando se requiera calcular f_{i+1}.

3.3.2. Un procedimiento de duplicado en el caso del PAV (Problema del Agente Viajero) simétrico

En esta sección, consideramos un procedimiento de duplicado para el PAV que es válido cuando se tiene una matriz de distancias simétricas, es decir, $d_{ij}=d_{ji}$ para toda i, j.

Si suponemos que N es par (se requieren modificaciones menores para N impar). Consideremos que $f_1, f_2, \ldots, f_{(N-2)/2}$ se han calculado como se indicó en las ecuaciones recursivas del ejemplo 3.3. Pensemos que un recorrido empieza y termina en la ciudad 1. Una ciudad, digamos j, será visitada en algún punto medio del recorrido, entonces, la longitud de los recorridos más cortos desde la ciudad 1 a la ciudad j, a través de un conjunto de ciudades intermedias $(N-2)/2$, está dada por $f_{(N-2)/2}$. Por lo tanto, la longitud del recorrido más corto está dada por:

$$\min_{j=2,3,..,N}\left\{\min_{S\subseteq N_j}\left\{f_{(N-2)/2}(j,S)+f_{(N-2)/2}(j,N_j-S\right\}\right\}$$

Con los procedimientos normales, comparamos los requerimientos computacionales del procedimiento de duplicado. Se requiere el siguiente número de sumas y comparaciones para el procedimiento de duplicado:

$$\text{Número de sumas } = (N-1)\sum_{i}^{(N-2)/2} i\binom{N-2}{i}+(N-1)\frac{1}{2}\binom{N-2}{(N-2)/2}$$

$$\text{Número de comparaciones} = (N-1)\sum_{i=1}^{(N-2)/2}(i-1)\binom{N-2}{i}+(N-1)\frac{1}{2}\binom{N-2}{(N-2)/2}-1$$

Donde: $\binom{N-2}{(N-2)/2} \Big/ 2$ es el número de particiones no ordenadas de un conjunto de $N-2$ objetos en dos conjuntos de $(N–2)/2$ objetos cada uno. En el caso de $N = 20$, se necesitan un total de aproximadamente 43 millones de operaciones. De esta manera, este procedimiento es considerablemente más eficiente, solamente, que la cantidad de almacenamiento requerida es la misma que para el otro procedimiento.

3.4. RECURSIVIDADES NO ADITIVAS

No se requiere que la recursividad sea aditiva, en varios casos, las recursividades son de tipo multiplicativo como se muestra en los siguientes ejemplos y nos enseñan la amplitud de aplicaciones que pueden resolverse a través de la PD.

Ejemplo 3.5. Un estudiante está tomando tres asignaturas y es importante que las apruebe todas. Si la probabilidad de reprobar francés es p_1, la probabilidad de reprobar inglés es p_2 y la probabilidad de reprobar estadística es p_3, luego, la probabilidad de reprobar las tres asignaturas es $p_1 * p_2 * p_3$. El estudiante definió que estudiaría cuatro horas. ¿Cómo podría minimizar la probabilidad de reprobar todas sus asignaturas?

La tabla 3.8 muestra la probabilidad de reprobar cada curso, dadas las horas de estudio dedicadas:

TABLA 3.8. Probabilidades de reprobación del estudiante

Horas	Francés	Inglés	Estadística
0	0.8	0.75	0.9
1	0.7	0.7	0.7
2	0.65	0.67	0.6
3	0.62	0.65	0.55
4	0.6	0.62	0.5

¿Qué tipo de estudiante es? Sea la etapa 1 estudiar francés, etapa 2 estudiar inglés y etapa 3 estudiar estadística. El estado corresponderá al número de horas que se esté estudiando para cada etapa, más las otras etapas. Sea $f_t(x)$ la probabilidad de reprobar t, más todos los cursos que siguen, suponiendo x horas disponibles. Sea $p_t(k)$ la probabilidad de reprobar el curso t, dadas k horas invertidas en él.

El estado final es:

$$f_3(x) = p_3(x)$$

La recursividad es:

$$f_t(x) = \min_{k:k\le x}\{p_t(k) f_{t+1}(x-k)\}$$

Resolviendo la recursividad se tiene:

ETAPA 3

TABLA 3.9. Cálculos para la etapa 3

X	$f_3(x)$
0	0.9
1	0.7
2	0.6
3	0.55
4	0.5

ETAPA 2

TABLA 3.10. Cálculos para la etapa 2

x	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$f_2(x)$
0	0.75(0.9) = 0.68					0.68
1	0.75(0.7) = 0.52	0.7(0.9) = 0.63				0.52
2	0.75(0.6) = 0.45	0.7(0.7) = 0.49	0.67(0.9) = 0.6			0.45
3	0.75(0.55) = 0.41	0.7(0.6) = 0.42	0.67(0.7) = 0.47	0.65(0.9) = 0.59		0.41
4	0.75(0.5) = 0.37	0.7(0.55) = 0.39	0.67(0.6) = 0.40	0.65(0.7) = 0.45	0.62(0.9) = 0.56	0.37

La forma óptima de asignar el tiempo entre estudiar inglés y estadística es gastar todo el tiempo en estadística.

ETAPA 1

TABLA 3.11. Cálculos para la etapa 1

x	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$f_1(x)$
4	0.8(0.37) = 0.3	0.7(0.41) = 0.287	0.65(0.45) = 0.293	0.62(0.52) = 0.32	0.6(0.68) = 0.41	0.287

La estrategia óptima es gastar una hora en francés y tres en estadística. La probabilidad de reprobar los tres cursos es cercana al 29%.

Ejemplo 3.6. Problema de confiabilidad (Taha, 2004).

En seguida, considere el diseño de un dispositivo electrónico que consta de tres componentes principales. Los tres componentes están dispuestos en serie, de manera que, la falla de uno de ellos hará que falle todo el dispositivo, la confiabilidad o probabilidad de que no haya ninguna falla del dispositivo se puede mejorar a través de la instalación de unidades de reserva de cada componente. El diseño requiere el uso de una o dos unidades de reserva, lo que significa que cada componente principal puede incluir hasta tres unidades en paralelo, como se puede ver en la figura siguiente:

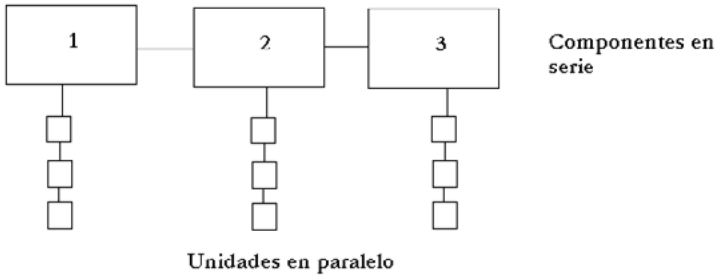


FIGURA 3.5. Diagrama del dispositivo electrónico

El capital total disponible para el diseño del dispositivo es de 10 000 unidades monetarias. Los datos de la confiabilidad $R_j(k_j)$ y el costo $c_j(k_j)$ de la j-ésima componente (j = 1, 2, 3), dadas las k_j unidades en paralelo, se resumen a continuación. El objetivo consiste en determinar el número de unidades en paralelo k_j en el componente j, que maximicen la confiabilidad del dispositivo sin exceder el capital asignado.

TABLA 3.12. Datos de costos y confiabilidad

	$j = 1$		$j = 2$		$j = 3$	
k_j	R_1	C_1	R_2	C_2	R_3	C_3
1	0.6	1	0.7	3	0.5	2
2	0.8	2	0.8	5	0.7	4
3	0.9	3	0.9	6	0.9	5

Por definición, la confiabilidad total R de un dispositivo de N componentes en serie y k_j unidades en paralelo en el componente j (j = 1, 2,…, N) es el producto de las confiabilidades individuales. Por lo tanto, el problema se escribe de la siguiente manera:

$$\text{maximizar } R = \prod_{j=1}^{N} R_j\left(k_j\right)$$

sujeta a

$$\sum_{j=1}^{N} c_j\left\{k_j\right\} \leq C$$

Donde C es el capital total disponible. Este problema es similar al del presupuesto de capital del capítulo anterior, con la diferencia de que la ecuación recursiva está basada en la descomposición multiplicativa y no en la aditiva. Por lo tanto, los elementos del modelo de la PD son:

1) La etapa j representa el componente principal j.

2) El estado y_j es el capital total asignado a los componentes j, j+1,…,N.

3) La alternativa k_j es el número de unidades paralelas asignadas al componente principal j.

Sea $f_j(y_j)$ la confiabilidad óptima total de los componentes j, j+1,…,N, dado el capital y_j, las ecuaciones recursivas se escriben como:

$$f_N\left(y_N\right) = \underset{\substack{k_N \\ c_N(k_N) \leq y}}{\text{máx}} \left\{R_N(k_N)\right\}$$

$$f_j\left(y_j\right) = \underset{\substack{k_j \\ c_j(k_j) \leq y}}{\text{máx}} \left\{R_j(k_j) f_{j+1}\left(y_j - c_j\left(k_j\right)\right)\right\} \qquad j = 1,2,...,N-1$$

Como ya se ha visto antes, el número de operaciones de cálculo en la etapa j depende directamente del número de los valores tomados para el estado y_j. Entonces, podemos dar límites más estrechos para los valores de y_j.

Si comenzamos con la etapa 3, ya que, el componente 3 debe incluir al menos una unidad en paralelo, vemos que y_3 debe ser cuando menos igual a $c_3(1) = 2$. Por el mismo razonamiento, y_3 no puede exceder a $10 - (3 + 1) = 6$, de lo contrario, el capital restante no será suficiente para proporcionar a los componentes principales 1 y 2, cuando menos, una unidad en paralelo a cada una. Entonces, $y_2 = 5, 6,\ldots,9$ y $y_1 = 6, 7,\ldots,10$.

ETAPA 3

$$f_3(y_3) = \max_{k=1,2,3}\{R_3(k_3)\}$$

TABLA 3.13. Cálculos para la etapa 3

	$R_3(k_3)$			Solución óptima	
	$k_3 = 1$	$k_3 = 2$	$k_3 = 3$	$f_3(y_3)$	k_3^*
y_3	$R = 0.5, c = 2$	$R = 0.7, c = 4$	$R = 0.9, c = 5$		
2	0.5	-	-	0.5	1
3	0.5	-	-	0.5	1
4	0.5	0.7	-	0.7	2
5	0.5	0.7	0.9	0.9	3
6	0.5	0.7	0.9	0.9	3

ETAPA 2

$$f_2(y_2) = \max_{k_2=1,2,3}\left\{R_2(k_2) f_3\left(y_2 - c_2(k_2)\right)\right\}$$

TABLA 3.14. Cálculos para la etapa 2

	$\left\{R_2(k_2) f_3\left(y_2 - c_2(k_2)\right)\right\}$			Solución óptima	
	$k_2 = 1$	$k_2 = 2$	$k_2 = 3$	$f_2(y_2)$	k_2^*
y_2	$R = 0.7, c = 3$	$R = 0.8, c = 5$	$R = 0.9, c = 6$		
5	0.7(0.5) = 0.35	-	-	0.35	1
6	0.7(0.5) = 0.35	-	-	0.35	1
7	0.7(0.7) = 0.49	0.8(0.5) = 0.40	-	0.49	1
8	0.7(0.9) = 0.63	0.8(0.5) = 0.40	0.9(0.5) = 0.45	0.63	1
9	0.7(0.9) = 0.63	0.8(0.7) = 0.56	0.9(0.5) = 0.45	0.63	1

ETAPA 1

$$f_1(y_1) = \max_{k_1=1,2,3} \left\{ R_1(k_1) f_2\left(y_1 - c_1(k_1)\right) \right\}$$

TABLA 3.15. Cálculos para la etapa 1

	$\left\{ \boldsymbol{R_1(k_1) f_2\left(y_1 - c_1(k_1)\right)} \right\}$			**Solución óptima**	
	$\boldsymbol{k_1 = 1}$	$\boldsymbol{k_1 = 2}$	$\boldsymbol{k_1 = 3}$	$\boldsymbol{f_1(y_1)}$	$\boldsymbol{k_1^*}$
$\boldsymbol{y_1}$	$\boldsymbol{R = 0.6, c = 1}$	$\boldsymbol{R = 0.8, c = 2}$	$\boldsymbol{R = 0.9, c = 3}$		
6	0.6(0.35) = 0.210	--	--	0.210	1
7	0.6(0.35) = 0.210	0.8(0.35) = 0.280	--	0.280	2
8	0.6(0.49) = 0.294	0.8(0.35) = 0.280	0.9(0.35) = 0.315	0.315	3
9	0.6(0.63) = 0.378	0.8(0.49) = 0.392	0.9(0.35) = 0.315	0.392	2
10	0.6(0.63) = 0.378	0.8(0.63) = 0.504	0.9(0.49) = 0.441	0.504	2

La solución con $C = 10$ es $(k_1^*, k_2^*, k_3^*) = (2, 1, 3)$ con $R = 0.504$, lo que significa que con una confiabilidad del 50 por ciento se puede trabajar instalando 2 unidades en paralelo en el componente 1, 1 unidad en el componente 2 y 3 para el tercero.

Ejemplo 3.7. Problema de la subdivisión óptima (Taha, 2004).

A continuación, considere el problema matemático de dividir una cantidad q (>0) en N partes. El objetivo es determinar la subdivisión óptima de que q maximizará el producto de las N partes.

Sea z_j la j-ésima parte de q ($j = 1,2,.., N$), por lo tanto, el problema se expresa como:

$$\text{maximizar } p = \prod_{j=1}^{N} z_j$$

sujeta a

$$\sum_{j=1}^{N} z_j = q, \qquad z_j \geq 0, \forall j$$

La formulación de la PD de este problema es como el de confiabilidad con la diferencia de que las variables z_j son continuas, condición que requiere el uso del cálculo para optimizar el problema de cada etapa. Los elementos de PD se definen como:

1) La etapa j representa la j-ésima parte de q.

2) El estado y_j es la parte de q que se asigna a las etapas $j, j+1,\ldots,N$.

3) La alternativa z_j es la parte de q asignada a la etapa j.

Sea $f_j(y_j)$ el valor óptimo de la función objetivo para las etapas j, $j+1,\ldots,N$, dado el estado y_j, por lo tanto, las ecuaciones recursivas están dadas por:

$$f_N(y_N) = \max_{z_N \le y_N} \{z_N\}$$

$$f_j(y_j) = \max_{z_j \le y_j} \left\{ z_j f_{j+1}\left(y_j - z_j\right) \right\} \qquad j = 1,2...,N-1$$

ETAPA N

$$f_N(y_N) = \max_{z_N \le y_N} \{z_N\}$$

Como z_N es una función lineal, entonces, el máximo lo alcanza en $z_N^* = y_N$, se puede resumir la solución óptima de esta etapa mediante el uso de una tabla como sigue:

TABLA 3.16. Solución óptima

	Solución óptima	
Estado	$f_N(y_N)$	z_N^*
y_N	y_N	y_N

ETAPA $N-1$

$$f_{N-1}(y_{N-1}) = \max_{z_{N-1} \le y_{N-1}} \left\{ z_{N-1} f_N\left(y_{N-1} - z_{N-1}\right) \right\}$$

Ya que, $f_N(y_N) = y_N$, tenemos:

$$f_N\left(y_{N-1} - z_{N-1}\right) = y_{N-1} - z_{N-1}$$

Por lo tanto, al sustituir por f_N , el problema para la etapa N–1 se reduce a maximizar:

$$h_{N-1} = z_{N-1} f_N\left(y_{N-1} - z_{N-1}\right) = z_{N-1}\left(y_{N-1} - z_{N-1}\right)$$

dada $z_{N-1} \leq y_{N-1}$

La figura siguiente ilustra lo que acarrea el problema de optimización trazando la función h_{N-1} en términos de z_{N-1} junto con la región factible $z_{N-1} \leq y_{N-1}$.

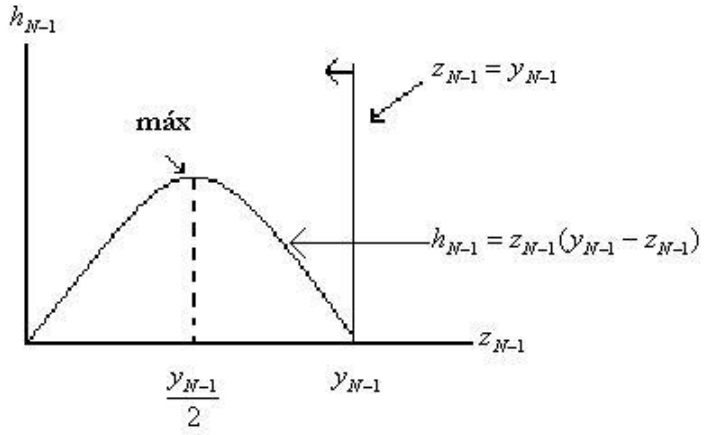


FIGURA 3.6. El problema de optimización

La solución óptima se encuentra en z_{N-1} *= $\frac{y_{N-1}}{2}$. Este punto se obtiene al diferenciar h_{N-1} con respecto a z_{N-1} en la región factible $z_{N-1} \leq y_{N-1}$.

El valor de $f_{N-1}(y_{N-1})$ se obtiene al sustituir $z_{N-1} = y_{N-1/}^{2}$ en h_{N-1}. La solución óptima resultante está dada por:

TABLA 3.17. Solución óptima

	Solución óptima	
Estado	$f_{N-1}(y_{N-1})$	z_{N-1}*
y_{N-1}	$(y_{N-1}/2)^2$	$y_{N-1/}^{2}$

ETAPA *J*

$$f_j(y_j) = \max_{z_j \leq y_j} \left\{ z_j f_{j+1}\left(y_j - z_j\right) \right\} \qquad j = 1,2...,N-1$$

Se puede usar la inducción para demostrar que la solución óptima en la etapa *j* se resume como:

TABLA 3.18. Solución óptima

	Solución óptima	
Estado	$f_j(y_j)$	Z_j^*
y_j	$\left(\frac{y_j}{N-j+1}\right)^{N-j+1}$	$\frac{y_j}{N-j+1}$

Por inspección de la solución general en la etapa *j*, se pueden obtener los valores óptimos de z_j, dado $y_1 = q$ de la manera siguiente:

$$y_1=q \rightarrow z_1=\frac{q}{N} \rightarrow y_2=\frac{N-1}{N}q \rightarrow y_j=\frac{N-j+1}{N}q \rightarrow z_j=\frac{q}{N}$$

Por lo tanto, la solución general es:

$$z_1^*=z_2^*=\ldots=z_j^*=\ldots=z_N^*=\frac{q}{N}$$

El valor óptimo de la función objetivo es:

$$p=f_1(q)=\left(\frac{q}{N}\right)^N$$

3.5. PROBLEMA DE DECISIÓN DE MÁRKOV

En un inicio, las cadenas de Márkov[21] se usaron para analizar procesos físicos y de meteorología, ya que, una de las primeras aplicaciones fue para predecir patrones de clima. Actualmente, su aplicación es muy extensa como en los precios, mantenimiento de la maquinaria, selección de los productos, longitud de filas en los servicios, manejo de inventarios y hasta en el comportamiento de animales en el laboratorio. Esto se basa en el hecho de que estos sistemas pueden estar en uno de los diferentes estados posibles y con el tiempo, ese estado puede cambiar. Si se encuentra que la transición de un estado a otro no está predeterminada, sino que ocurre en función de ciertas probabilidades que dependen de la

[21] Andréi Andréyevich Márkov (1856-1922) matemático ruso que inventó el método.

historia del sistema, entonces, se tiene un *proceso estocástico*. Pero, si además estas probabilidades de transición dependen solamente de la historia inmediata del sistema, es decir, si el estado del sistema en una observación cualquiera depende solo de su estado en la observación inmediata anterior, entonces, el proceso es un *proceso de Márkov*.

Una variedad de procesos de Márkov existen, mismos que no se abordan en estos apuntes, pues no es la finalidad de los mismos, solo cabe mencionar que las probabilidades asociadas a los diferentes estados se conocen como probabilidades de transición y con dichas probabilidades se construye una matriz de transición como se define a continuación.

Una *matriz de transición* $P = [p_{ij}]$ es una matriz cuadrada con entradas no negativas y donde la suma de cada columna es igual a la unidad.

En esta sección veremos cómo se resuelve un problema de cadenas de Márkov usando PD, esto será a través de un ejemplo que, a pesar de su sencillez, hace la paráfrasis de varias aplicaciones importantes en las áreas de inventarios, reemplazo, manejo de la circulación de efectivo y regulación de la capacidad de un depósito de agua.

Ejemplo 3.8. El problema de la jardinera. (Taha, 2004).

Todos los años, al inicio de la estación de cultivo, una jardinera realiza pruebas químicas para revisar la condición de la parcela. Dependiendo de los resultados de las pruebas, puede clasificar la productividad del jardín para la nueva temporada como buena, regular o deficiente.

Con el paso de los años, la mujer observó que la productividad del año en curso puede suponerse dependiente solo de la condición del terreno del año anterior. Por lo tanto, puede representar las probabilidades de transición en el periodo de un año, de un estado de productividad a otro y en términos de la siguiente cadena de Márkov:

$$\text{Estado del sistema este año} \begin{Bmatrix} 1 \\ 2 \\ 3 \end{Bmatrix} \overbrace{\begin{bmatrix} 0.2 & 0.5 & 0.3 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{bmatrix}}^{\begin{matrix}\text{Estado del sistema} \\ \text{para el próximo año} \\ 1 \quad 2 \quad 3\end{matrix}} = P^1$$

La representación supone la siguiente correspondencia entre la productividad y los estados de la cadena:

TABLA 3.19. Productividad y estados de la cadena

Productividad (Condición del terreno)	**Estado del sistema**
Buena	1
Regular	2
Deficiente	3

Las probabilidades de transición en P^1 indican que la productividad de un año en curso puede no ser mejor que la del año anterior. Por ejemplo, si la condición del terreno para este año es regular (estado 2), la productividad del año siguiente puede seguir regular con probabilidad 0.5 o volverse deficiente (estado 3), también, con probabilidad 0.5.

La mujer puede alterar las probabilidades de transición P^1 tomando otros cursos de acción que tenga a su disposición. Comúnmente, puede decidir fertilizar el jardín para mejorar la condición del terreno, que producirá la siguiente matriz de transición P^2:

$$P^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0.3 & 0.6 & 0.1 \\ 0.1 & 0.6 & 0.3 \\ 0.05 & 0.4 & 0.55 \end{bmatrix} \end{array}$$

Al aplicar el fertilizante, es posible mejorar la condición del terreno con respecto a la del año pasado.

Para poner en perspectiva el problema de la decisión, la jardinera asocia una función de rendimiento (o estructura de recompensa) con la transición de un estado a otro. La función de rendimiento expresa la ganancia o pérdida durante un periodo de un año, dependiendo de los estados entre los que se haga la transición. Como la mujer tiene las opciones de utilizar o no fertilizante, se espera que su ganancia y pérdidas varíen según la decisión que ella tome.
Las matrices R^1 y R^2 resumen las funciones de rendimiento en cientos de unidades monetarias asociadas con las matrices P^1, P^2, respectivamente. Por lo tanto, R^1 se aplica cuando no se usa fertilizante; en caso contrario, se puede utilizar R^2 en la representación de la función de rendimiento.

$$R^1 = \left\| r_{ij}^1 \right\| = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 7 & 6 & 3 \\ 0 & 5 & 1 \\ 0 & 0 & -1 \end{bmatrix} \end{array}$$

$$R^2 = \left\| r_{ij}^2 \right\| = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 6 & 5 & -1 \\ 7 & 4 & 0 \\ 6 & 3 & -2 \end{bmatrix} \end{array}$$

Note que los elementos r_{ij}^2 de R^2 toman en cuenta el costo por aplicar el fertilizante. Por ejemplo, si el sistema estuviera en el estado 1 y se mantuviera en ese estado durante el año siguiente, su ganancia sería $r_{11}^2 = 6$ en comparación con $r_{11}^2 = 7$ cuando no se emplea fertilizante.

¿Qué tipo de problema de decisión tiene la jardinera? Primero, queremos saber si la actividad de cultivo seguirá realizándose un número limitado de años o, para fines prácticos, por tiempo indefinido. Estas situaciones se conocen como problemas de decisión de *etapa finita* y de *etapa infinita*. En ambos casos, la jardinera necesitaría determinar el mejor curso de acción que debe seguir (fertilizar o no fertilizar el terreno) dado el resultado de las pruebas químicas (estado del sistema). El proceso de optimización estará basado en la maximización del ingreso esperado. Quizá, la jardinera, también, esté interesada en evaluar el ingreso esperado resultante por seguir un curso de acción especificado, siempre que ocurra un estado dado del sistema Por ejemplo, ella puede decidir fertilizar siempre que la condición del terreno sea deficiente (estado 3). En este caso, el proceso de la toma de decisión se dice estar representado por una política estacionaria.

Además, debemos observar que cada política estacionaria debe estar asociada con matrices de transición y rendimiento diferentes que, en general, pueden construirse a partir de las matrices P^1, P^2, R^1, y R^2. Por ejemplo, para la política estacionaria que pide se aplique fertilizante solo cuando la condición del terreno sea deficiente (estado 3), las matrices de transición y rendimiento resultantes P y R están dadas por:

$$P = \begin{array}{c} \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0.2 & 0.5 & 0.3 \\ 0 & 0.5 & 0.5 \\ 0.05 & 0.4 & 0.55 \end{bmatrix} \end{array} \qquad R = \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 7 & 6 & 3 \\ 0 & 5 & 1 \\ 6 & 3 & -2 \end{bmatrix} \end{array}$$

Estas matrices difieren de P^1 y R^1 solo en los renglones terceros que se toman, directamente, de P^2 y R^2. La razón es que P^2 y R^2 son las matrices que se generan cuando se aplica fertilizante en todos y cada uno de los estados.

3.5.1. Modelo de PD (Programación Dinámica) de etapa finita

Ahora, suponga que la jardinera planea *retirarse* de su pasatiempo en N años. Por lo tanto, está interesada en determinar su curso de acción óptimo para cada año (fertilizar o no fertilizar el terreno) sobre un horizonte de planeación finito. Aquí, la optimilidad se define, de manera que, la jardinera acumulará el más alto ingreso esperado al cabo de N años.

Sea $k = 1$ y 2 los dos cursos de acción (opciones) disponibles para ella. Las matrices P^k y R^k representan las probabilidades de transición y la función de remuneración para la alternativa k.

$$P^1 = \left\| p_{ij}^1 \right\| = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0.2 & 0.5 & 0.3 \\ 0 & 0.5 & 0.5 \\ 0 & 0.0 & 1 \end{bmatrix} \end{array} \qquad R^1 = \left\| r_{ij}^1 \right\| = \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 7 & 6 & 3 \\ 0 & 5 & 1 \\ 0 & 0 & -1 \end{bmatrix} \end{array}$$

$$P^2 = \left\| P_{ij}^2 \right\| = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0.3 & 0.6 & 0.1 \\ 0.1 & 0.6 & 0.3 \\ 0.05 & 0.4 & 0.55 \end{bmatrix} \end{array} \qquad R^2 = \left\| r_{ij}^2 \right\| = \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 6 & 5 & -1 \\ 7 & 4 & 0 \\ 6 & 3 & -2 \end{bmatrix} \end{array}$$

Y recuerde que el sistema tiene tres estados: bueno (estado 1), regular (estado 2) y malo o deficiente (estado 3).

El problema de la jardinera lo podemos expresar como un modelo de PD de estado finito de la manera siguiente. Para hacer una generalización, suponga que el número de estados para cada etapa (año) es m (= 3 en el ejemplo del jardinero) y se define como:

$f_n(i)$ = ingreso *esperado* óptimo de las etapas n, n +1,…, N, dado que el estado del sistema (condición del terreno) al inicio del año n es i.

La ecuación recursiva hacia atrás, que relaciona a f_n y f_{n+1}, puede escribirse como sigue y se puede ver en la figura 3.7.

$$f_n(i) = \max_K \left\{ \sum_{j=1}^{m} p_{ij}^k \left[r_{ij}^k + f_{n+1}(j) \right] \right\}, \qquad n = 1,2,..., N$$

donde $f_{N+1}(j) = 0$ para toda j

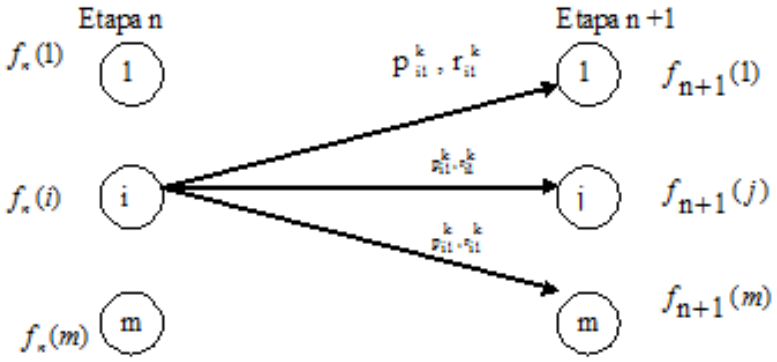


FIGURA 3.7. Relaciones recursivas para el problema de la jardinera

Una justificación para la ecuación es que el ingreso acumulado $r_{ij}^k + f_{n+1}(j)$, que resulta de llegar al estado j en la etapa $n + 1$ desde el estado i en la etapa n, ocurre con la probabilidad p_{ij}^k. De hecho, si v_i^k. representa el rendimiento esperado resultante de una transición desde el estado i, dada la alternativa k, entonces, v_i^k. puede expresarse como:

$$v_i^k = \sum_{j=1}^{m} p_{ij}^k r_{ij}^k .$$

La ecuación recursiva de PD puede escribirse como:

$$f_N(i) = \max_k \left\{ v_i^k \right\}$$

$$f_n(i) = \max_k \left\{ v_i^k + \sum_{j=1}^{m} p_{ij}^k f_{n+1}(1) \right\}, \qquad n = 1,2,...,N-1$$

Antes de demostrar cómo se utiliza la ecuación recursiva para resolver el problema de la jardinera, ilustramos el cálculo de v_i^k que es parte de la ecuación recursiva. Por ejemplo, suponga que no se utiliza fertilizante ($k = 1$), entonces:

$$v_1^1 = 0.2\ (7) + 0.5\ (6)\ + 0.3\ (3) = 5.3$$

$$v_2^1 = 0\ (0) + 0.5\ (5) + 0.5\ (1) = 3$$

$$v_3^1 = 0\ (0) + 0\ (0) + 1\ (-1) = -1$$

Estos valores muestran que, si se advierte que la condición del terreno es buena (estado 1) al inicio del año, se espera que una sola transición genere 5.3 para el año. En forma análoga, si la condición del terreno es regular (deficiente), el ingreso esperado es 3(–1).

Ahora, con esta explicación se resuelve el problema de la jardinera.

Y, considerando los datos de las matrices P^1, P^2, R^1 y R^2 con un horizonte de planeación de tres años, es decir $N = 3$, se tienen los siguientes cálculos:

TABLA 3.20. Datos para el problema de la jardinera

i	v_i^1.	v_i^2.
1	5.3	4.7
2	3	3.1
3	–1	0.4

ETAPA 3

TABLA 3.21. Cálculos para la tercera etapa

	v_i^k.		Solución óptima	
i	$k = 1$	$k = 2$	$f_3(i)$	k^*
1	5.3	4.7	5.3	1
2	3	3.1	3.1	2
3	–1	0.4	0.4	2

ETAPA 2

TABLA 3.22. Cálculos para la segunda etapa

	$v_i^k + p_{i1}{}^k f_3.(1) + p_{i2}{}^k f_3(2) + p_{i3}{}^k f_3(3)$		Solución óptima	
I	***k* = 1**	***k* = 2**	$f_2(i)$	k^*
1	5.3 + 0.2 (5.3) + 0.5 (3.1) + 0.3 (0.4) = 8.03	4.7 + 0.3 (5.3) + 0.6 (3.1) + 0.1 (0.4) = 8.19	8.19	2
2	3 + 0 (5.3) + 0.5 (3.1) + 0.5 (0.4) = 4.75	3.1 + 0.1 (5.3) + 0.6 (3.1) + 0.3 (0.4) = 5.61	5.61	2
3	–1 + 0 (5.3) + 0 (3.1) + 1 (0.4) = – 0.6	0.4 + 0.05 (5.3) + 0.4 (3.1) + 0.55 (0.4) = 2.13	2.13	2

ETAPA 1

TABLA 3.23. Cálculos para la primera etapa

	$v_i^k + p_{i1}{}^k f_2.(1) + p_{i2}{}^k f_2(2) + p_{i3}{}^k f_2(3)$		Solución óptima	
I	***k* = 1**	***k* = 2**	$f_1(i)$	k^*
1	5.3 + 0.2 (8.19) +0.5 (5.61) +0.3 (2.13) = 10.38	4.7 + 0.3 (8.19) + 0.6 (5.61) + 0.1 (2.13) = 10.74	10.74	2
2	3 + 0 (8.19) + 0.5 (5.61) + 0.5 (2.13) = 6.87	3.1 + 0.1 (8.19) + 0.6 (5.61) + 0.3 (2.13) = 7.92	7.92	2
3	–1 + 0 (8.19) + 0 (5.61) + 1 (2.13) = 1.13	0.4 + 0.05 (8.19) + 0.4 (5.61) + 0.55 (2.13)= 4.23	4.23	2

La solución óptima indica que para los años 1 y 2, la jardinera debe fertilizar el terreno (k^*=2) sin importar el estado del sistema (condición del terreno revelada por las pruebas químicas). Sin embargo, en el año 3, ella debe aplicar fertilizante solo si el sistema se encuentra en el estado 2 o 3 (condición del terreno regular o deficiente). Los ingresos totales esperados de los tres años son $f_1(1)$ = 10.74 si el estado del sistema en el año 1 es bueno, $f_1(2)$ = 7.92 si es regular y $f_1(3)$ = 4.23 si es deficiente.

La solución de PD que se dio antes se conoce, algunas veces, como el enfoque de iteración de valor, ya que, por la naturaleza real de la ecuación recursiva, los valores $f_n(i)$ se determinan en forma iterativa.

El problema de la jardinera (de horizonte finito), que acabamos de resolver, se puede generalizar de dos formas. Primero, las probabilidades de transición y sus funciones de rendimiento no necesitan ser las mismas para cada año. Segundo, se puede aplicar un factor

de descuento al ingreso esperado de las etapas sucesivas, de manera que, los valores $f_1(i)$ representen el valor presente de los ingresos esperados de todas las etapas.

La primera generalización requeriría simplemente que los valores de rendimiento r_{ij}^{k} y las probabilidades de transición p_{ij}^{k} sean, además, funciones de la etapa *n*. En este caso, las ecuaciones recursivas se ven como:

$$f_N(i) = \max_k \left\{ v_i^{k,N} \right\}$$

$$f_n(i) = \max_k \left\{ v_k^{k,N} + \sum_{j=1}^{m} p_{ij}^{k,n} f_{n+1}(j) \right\}, \qquad n = 1,2,..,N-1$$

donde

$$v_i^{k,n} = \sum_{j=1}^{m} p_{ij}^{k,n} r_{ij}^{k,n}$$

La segunda generalización se lleva a cabo de la siguiente manera. Sea α (<1) el factor de descuento por año que normalmente se calcula como α = 1/ (1 + t), donde *t* es la tasa de interés anual. Por lo tanto, *D* unidades monetarias de aquí a un año es equivalente a αD unidades monetarias ahora, esto se verá con más detalle en el capítulo 4. La introducción del factor de descuento modificará la ecuación recursiva original como sigue:

$$f_N(i) = \max_K \left\{ v_i^{k} \right\}$$

$$f_n(i) = \max_k \left\{ v_i^{k} + \alpha \sum_{j=1}^{m} p_{ij}^{k} f_{n+1}(j) \right\}, \qquad n = 1,2,..,N-1$$

La aplicación de esta ecuación recursiva es exactamente similar a la original salvo por la multiplicación del factor de descuento. En general, el uso de un factor de descuento puede generar una decisión óptima diferente en comparación con el caso cuando no se utiliza un descuento.

La ecuación recursiva se puede utilizar para evaluar cualquier política estacionaria para el problema de la jardinera. Suponiendo que no se utiliza descuento, es decir (α = 1), la ecuación recursiva para evaluar una política estacionaria es:

$$f_n(i) = v_i + \sum_{j=1}^{m} p_{ij} f_{n+1}(j)$$

Donde p_{ij} es el (i, j)-ésimo elemento de la matriz de transición asociada con la política y v_i es el ingreso de transición en un paso esperado de la política.

3.6. CONCLUSIONES

Los problemas abordados en este capítulo son problemas clásicos de la optimización y en los casos en donde ha sido posible, se ha hecho una validación de los modelos de PD usando modelos y métodos de solución alternativos, si el lector tiene más herramientas para validar los diferentes modelos es algo muy recomendable, ya que, siempre es un paso importante en la modelación.

Para los modelos donde se cuenta con recursividades no aditivas y, por lo tanto, no es posible hacer uso de la PLE, también, se puede usar PNL (Programación No Lineal) u otras herramientas para la investigación de operaciones y sería el mismo caso de las cadenas de Márkov. El problema podría resolverse directamente usando cálculo matricial, pero, haría muy extenso el contenido del capítulo.

3.7. NOTAS HISTÓRICAS

La primera vez que se usó el término PAV en los círculos matemáticos pudo ser en los años 1931-1932 por Karl Menger. Pero en 1832, un libro, que fue escrito por un agente viajero veterano y editado en Alemania titulado *El agente viajero* narra cómo debe ser y qué debe hacerse para cumplir sus comisiones y tener éxito en los negocios. Aunque, la obra trata de muchos otros temas en su último capítulo alcanza lo que es la esencia del problema: "Para la selección apropiada de un buen recorrido, se puede ganar mucho tiempo para lo cual hacemos algunas sugerencias… El aspecto más importante es cubrir la mayor cantidad de lugares sin visitar alguno dos veces…".

No sabemos con exactitud, quién trajo el nombre del problema en los círculos matemáticos, pero no hay duda de que Merrill Flood es responsable de su publicación en la comunidad de Investigación de Operaciones. Flood menciona que fue A. W. Tucker quien en 1937 lo mencionó al hablar sobre el estudio de la ruta de un autobús escolar. Pero, Tucker dice, a su vez, que la historia la escuchó de Hassler Whitney, lo cual ocurrió entre 1931 y 1932.

> El *problema de la mochila* es uno de los 21 problemas, NP-completos de Richard Karp establecidos por el informático teórico en un famoso artículo de 1972. Ha sido intensamente estudiado desde mediados del siglo XX y se hace referencia a él en el año 1897, un artículo de George Mathews Ballard.
>
> Si bien, la formulación del problema es sencilla, su resolución es más compleja. Algunos algoritmos existentes pueden resolverlo en la práctica para casos de un gran tamaño. Sin

embargo, la estructura única del problema y el hecho de que se presente como un subproblema de otros problemas más generales, lo convierte en un problema frecuente en la investigación.[22]

Richard Karp

Karp nació en Boston, Massachusetts. Recibió su licenciatura por la Universidad de Harvard en 1955, su máster en 1956, y su Ph. D. en matemática aplicada en 1959. A partir de entonces trabajó en el Thomas J. Watson Research Center de IBM. En 1968 ingresó como profesor de Ciencias de la Computación, Matemáticas e Investigaciones Operacionales de la Universidad de California, Berkeley. Aparte de un periodo de 4 años en el que fue profesor en la Universidad de Washington, ha permanecido en Berkeley. Karp también ganó la Medalla Benjamin Franklin de 2004 en Ciencias de la Computación y Cognitivas por sus contribuciones al campo de la complejidad computacional.

La razón por la que se le otorgó el Premio Turing fue:

Por sus continuas contribuciones a la teoría de algoritmos, incluyendo el desarrollo de algoritmos eficientes para el flujo de redes y otros problemas de optimización combinatoria, la demostración de equivalencia de la noción intuitiva de eficiencia logarítmica con la computabilidad en tiempo polinómico y, principalmente, sus contribuciones a la teoría de NP-completitud. Karp la metodología hoy común para probar que ciertos problemas son NP-completos que ha llevado a determinar que muchos problemas teóricos y prácticos son computacionalmente difíciles.

En 1971 co-desarrolló junto con Jack Edmonds el Algoritmo de Edmonds-Karp para resolver problemas de maximización de flujo en redes. En 1972 publicó su famosa lista de 21 problemas NP-completos. En 1987, junto con Michael O. Rabin desarrolló el Algoritmo Rabin-Karp de búsqueda de cadenas.

Ha hecho muchos otros importantes descubrimientos en las ciencias de la computación e investigación operacional, en el área de optimización combinatoria. En 2006, cuando se escribió este artículo, su principal interés incluye la bioinformática.[23]

[22] "Problema de la mochila", 2015. Recuperado de: http://es.wikipedia.org/wiki/Problema_de_la_mochila
[23] "Richard Karp", 2014. Recuperado de: http://es.wikipedia.org/wiki/Richard_Karp


А. А. Марков (1886).

Andréi Andréyevich Márkov
(Андре́й Андре́евич Ма́рков) (14 de junio de 1856 - 20 de julio de 1922) fue un matemático ruso conocido por sus trabajos en la teoría de los números y la teoría de probabilidades.

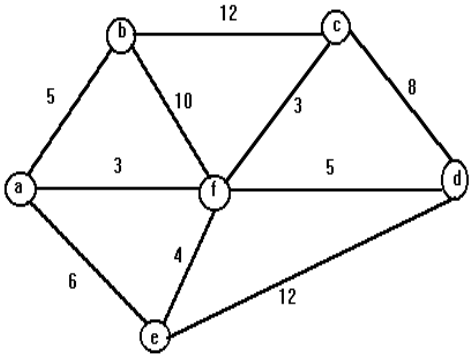
Márkov nació en Riazán, Rusia. Antes de los 10 años su padre, un funcionario estatal, fue trasladado a San Petersburgo donde Andréi entró a estudiar en un instituto de la ciudad. Desde el principio mostró cierto talento para las matemáticas y cuando se graduó en 1874 ya conocía a varios matemáticos de la Universidad de San Petersburgo, donde ingresó tras su graduación. En la universidad fue discípulo de Pafnuti Chebyshov y tras realizar sus tesis de maestría y doctorado, en 1886 accedió como adjunto a la Academia de Ciencias de San Petersburgo, a propuesta del propio Chebyshov. Diez años después Márkov había ganado el puesto de académico regular. Desde 1880, tras defender su tesis de maestría, Márkov impartió clases en la universidad y, cuando el propio Chebyshov dejó la universidad tres años después, fue Márkov quien le sustituyó en los cursos de teoría de probabilidad. En 1905, tras 25 años de actividad académica, Márkov se retiró definitivamente de la universidad, aunque siguió impartiendo algunos cursos sobre la teoría de la probabilidad.

Aparte de su perfil académico, Andréi Márkov fue un convencido activista político. Se opuso a los privilegios de la nobleza zarista y llegó a rechazar las condecoraciones del propio zar en protesta por algunas decisiones políticas relacionadas con la Academia de Ciencias. Hasta tal punto llegó su implicación en la política que llegó a ser conocido con el sobrenombre de "el académico militante".

Márkov arrastró durante toda su vida problemas relacionados con una malformación congénita en la rodilla que le llevaría varias veces al quirófano y que, con el tiempo, fue la causa de su muerte cuando el 20 de julio del año 1922 una de las muchas operaciones a las que se sometió le produjo una infección generalizada de la que no pudo recuperarse.[24]

3.8. EJERCICIOS PROPUESTOS

1) Suponga que un autobús debe recorrer cada uno de los seis poblados indicados a continuación, visitando cada poblado una vez y regresando al punto de partida. El tiempo de recorrido en este caso es simétrico, es decir, $t_{ij}=t_{ji}$. ¿Qué secuencia de visita se debe llevar a cabo con objeto de minimizar el tiempo de recorrido total? Plantee el modelo de PD y resuelva.



2) Resuelva el problema de la mochila 0 – 1 con los siguientes datos:

Objetos	Peso	Valor
1	8	80
2	6	48
3	2	14
4	3	18
5	2	10
Capacidad	11	

3) Una empresa revisa cada año el estado de un producto importante y debe decidir, si tiene éxito (estado 1) o no lo tiene (estado 2). Para esto, tiene que decidir, si anuncia o no el producto para impulsar las ventas. Las matrices P^1 y P^2 representan las probabilidades de transición con y sin publicidad durante cualquier año. Los ingresos correspondientes están dados por las matrices R^1 y R^2. Determine la política óptima durante los próximos tres años.

$$P^1 = \begin{array}{c} \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} 0.9 & 0.1 \\ 0.6 & 0.4 \end{pmatrix} \end{array} \qquad R^1 = \begin{array}{c} \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} 2 & -1 \\ 1 & -3 \end{pmatrix} \end{array}$$

$$P^2 = \begin{array}{c} \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{pmatrix} \end{array} \qquad R^2 = \begin{array}{c} \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} 4 & 1 \\ 2 & -1 \end{pmatrix} \end{array}$$

[24] "Andréi Márkov", 2013. Recuperado de: http://es.wikipedia.org/wiki/Andr%C3%A9i_M%C3%A1rkov

CAPÍTULO 4

MODELOS DE INVENTARIOS

4.1. INTRODUCCIÓN[25]

Los inventarios se relacionan con el mantenimiento de las cantidades suficientes de bienes (por ejemplo, refacciones y materias primas) que garanticen una operación fluida en un sistema de producción o en una actividad comercial. Los inventarios se han considerado por parte del comercio y la industria como un mal necesario; desde ese punto de vista, la única manera efectiva de manejar los inventarios es minimizar su impacto adverso, encontrando un justo medio entre los dos casos extremos.

El objetivo final de cualquier modelo de inventarios es el de dar respuesta a las dos preguntas siguientes:

1) ¿Qué cantidad de artículos deben pedirse?

2) ¿Cuándo deben pedirse?

La respuesta a la primera pregunta se expresa en términos de lo que llamamos *cantidad de pedido.* Esta representa la cantidad óptima que debe ordenarse cada vez que se haga un pedido y puede variar con el tiempo, dependiendo de la situación que se considere. La respuesta a la segunda interrogante depende del tipo de sistema de inventarios. Si el sistema requiere *revisión periódica* en intervalos de tiempo iguales (por ejemplo, cada semana o cada mes), el tiempo para adquirir un nuevo pedido suele coincidir con el inicio de cada intervalo de tiempo. Por otra parte, si el sistema es del tipo de *revisión continua*, el nivel de inventario en el cual debe colocarse un nuevo pedido suele especificar un *punto para un nuevo pedido.*

[25] Taha, Hamdy A., *Investigación de operaciones,* 7.ª ed., Pearson, 2004.

Por lo tanto, podemos expresar la solución del problema general de inventarios de la manera siguiente:

1) Caso de revisión periódica. Recepción de un nuevo pedido por la cantidad especificada y por la cantidad de pedido en intervalos de tiempo iguales.

2) Caso de revisión continua. Cuando el nivel del inventario llega al punto para un nuevo pedido, se coloca un nuevo pedido cuyo tamaño sea igual a la cantidad del pedido.

La cantidad y el punto de un nuevo pedido suelen determinarse, normalmente, minimizando el costo del inventario total que se puede expresar como una función de estas dos variables. Podemos resumir el costo total de un modelo general de inventarios como la función de sus componentes principales en la forma siguiente:

$$\begin{pmatrix}\text{costo de}\\ \text{inventario}\\ \text{total}\end{pmatrix} = \begin{pmatrix}\text{costo de}\\ \text{compra}\end{pmatrix} + \begin{pmatrix}\text{costo}\\ \text{fijo}\end{pmatrix} + \begin{pmatrix}\text{costo de}\\ \text{almacenamiento}\end{pmatrix} + \begin{pmatrix}\text{costo de}\\ \text{escasez}\end{pmatrix}$$

El *costo de compra* se vuelve un factor importante cuando el precio de 1 unidad de mercancía depende del tamaño del pedido. Esta situación se expresa normalmente en términos de un *descuento por cantidad* o una *reducción del precio* donde el precio unitario del artículo disminuye con el incremento de la cantidad ordenada.

El *costo fijo* representa el gasto fijo (no variable) en que se incurre cuando se hace un pedido. Por lo tanto, para satisfacer la demanda en un periodo, el pedido (más frecuente) de cantidades menores dará origen a un costo fijo mayor durante el periodo que, si se satisficiera la demanda haciendo pedidos mayores (y, por lo tanto, menos frecuentes).

El *costo de almacenamiento* que representa los costos de almacenamiento de los productos en bodega (por ejemplo, interés sobre el capital invertido, almacenamiento, manejo, depreciación y mantenimiento), normalmente aumenta con el nivel del inventario.

El *costo de escasez* es una penalización en la que se incurre cuando se termina la existencia de un producto que se necesita. Por lo general, incluye costos que se acreditan a la pérdida de la benevolencia del cliente y, también, a la pérdida potencial del ingreso.

En la figura siguiente, se ilustra la variación de los cuatro componentes del costo del modelo del inventario general como función del nivel de inventario. El nivel de inventario óptimo corresponde al costo total mínimo de los cuatro componentes. Sin embargo, se puede observar

que un modelo de inventarios no necesita incluir a los cuatro tipos de costos, ya sea porque, algunos son insignificantes o porque complican mucho el modelo; en la práctica, podemos suprimir un componente de costo solo, si su efecto en el modelo del costo total es insignificante.

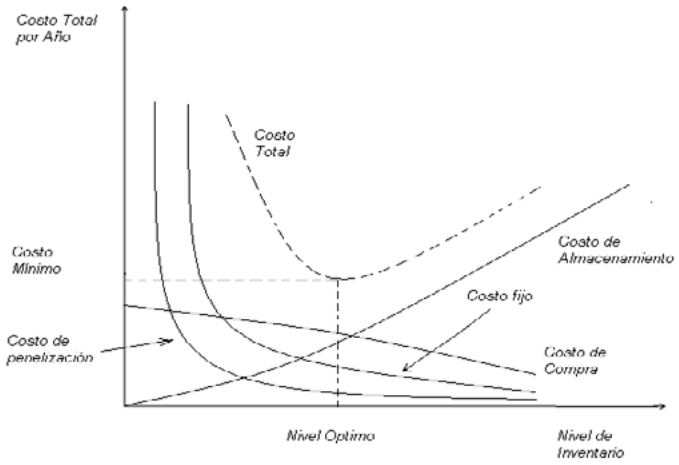


FIGURA 4.1**.** Costos en el modelo de inventarios

El modelo de inventarios anterior parece ser lo suficientemente simple, entonces, ¿Por qué tenemos grandes variedades de modelos cuyos métodos de solución van desde un cálculo simple hasta las refinadas aplicaciones de la PD (Programación Dinámica)?

La respuesta radica principalmente en que la demanda sea determinista o probabilista. En la figura siguiente, se describen los diferentes tipos de demanda como se toman normalmente en los modelos de inventarios.

FIGURA 4.2. Tipos de demanda

Una *demanda determinista* puede ser *estática* en el sentido de que la tasa de consumo permanece constante durante el transcurso del tiempo o *dinámica* donde la demanda se conoce con certeza, pero varía de un periodo al siguiente.

La *demanda probabilista* o *estocástica* tiene dos clasificaciones análogas: el caso *estacionario*, en el cual la función de la densidad de la probabilidad de la demanda se mantiene sin cambio en el tiempo y el caso *no estacionario* donde la función de la densidad de la probabilidad varía con el tiempo.

La representación más precisa de la demanda quizá pueda hacerse a través de distribuciones no estacionarias probabilísticas, sin embargo, desde el punto de vista matemático, el modelo de inventarios resultante será más bien complejo, en especial a medida que aumenta el horizonte de tiempo del problema.

Aunque, el tipo de demanda es un factor principal en el diseño del modelo de inventarios; también, los siguientes factores pueden influir en la forma como se formula el modelo:

- Demoras en la entrega o tiempos guía
- Reabasto del almacén
- Horizonte de tiempo
- Abastecimiento múltiple
- Número de artículos

4.2. MODELOS DETERMINISTAS

Modelo estático de un solo artículo

El tipo más simple del modelo de inventarios ocurre cuando la demanda es constante en el tiempo con reabastecimiento instantáneo y sin escasez. La figura siguiente ilustra la variación del nivel de inventario. Se supone que la demanda ocurre con la tasa D (por unidad de tiempo). El nivel más alto del inventario ocurre cuando se entrega la cantidad ordenada y. (La demora en la entrega se supone una constante conocida). El nivel del inventario alcanza el nivel cero y/D unidades de tiempo, después que se recibe la cantidad pedida y.

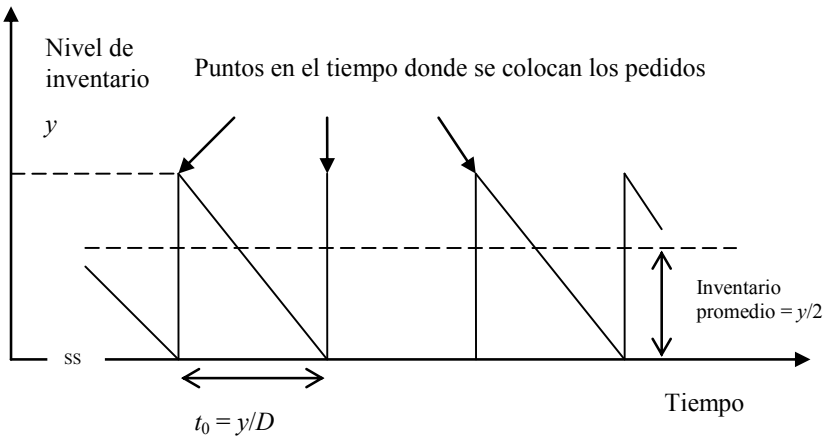


FIGURA 4.3. Modelo estático de un solo artículo

Entre más pequeña es la cantidad *y* ordenada, más frecuente será la colocación de nuevos pedidos. Sin embargo, se reducirá el nivel promedio del inventario mantenido en almacén. Por otra parte, pedidos de mayor cantidad indican un nivel de inventario más grande, pero una colocación menos frecuente de pedidos, como se puede ver en la figura 4.4. Debido a que existen costos asociados al colocar los pedidos y mantener el inventario en el almacén, se selecciona la cantidad *y* para permitir un compromiso entre los dos tipos de costos. Esta es la base para formular el modelo de inventarios.

Entonces, sea *K* el costo fijo originado cada vez que se coloca un pedido y suponga que el costo por mantener 1 unidad de inventario (por unidad de tiempo) es *h.* Por lo tanto, el CTU (Costo Total por Unidad) de tiempo como función de *y* puede expresarse así:

$$CTU(y) = \frac{K}{y/D} + h\left(\frac{y}{2}\right)$$

= (costo fijo/unidad de tiempo) + (costo por mantener inventario/unidad de tiempo)

Como se ve en la figura 4.3, la longitud de cada ciclo de inventario es $t_0 = y/D$ y el inventario promedio en el almacén es $y/2$.

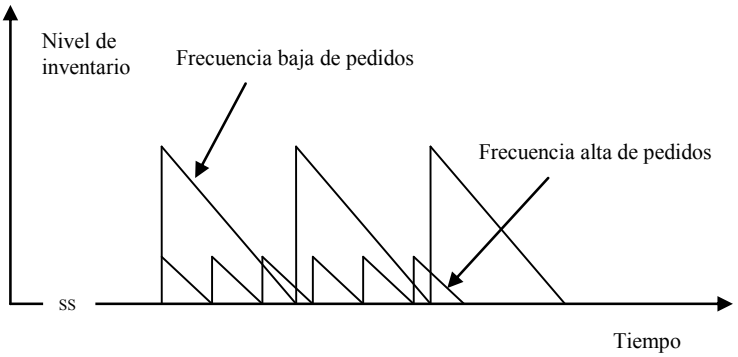


FIGURA 4.4. Diferentes frecuencias en pedidos

El valor óptimo de y se obtiene minimizando CTU (y) con respecto a y. Por consiguiente, suponiendo que y es una variable continua se deduce que:

$$\frac{dCTU(y)}{dy} = -\frac{KD}{y^2} + \frac{h}{2} = 0$$

de donde se obtiene la cantidad de pedido óptimo:

$$y^* = \sqrt{\frac{2KD}{h}}$$

La segunda derivada en y^* es estrictamente positiva, lo cual demuestra que sí minimiza a CTU (y). La cantidad pedida se denomina *tamaño del lote económico de Wilson* o CPE (Cantidad Pedida Económica).

La política óptima del modelo requiere ordenar y^* unidades por cada $t_0^* = y^*/D$ unidades de tiempo. El costo óptimo CTU (y^*) se obtiene por sustitución directa como $\sqrt{2KD/h}$.

En la mayoría de las situaciones prácticas se tiene un tiempo de fabricación positivo L (o de retraso) desde el punto en el cual, se coloca la orden hasta que realmente se entrega. La política de pedidos del modelo anterior, por consiguiente, debe especificar el punto de reorden.

En la figura 4.5, se ilustra la situación donde en el punto de reorden ocurren *L* unidades de tiempo antes de lo esperado para la entrega. En la práctica, esto es equivalente a observar, continuamente, el nivel del inventario hasta que se alcance el punto de reorden, por esta razón, este modelo se conoce como *modelo de revisión continua.* Conforme el sistema se *estabiliza,* el tiempo de fabricación *L* se puede tomar siempre menor que el ciclo t_0*, como se muestra en el ejemplo siguiente:

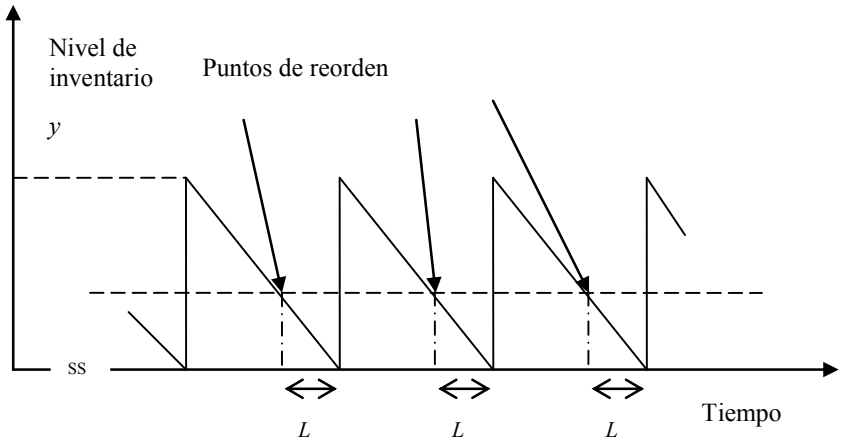


FIGURA 4.5. Reordenar con *L* unidades de tiempo antes de la entrega

Ejemplo 4.1. La demanda diaria *D* para una mercancía es aproximadamente de 100 unidades. Cada vez que se coloca un pedido, se origina un costo fijo *K* de $100.00. El costo diario *h* por mantener el inventario por unidad es de $0.02. Si el tiempo de fabricación es de 12 días, determine el tamaño del lote y el punto de reorden.

De las fórmulas anteriores, el tamaño del lote económico es:

$$y^* = \sqrt{\frac{2KD}{h}} = \sqrt{\frac{2\times 100\times 100}{0.02}} = 1\,000 \text{ unidades}$$

Por lo tanto, la longitud óptima del ciclo asociada está dada por:

$$t_0^* = \frac{y^*}{D} = \frac{1\,000}{100} = 10 \text{ días}$$

Puesto que el tiempo de fabricación es de 12 días y la longitud del ciclo es de 10 días, el volver a pedir ocurre cuando el nivel de inventario es suficiente para satisfacer la demanda para dos días (= 12 – 10). Por consiguiente, la cantidad $y^* = 1\,000$ se ordena cuando el nivel de inventario alcanza $2 \times 100 = 200$ unidades. Se puede observar que el tiempo *efectivo* de fabricación se toma igual a 2 días en lugar de 12 días. Esto ocurre, porque el tiempo es mayor que t^*_0.

Desafortunadamente, las hipótesis del modelo anterior se cumplen en raras ocasiones en la vida real, principalmente, porque la demanda es estocástica. Sin embargo, en la práctica ha surgido un procedimiento bajo una idea muy simple, superponer un almacenamiento de amortiguación (constante) sobre el nivel de inventario en todo el horizonte de planeación. El tamaño del amortiguamiento se determina, de tal manera que, la probabilidad de estar sin artículos en el inventario durante el tiempo de fabricación L no exceda un valor especificado. Suponga que $f(x)$ es la función de densidad de la demanda durante el tiempo de fabricación. Además, infiera que la probabilidad de no tener artículos en el almacén durante L no debe exceder de α. Entonces, el tamaño B de amortiguamiento se determina a partir de:

$$P\{x \geq B + LD\} \leq \alpha$$

Donde LD representa el consumo durante L. La variación del inventario con el amortiguamiento se muestra en la figura siguiente:

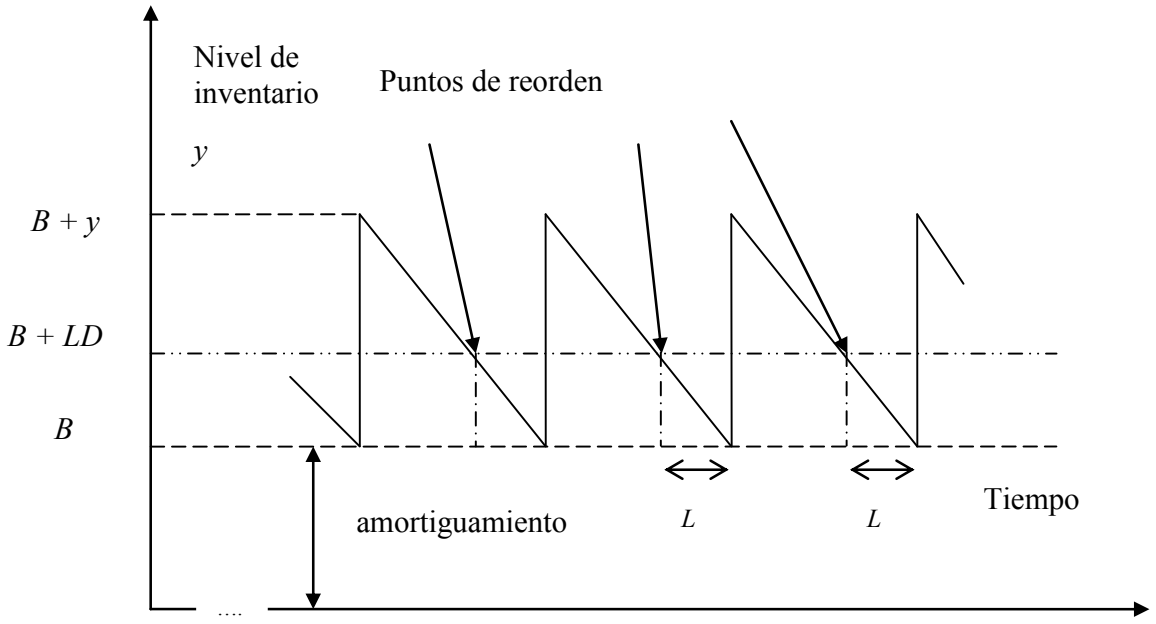


FIGURA 4.6. Puntos de reorden y amortiguamiento

Ejemplo 4.2. Suponga que la demanda en el ejemplo 4.1 es realmente una aproximación de una situación probabilística, en la cual la demanda diaria es normal con una media $\mu = 100$ y una desviación estándar de $\sigma = 10$. Determine el tamaño del almacenamiento amortiguante, de modo que, la probabilidad de estar sin artículos durante el tiempo de fabricación sea, a lo más, de 0.05.

Del ejemplo 4.1, el tiempo mencionado es igual a 2 días. Puesto que la demanda diaria es normal, la demanda x_L en el tiempo de demora, también, es normal con media $\mu_L = 2 \times 100 = 200$ unidades y desviación estándar $\sigma_L = \sqrt{2 \times 10^2} = 14.14$. La figura 4.7 muestra la relación entre la distribución de x_L y el tamaño de amortiguamiento B.

Entonces, se deduce que:

$$P\{x_L \geq \mu_L + B\} \leq \alpha$$

o bien,

$$P\left\{\frac{x_L - \mu_L}{\sigma_L} \geq \frac{B}{\sigma_L}\right\} \leq \alpha$$

en este caso:

$$P\left\{\frac{x_L - \mu_L}{\sigma_L} \geq \frac{B}{14.14}\right\} \leq 0.05$$

De las tablas estándar se tiene $B/14.14 \geq 1.64$, o bien, $B \geq 23.2$.

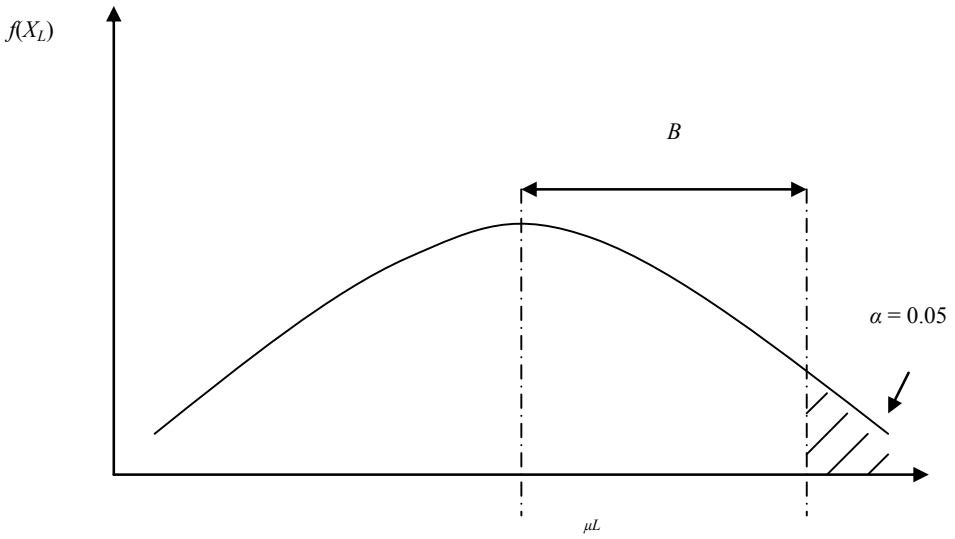


FIGURA 4.7. Distribución x_L y el amortiguamiento B

En este ejemplo es interesante observar que el tamaño de B es independiente de la media μ_L. Esto generalmente es de esperarse, ya que, el factor importante es la desviación estándar, si la desviación estándar es cero (caso determinístico), el tamaño de amortiguamiento debe ser cero.

Modelo estático de un solo artículo con diferentes precios

En los modelos de la sección anterior, el costo de compra por unidad se ignora en el análisis, porque es constante y, por lo tanto, no debe afectar el nivel del inventario. Sin embargo, muchas veces sucede que el precio de compra por unidad depende de la unidad comprada. Generalmente, esto ocurre en forma de rebajas de precio o descuentos según la cantidad y en, tales casos, el precio de compra, sí debe considerarse en el modelo de inventarios. Entonces, se tiene que si el tamaño del pedido y excede un límite dado q, el precio unitario de compra c se expresa:

$$c = \begin{Bmatrix} c_1, & \text{si } y \le q \\ c_2, & \text{si } y > q \end{Bmatrix}, \quad c_1 > c_2$$

por consiguiente,

$$\text{el costo de compra por unidad de tiempo} = \begin{Bmatrix} \dfrac{c_1 y}{t_0} = \dfrac{c_1 y}{\left(\dfrac{y}{D}\right)} = Dc_1, y \le q \\ \dfrac{c_2 y}{t_0} = \dfrac{c_2 y}{\left(\dfrac{y}{D}\right)} = Dc_2, y > q \end{Bmatrix}$$

Y, usando la misma notación de la sección anterior se tiene:

$$CTU(y) = \begin{Bmatrix} CTU_1(y) = Dc_1 + \dfrac{KD}{y} + \dfrac{h}{2} y, y \le q \\ CTU_2(y) = Dc_2 + \dfrac{KD}{y} + \dfrac{h}{2} y, y > q \end{Bmatrix}$$

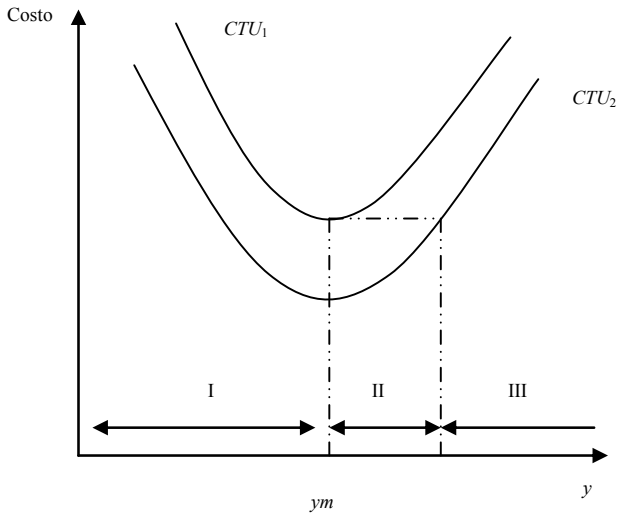


FIGURA 4.8. Función de costo de inventario con reducciones de precio

Las funciones CTU_1 y CTU_2 se pueden ver en la figura 4.8 debido a que las dos funciones solo difieren en una constante y sus mínimos deben coincidir en:

$$y_m = \sqrt{\frac{2KD}{h}}$$

La determinación de la cantidad de pedido óptima y^* depende de dónde queda el punto de reducción de precios q, con respecto a las zonas I, II y III delineadas en la figura 4.8 por los intervalos $(0, y_m)$, (y_m, Q) y (Q, ∞), respectivamente. El valor de Q $(>y_m)$ se determina a partir de la ecuación:

$$CTU_2(Q) = CTU_1(y_m)$$

o bien,

$$c_2 D + \frac{KD}{Q} + \frac{hQ}{2} = CTU_1(y_m)$$

esta se puede simplificar de la manera siguiente:

$$Q^2 + \left(\frac{2(c_2 D - CTU_1(y_m))}{h}\right) Q + \frac{2KD}{h} = 0$$

La figura 4.9 muestra que la cantidad óptima deseada y^* es:

$$y^* = \begin{Bmatrix} y_m, \text{ si } q \text{ se encuentra en las zonas I o III} \\ q, \text{ si se encuentra en la zona II} \end{Bmatrix}$$

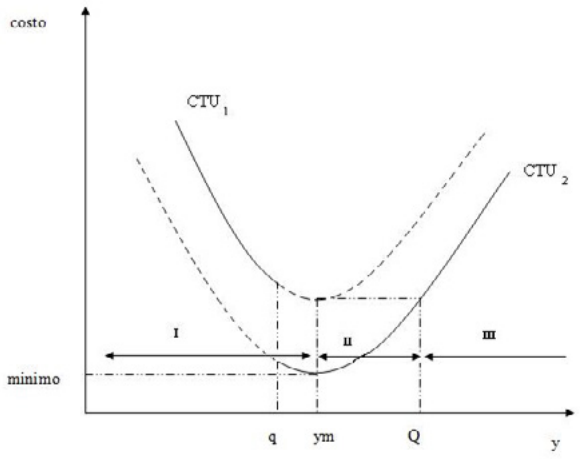
Los pasos para determinar y^* son:

PASO 1: determine $y_m = \sqrt{\frac{2KD}{h}}$, si q está en las zona I, entonces, $y^* = y_m$ de lo contrario vaya al paso 2.

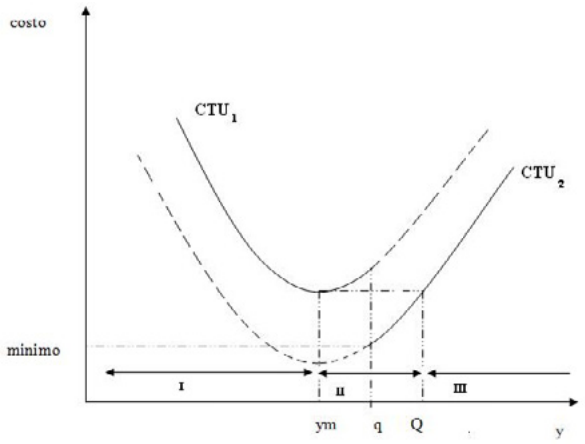
PASO 2: determine $Q(>y_m)$ a partir de la ecuación Q.

$$Q^2 + \left(\frac{2(c_2 D - CTU_1(y_m))}{h} \right) Q + \frac{2KD}{h} = 0$$

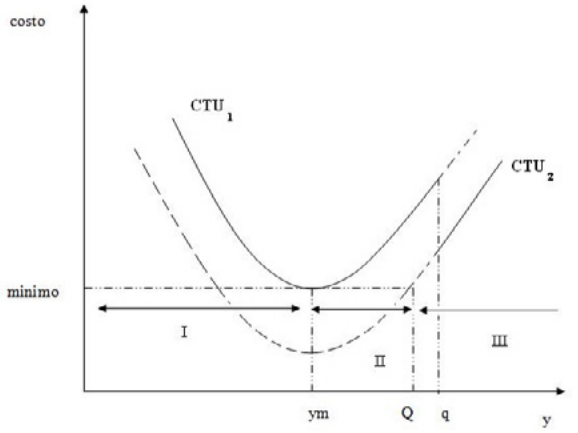
Defina las zonas II y III. Si q está en la zona II, $y^* = q$. De lo contrario, q está en la zona III y, entonces, $y^* = y_m$.



CASO 1. q queda dentro de la zona I, $y^* = y_m$



CASO 2. q queda dentro de la zona II, $y^* = q$



CASO 3. q queda dentro de la zona III, $y^* = y_m$

FIGURA 4.9. Solución óptima de los problemas de inventario con reducciones de precio

Ejemplo 4.3. Fastcar se especializa en cambios de aceite rápidos. El taller compra aceite automotriz a granel a $3.00 el galón con un precio de descuento de $2.50, si la cantidad de pedido es de más de 1 000 galones. El taller atiende, aproximadamente, 150 automóviles por día y cada cambio de aceite requiere 1.25 galones. Fastcar guarda el aceite a granel a un costo de $0.02 por galón, por día. Incluso, el costo de colocar un pedido es de $20.00. El tiempo de espera es de 2 días para la entrega. Determine la política de inventario óptima.

El consumo de aceite por día es:

$$D = 150 \text{ autos por día} \times 1.25 \text{ galones por auto} = 187.5 \text{ galones por día}$$

También, tenemos que:

h = \$0.02 por galón, por día
K = \$20.00 por pedido
L = 2 días
c_1 = \$3.00 por galón
c_2 = \$2.50 por galón
q = 1 000 galones

PASO 1:

calcule:

$$y_m = \sqrt{\frac{2KD}{h}} = \sqrt{\frac{2x20x187.5}{0.02}} = 612.37 \text{ galones}$$

como q = 1 000 es mayor que y_m = 612.37, nos vamos al paso 2.

PASO 2:

determine: Q

$$CTU_1(y_m) = c_1 D + \frac{KD}{y_m} + \frac{hy_m}{2}$$

$$= 3(187.5) + \frac{20(187.5)}{612.37} + \frac{0.02(612.37)}{2} = 574.75$$

por consiguiente, la ecuación Q se calcula como sigue:

$$Q^2 + \left(\frac{2(2.5(187.5) - 574.75)}{0.02} \right) \quad Q + \frac{2(20)(187.5)}{0.02} = 0$$

o bien,

$$Q^2 - 10\,599.74Q + 375\,000 = 0$$

La solución $Q = 10\ 564.25$ ($>y_m$) define las zonas como:

$$\begin{aligned}\text{Zona I} &= (0, 612.37)\\ \text{Zona II} &= (612.37, 10\ 564.25)\\ \text{Zona III} &= (10\ 564.25, \infty)\end{aligned}$$

Por lo tanto, se tiene que $q = 1\ 000$ queda en la zona II, la cual produce la cantidad de pedido óptima $y^* = q = 1\ 000$ galones.

Después de un tiempo de espera de dos días, el punto de volver a pedir es $2D = 2\ (187.5) = 375$ galones. Por lo tanto, la política de inventario óptima es: "pedir 1 000 galones cuando el nivel de inventario se reduzca a 375 galones".

4.3. Modelos dinámicos de la Cantidad Económica de Pedido o EOQ (Economic Order Quantity)

Los modelos en esta sección difieren de los anteriores en dos aspectos:

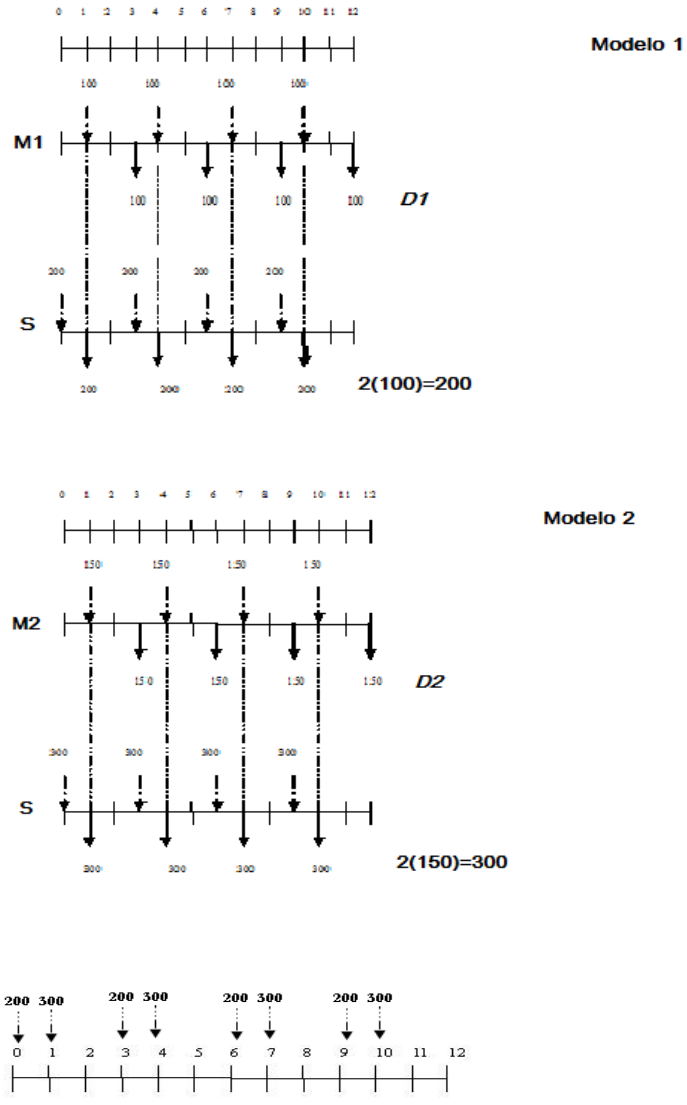
1) El nivel de inventario se revisa periódicamente a lo largo de un número finito de periodos iguales.

2) La demanda por periodo, aun cuando, sea determinística es dinámica en cuanto a las variaciones de un periodo y el siguiente.

Un caso en donde ocurre la demanda determinística dinámica es en la Planeación de Requerimiento de Materiales o MRP (Materials Requirements Planning). La idea de la MRP se describe a través de un ejemplo: suponga que las demandas trimestrales durante el año siguiente para dos modelos finales M_1 y M_2 de un producto dado son 100 y 150 unidades, respectivamente. Al final de cada trimestre, se entregan los lotes trimestrales. El tiempo de espera de la producción es de dos meses para M_1 y de un mes para M_2. Cada unidad de M_1 y M_2 utiliza 2 unidades de un subensamble S. El tiempo de espera para la producción de S es de un mes.

La figura 4.10 muestra los programas de producción para M_1 y M_2. Los programas se inician con la demanda trimestral de los dos modelos (mostrada por flechas sólidas) que ocurre al final de los meses 3, 6, 9 y 12. Dados los tiempos de espera para M_1 y M_2, las flechas de rayas muestran los inicios planeados de cada lote de producción. Para iniciar a tiempo la producción

de los dos modelos, la entrega del subensamble *S* debe coincidir con la ocurrencia de las flechas de rayas M_1 y M_2. Esta información se muestra por medio de las flechas sólidas en la gráfica *S*, donde la demanda *S* resultante es de 2 unidades por unidad de M_1 y M_2. Utilizando un tiempo de espera de acuerdo con estos dos programas, la demanda combinada de *S* correspondiente a M_1 y M_2 se puede determinar, entonces, como se muestra en la parte inferior de la figura 4.10. La demanda variable, pero conocida resultante de *S,* es típica de la situación donde se aplica la EOQ dinámica.



Requerimientos combinados de *S* para los modelos

FIGURA 4.10**.** Programas de producción para M_1 y M_2

Se consideran dos modelos, el primero asume que no hay un costo de preparación (de pedido) y el segundo admite que si lo hay, esta pequeña variación hace la diferencia en la complejidad del modelo.

Modelo de la EOQ sin costo de preparación

Este modelo implica un horizonte de planeación de *n* periodos iguales. Cada periodo tiene una capacidad de producción limitada con uno o más niveles de producción (por ejemplo, el tiempo regular y el tiempo extra representan dos niveles de producción). Un periodo actual puede producir más que su demanda inmediata para satisfacer la necesidad de periodos posteriores, en cuyo caso ocurre un costo de retención.

Las suposiciones generales del modelo son:

1) No se incurre en costo de preparación en ningún periodo.

2) No se permite que hayan faltantes.

3) La función del costo de producción unitario en cualquier periodo es constante o tiene costos marginales crecientes (convexos).

4) El costo de retención unitario en cualquier periodo es constante.

La ausencia de faltantes significa que la producción demorada para periodos futuros no puede satisfacer la demanda en un periodo actual. Esta suposición requiere que la capacidad de producción acumulada para los periodos 1, 2,… e *i* sea igual, al menos, a la demanda acumulada durante los mismos periodos.

La figura 4.11 muestra la función del costo de producción unitario con márgenes crecientes. Por ejemplo, la producción durante el tiempo regular y el tiempo extra corresponden a dos niveles, donde el costo de producción unitario, durante el tiempo extra, excede al del tiempo regular.

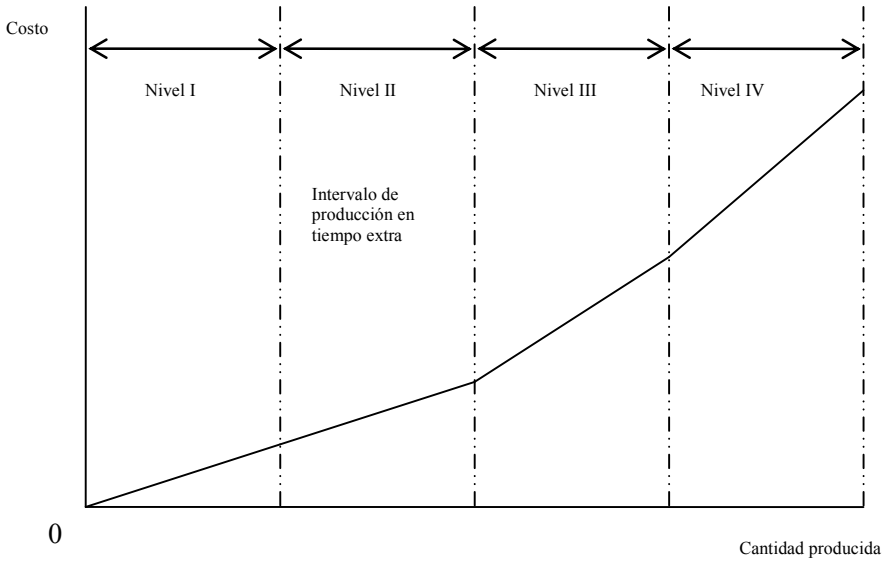


FIGURA 4.11. Función del costo de producción unitario convexa

El problema de *n* periodos puede formularse como un modelo de transporte con *kn* orígenes y *n* destinos, donde *k* es el número de niveles de producción por periodo (por ejemplo $k = 2$, si cada periodo utiliza el tiempo regular y tiempo extra). La capacidad de producción de cada uno de los *kn* orígenes del nivel de la producción es igual a las cantidades de la oferta. Las cantidades demandadas se especifican por la demanda de cada periodo. El costo del *transporte* unitario de un origen a un destino es la suma de los costos de producción y retención o inventario que se aplican por unidad. La solución del problema como un modelo de transporte determina las cantidades de la producción a un costo mínimo en cada nivel de la producción.

El modelo de transporte resultante puede resolverse sin usar la técnica del transporte y la validez del nuevo algoritmo se fundamenta en suponer que no hay faltantes y, así, se tiene una función del costo de la producción convexa.

Ejemplo 4.4. Metalco produce equipos contra incendio que se utilizan en chimeneas domésticas durante los meses de diciembre a marzo. Al inicio, la demanda es lenta, alcanza su máximo a mediados de la temporada y baja hacia el final. Debido a la popularidad del producto, Metalco puede utilizar tiempo extra para satisfacer la demanda. La tabla siguiente proporciona las capacidades de producción y las demandas durante los cuatro meses de invierno.

TABLA 4.1. Capacidad de producción y demanda

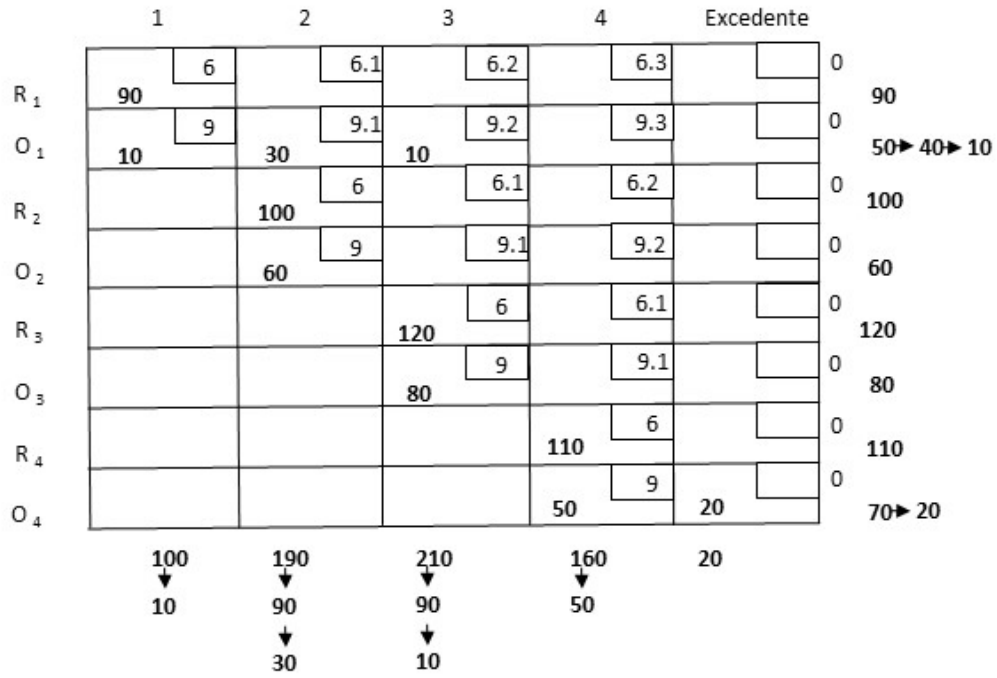
	Capacidad		
Mes	**Tiempo regular (unidades)**	**Tiempo extra (unidades)**	**Demanda (unidades)**
1	90	50	100
2	100	60	190
3	120	80	210
4	110	70	160

El costo de producción unitario en cualquier periodo es de $6.00 durante el tiempo regular y de $9.00 durante el tiempo extra. El costo de retención por unidad y por mes es de $0.10. Para asegurarnos de que el modelo tenga solución factible, cuando no se permiten faltantes, la oferta acumulada de cada mes no puede ser menor que la demanda acumulada, como se muestra en la tabla siguiente.

La tabla siguiente resume el modelo y la solución. Los símbolos R_i y O_i representan niveles de producción durante el tiempo regular y durante el tiempo extra en el periodo i, i = 1,2,3,4. Debido a que la oferta acumulada en el periodo 4 excede a la demanda acumulada, se agrega un destino ficticio para balancear el modelo como se muestra en la tabla. Todas las rutas de *transporte*, desde un periodo anterior a uno actual, están bloqueadas porque no se permiten faltantes.

El costo del *transporte unitario* es la suma de los costos de producción y retención aplicables. Por ejemplo, el costo unitario, del periodo R_1 al periodo 1, es igual al costo de producción unitario únicamente (= $6.00), en tanto que, el costo unitario de O_1 al periodo 4 es igual al costo de producción unitario en O_1 más el costo de retención unitario, desde el periodo 1 hasta el periodo 4, es decir, 9 + (0.1 + 0.1 + 0.1) = 9.3. El costo unitario para cualquier destino excedente es cero.

TABLA 4.2. Modelo y solución del problema



El modelo se resuelve iniciando en la columna 1 y terminando en la columna excedente. Para cada columna, la demanda se satisface dando prioridad a sus rutas más económicas. Para la columna 1, la ruta (R_1, 1) es la más económica y, por lo tanto, se le asigna la cantidad factible máxima = mín (90, 100) = 90 unidades. Esta asignación deja 10 unidades no satisfechas en la columna 1. La siguiente ruta, más económica en la columna 1, es (O_1, 1), a la cual se le asigna 10 (= mín (50, 10)). Ahora, la demanda durante el periodo 1 está satisfecha.

Si pasamos a la columna 2, las asignaciones en esta columna ocurren en el orden siguiente: 100 unidades a (R_2, 2), 60 unidades a (O_2, 2) y 30 unidades a (O_1, 2). Los costos unitarios de esta asignación son \$6.00, \$9.00 y \$9.10, respectivamente. No utilizamos la ruta (R_1, 2), cuyo costo unitario es de \$6.10, porque toda la oferta de R_1 ya se asignó al periodo 1. Así, se sigue con las demandas de la columna 3 y de la columna 4. La solución óptima se puede ver en negritas en la tabla anterior y se desglosa en la siguiente tabla.

TABLA 4.3. Solución óptima para el problema

Etapa	Programa de producción
Tiempo regular 1	Producir 90 unidades durante la etapa 1
Tiempo extra 1	Producir 50 unidades: 10 durante la etapa 1, 30 durante la 2 y 10 durante la 3
Tiempo regular 2	Producir 100 unidades durante la etapa 2
Tiempo extra 2	Producir 60 unidades durante la etapa 2
Tiempo regular 3	Producir 120 unidades durante la etapa 3
Tiempo extra 3	Producir 80 unidades durante la etapa 3
Tiempo regular 4	Producir 110 unidades durante la etapa 4
Tiempo extra 4	Producir 50 unidades durante la etapa 4, con 20 unidades de capacidad ociosa
El costo asociado es: (90 * $6.00) + (10 * $9.00) + (30 * $9.10) + (100 * $6.00) + (60 * $9.00) + (10 * $9.20) + (120 * $6.00) + (80 * $9.00) + (110 * $6.00) + (50 * $9.00) = $4 685.00	

Modelo de la EOQ con costo de preparación

En esta situación no se permiten faltantes y se incurre en un costo de preparación cada vez que se inicia un nuevo lote de producción, como se puede ver en la figura 4.12.

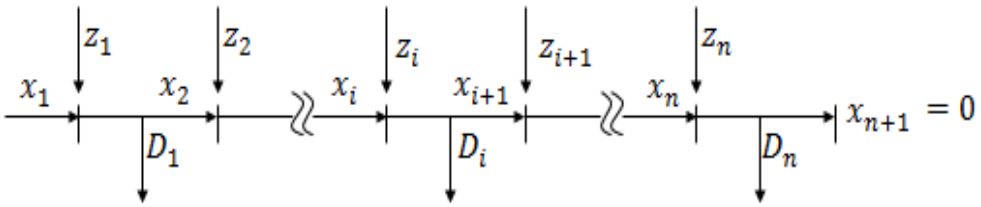


FIGURA 4.12. Modelo dinámico de inventario con costo de preparación

La figura resume, esquemáticamente, la situación del inventario. Los símbolos mostrados en la figura 4.12 se definen para el periodo $i, i = 1, 2, \dots, n$, como:

z_i = cantidad pedida
D_i = demanda durante el periodo
x_i = inventario al inicio del periodo i

Los elementos de los costos de la situación se definen como:

K_i = costo de preparación en el periodo i
h_i = costo de retención del inventario unitario del periodo i a $i + 1$

La función del costo de producción asociado para el periodo i es:

$$C_i(z_i) = \begin{cases} 0, & z_i = 0 \\ K_i + c_i(z_i), & z_i > 0 \end{cases}$$

La función $c_i(z_i)$ es la función del costo de producción marginal, dada z_i.

4.4. ALGORITMO DE PDG (PROGRAMACIÓN DINÁMICA GENERAL)

Sin faltantes, el modelo de inventario se basa en minimizar la suma de los costos de la producción y retención o inventario en los n periodos. A fin de simplificar, supondremos que el costo de retención en el periodo i se basa en el inventario al final del periodo, definido como:

$$x_{i+1} = x_i + z_i - D_i$$

Para la ecuación recursiva hacia adelante o de avance, el *estado* en la *etapa* (periodo) i se define como x_{i+1} o nivel del inventario al final del periodo. En el caso extremo, el inventario restante x_{i+1} puede satisfacer la demanda en todos los periodos restantes, es decir:

$$0 \leq x_{i+1} \leq D_{i+1} + \cdots + D_n$$

Sea $f_i(x_{i+1})$ el costo mínimo del inventario para los periodos 1, 2, … , e i, dado el inventario al final del periodo x_{i+1}.

La ecuación recursiva hacia adelante es:

$$f_1(x_2) = \min_{z_1 = D_1 + x_2 - x_1} \{C_1(z_1) + h_1 x_2\}$$

$$f_i(x_{i+1}) = \min_{0 \le z_i \le D_i + x_{i+1}} \{C_i(z_i) + h_i x_{i+1} + f_{i-1}(x_{i+1} + D_i - z_i)\}, i = 2,3, \dots, n$$

Durante el periodo 1, z_1 observe que es exactamente igual a $D_1 + x_2 - x_1$. Para $i > 1$, z_i puede ser cero, porque D_i puede satisfacerse a partir de la producción en periodos precedentes.

Ejemplo 4.5. La siguiente tabla proporciona los datos de una situación de inventario de tres periodos.

TABLA 4.4. Datos de inventario en tres etapas

Periodo i	**Demanda** D_i **(unidades)**	**Costo de preparación** K_i **($)**	**Costo de retención** h_i **($)**
1	3	3	1
2	2	7	3
3	4	6	2

La demanda ocurre en unidades discretas y el inventario de inicio es $x_1 = 1$ unidad. El costo de producción unitario $C_i(z_i)$ es de $10.00 para las primeras 3 unidades y de $20.00 para cada unidad adicional, por lo que, el costo de esas unidades adicionales lleva, también, el costo de 3 (10) = 30 por las 3 primeras unidades, es decir:

$$C_i(z_i) = \begin{cases} 10z_i, & 0 \le z_i \le 3 \\ 30 + 20(z_i - 3), & z_i \ge 4 \end{cases}$$

donde:

z_i = cantidad pedida
D_i = demanda durante el periodo
x_i = inventario al inicio del periodo i
K_i = costo de preparación en el periodo i
h_i = costo de retención de inventario unitario del periodo i a $i + 1$

Bien, determine la política de inventario óptima:

Para usar la misma terminología empleada en los capítulos anteriores, utilizaremos etapa en lugar de periodo.

ETAPA 1. $D_1 = 3, 0 \leq x_2 \leq 2 + 4 = 6,\ z_1 - D_1 + x_1 = x_2 + 2$

$C_1(z_1) + h_1 x_2$										
		$z_1 = 2$	3	4	5	6	7	8	Solución óptima	
x_2	$h_1 x_2$	$C_1(z_1) = 23$	33	53	73	93	113	133	$f_1(x_2)$	z_1^*
0	0	23							23	2
1	1		34						34	3
2	2			55					55	4
3	3				76				76	5
4	4					97			97	6
5	5						118		118	7
6	6							139	139	8

Y, observe que debido a que $x_1 = 1$, el valor mínimo de z_1 es $D_1 - x_1 = 3 - 1 = 2$

ETAPA 2. $D_2 = 2,\ 0 \leq x_3 \leq 4,\ 0 \leq z_2 \leq D_2 + x_3 = x_3 + 2$

$C_2(z_2) + h_2 x_3 + f_1(x_3 + D_2 - z_2)$										
		$z_2 = 0$	1	2	3	4	5	6	Solución óptima	
x_3	$h_2 x_3$	$C_2(z_2)0$	17	27	37	57	77	97	$f_2(x_3)$	z_2^*
0	0	0+55= 55	17+34= 51	27+23= 50					50	2
1	3	3+76= 79	20+55= 75	30+34= 64	40+23= 63				63	3
2	6	6+97= 103	23+76= 99	33+55= 101	43+34= 77	63+23= 86			77	3
3	9	9+118= 127	26+97= 123	36+76= 112	46+55= 101	66+34= 100	86+23= 109		100	4
4	12	12+139= 151	29+118= 147	39+97= 136	49+76= 125	69+55= 124	89+34= 123	109+23= 132	123	5

ETAPA 3. $D_3 = 4,\ x_4 = 0,\ 0 \leq z_3 \leq D_3 + x_4 = 4$

$C_3(z_3) + h_3x_4 + f_2(x_4 + D_3 - z_3)$								
		$z_3 = 0$	1	2	3	4	Solución óptima	
x_4	h_3x_4	$C_3(z_3) = 0$	16	26	36	56	$f_3(x_4)$	z_3^*
0	0	0+123= 123	16+100= 116	26+77= 103	36+63= 99	56+50= 106	99	3

La solución óptima se lee como sigue:

$$(x_4 = 0) \rightarrow [z_3 = 3] \rightarrow (x_3 = 0 + 4 - 3 = 1) \rightarrow [z_2 = 3] \rightarrow (x_2 = 1 + 2 - 3 = 0) \rightarrow [z_1 = 2]$$

Por lo tanto, la solución óptima es $z_1^* = 2,\ z_2^* = 3,\ z_3^* = 3$, con un costo total de $99.00.

Lo que significa que para el periodo 1 se piden 2 unidades, porque se cuenta con un inventario inicial de 1 unidad, entonces, la demanda se satisface. Para el periodo 2 se piden 3 con lo que se cubre la demanda de 2 unidades y se deja 1 en inventario para el periodo siguiente, por lo que, en el periodo 3 se piden 3 y con esto satisfacemos las demanda de 4 unidades con un costo total de $99.00.

4.5. ALGORITMO DE PD (PROGRAMACIÓN DINÁMICA) CON COSTOS MARGINALES CONSTANTES O DECRECIENTES

La PDG expuesta antes es aplicable con cualquier función de costos. Esta generalización dicta que el estado x_1 y las alternativas z_i en la etapa i asuman valores en incrementos de 1, lo que podría dar lugar a tablas grandes cuando las cantidades demandadas son grandes.

Un caso especial del modelo de PDG promete reducir el volumen de los cálculos. En esta situación especial, tanto el costo de producción unitario como los costos de retención unitarios son funciones *no crecientes* (cóncavas) de la cantidad de producción y el nivel del inventario, respectivamente. Esta situación suele ocurrir cuando la función del costo unitario es constante o si se permite el descuento por cantidad. En las condiciones dadas se puede demostrar que:

1) Ya que un inventario inicial cero (x_i) es óptimo para satisfacer la demanda en cualquier periodo i o con una nueva producción con inventario entrante, pero nunca con ambos, es

decir, $z_i x_i = 0$. (En el caso de inventario inicial positivo, $x_i > 0$, la cantidad puede amortizarse con las demandas de los periodos sucesivos hasta que se agote).

2) La cantidad de producción óptima z_i, durante el periodo i, debe ser cero o satisfacer la demanda exacta de uno o más periodos subsiguientes contiguos.

Ejemplo 4.6. Un modelo de inventario de cuatro periodos opera con los siguientes datos:

TABLA 4.5. Datos para un inventario de cuatro periodos

Periodo i	**Demanda D_i (unidades)**	**Costo de preparación K_i ($)**
1	76	98
2	26	114
3	90	185
4	67	70

El inventario inicial x_i es de 15 unidades, el costo de producción unitario es de $2.00 y el costo de retención unitario es de $1.00 durante todos los periodos. (Para simplificar, los costos de producción y retención unitarios son los mismos durante todos los periodos).

La solución se determina por el algoritmo hacia adelante ya proporcionado, excepto que los valores de x_{i+1} y z_i, ahora, suponen sumas *concentradas* en lugar de con incrementos de uno. Debido a que $x_1 = 15$, la demanda del primer periodo se ajusta a 76 – 15 = 61 unidades.

Esquemáticamente, se puede ver en el siguiente diagrama:

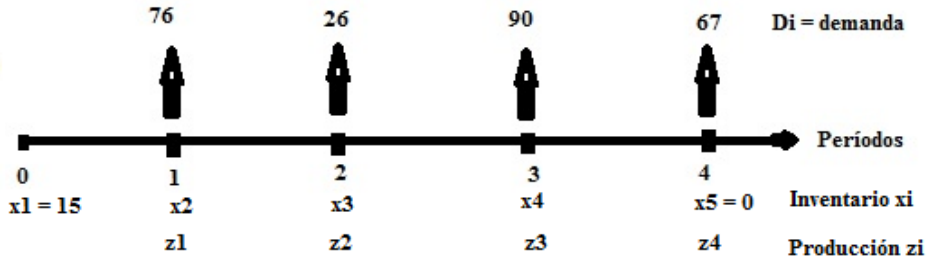


FIGURA 4.13. Diagrama del planteamiento con PD

ETAPA 1. $D_1 = 61$

$C_1(z_1) + h_1x_2$							
		$z_1 = 61$	87	177	244	Solución óptima	
x_2	h_1x_2	$C_1(z_1) = 220$	272	452	586	$f_1(x_2)$	z_1^*
0	0	220				220	61
26	26		298			298	87
116	116			568		568	177
183	183				769	769	244
Pedir en 1 para		1	1,2	1,2,3	1,2,3,4		

ETAPA 2. $D_2 = 26$

$C_2(z_2) + h_2x_3 + f_1(x_3 + D_2 - z_2)$							
		$z_2 = 0$	26	116	183	Solución óptima	
x_3	h_2x_3	$C_2(z_2)$	166	346	480	$f_1(x_2)$	z_2^*
0	0	0 + 298 = 298	166 + 220 = 386			298	0
90	90	90 + 568 = 658		436 + 220 = 656		656	116
157	157	157 + 769 = 926			637 + 220 = 857	857	183
Pedir en 2 para		-	2	2,3	2,3,4		

ETAPA 3. $D_3 = 90$

$C_3(z_3) + h_3x_4 + f_2(x_4 + D_3 - z_3)$						
		$z_3 = 0$	90	157	Solución óptima	
x_4	h_3x_4	$C_3(z_3) = 0$	365	499	$f_3(x_4)$	z_3^*
0	0	0 + 656 = 656	365 + 298 = 663		656	0
67	67	67 + 857 = 924		566 + 298 = 864	864	157
Pedir en 3 para		-	3	3,4		

ETAPA 4. $D_4 = 67$

$C_4(z_4) + h_4x_5 + f_3(x_5 + D_4 - z_4)$					
		$z_4 = 0$	67	Solución óptima	
x_5	h_4x_5	$C_4(z_4) = 0$	204	$f_4(x_5)$	z_4^*
0	0	0 + 864 = 864	204 + 656 = 860	860	67
Pedir en 4 para		-	4		

La política óptima se determina a partir de las tablas como sigue:

$$(x_5 = 0) \rightarrow [z_4 = 67] \rightarrow (x_4 = 0) \rightarrow [z_3 = 0] \rightarrow (x_3 = 90) \rightarrow [z_2 = 116] \rightarrow (x_2 = 0) \rightarrow [z_1 = 61]$$

Esto da $z_1^* = 61,\ z_2^* = 116,\ z_3^* = 0,$ y $z_4^* = 67$ a un costo total de $860.00. Lo que significa que se produce en los periodos 1, 2 y 4 para cubrir las demandas de dichos periodos y en el periodo 2 se produce para el periodo 3. Esta solución se puede visualizar en el siguiente diagrama:

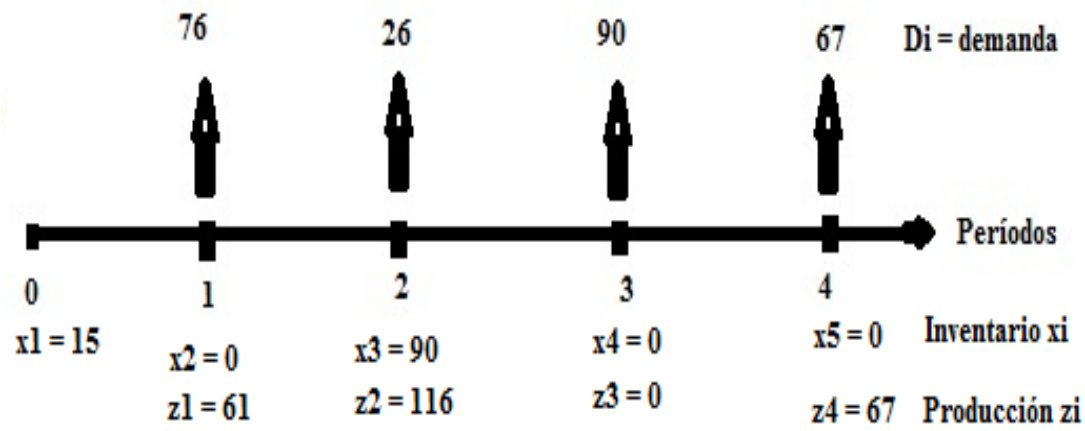


FIGURA 4.14. Diagrama de solución usando PD

Para verificar el modelo de PD, este mismo problema de inventario se puede plantear como un problema de PLEMB (Programación Lineal Entera Mixta Binaria) como se expone a continuación.

Sean:

Xi = cantidad de inventario para la etapa *i*
zi = cantidad de artículos producidos en la etapa *i*
yi = variable binaria que sirve para decidir cuándo se produce para satisfacer la demanda en la etapa *i*

Escribiendo este problema en LINGO se tiene:

```
min= 2*z1+2*z2+2*z3+2*z4+98*y1+114*y2+185*y3+70*y4+x1+x2+x3+x4;


15–x1+z1=76;
x1+z2–x2=26;
x2+z3–x3=90;
x3+z4–x4=67;
x4=0;
z1<=183*y1;
z2<=183*y2;
z3<=157*y3;
z4<=67*y4;


@bin(y1);
@bin(y2);
@bin(y3);
@bin(y4);
```

Esquemáticamente, este planteamiento se puede ver en el siguiente diagrama:

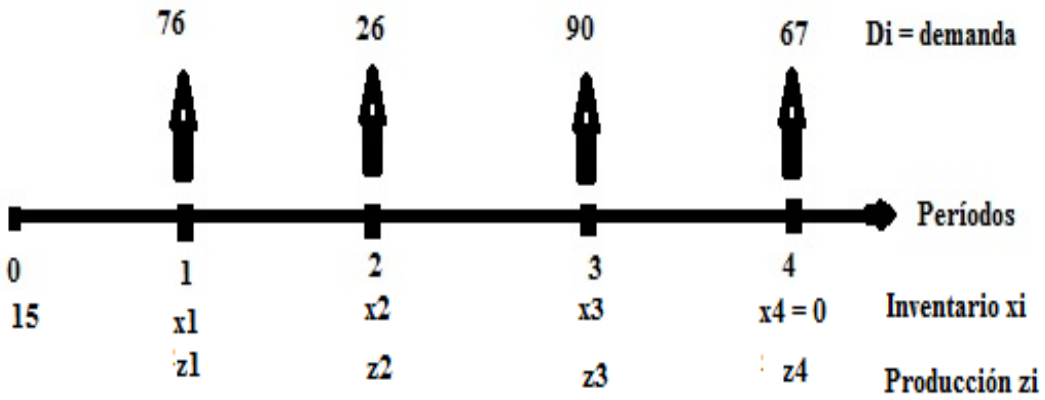


FIGURA 4.15. Diagrama del planteamiento con PLEM (Programación Lineal Entera Mixta)

Cuya solución está dada en la siguiente tabla:

```
Global optimal solution found.
Objective value:      860.0000


Variable    Value
  Z1       61.00000
  Z2       116.000000
  Z3       0.0000
  Z4       67.00000
  Y1       1.000000
  Y2       1.000000
  Y3       0.000000
  Y4       1.000000
  X1       0.00000
  X2       90.00000
  X3       0.000000
  X4       0.000000
```

Como se puede verificar, el resultado óptimo es el mismo y las variaciones en las variables son, porque en este modelo no se consideran como en el caso dinámico de manera recursiva, sino que se plantean en cada periodo y se pueden ver de la siguiente manera:

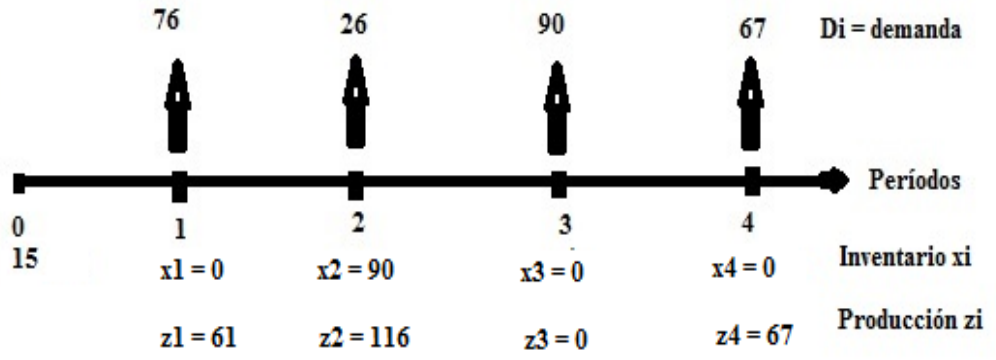


FIGURA 4.16. Diagrama de solución usando PLEM

Una ligera variación en el modelo de PD es cuando los costos son cóncavos.

Dado que $z = (z_1, z_2, ..., z_n)$ se define el conjunto de todas las políticas de producción factibles como:

$$S = \left\{ z : z_i \geq 0 \quad \text{y} \quad \sum_{j=1}^{i} z_j - \sum_{j=1}^{i} d_j \geq 0 \quad \text{para } i = 1,2,..,n \qquad \sum_{j=1}^{n} z_j - \sum_{j=1}^{n} d_j = 0 \right\}$$

Como la función de costos es cóncava, se cumple el teorema que dice que existe una política óptima en un punto extremo de *S*, por lo tanto, para cada *i*: $x_{i-1} = 0$ o $z_i = 0$ o ambos. En otros términos $x_{i-1}\, z_i = 0$ lo que, también, se conoce como la *propiedad de punto de regeneración* (p.p.r.).

Con base en esto, podemos afirmar que para la función de costos se cumple lo siguiente:

Para $j \leq i$, sea $c(j, i)$ el costo total en los periodos de *j* a *i,* dado que los periodos $j–1$ e *i* cumplen con p.p.r., y la única producción durante los periodos *j* al *i* es en el periodo *j*, entonces, $c(j, i)$ está dado por:

$$c(j,i) = c_j\left(\sum_{s=j}^{i} d_s\right) + \sum_{s=j+1}^{i} c_s(0) + \sum_{s=j}^{i-1} h_s\left(\sum_{t=s+1}^{i} d_t\right)$$

De acuerdo a esta expresión se tiene que una política óptima se caracteriza con la siguiente función de valor óptimo:

f_i = al mínimo costo para los periodos 1 a *i,* dado que se debe tener un inventario de 0 artículos al final del periodo *i*.

Donde f_i satisface la siguiente ecuación recursiva:

$$f_i = \min_{1 \leq j \leq i}\left[f_{j-1} + c(j,i) \right] \qquad i = 1,2..,n$$

Y con la condición de frontera:

$$f_0 = 0$$

También, esta ecuación recursiva se puede usar para encontrar la ruta más corta en una red acíclica. En el siguiente ejemplo, se consideran estos costos y esta recursividad.

Ejemplo 4.7. Considere un modelo de inventario de cuatro periodos con los datos siguientes:

Demandas por periodo:

$$d_1 = 1 \qquad d_2 = 2 \qquad d_3 = 2 \qquad d_4 = 1$$

Los costos de producción:

$$c(0) = 0 \qquad c(1) = 5 \qquad c(2) = 9 \qquad c(3) = 12 \qquad c(4) = 14$$
$$c(5) = 16 \qquad c(6) = 18$$

Y el costo de inventario es igual a $h(x) = x$ para $x \geq 0$, entonces, la solución al problema es la siguiente.

Si estimamos que el costo $c(j, i)$ reside en el costo para producir los artículos de toda la demanda i en el periodo j, se tiene lo siguiente:

Para el periodo o etapa 1, los costos son:

$c(1, 1) = c(1) = 5$

$c(1, 2) = c(3) + h(2) = 12 + 2 = 14$

$c(1, 3) = c(5) + h(4) + h(2) = 16 + 4 + 2 = 22$

$c(1, 4) = c(6) + h(5) + h(3) + h(1) = 18 + 5 + 3 + 1 = 27$

Esquemáticamente, se puede ver de la siguiente manera:

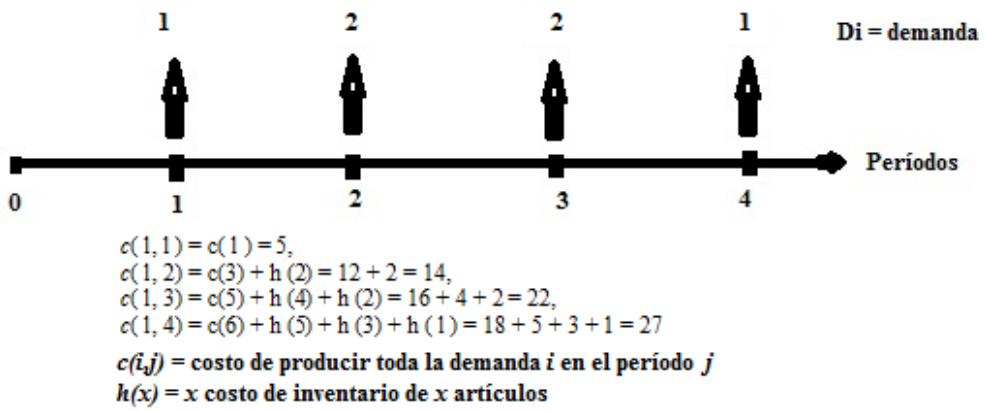


FIGURA 4.17. Diagrama de costos para el primer periodo

Para la etapa 2, siguiendo el mismo razonamiento se tiene:

$c(2, 2) = c(2) = 9$
$c(2, 3) = c(4) + h(2) = 14 + 2 = 16$
$c(2, 4) = c(5) + h(3) + h(1) = 16 + 3 + 1 = 20$

Para la etapa 3:

$c(3, 3) = c(2) = 9$
$c(3, 4) = c(3) + h(1) = 12 + 1 = 13$

Para la etapa 4:
$c(4, 4) = c(1) = 5$

Para aplicar estos costos en las ecuaciones recursivas hacia adelante, se plantea la condición de frontera como:

$f_0 = 0$

Y para las etapas $i = 1, 2, 3, 4$ se tiene:

$f_1 = f_0 + c(1, 1) = 0 + 5 = 5,$

La producción para la etapa $j = 1$ $\quad j(1) = 1;$

$f_2 = \text{mín}\,[f_0 + c(1, 2),\ f_1 + c(2, 2)] = \text{mín}[0 + 14,\ 5 + 9] = 14,$

La producción para la etapa $j = 2$ $\quad j(2) = 1$ o $2;$

$f_3 = \text{mín}\,[f_0 + c(1, 3),\ f_1 + c(2, 3),\ f_2 + c(3, 3)] = \text{mín}[0 + 22,\ 5 + 16,\ 14 + 9] = 21,$

La producción para la etapa $j = 3$ $\quad j(3) = 2$, es en la etapa 2;

$$f_4 = \text{mín}\,[f_0 + c(1, 4),\ f_1 + c(2, 4),\ f_2 + c(3, 4),\ f_3 + c(4, 4)]$$
$$= \text{mín}\,[0 + 27,\ 5 + 20,\ 14 + 13,\ 21 + 5] = 25,$$

La producción para la etapa $j = 4$ $\quad j(4) = 2$, es en la etapa 2; entonces, las etapas donde hay producción son la 1 y la 2 con una política de producción óptima de:

$$z_1 = 1 \qquad z_2 = 5 \qquad z_3 = 0 \qquad z_4 = 0$$

y un costo total de $25.00.

La solución se ilustra en el siguiente diagrama:

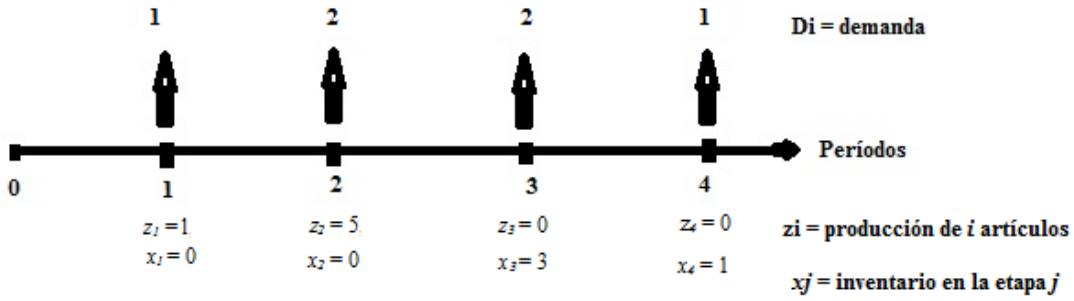


FIGURA 4.18. Solución para las cuatro etapas

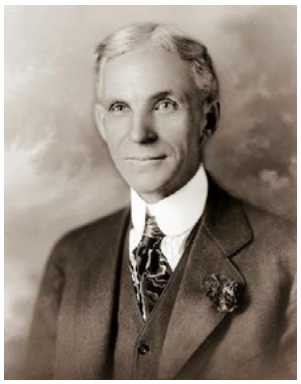
4.6. CONCLUSIONES

La teoría de inventarios es muy basta y no se abordan todos los modelos que existen en este capítulo, el objetivo es introducir al lector al tema y a través de ejemplos mostrar los modelos

que se resuelven usando PD. Como ya se comentó antes en estos apuntes, se presenta una solución para un ejemplo usando PLEM como una manera de validar el modelo dinámico, sin embargo, para el lector que no esté familiarizado con la modelación usando la PLE (Programación Lineal Entera) puede buscar en la literatura otras alternativas de solución, como lo son algunos heurísticos. De cualquier manera, en el anexo C de estos apuntes se presentan algunos modelos de PLE.

4.7. NOTAS HISTÓRICAS

La **Cantidad Económica de Pedido** (conocida en inglés como ***Economic Order Quantity*** o por las siglas ***EOQ***), es el modelo fundamental para el control de inventarios. Es un método que, tomando en cuenta la demanda determinista de un producto (es decir, una demanda conocida y constante), el costo de mantener el inventario, y el costo de ordenar un pedido, produce como salida la cantidad óptima de unidades a pedir para minimizar costos por mantenimiento del producto. El principio del EOQ es simple, y se basa en encontrar el punto en el que los costos por ordenar un producto y los costos por mantenerlo en inventario son iguales.



Este modelo fue desarrollado en 1913 por **Ford Whitman Harris**, un ingeniero que trabajaba en Westinghouse Corporation, aunque el artículo original en el que se presentaba el modelo fue incorrectamente citado durante muchos años.

Posteriormente la publicación de Harris fue analizada a profundidad y aplicada extensivamente por el consultor **R. H. Wilson**, quien publicó un artículo en 1934 que popularizó el modelo. Por esta razón, este también suele ser conocido como el **Modelo de Wilson**.[26]

Historia de la administración de inventarios

…En el pasado para administrar una tienda, los comerciantes escribían las compras, o miraban a cuántas unidades se habían ido al final del día y luego hicieron todo lo posible para prever las necesidades futuras. La experiencia y la intuición son habilidades clave, pero seguían siendo un método inexacto, incluso cuando se aplica a las operaciones que eran bastante pequeñas para los estándares de hoy en día.

[26] "Cantidad Económica de Pedido", 2015. Disponible en: http://es.wikipedia.org/wiki/Cantidad_Econ%C3%B3mica_de_Pedido

Después de la Revolución Industrial, la eficiencia y la producción en masa se convirtieron en los principales objetivos de las empresas, junto con una mejora de la experiencia del cliente en el punto de venta. Un equipo de la Universidad de Harvard diseñó el primer sistema de "check-out" moderno a principios de 1930. Se utilizaban tarjetas perforadas que correspondían con artículos del catálogo. Una computadora podría leer las tarjetas perforadas y pasar la información a la bodega, para surtir el pedido del cliente en espera. Debido al sistema automatizado, las máquinas también podrían generar registros de facturación y gestión de inventario. En ese tiempo el sistema resultó ser demasiado caro para su uso, y actualmente hay una versión del mismo que está en uso en algunas tiendas, donde los comerciantes colocan tarjetas con información sobre los productos en el pasillo que los clientes pueden seleccionar y llevar a la cola de la caja. Esto por lo general se aplica a los artículos que son caros o grandes y para artículos controlados, como los medicamentos.

Los comerciantes sabían que necesitaban un mejor sistema, y los investigadores crearon el precursor del sistema de códigos de barras moderno a finales de 1940 y principios de 1950. Se usaba tinta sensible a la luz ultravioleta y un lector para marcar artículos a la venta. Una vez más, el sistema era demasiado engorroso y carecía de la potencia de cálculo necesario para hacer que funcionara.

El desarrollo de la tecnología láser asequible en la década de 1960 revivió el concepto. Los láseres permiten rápidos lectores o escáneres más pequeños y más baratos. El código de barras moderno, o el Código Universal de Producto (UPC), se creó justo antes de la década de 1970. Como el poder de cómputo mejoró, con ello el poder de los códigos UPC para ayudar a rastrear y gestionar el inventario también mejoró de manera exponencial.

Durante mediados y finales de 1990, los comerciantes comenzaron a implementar los sistemas modernos de gestión de inventario, lo que fue posible en gran parte por los avances en la tecnología informática y el *software*. Los sistemas funcionan en un proceso circular, desde el seguimiento de la compra del artículo para dar seguimiento al inventario, ver si es necesario para volver a ordenar y se regresa de nuevo.

En los últimos años, otra tecnología prometedora para el seguimiento de inventario que se ha hecho popular en tiendas, almacenes y fábricas es la identificación por radiofrecuencia, o RFID, que utiliza un microchip para transmitir información de productos -como el tipo, fabricante y número de serie- a un escáner u otro dispositivo de levantamiento de datos. Es superior a los códigos de barras de varias maneras. Por ejemplo, un escáner lee la información de un RFID desde varios metros de distancia, por lo que es ideal para el seguimiento de artículos apilados en estantes altos en los almacenes. También puede codificar más datos que un código de barras y en algunos sistemas informa a los comerciantes, si un artículo está fuera de lugar en la tienda, proporcionando excelentes características anti-robo.

Otro medio popular de control automatizado de inventario es el inventario gestionado por el proveedor. En esta disposición, el vendedor es responsable de mantener sus productos en

existencia en el estante de una tienda, y el trabajo del proveedor y el minorista es estar en estrecha colaboración y compartir información de propiedad.

Este sistema también tiene muchas ventajas para los vendedores. Se les permite asegurarse de que sus productos están disponibles y se muestran debidamente. También los pone en estrecho contacto con el minorista y sus datos de ventas. La retroalimentación que el vendedor recibe puede desempeñar un papel importante en su comercialización, investigación y desarrollo.[27]

4.8. EJERCICIOS PROPUESTOS

1) Considere el problema de inventario de tres periodos con los siguientes datos:

Periodo i	**Demanda** w_i	**Costo fijo** K_i	**Costo de inventario** h_i
1	3 unidades	$3.00	$1.00
2	2 unidades	7.00	3.00
3	4 unidades	6.00	2.00

El inventario inicial x_1 es igual a 1. Suponga que el costo de compra marginal es de $10.00 por unidad por las primeras 3 unidades y de $20.00 por cada unidad adicional, entonces:

$$c_i(u_i) = \begin{Bmatrix} 10u_i & 0 \le u_i \le 3 \\ 30 + 20(u_i - 3) & u_i \ge 3 \end{Bmatrix}$$

a) Solucione usando un inventario inicial $x_1 = 4$.

b) Resuelva empleando ecuaciones recursivas de retroceso.

[27] "How Inventory Management Systems Work", trad. Idalia Flores de la Mota. Disponible en: http://money.howstuffworks.com/how-inventory-management-systems-work1.htm

2) Solucione el siguiente problema de inventario determinista de cuatro periodos:

Periodo i	**Demanda** w_i	**Costo fijo** K_i	**Costo de inventario** h_i
1	5 unidades	$5.00	$1.00
2	7 unidades	7.00	1.00
3	11 unidades	9.00	1.00
4	3 unidades	7.00	1.00

El costo de compra por unidad es de 1 para las primeras 6 unidades y 2 para cualquier unidad adicional.

3) Resuelva el ejemplo visto en este capítulo con la siguiente variante: el inventario inicial es $x_1 = 80$ unidades.

Periodo i	**Demanda** w_i	**Costo fijo** K_i
1	76 unidades	$ 98.00
2	26 unidades	114.00
3	90 unidades	185.00
4	67 unidades	70.00

El costo de mantenimiento de inventario por unidad por periodo es constante e igual a $1.00. También, el costo de compra por unidad es igual a $2.00 para todos los periodos.

4) En el ejercicio 3, pruebe a escribirlo como un problema de PLEM.

5) Ya que el siguiente problema de inventario está planteado como un problema de PLEM, resuélvalo usando PD. Puede verificar el resultado resolviendo el modelo con LINGO.

Mes	Max. Producción	Demanda	Costo Producción	Costo Inventario
Enero	120	100	60	15
Febrero	120	130	60	15
Marzo	150	160	55	20
Abril	150	160	55	20
Mayo	150	140	50	20
Junio	150	140	50	20

Se supone un inventario inicial de 50 unidades: $10 = 50$[28].

a) Variables de decisión:

Xt: Unidades a producir en el mes *t* (*t* = 1,..,6 con *t* = 1 => enero; *t* = 6 => junio)

It: Unidades a almacenar en inventario al final del mes *t* (*t* = 1,…,6 con *t* = 1 => enero; *t* = 6 => junio).

b) Función objetivo:

Minimizar los costos de producción e inventario durante el periodo de planificación definido por:

$$\text{Minimizar } Z = 60X1 + 60X2 + 55X3 + 55X4 + 50X5 + 50X6 + 15I1 + 15I2 + 20I3 + 20I4 + 20I5 + 20I6$$

c) Restricciones:

Satisfacer los requerimientos de la demanda (conocida como restricción de balance de inventario):

$X1 + 50 - I1 = 100$ (enero)
$X2 + I1 - I2 = 130$ (febrero)
$X3 + I2 - I3 = 160$ (marzo)
$X4 + I3 - I4 = 160$ (abril)
$X5 + I4 - I5 = 140$ (mayo)
$X6 + I5 - I6 = 140$ (junio)

[28] "Gestión de operaciones". Disponible en: http://www.gestiondeoperaciones.net/programacion_lineal/problema-de-produccion-e-inventario-resuelto-con-solver-de-excel/

Respetar la capacidad máxima de producción mensual:

$X1 \leq 120$
$X2 \leq 120$
$X3 \leq 150$
$X4 \leq 150$
$X5 \leq 150$
$X6 \leq 150$

Condiciones de no negatividad:
$Xt \geq 0 \quad It \geq 0$ para todo t.

CAPÍTULO 5

PROGRAMACIÓN DINÁMICA ESTOCÁSTICA

5.1. INTRODUCCIÓN

En los capítulos previos, hemos supuesto que el costo y los cambios en los estados resultantes de cada decisión se conocen con certeza. Desafortunadamente, en la vida real no es así, la incertidumbre en los datos es lo más común. Uno de los atractivos más importantes de la PD (Programación Dinámica) es que se puede adaptar de forma matemática y computacional a situaciones estocásticas. En el caso estocástico, los estados y los beneficios en cada etapa son probabilísticos, estos casos se originaron, en especial, en modelos estocásticos de inventarios, mismos que se tratan al final del capítulo. Para efectos de sencillez, comenzamos con un ejemplo de la ruta más corta.

Ejemplo 5.1. Ruta más corta estocástica (Dreyfus y Law, 1977).

Se tiene que instruir a un viajante para ir de la ciudad *A* a la ciudad *B* al mínimo costo, la figura 5.1 muestra la red donde se tienen los costos asociados por tramos. Si se hace el recorrido en forma diagonal hacia arriba, se designa por *U*, con una probabilidad de ¾ para que el viajante recuerde nuestro consejo y de ¼ para que lo olvide. De la misma manera, si nuestra instrucción es que se mueva diagonalmente hacia abajo (decisión *D*), lo hará con la probabilidad de ¾, pero en su lugar, se podría mover hacia arriba con la probabilidad de ¼. Este comportamiento lo seguirá en cada vértice, sin importar qué tan buena sea su memoria inicial, por este motivo, no sabemos con precisión cuál será su trayectoria a pesar de nuestro consejo, pero ciertamente, nuestras instrucciones determinan las probabilidades de varios resultados. En este punto, se pueden suponer dos posibles objetivos: minimizar el costo esperado del viaje considerando que el desmemoriado viajero repite su recorrido varias veces y, así, obtener diferentes costos en los diferentes recorridos, en diferentes tramos, por lo que, el costo promedio se minimizaría. El segundo objetivo consistiría en maximizar la

probabilidad de que la trayectoria que sigue nuestro viajero cuesta menos que, un cierto número *Z*, cantidad que puede gastar.

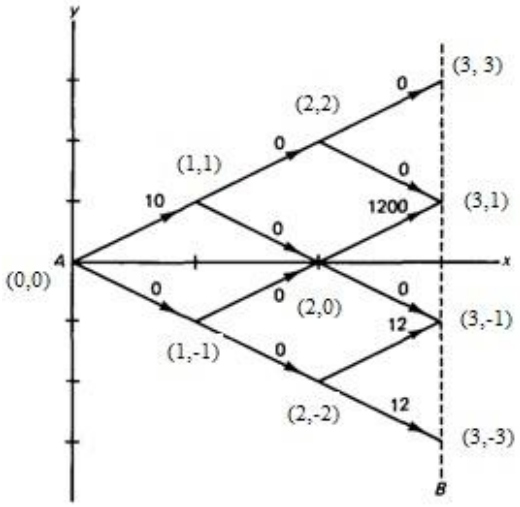


FIGURA 5.1. Red para la ruta más corta

Descripción de una solución en el caso estocástico

En el caso de la ruta más corta determinística se usó la PD, primero para determinar la función de la política óptima que dará una decisión en cada vértice y, entonces, deducir de ahí la secuencia óptima de las decisiones desde el vértice inicial *A*. La política y la secuencia eran dos representaciones de la misma solución óptima. En el caso estocástico, una política y una secuencia son cosas distintas, ya que, una secuencia de decisiones dicta las subsecuentes, independientemente, del resultado de las anteriores, mientras que el resultado de una política es dependiente.

Por ejemplo: la secuencia *ve hacia arriba, luego hacia abajo* y *luego hacia arriba* U-D-U que probablemente no sea óptima, significa que nuestro viajero tratará de recordar en el paso 2, ir hacia abajo con una probabilidad de ¾, no importa, si el viajero ejecutó correctamente la primera instrucción *ir hacia arriba* o no. Si en el paso 1 se está en (1, 1), entonces, se va hacia abajo, pero, si está en (1, –1), porque, olvidó nuestra instrucción y luego trata de recordarla, estaremos en condiciones muy diferentes.

Si se presenta la solución como política o como una secuencia, depende de lo que se nos instruya qué hacer, la secuencia óptima es más fácil de establecer y no requiere que el viajero observe lo que sucede en cada etapa, (lo que significa que observe el estado o la coordenada).

Sin embargo, la política óptima siempre dará un costo promedio tan pequeño como sea posible, mientras que la secuencia óptima es más flexible. La secuencia óptima se puede pensar como una política óptima restringida a la estipulación de que todas las decisiones dadas en una etapa deben ser las mismas no importando el estado. En términos de la teoría de control, se dice que una solución especificada, por una secuencia de decisiones, es un *control de lazo abierto* y para una solución especificada, por una política, se conoce como *control de retroalimentación*. La PD aborda esta última como veremos en la siguiente sección.

5.2. SOLUCIÓN NUMÉRICA DE LA RUTA MÁS CORTA ESTOCÁSTICA

Para considerar un control de lazo abierto del problema de la ruta más corta en la figura 5.1, se valoran las ocho posibles secuencias de las tres decisiones y se escoge una con el menor costo esperado. Por ejemplo, la secuencia *D*-*U*-*D* que optimiza la versión determinística del problema tiene una probabilidad de 27/64 = (¾) (¾) (¾) de que siga la trayectoria consistente de ir hacia abajo, hacia arriba y luego hacia abajo, esta trayectoria tiene un costo cero. La trayectoria *U*-*U*-*D* se sigue con una probabilidad de 9/64 = (¼) (¾) (¾) y un costo de $10.00. Desafortunadamente, también hay una probabilidad de 9/64 para una trayectoria *D*-*U*-*U* con un costo de $1 200.00. Multiplicando cada uno de los ocho costos con sus respectivas probabilidades y sumando para encontrar el costo promedio E_{DUD}, se tiene lo siguiente:

$$E_{DUD} = \frac{27}{64}0 + \frac{9}{64}(10 + 12 + 1\,200) + \frac{3}{64}(12 + 10 + 10) + \frac{1}{64}1\,210 = 192.25$$

La secuencia *U*-*U*-*D* tiene un costo mínimo esperado de $120.18. Se recomienda hacer los cálculos.

Por otro lado, la política de control de retroalimentación se puede calcular recursivamente como en los casos determinísticos. Se empieza por definir la función del valor óptimo esperado por:

$S(x, y)$ = costo esperado del proceso remanente, comenzamos en el vértice (x, y) y usamos la política de retroalimentación de control.

Entonces, si seleccionamos la decisión *U* en (x, y) con probabilidad de ¾ y hacemos la transición a $(x + 1, y + 1)$ con costo para el primer paso de $a_u(x, y)$ y el costo remanente esperado de $S(x + 1, y + 1)$. Con la probabilidad de ¼ hacemos la transición a $(x + 1, y - 1)$ con un costo del primer paso $a_d(x, y)$ y con un costo esperado remanente de $S(x + 1, y - 1)$.

Las probabilidades se invierten para la decisión inicial *D*, en consecuencia, la versión estocástica del principio de optimalidad queda definida de la siguiente manera:

$$S(x,y)=\text{mín}\begin{Bmatrix} U:\frac{3}{4}\left[a_u(x,y)+S(x+1,y+1)\right]+\frac{1}{4}\left[a_d(x,y)+S(x+1,y-1\right] \\ D:\frac{1}{4}\left[a_u(x,y)+S(x+1,y+1)\right]+\frac{3}{4}\left[a_d(x,y)+S(x+1,y-1\right] \end{Bmatrix} \qquad (5.1)$$

las condiciones de frontera son:

$$S(3,\,3)=0 \qquad S(3,\,1)=0 \qquad S(3,-1)=0 \qquad S(3,-3)=0 \qquad (5.2)$$

Así, tenemos que es un problema de tres etapas:

ETAPA 3

$$S(2,2)=\text{mín}\begin{Bmatrix} U:\frac{3}{4}\left[0+S(3,3)\right]+\frac{1}{4}\left[0+S(3,1)\right]=0^{*} \\ D:\frac{1}{4}\left[0+S(3,3)\right]+\frac{3}{4}\left[0+S(3,1\right]=\;0^{*} \end{Bmatrix}$$

$$S(2,0)=\text{mín}\begin{Bmatrix} U:\frac{3}{4}\left[1\;200+S(3,1)\right]+\frac{1}{4}\left[0+S(3,-1)\right]=\frac{3}{4}\left(1\;200+0\right)+0=\frac{3\,600}{4}=900 \\ D:\frac{1}{4}\left[1\;200+S(3,1)\right]+\frac{3}{4}\left[0+S(3,-1)\right]=\frac{1\;200}{4}+0=300^{*} \end{Bmatrix}$$

$$S(2,-2)=\text{mín}\begin{Bmatrix} U:\frac{3}{4}\left[12+S(3,-1)\right]+\frac{1}{4}\left[12+S(3,-3)\right]=\frac{36}{4}+\frac{12}{4}=12^{*} \\ D:\frac{1}{4}\left[12+S(3,-1)\right]+\frac{3}{4}\left[12+S(3,-3)\right]=\frac{12}{4}+\frac{36}{4}=12^{*} \end{Bmatrix}$$

ETAPA 2

$$S(1,1)=\text{mín}\begin{Bmatrix} U:\frac{3}{4}\left[0+S(2,2)\right]+\frac{1}{4}\left[0+S(2,0)\right]=\frac{3}{4}\left(0\right)+\frac{1}{4}\left(300\right)=\;\;75^{*} \\ D:\frac{1}{4}\left[0+S(2,2)\right]+\frac{3}{4}\left[0+S(2,0)\right]=\frac{1}{4}\left(0\right)+\frac{3}{4}\left(300\right)=225 \end{Bmatrix}$$

$$S(1,-1)=\text{mín}\left\{\begin{array}{l} U:\frac{3}{4}\left[0+S(2,0)\right]+\frac{1}{4}\left[0+S(2,-2)\right]=\frac{3}{4}\left(300\right)+\frac{1}{4}\left(12\right)=225+3=228 \\ D:\frac{1}{4}\left[0+S(2,0)\right]+\frac{3}{4}\left[0+S(2,-2)\right]=\frac{1}{4}\left(300\right)+\frac{3}{4}\left(12\right)=75+9= \quad 84^{*} \end{array}\right\}$$

ETAPA 1

$$S(0,0)=\text{mín}\left\{\begin{array}{l} U:\frac{3}{4}\left[10+S(1,1)\right]+\frac{1}{4}\left[0+S(1,-1)\right]=\frac{3}{4}\left(10+75\right)+\frac{1}{4}\left(0+84\right)=63.75+21=84.75 \\ D:\frac{1}{4}\left[10+S(1,1)\right]+\frac{3}{4}\left[0+S(1,-1)\right]=\frac{1}{4}\left(85\right)+\frac{3}{4}\left(84\right)=21.25+63= \quad 84.25^{*} \end{array}\right\}$$

Con el uso de estas ecuaciones tenemos los resultados de la figura 5.2 donde los valores de la función de control óptimo esperado se muestran en círculos y las decisiones óptimas (no necesariamente de transición que son eventos aleatorios) están dados en los cuadros. El costo esperado usando la política de retroalimentación de control óptimo es $84.25.

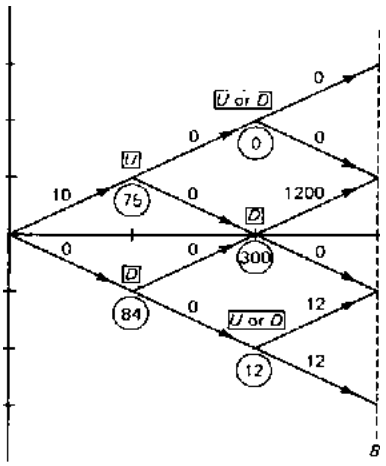


FIGURA 5.2. Solución del problema de la ruta más corta estocástica

La política de retroalimentación de control óptimo es el conjunto de decisiones en donde es una por cada etapa y estado que minimizan el costo esperado, comenzando de (0, 0) que denotamos por $V(0, 0)$. El costo esperado $V(0, 0)$ es la suma sobre todas las trayectorias del producto de las probabilidades para cada trayectoria y sus costos. Entonces, el costo $S(0, 0)$ calculado recursivamente a partir de la ecuación (5.1) y las condiciones de frontera (5.2) es el

costo mínimo esperado $V(0, 0)$ y, por lo tanto, el principio de optimalidad es válido para problemas estocásticos.

Si definimos $p(i, j)$ como la probabilidad de ir, diagonalmente, hacia arriba en el vértice (i, j) y $q(i, j) = 1 - p(i,j)$, entonces, escogemos ¾ o ¼ para cada uno de los seis vértices por definición:

$$V(0,0) = \min_{\substack{p(0,0),p(1,1)\\ p(1,-1),p(2,2)\\ p(2,0),p(2,-2)}} \left\{ \begin{array}{l} p(0,0)\,p(1,1)\,p(2,2)\left[a_u(0,0)+a_u(1,1)+a_u(2,2)\right]+ \\ +p(0,0)\,p(1,1)\,q(2,2)\left[a_u(0,0)+a_u(1,1)+a_d(2,2)\right]+ \\ +p(0,0)\,q(1,1)\,p(2,0)\left[a_u(0,0)+a_d(1,1)+a_u(2,0)\right]+ \\ +p(0,0)\,q(1,1)\,q(2,0)\left[a_u(0,0)+a_d(1,1)+a_d(2,0)\right]+ \\ +q(0,0)\,p(1,-1)\,p(2,0)\left[a_d(0,0)+a_u(1,-1)+a_u(2,0)\right]+ \\ +q(0,0)\,p(1,-1)\,q(2,0)\left[a_d(0,0)+a_u(1,-1)+a_d(2,0)\right]+ \\ +q(0,0)\,q(1,-1)\,p(2,-2)\left[a_d(0,0)+a_d(1,-1)+a_u(2,-2)\right]+ \\ +q(0,0)\,q(1,-1)\,q(2,-2)\left[a_d(0,0)+a_d(1,-1)+a_d(2,-2)\right] \end{array} \right\}$$

Si agrupamos los términos y se escribe la minimización frente a los términos que involucran la variable minimizada, se tiene:

$$\begin{aligned} V(0,0) &= \min_{p(0,0)}\left\{p(0,0)\,a_u(0,0)+q(0,0)a_d(0,0)\right\}+ \\ &+\min_{p(1,1),p(1,-1)}\left\{p(0,0)\,p(1,1)\,a_u(1,1)+p(0,0)\,q(1,1)\,a_d(1,1)+q(0,0)\,p(1,-1)\,a_u(1,-1)+q(0,0)\,q(1,-1)\,a_d(1,-1)\right\}+ \\ &+\min_{p(2,2)}\left\{p(0,0)\,p(1,1)\,p(2,2)\,a_u(2,2)+p(0,0)\,p(1,1)\,q(2,2)\,a_d(2,2)\right\}+ \\ &+\min_{p(2,0)}\left\{p(0,0)\,q(1,1)+q(0,0)\,p(1,-1)\,p(2,0)\,a_u(2,0)+p(0,0)\,q(1,1)+q(0,0)\,p(1,-1)\,q(2,0)\,a_d(2,0)\right\}+ \\ &+\min_{p(2,-2)}\left\{q(0,0)\,q(1,-1)\,p(2,-2)\,a_u(2,-2)+q(0,0)\,q(1,-1)\,q(2,-2)\,a_d(2,-2)\right\} \end{aligned}$$

Y, nuevamente, reagrupando y sustituyendo S dado por (5.1) y (5.2) se tiene:

$$\begin{aligned} V(0,0) &= \min_{p(0,0)}\left\{p(0,0)\,a_u(0,0)+q(0,0)a_d(0,0)\right\}+ \\ &+\min_{p(1,1)}\left\{p(0,0)\,p(1,1)[a_u(1,1)S(2,2)+p(0,0)\,q(1,1)[a_d(1,1)+S(2,0)]\right\}+ \\ &+\min_{p(1,-1)}\left\{q(0,0)\,p(1,-1)[a_u(1.-1)+S(2,0)+q(0,0)\,q(1,-1)[a_d(1,-1)+S(2,-2)]\right\}= \\ &=\min_{p(0,0)}\left\{p(0,0)[a_u(0,0)+S(1,1)]+q(0,0)[a_d(0,0)\right\}+S(1,-1)]=S(0,0) \end{aligned}$$

Esto completa la prueba de la validación del principio de optimalidad para este problema.

5.3. PROBLEMAS DE INVERSIÓN

Ejemplo 5.2. Problema de inversión (Taha, 2004).

Una persona desea invertir hasta $\$C$ miles de pesos en el mercado accionario durante los n años siguientes. El plan de inversión es comprar acciones al inicio del año y venderlas al final del mismo año. Entonces, se puede usar el dinero acumulado para reinvertirlo (todo o en parte) al comienzo del próximo año. El grado de riesgo en la inversión se representa expresando el ingreso en forma probabilística. Un estudio de mercado indica que el retorno sobre la inversión está afectado por m condiciones (favorables o desfavorables) del mercado, y que la condición i produce un ingreso r_i con probabilidad p_i, $i = 1, 2\ldots, m$.

¿Cómo se debe invertir la cantidad C para obtener la máxima acumulación al final de los n años?

Se definen:
x_i = cantidad de fondos disponibles para invertir al inicio del año i ($x_1 = C$).
y_i = cantidad invertida realmente al comenzar el año i ($y_i \leq x_i$).

El modelo de PD se describe como sigue:

- La etapa i está representada por el año i.
- Las alternativas en la etapa i son y_i.
- El estado en la etapa i es x_i.

Ya sean:

$f_i(x_i)$ = fondos esperados máximos para los años i, $i + 1$,.., n dada x_i al comenzar el año i.

Para la condición k del mercado se tiene:

$$x_{i+1} = (1 + r_k)y_i + (x_i - y_i) = x_i + r_k y_i, \quad k = 1, 2, \ldots, m.$$

Como la condición del mercado k se representa con la probabilidad p_k, la ecuación recursiva de PD se expresa como sigue:

$$f_i(x_i) = \max_{0 \leq y_i \leq x_i} \left\{ \sum_{k=1}^{m} p_k f_{i+1}(x_i + r_k y_i) \right\}, i = 1, 2, \ldots, n$$

En donde la condición de frontera es $f_{n+1}(x_{n+1}) = x_{n+1}$, porque, después del año n no se hace inversión. Por lo anterior, se tiene:

$$f_n(x_n) = \max_{0 \le y_n \le x_n} \left\{ \sum_{k=1}^{m} p_k (x_n + r_k y_n) \right\} = x_n + \max_{0 \le y_n \cdot \le x_n} \left\{ \left(\sum_{k=1}^{m} p_k r_k \right) y_n \right\}$$

la esperanza de r se expresa como:

$$\bar{r} = \sum_{k=1}^{m} p_k r_k$$

y, por lo tanto, se tiene:

$$y_n = \begin{cases} 0, & \text{si} \quad \bar{r} \le 0 \\ x_n, & \text{si} \quad \bar{r} > 0 \end{cases} \qquad (5.3)$$

$$f_n(x_n) = \begin{cases} \left(1 + \bar{r}\right) x_n, & \text{si} \quad \bar{r} > 0 \\ x_n, & \text{si} \quad r \le 0 \end{cases} \qquad (5.4)$$

Ya establecidas las ecuaciones se tiene que, si C = \$10 000 y son 4 años con un 50% de probabilidades de que el dinero aumente al doble: 20% por salir a mano y 30% por perder la cantidad invertida, ¿Cuál es una estrategia óptima de inversión?

Así, tenemos:

$C = 10\,000$	$n = 4$	$m = 3$
$p_1 = 0.50$	$p_2 = 0.20$	$p_3 = 0.30$
$r_1 = 1$	$r_2 = 0$	$r_3 = -1$

ETAPA 4

$$\bar{r} = (0.5)1 + (0.2)0 + (0.3)(-1) = 0.2$$

entonces,

$$f_4(x_4) = 1.2\, x_4$$

La solución óptima se resume en la tabla siguiente:

TABLA 5.1. Cálculos para la etapa 4

	Solución óptima	
Estado	$f_4(x_4)$	y_4^*
x_4	$1.2x_4$	x_4

ETAPA 3

$$\begin{aligned} f_3(x_3) &= \max_{0\le y_3\le x_3}\left\{p_1 f_4\left(x_3 + r_1 y_3\right) + p_2 f_4\left(x_3 + r_2 y_3\right) + p_3 f_4\left(x_3 + r_3 y_3\right)\right\} = \\ &= \max_{0\le y_3\le x_3}\left\{(0.5)1.2\left(x_3 + y_3\right) + (0.2)1.2\left(x_3 + 0y_3\right) + (0.3)1.2\left[x_3 + (-1)y_3\right]\right\} = \\ &= \max_{0£ y_3 £ x_3}\left\{1.2x_3 + 0.24y_3\right\} = 1.44x_3 \end{aligned}$$

Y la solución se resume en la tabla siguiente:

TABLA 5.2. Cálculos para la etapa 3

	Solución óptima	
Estado	$f_3(x_3)$	y_3^*
x_3	$1.44x_3$	x_3

ETAPA 2

$$\begin{aligned} f_2(x_2) &= \max_{0\le y_2\le x_2}\left\{p_1 f_3\left(x_2 + r_1 y_2\right) + p_2 f_3\left(x_2 + r_2 y_2\right) + p_3 f_3\left(x_2 + r_3 y_2\right)\right\} = \\ &= \max_{0\le y_3\le x_2}\left\{(0.5)1.44\left(x_2 + y_2\right) + (0.2)1.44\left(x_2 + 0y_2\right) + (0.3)1.44\left[x_2 + (-1)y_2\right]\right\} = \\ &= \max_{0\le y_3\le x_2}\left\{1.44x_2 + 0.288y_2\right\} = 1.728x_2 \end{aligned}$$

Siendo así, se tiene:

TABLA 5.3. Cálculos para la etapa 2

	Solución óptima	
Estado	$f_2(x_2)$	y_2^*
x_2	$1.728x_2$	x_2

ETAPA 1

$$
\begin{aligned}
f_1(x_1) &= \max_{0\le y_1\le x_1}\left\{p_1 f_2\left(x_1+r_1y_1\right)+p_2 f_2\left(x_1+r_2y_1\right)+p_3 f_2\left(x_1+r_3y_1\right)\right\}= \\
&= \max_{0\le y_1\le x_1}\left\{(0.5)1.728\left(x_1+y_1\right)+(0.2)1.728\left(x_1+0y_1\right)+(0.3)1.728\left[x_1+(-1)y_1\right]\right\}= \\
&= \max_{0\le y_1\le x_1}\left\{1.728x_1+0.3456y_1\right\}=2.0736\,x_1
\end{aligned}
$$

y se obtiene:

TABLA 5.4. Cálculos para la etapa 1

	Solución óptima	
Estado	$f_1(x_1)$	y_1^*
x_1	$2.073x_1$	x_1

Así, la política óptima de inversión se puede resumir de la siguiente manera:

$y_i = x_i$ para i de 1 a 4 significa que se deben invertir todos los fondos disponibles al iniciar cada año. Los fondos acumulados al finalizar los 4 años son de 2.0736, $x_1 = 2.0736$ (10 000) = $20 736.00. En general, el problema tiene una solución como se expresó en las ecuaciones (5.3) y (5.4).

Ejemplo 5.3. Costos inciertos, estado siguiente seguro. (Winston, 2005).

Por el precio de 1 litro de agua, una cadena de supermercados SPER compró 6 litros de agua de una expendedora local. Cada litro de agua se vende en las tres tiendas de la cadena en $2.00 el litro. La expendedora debe comprar de nuevo en $.50 el litro que se queda al final del día. Desafortunadamente, para SPER, la demanda para cada una de las tres tiendas es incierta.

De acuerdo con los registros de los datos de la demanda anterior, se sabe que la demanda diaria en cada tienda es como se muestra en la tabla siguiente.

TABLA 5.5. Datos del problema de SPER

	Demanda diaria (litros)	**Probabilidad**
Tienda 1	1	0.60
	2	0
	3	0.40
Tienda 2	1	0.50
	2	0.10
	3	0.40
Tienda 3	1	0.40
	2	0.30
	3	0.30

SPER quiere asignar los 6 litros a las tres tiendas para maximizar la ganancia diaria neta esperada (ingresos menos costos) obtenida del agua.

Solución:

Con excepción de que la demanda y, por consiguiente, el ingreso son inciertos, este problema es similar a los vistos en capítulos anteriores. Como los costos de compra diarios son siempre de $6.00, podemos centrarnos en el problema de distribuir el agua para maximizar el ingreso diario esperado obtenido de los 6 litros.

Ya sean:

$r_t(l_t)$ = ingreso esperado obtenido de l litros asignados a la tienda t.
$f_t(x)$ = ingreso esperado máximo obtenido de x litros asignados a las tiendas t (t = 1,2,3).

Por definición $f_3(x)$ debe ser el ingreso esperado obtenido al asignar x litros de agua a la tienda 3, por lo cual, $f_3(x) = r_3(x)$.

Y para t = 1,2 se tendrá:

$$f_t(x) = \text{máx}_{lt}\{r_t(l_t) + f_{t+1}(x - l_t)\} \qquad (5.5)$$

donde l_t debe ser un miembro de $\{0, 1, \ldots, x\}$.

De la ecuación recursiva se deduce que debido a cualquier elección de l_t (número de litros asignados a la tienda t), el ingreso esperado obtenido de la tienda $t = 1, 2$ será la suma del ingreso esperado obtenido de la tienda t, si l_t litros se asignan a la tienda t, más el ingreso máximo esperado de las tiendas $t = 2, 3$ cuando se asignan $x - l_t$ litros a esas tiendas.

Para calcular la asignación óptima de agua a las tiendas, se empieza por calcular para la tienda 3, en la etapa 3, lo siguiente:

ETAPA 3

Como se mencionó antes $f_3(x) = r_3(x)$, por lo que, calculamos las $r_3(x)$ con $x = 0, 1, 2, 3$, ya que, no se necesita calcular más de 3 litros por tienda.

$r_3(2)$ es el ingreso esperado obtenido, si se asignan 2 litros a la tienda 3; si la demanda en la tienda 3 es, por lo menos, 2 o más litros, se venderán ambos litros asignados a la tienda 3 obteniendo un ingreso de \$4.00. Entonces, si la demanda en la tienda 3 es de 1 litro, ese litro se vende con un ingreso de \$2.00 y se devuelve un litro por \$0.50.

De esta manera, si la demanda en la tienda 3 es de 1 litro se obtiene un ingreso de \$2.50, pero, como hay una probabilidad de \$0.60 de que la demanda en la tienda 3 sea de 2 o 3 litros y una probabilidad de \$0.40 de que sea de 1 litro, entonces, se deduce que:

Si se asignan 2 litros se puede tener:
demanda de 1 y se regresa 1, demanda de 2 o 3 y no se regresa nada, con lo que se obtiene:

$$r_3(2) = (0.30 + 0.30)\ 4 + (0.40)\ (2.50) = \$3.40$$

Por lo tanto, si se asignan 3 litros se puede tener:

Demanda de 1 y se regresan 2.
Demanda de 2 y se regresa 1.
Demanda de 3 y no se regresa nada, con lo cual se tiene:

$$r_3(3) = 3(0.40) + (0.30)\ (4.5) + (0.3)6 = \$4.35$$

Así, se obtiene:

$r_3(0) = \$0$
$r_3(1) = \$2.00$

Para las otras tiendas, se tienen los siguientes cálculos, por favor verifíquelos:

$r_2(0) = \$0$
$r_2(1) = \$2.00$
$r_2(2) = \$3.25$
$r_2(3) = \$4.35$

$r_1(0) = \$0$
$r_1(1) = \$2.00$
$r_1(2) = \$3.10$
$r_1(3) = \$4.20$

Se usa la ecuación (5.5) para determinar una asignación óptima de agua a las tiendas. Sea $g_t(x)$ una asignación de agua a la tienda t que se obtiene por medio de $f_t(x)$.

$f_3(0) = r_3(0) = 0$	$g_3(0) = 0$
$f_3(1) = r_3(1) = 2$	$g_3(1) = 1$
$f_3(2) = r_3(2) = 3.40$	$g_3(2) = 2$
$f_3(3) = r_3(3) = 4.35$	$g_3(3) = 3$

No es necesario calcular para 4, 5 o 6 litros, ya que, la asignación óptima nunca tendrá más de 3 litros para asignarlos a una sola tienda ni la demanda es de más de 3 litros por tienda.

La ecuación (5.5) se usa para las dos etapas siguientes:

ETAPA 2

$f_2(0) = r_2(0) + f_3(0-0) = 0$ $\qquad$ $g_2(0) = 0$

$$f_2(1) = \text{máx}\begin{Bmatrix} r_2(0) + f_3(1-0) = 2.00^* \\ r_2(1) + f_3(1-1) = 2.00^* \end{Bmatrix} \qquad g_2(1) = 0 \quad \text{o} \quad 1$$

$$f_2(2) = \text{máx}\begin{Bmatrix} r_2(0) + f_3(2-0) = 0 + 3.40 = 3.40 \\ r_2(1) + f_3(2-1) = 2.00 + 2.00 = 4.00^* \\ r_2(2) + f_3(2-2) = 3.25 + 0 = 3.25 \end{Bmatrix} \qquad g_2(2) = 1$$

$$f_2(3) = \text{máx}\begin{Bmatrix} r_2(0) + f_3(3-0) = 0 + 4.35 = 4.35 \\ r_2(1) + f_3(3-1) = 2.00 + 3.40 = 5.40^* \\ r_2(2) + f_3(3-2) = 3.25 + 2.00 = 5.25 \\ r_2(3) + f_3(3-3) = 4.35 + 0 = 4.35 \end{Bmatrix} \qquad g_2(3) = 1$$

En el cálculo de $f_2(4)$, $f_2(5)$, $f_2(6)$ es innecesario considerar alguna asignación de más de 3 litros a la tienda 2 o alguna que deje más de 3 litros a la tienda 3.

$$f_2(4) = \text{máx}\begin{Bmatrix} r_2(1) + f_3(4-1) = 2.00 + 4.35 = 6.35 \\ r_2(2) + f_3(4-2) = 3.25 + 3.40 = 6.65^* \\ r_2(3) + f_3(4-3) = 4.35 + 2.00 = 6.35 \end{Bmatrix} \qquad g_2(4) = 2$$

$$f_2(5) = \text{máx}\begin{Bmatrix} r_2(2) + f_3(5-2) = 3.25 + 4.35 = 7.60 \\ r_2(3) + f_3(5-3) = 4.35 + 3.40 = 7.75^* \end{Bmatrix} \qquad g_2(5) = 3$$

$$f_2(6) = r_2(3) + f_3(6–3) = 4.35 + 4.35 = 8.70^* \qquad g_2(6) = 3$$

ETAPA 1

$$f_1(6) = \text{máx}\begin{Bmatrix} r_1(0) + f_2(6-0) = 0 + 8.70 = 8.70 \\ r_1(1) + f_2(6-1) = 2.00 + 7.75 = 9.75^* \\ r_1(2) + f_2(6-2) = 3.10 + 6.65 = 9.75^* \\ r_1(3) + f_2(6-3) = 4.20 + 5.40 = 9.60 \end{Bmatrix} \qquad g_1(6) = 1 \text{ o } 2$$

Así, se pueden asignar 1 o 2 litros a la tienda 1. Si se eligiera de manera arbitraria asignar 1 litro a la tienda 1, entonces, tendríamos 6 – 1 = 5 litros para las tiendas 2 y 3. Puesto que, $f_2(5)$ se obtiene a través de $g_2(5) = 3$, se asignan 3 litros a la tienda 2. Entonces, 5 – 3 = 2 litros que están disponibles para la tienda 3. Puesto que, $g_3(2) = 2$ se asignan 2 litros a la tienda 3. Observe que, aunque esta política obtiene un ingreso máximo esperado $f_1(6)$ = \$9.75, el

ingreso total recibido actualmente en un día determinado podría ser más o menos \$9.75. Por ejemplo, si la demanda en cada tienda fuera de 1 litro, el ingreso total sería $3(2.00) + 3(0.5)$ = \$7.50, en tanto que, si la demanda en cada tienda fuera de 3 litros, toda el agua se vendería a \$2.00 el litro y el ingreso total sería de 6(2.00) = \$12.00.
Este resultado lo podemos visualizar en la siguiente figura:

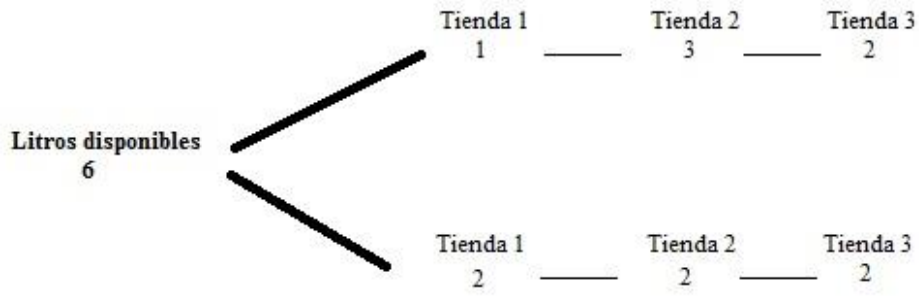


FIGURA 5.3. Soluciones del problema para asignar agua

5.4. MODELOS DE INVENTARIOS ESTOCÁSTICOS

En el siguiente ejemplo, se considera un modelo de inventario con demanda incierta y, por lo tanto, en términos de la PD (Programación Dinámica) estocástica el estado durante la siguiente etapa es incierto, dados el estado actual y la decisión.

Ejemplo 5.4. (Winston, 2005).

En el siguiente ejemplo considere el inventario de tres periodos. Al comienzo de cada periodo, una empresa debe determinar cuántas unidades debe producir durante el periodo actual. Durante un periodo en el que se producen x unidades, se incurre en un costo de producción $c(x)$, donde $c(0) = 0$ y para $x>0$, $c(x) = 3 + 2x$. La producción durante cada periodo está limitada a lo sumo a 4 unidades. Después de que ocurre la producción, se observa la demanda aleatoria del periodo; la demanda de cada periodo tiene las mismas probabilidades de que sean 1 o 2 unidades. Después de satisfacer la demanda del periodo actual de la producción o inventario actual, se evalúa el inventario de fin de periodo de la empresa y se estima un costo de inventario de 1 dólar por unidad. Como resultado de la capacidad limitada, el inventario al final de cada etapa no puede exceder a 3 unidades y se requiere que toda la demanda se satisfaga a tiempo. Cualquier inventario disponible al final del periodo 3 se puede vender en 2.00 dólares por unidad. Al inicio del periodo 1, la empresa tiene 1 unidad de

inventario. Determine la política de producción que minimiza el costo neto esperado en que se incurre durante los tres periodos.

Se define $f_t(i)$ como el costo neto esperado en el que se incurre en los periodos t = 1, 2, 3, cuando el inventario al inicio del periodo t es de i unidades, entonces:

$$f_3(i) = \min_x \begin{Bmatrix} c(x) + \left(\frac{1}{2}\right)(i+x-1) + \left(\frac{1}{2}\right)(i+x-2) - \\ -\left(\frac{1}{2}\right)2(i+x-2) - \left(\frac{1}{2}\right)2(i+x-1) \end{Bmatrix} \tag{5.6}$$

donde x pertenece al conjunto de {0, 1, 2, 3, 4} y x debe satisfacer:

$$(2 - i) \leq x \leq (4 - i)$$

De la ecuación (5.6) se deduce que, si se producen x unidades durante el periodo 3, el costo neto durante ese periodo será el siguiente.

Costo neto para el periodo 3:

Costo de producción esperado + costo de inventario esperado – costo de salvamento esperado.

Si se producen x unidades, el costo del periodo 3 será $i + x - 1$ y una probabilidad de 0.5 de que sea $i + x - 2$, por consiguiente, el *costo de retención del periodo 3* será:

$$\left(\frac{1}{2}\right)(i+x-1) + \left(\frac{1}{2}\right)(i+x-2) = i + x - \frac{3}{2}$$

Un razonamiento similar muestra que el *costo de salvamento esperado* (un costo negativo) al final del periodo 3 será:

$$\left(\frac{1}{2}\right)2(i+x-1) + \left(\frac{1}{2}\right)2(i+x-2) = 2i + 2x - 3, \qquad x \geq 2 - i$$

De manera similar, para asegurar que el inventario al final de tres periodos no exceda a 3 unidades, se debe tener que $i + x - 1 \leq 3$ o, bien, $x \leq 4 - i$.

Para $t = 1, 2$, se puede derivar la relación recursiva para $f_t(i)$ al observar que para cualquier nivel de producción x del mes t, los costos esperados en que se incurre durante los periodos t, $t + 1,\ldots, 3$ son la suma de los costos esperados en que se incurre durante los periodos $t+1$, $t + 2,\ldots,3$. Como antes, si se producen x unidades durante el mes t, el *costo esperado* durante ese mes será:

$$c(x)+\left(\frac{1}{2}\right)(i+x-1)+\left(\frac{1}{2}\right)(i+x-2)$$

Durante los periodos 1 y 2, observe que no se recibe el valor de salvamento. Si durante el mes t se producen x unidades, el costo esperado durante los periodos $t + 1$, $t + 2$, …, 3 se calcula de la siguiente manera: la mitad del tiempo, la demanda durante el periodo t será 1 unidad y el inventario al inicio del periodo $t + 1$ será $i + x - 1$. En esta situación los costos esperados en que se incurre en los periodos $t + 1$, $t + 2,\ldots, 3$, suponiendo que actuamos de manera óptima durante esos periodos es $f_{t+1}(i + x - 1)$.

De manera similar, hay una probabilidad de 0.5 de que el inventario al inicio del periodo $t + 1$ sea $i + x - 2$, en cuyo caso, el costo esperado para los periodos $t + 1$, $t + 2,\ldots, 3$ será $f_{t+1}(i + x - 2)$. En resumen, el *costo esperado durante los periodos* $t + 1$, $t + 2,\ldots, 3$ será:

$$\left(\frac{1}{2}\right)f_{t+1}(i+x-1)+\left(\frac{1}{2}\right)f_{t+1}(i+x-2)$$

De esta forma, las ecuaciones recursivas para $t = 1, 2$ son:

$$f_t(i)=\min_x[c(x)+\left(\frac{1}{2}\right)(i+x-1)+\left(\frac{1}{2}\right)(i+x-2)+\left(\frac{1}{2}\right)f_{t+1}(i+x-1)+\left(\frac{1}{2}\right)f_{t+1}(i+x-2)] \qquad (5.7)$$

donde $x \in \{0, 1, 2, 3, 4\}$ y satisface a $(2 - i) \leq x \leq (4 - i)$.

Generalizando, del razonamiento que dio lugar a (5.7) se tiene la observación importante para la formulación de la PD: suponga que los estados posibles durante el periodo $t + 1$ son s_1, $s_2,\ldots, s_n$ y la probabilidad de que el estado del periodo $t + 1$ sea s_i es p_i. Entonces, el costo mínimo esperado en que se incurre en los periodos $t + 1$, $t + 2,\ldots$, etc., el problema es:

$$\sum_{i=1}^{n} p_i f_{t+1}(s_i)$$

donde $f_{t+1}(s_i)$ es el costo mínimo esperado en que se incurre desde el periodo $t + 1$ hasta el fin del problema, dado que el estado durante el periodo t es s_i.

Se define $x_t(i)$ como un nivel de producción del periodo t que obtiene el mínimo en (5.7) para $f_t(i)$. Ahora, trabajamos hacia atrás hasta que se determina $f_1(t)$. Como el inventario al final de cada periodo debe ser no negativo y no puede exceder 3 unidades, el estado durante cada periodo debe ser 0, 1, 2 o 3.

ETAPA 3

i	x	$c(x)$	**Costo de retención esperado** $i + x - (3/2)$	**Valor de salvamento esperado** $2i + 2x - 3$	**Costo total esperado**	$f_3(i)$ $x_3(i)$
3	0	0	3/2	3	–3/2*	$f_3(3) = -3/2$
3	1	5	5/2	5	5/2	$x_3(3) = 0$
2	0	0	1/2	1	–1/2*	$f_3(2) = -1/2$
2	1	5	3/2	3	7/2	$x_3(2) = 0$
2	2	7	5/2	5	9/2	
1	1	5	1/2	1	9/2*	$f_3(1) = 9/2$
1	2	7	3/2	3	11/2	$x_3(1) = 1$
1	3	9	5/2	5	13/2	
0	2	7	1/2	1	13/2*	$f_3(0) = 13/2$
0	3	9	3/2	3	15/2	$x_3(0) = 2$
0	4	11	5/2	5	17/2	

ETAPA 2

i	x	$c(x)$	**Costo de retención esperado** $i + x - (3/2)$	**Costo futuro esperado** $(1/2) f_3(i + x - 1) + (1/2) f_3(i + x - 2)$	**Costo total esperado periodos 2, 3**	$f_3(i)$ $x_3(i)$
3	0	0	3/2	2	7/2*	$f_2(3) = 7/2$
3	1	5	5/2	–1	13/2	$x_2(3) = 0$
2	0	0	1/2	11/2	6*	$f_2(2) = 6$
2	1	5	3/2	2	17/2	$x_2(2) = 0$
2	2	7	5/2	–1	17/2	
1	1	5	1/2	11/2	11	$f_2(1) = 21/2$
1	2	7	3/2	2	21/2*	$x_2(1) = 2,3$
1	3	9	5/2	–1	21/2*	
0	2	7	1/2	11/2	13	$f_2(0) = 25/2$
0	3	9	3/2	2	25/2*	$x_2(0) = 3,4$
0	4	11	5/2	–1	25/2*	

ETAPA 1

x	$c(x)$	**Costo de retención esperado** $i + x - (3/2)$	**Costo futuro esperado** $(1/2) f_2(i + x - 1) + (1/2) f_2(i + x - 2)$	**Costo total esperado periodos 1, 2,3**	$f_1(i)$ $x_1(i)$
1	5	1/2	23/2	17	$f_1(1) = 65/4$
2	7	3/2	33/4	67/4	$x_1(1) = 3$
3	9	5/2	19/4	65/4*	

Se empieza produciendo $x_1(1) = 3$ unidades durante el periodo 1, sin embargo, no se puede determinar el nivel de producción del periodo 2 hasta que se observa la demanda del periodo 1. Tampoco, es posible determinar el nivel de producción del periodo 3 hasta que se observa la demanda del periodo 2. Para ilustrar la idea, se determina el programa óptimo de producción, si tanto la demanda del periodo 1 como la del periodo 2 son 2 unidades. Puesto que, $x_1(1) = 3$ durante el periodo 1 se producirán 3 unidades. Entonces, en el periodo 2 se comenzará con un inventario de $1 + 3 + 2 = 2$ unidades, así que, se deben producir $x_2(2) = 0$ unidades. Después, se satisface la demanda de 2 unidades del periodo 2, el periodo 3 comienza con $2 - 2 = 0$ unidades disponibles. Por consiguiente, durante el periodo 3 se producirán $x_3(0) = 2$ unidades.

En los siguientes ejemplos (Dreyfus y Law, 1977), vamos a suponer ciertas probabilidades al tratar el problema de los inventarios como es el caso de una demanda estocástica. La longitud del periodo de tiempo n puede ser una semana, un mes o un año, el número de periodos para los que una compañía quisiera programar su inventario es, en este caso, n y se conoce como el *horizonte de planeación.*

La demanda del producto para el periodo i se denota por D_i que es una variable aleatoria no negativa con una función de probabilidad asociada $p_i(d)$ y significa la probabilidad de que $D_i = d$, cuando se supone que $D_1, D_2, \ldots, D_n$ son independientes. Al inicio de cada periodo la compañía revisa el nivel del inventario y decide cuántos artículos ordenar de un proveedor. Una alternativa es que la compañía produzca sus propios artículos, en cuyo caso, al inicio de cada periodo se tiene que decidir cuánto producir y/o cuánto pedir al proveedor. Sea x_i el inventario disponible que se ordenará antes de iniciar el periodo i. Si se pone una orden de z_i artículos (un entero no negativo) llega en el periodo $i + \lambda$ donde λ es un entero no negativo y se conoce como el *retraso en la entrega*. Se supone que $\lambda < n$ y que la compañía no ordenará en periodos $n - \lambda + 1, n - \lambda + 2, \ldots, n$. Sea y_i la cantidad de inventario disponible y a ordenar después del periodo i, entonces, $y_i = x_i + z_i$.

En algunos casos hay restricciones en la cantidad ordenada z_i o en el nivel del inventario y_i. Sea w_i la cantidad de inventario actualmente disponible (no incluye lo que se ordena), después, de que se ha entregado una orden para el periodo $i - \lambda$, z_{i-k}, pero, antes de que la demanda ocurra. Entonces, w_i es la cantidad del inventario actualmente disponible para cumplir la demanda del periodo i. Si $\lambda = 0$, entonces, $w_i = y_i$ y, si $\lambda > 0$, entonces:

$$w_{i+k} = y_i - (d_i + d_{i+1} + \ldots + d_{i+\lambda-1})$$

donde d_i es la demanda del periodo i. La sucesión de eventos (durante un periodo de tiempo) para $\lambda \geq 0$ se muestra en la figura 5.4.

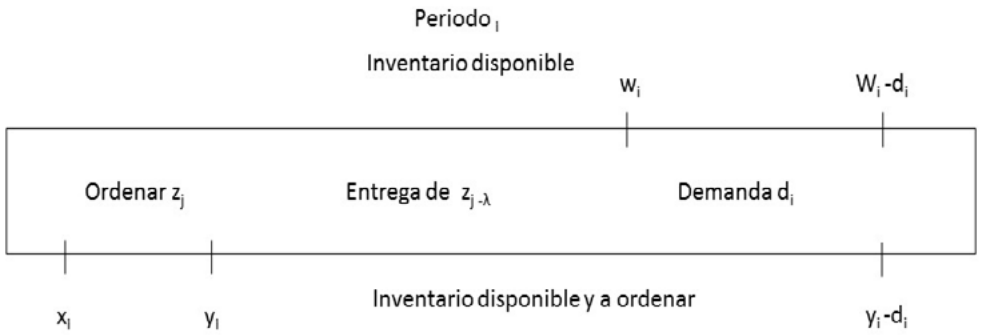


FIGURA 5.4. Eventos durante un periodo i

Para determinar x_{i+1} es necesario hacer algunas suposiciones de lo que sucede cuando $d_i > w_i$. Estimemos dos casos diferentes. Si toda la demanda no se satisface al final del periodo i y eventualmente se satisface para futuras entregas, entonces, se llama el *caso con retraso*. En este caso, $x_{i+1} = y_i - d_i$ puede ser negativo. Si la demanda no se satisface al final de periodo i, se supone perdida, entonces, se llama el caso de *ventas perdidas*. En este caso, si $\lambda = 0$, entonces, $x_{i+1} = \text{máx}\ (y_i - d_i, 0)$ y, si $\lambda > 0$, entonces, como veremos, posteriormente, la situación es más compleja. En el caso de las ventas perdidas, se supone que la demanda no satisfecha en el periodo i se pierde, después de que $w_i - d_i$ y $y_i - d_i$ se han determinado antes de que inicie el periodo $i + 1$, tampoco, la expresión $w_{i+\lambda}$ dada anteriormente es válida.

Para operar el sistema de inventarios hay tres costos asociados. Si z artículos se ordenan en el periodo i, entonces, hay un costo por ordenar de $c_i(z)$ que incurre en el tiempo de la entrega en el periodo $i + \lambda$ (también, es posible suponer que incurre en el costo por ordenar en el tiempo cuando se pone la orden). Si el inventario neto disponible al final del periodo i es $w_i - d_i$, es no negativo, entonces, hay un costo por llevar el inventario de $c_i(w_i - d_i)$. El costo por llevar el inventario incluye costos, tales como, renta de almacén, seguros, impuestos y mantenimiento.

También, se incluye el costo por tener capital ligado al inventario en lugar de poder invertirlo en otras cosas.

Si $w_i - d_i$ es negativo (significa que se tiene la demanda no satisfecha al final del periodo i), entonces, hay un costo de déficit o penalización $\Pi_i(d_i - w_i)$ y un costo mínimo por inventario de $h_i(0)$. Se supone que $\Pi_i(0) = 0$. El costo por déficit incluye el costo por mantenimiento de los casos con retraso y el costo debido a la pérdida por el caso de las ventas perdidas. También, algunos sistemas de inventarios incluyen un costo debido por clientes perdidos.

Ahora, supondremos que el costo incurrido en el periodo j es equivalente al costo que se incurra en periodos futuros. Sin embargo, si la longitud del periodo es suficientemente larga y/o la tasa de interés en el capital invertido es suficientemente alta, no será una buena suposición.

Por el periodo de tiempo en el capital invertido, i es la tasa de interés, es decir, cada peso invertido al inicio del periodo j tiene un valor de $1 + i$ pesos al inicio del periodo $j + 1$. Sea $\alpha = 1/(1 + i)$, la cantidad α es llamada un factor de descuento y se considera $0 < \alpha \leq 1$. Tenemos en efecto la suposición de que $\alpha = 1$ en los casos vistos con anterioridad. Entonces, un costo c en el periodo $j + 1$ tiene un valor de descuento de αc en el periodo j. De manera similar, un costo c en el periodo $j + k$ tiene un valor de descuento de $\alpha^k c$ en el periodo j. Esto es equivalente a afirmar que una inversión de $\alpha^k c$ pesos en el periodo j, a una tasa de interés i, dará c pesos, k periodos más tarde con $c = \alpha^k c\,(1+ i)^k$.

Una política para ordenar es una regla que nos dice cómo decidir, cuánto se va a ordenar al inicio de cada periodo, correspondiente a cada política para ordenar hay un costo descontado total esperado para n periodos de tiempo. En ese sentido, el objetivo de la compañía es seleccionar una política óptima para ordenar, es decir, una que minimice el costo descontado total esperado.

Si $p_i(d)$ está dada como la probabilidad de que $D_i = d$ $(i = 1, 2,\ldots,n)$ donde D_i son variables aleatorias independientes, entonces, la probabilidad de:

$$D_i + D_{i+1} + \ldots + D_j = d \qquad (i = 1, 2,\ldots, n; \quad j = i, i + 1,\ldots,n)$$

Se conoce como la convolución de $p_i(d)$, $p_{i+1}(d),..,p_j(d)$ y se denota por $p_{i,j}(d)$. Así, podemos calcular recursivamente $p_{i,j}(d)$ (para $d = 0, 1, \ldots$) como sigue:

$$p_{i,i}(d) \equiv p_i(d)$$

$$p_{i,k}(d) = \sum_{l=0}^{d} p_{i,k-1}(l)\, p_k(d-l) \qquad k = i+1,\ i+2,\ldots, j$$

5.4.1. MODELOS CON CERO RETRASO DE ENTREGA

En esta sección veremos cómo se puede determinar una política óptima por ordenar, cuando el retraso λ en la entrega es cero o es despreciable por el tamaño relativo a la longitud de su periodo de tiempo.

Así, comenzaremos con el caso de déficit de la siguiente manera, supongamos que, si hay un inventario no negativo de v artículos al final del periodo n (el inicio del periodo n + 1), entonces, los v artículos se pueden regresar y la compañía recibe un valor de salvamento de $s(v)$ (se incurre en un costo $-s(v)$). Inversamente, si existe un déficit de v artículos al final del periodo n, luego, ese déficit se puede satisfacer comprando v artículos a un costo de $b(v)$. Si $t(v)$ es el costo terminal, cuando hay un inventario de v (v =…,–2, –1, 0, 1, 2,…) artículos al final del periodo n y $t(n)$ está dado por:

$$t(v) = \begin{Bmatrix} -s(v), & \text{si } v \geq 0 \\ b(-v), & \text{si } v < 0 \end{Bmatrix}$$

Al inicio del periodo i, estime que se tienen x artículos disponibles (no hay artículos por ordenar, si consideramos que $\lambda = 0$), si ordenamos $y - x$ artículos en el periodo i, entonces, se incurre en un costo por ordenar dado por $c_i(y - x)$ y se tienen y artículos disponibles para satisfacer la demanda. Sea $L_i^0(w, d)$ el costo por llevar el inventario y por el déficit en el periodo i cuando w artículos ($w = y$, ya que, $\lambda = 0$) están disponibles y hay una demanda d. De manera similar, sea $L_i^0(w)$ el costo esperado por llevar el inventario y el déficit cuando se tienen w artículos disponibles para satisfacer la demanda. Entonces, $L_i^0(w, d)$ y $L_i^0(w)$ están dados por:

$$L_i^0(w,d) = \begin{Bmatrix} h_i(w-d), & \text{si } w-d \geq 0 \\ \prod_i (d-w) + h_i(0), & \text{si } w-d \leq 0 \end{Bmatrix}$$

$$L_i^0(w) = \sum_{d=0}^{\infty} L_i^0(w,d) p_i(d) \tag{5.8}$$

Ahora, tengamos en cuenta la formulación de la PD para este caso. Conjeturemos que los costos futuros se descuentan a una tasa de α por periodo y que la compañía quiere determinar

una política por ordenar que minimice el costo total descontado para los n periodos del horizonte de planeación. Defina la función del valor esperado óptimo $f_i(x)$ como sigue:

$f_i(x)$ = el costo descontado total esperado (descontado al periodo i) para los periodos i a n (más el costo final), si la cantidad de inventario a la mano del periodo i es x (no hay artículos para ordenar, ya que, $\lambda = 0$) y se sigue una política óptima para ordenar. (5.9)

Si $D_i = d$, entonces, $x_{i+1} = y - d$ y es fácil ver que $f_i(x)$ satisface la relación de recurrencia siguiente:

$$f_i(x) = \min_{y \geq x} \left\{ c_i(y-x) + L_i^0(y) + \alpha \sum_{d=0}^{y} f_{i+1}(y-d)\, p_i(d) \right\} \quad i = 1, 2, ..., n \tag{5.10}$$

Para cada valor de i, $f_i(x)$ se calcula para cada posible valor de x. Si $y_i(x) - x$ es la cantidad óptima a ordenar cuando $x_i = x$, entonces, $y_i(x)$ es igual al valor de y que minimiza el lado derecho de (5.9) la condición de frontera apropiada para este caso es:

$$f_{n+1}(x) = t(x) \tag{5.11}$$

Si x_1 es el inventario inicial específico, entonces, $f_1(x_1)$ es la respuesta al problema de n periodos.

Ejemplo 5.5. Una compañía quiere programar sus inventarios para los siguientes tres periodos de tiempo y se tienen los siguientes datos:

Las probabilidades asociadas a la demanda son:

$$p(0) = ¼ \qquad p(1) = ½ \qquad p(2) = ¼$$

Los costos por ordenar son:

$c(0) = 0$ $\quad c(1) = 3$ $\quad c(2) = 5$ $\quad c(3) = 6$ $\quad c(z) = \infty$ para $z \geq 4$

Los costos por inventario h y déficit $\prod$ son:

$$h(v) = v \text{ para } v \geq 0 \qquad \prod(v) = -5v \text{ para } v \geq 0$$

Valor de salvamento: $s(v) = 2v$ para $v \geq 0$

Costo por comprar v artículos: $b(v) = 4v$ para $v \geq 0$

Note que todas las funciones del costo y la distribución de la demanda son independientes del tiempo. También, damos por sentado un inventario inicial $x_1 = 0$ y una tasa de descuento $\alpha = 1$.

La solución de PD es la siguiente:

$$f_4(x) = -2x \text{ para } x = 0, 1, 2,..., 9, \text{ el inventario máximo es 9}$$

$$f_4(x) = -4x \text{ para } x = -1, -2,\ldots, -6, \text{ el déficit máximo es 6}$$

Esto se puede ver con más claridad en la figura siguiente:

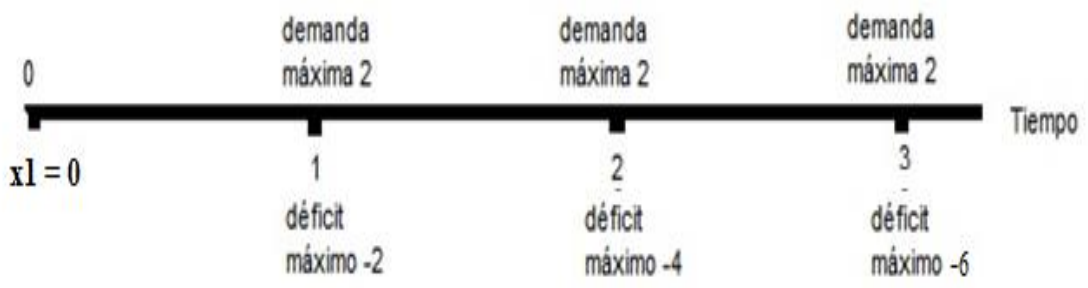


FIGURA 5.5. Periodos de tiempo con demanda y déficit máximos por año

ETAPA 3

Para esta etapa, calcularemos $f_3(x)$ para x que va entre 6 y –4 y donde se tiene:

$$f_3(6) = \min_{6 \leq y \leq 9} \left\{ c(y-6) + L^0(y) + \sum_{d=0}^{2} f_4(y-d)\, p(d) \right\} =$$

$$= \text{mín}\begin{Bmatrix} c(6-6)+L^0(6)+[f_4(6-0)p(0)+f_4(6-1)p(1)+f_4(6-2)p(2)], \\ c(7-6)+L^0(7)+\left[f_4(7-0)p(0)+f_4(7-1)p(1)+f_4(7-2)p(2)\right], \\ c(8-6)+L^0(8)+\left[f_4(8-0)p(0)+f_4(8-1)p(1)+f_4(8-2)p(2)\right], \\ c(9-6)+L^0(9)+\left[f_4(9-0)p(0)+f_4(9-1)p(1)+f_4(9-2)p(2)\right] \end{Bmatrix} =$$

$$= \text{mín}\begin{Bmatrix} c(0)+[h(6)\frac{1}{4}+h(5)\frac{1}{2}+h(4)\frac{1}{4}]+\left[f_4(6)\frac{1}{4}+f_4(5)\frac{1}{2}+f_4(4)\frac{1}{4}\right], \\ c(1)+[h(7)\frac{1}{4}+h(6)\frac{1}{2}+h(5)\frac{1}{4}]+\left[f_4(7)\frac{1}{4}+f_4(6)\frac{1}{2}+f_4(5)\frac{1}{4}\right], \\ c(2)+[h(8)\frac{1}{4}+h(7)\frac{1}{2}+h(6)\frac{1}{4}]+\left[f_4(8)\frac{1}{4}+f_4(7)\frac{1}{2}+f_4(6)\frac{1}{4}\right], \\ c(3)+[h(9)\frac{1}{4}+h(8)\frac{1}{2}+h(7)\frac{1}{4}]+\left[f_4(9)\frac{1}{4}+f_4(8)\frac{1}{2}+f_4(7)\frac{1}{4}\right] \end{Bmatrix}$$

$$= \text{mín}\{0+5-10,\ 3+6-12,\ 5+7-14,\ 6+8-16\} = \text{mín}\{-5,-3,-2,-2\} = -5$$

$$f_3(6) = -5 \qquad y_3(6) = 6$$

Para $x = 5$ se tiene:

$$f_3(5) = \min_{5\le y\le 8}\{c(y-5) + L^0(y) + \sum_{d=0}^{2} f_4(y-d)p(d)\}$$

$$= \text{mín}\begin{Bmatrix} c(5-5)+\left[h(6)\frac{1}{4}+h(4)\frac{1}{2}+h(3)\frac{1}{4}\right]+\left[f_4(5)\frac{1}{4}+f_4(4)\frac{1}{2}+f_4(3)\frac{1}{4}\right] \\ c(6-5)+\left[h(6)\frac{1}{4}+h(5)\frac{1}{2}+h(4)\frac{1}{4}\right]+\left[f_4(6)\frac{1}{4}+f_4(5)\frac{1}{2}+f_4(4)\frac{1}{4}\right] \\ c(7-5)+\left[h(7)\frac{1}{4}+h(6)\frac{1}{2}+h(5)\frac{1}{4}\right]+\left[f_4(7)\frac{1}{4}+f_4(6)\frac{1}{2}+f_4(5)\frac{1}{4}\right] \\ c(8-5)+\left[h(8)\frac{1}{4}+h(7)\frac{1}{2}+h(6)\frac{1}{4}\right]+\left[f_4(8)\frac{1}{4}+f_4(7)\frac{1}{2}+f_4(6)\frac{1}{4}\right] \end{Bmatrix}$$

$$= \text{mín}\begin{Bmatrix} 0+\left[\frac{5}{4}+\frac{4}{2}+\frac{3}{4}\right]+\left[-\frac{10}{4}-4-\frac{6}{4}\right] = 0+4-8 = -4 \\ 3+\left[\frac{3}{2}+\frac{5}{2}+1\right]+\left[-3-\frac{10}{2}-2\right] = 3+5-10 = -2 \\ 5+\left[\frac{7}{4}+3+\frac{5}{4}\right]+\left[-\frac{14}{4}-6-\frac{10}{4}\right] = 5+6-12 = -1 \\ 6+\left[2+\frac{7}{2}+\frac{3}{2}\right]+\left[-\frac{16}{4}-\frac{14}{2}-\frac{12}{4}\right] = 6+7-14 = -1 \end{Bmatrix}$$

$$f_3(5) = -4 \qquad y_3(5) = 5$$

Con $x = 0$

$$f_3(0) = \min_{0 \le y \le 3} \{c(y-0) + L^0(y) + \sum_{d=0}^{2} f_4(y-d)p(d)$$

$$= \text{mín} \begin{Bmatrix} c(0-0) + \left[h(0)\frac{1}{4} + h(-1)\frac{1}{2} + h(-2)\frac{1}{4}\right] + \left[f_4(0)\frac{1}{4} + f_4(-1)\frac{1}{2} + f_4(-2)\frac{1}{4}\right] \\ c(1-0) + \left[h(1)\frac{1}{4} + h(0)\frac{1}{2} + h(-1)\frac{1}{4}\right] + \left[f_4(1)\frac{1}{4} + f_4(0)\frac{1}{2} + f_4(-1)\frac{1}{4}\right] \\ c(2-0) + \left[h(2)\frac{1}{4} + h(1)\frac{1}{2} + h(0)\frac{1}{4}\right] + \left[f_4(2)\frac{1}{4} + f_4(1)\frac{1}{2} + f_4(0)\frac{1}{4}\right] \\ c(3-0) + \left[h(3)\frac{1}{4} + h(2)\frac{1}{2} + h(1)\frac{1}{4}\right] + \left[f_4(3)\frac{1}{4} + f_4(2)\frac{1}{2} + f_4(1)\frac{1}{4}\right] \end{Bmatrix}$$

$$= \text{mín} \begin{Bmatrix} 0 + \left[0 + \frac{5}{2} + \frac{5}{2}\right] + [0+2+2] = 0 + 5 + 4 = 9 \\ 3 + \left[\frac{1}{4} + 0 + \frac{5}{4}\right] + \left[-\frac{2}{4} + 0 + 1\right] = 3 + \frac{6}{4} + \frac{1}{2} = 5 \\ 5 + \left[\frac{2}{4} + \frac{1}{2} + 0\right] + \left[-\frac{4}{4} - \frac{2}{2} - 0\right] = 5 + 1 - 2 = 4 \\ 6 + \left[\frac{3}{4} + 1 + \frac{1}{4}\right] + \left[-\frac{6}{4} - \frac{4}{2} - \frac{2}{4}\right] = 6 + 2 - 4 = 4 \end{Bmatrix}$$

$$f_3(0) = 4 \qquad y_3(0) = 2 \text{ o } 3$$

Para $x = -1$ y con una demanda $d = 0,1,2$, los valores de y son: $y = -1,0,1,2$

$$f_3(-1) = \min_{\substack{-1 \le y \le 2 \\ y \ge x}} \{c(y+1) + L^0(y) + \sum_{d=0}^{2} f_4(y-d)p(d)\}$$

$$= \text{mín}\left\{\begin{array}{c} c(-1+1)+\left[\prod(1)\frac{1}{4}+\prod(2)\frac{1}{2}+\prod(3)\frac{1}{4}\right]+\left[f_4(-1)\frac{1}{4}+f_4(-2)\frac{1}{2}+f_4(-3)\frac{1}{4}\right] \\ c(0+1)+\left[h(0)\frac{1}{4}+h(-1)\frac{1}{2}+h(-2)\frac{1}{4}\right]+\left[f_4(0)\frac{1}{4}+f_4(-1)\frac{1}{2}+f_4(-2)\frac{1}{4}\right] \\ c(1+1)+\left[h(1)\frac{1}{4}+h(0)\frac{1}{2}+h(-1)\frac{1}{4}\right]+\left[f_4(1)\frac{1}{4}+f_4(0)\frac{1}{2}+f_4(-1)\frac{1}{4}\right] \\ c(2+1)+\left[h(2)\frac{1}{4}+h(1)\frac{1}{2}+h(0)\frac{1}{4}\right]+\left[f_4(2)\frac{1}{4}+f_4(1)\frac{1}{2}+f_4(0)\frac{1}{4}\right] \end{array}\right\}$$

$$= \text{mín}\left\{\begin{array}{c} c(0)+\left[\frac{5}{4}+5+\frac{15}{4}\right]+\left[\frac{4}{4}+4+3\right]=0+10+8=\ 18 \\ c(1)+\left[0+\frac{5}{2}+\frac{5}{2}\right]+[0+2+2]=3+5+4=12 \\ c(2)+\left[\frac{1}{4}+0+\frac{5}{4}\right]+\left[-\frac{1}{2}\mp 0+1\right]=5+\frac{3}{2}+\frac{1}{2}=\ 7 \\ c(3)+\left[\frac{1}{2}+\frac{1}{2}+0\right]+[-1-1+0]=6+1-2=\ 5 \end{array}\right\}$$

$$f_3(-1)=5 \qquad y_3(-1)=2$$

Nunca se ordena cuando *x* excede la demanda más alta para el proceso remanente. Los mismos cálculos se hacen para los demás valores de *x* (se sugiere hacerlos) y se tiene:

$f_3(4) = -3$	$y_3(4) = 4$
$f_3(3) = -2$	$y_3(3) = 3$
$f_3(2) = -1$	$y_3(2) = 2$
$f_3(1) = 2$	$y_3(1) = 1{,}2$
$f_3(-2) = 8$	$y_3(-2) = 1$
$f_3(-3) = 15$	$y_3(-3) = 0$
$f_3(-4) = 24$	$y_3(-4) = -1$

ETAPA 2

Para esta etapa, x va de 3 a –2 y, entonces, los valores de y se calculan de la manera siguiente:

$$
\begin{aligned}
y_2 &= x_3 + z_2 \\
6 &= 3 + 3 \\
5 &= 2 + 3 \\
4 &= 1 + 3 \\
3 &= 0 + 3 \\
2 &= -1 + 3 \\
1 &= -2 + 3
\end{aligned}
$$

Para $x = 3$ y con una demanda igual a $d = 0,1,2$ los valores de y son: $y = 3,4,5,6$

$$f_2(3) = \min_{3 \le y \le 6} \{c(y-3) + L^0(y) + \sum_{d=0}^{2} f_4(y-d)p(d)\}$$

$$= \text{mín} \begin{Bmatrix} c(3-3) + \left[h(3)\frac{1}{4} + h(2)\frac{1}{2} + h(1)\frac{1}{4}\right] + \left[f_3(3)\frac{1}{4} + f_3(2)\frac{1}{2} + f_3(1)\frac{1}{4}\right] \\ c(4-3) + \left[h(4)\frac{1}{4} + h(3)\frac{1}{2} + h(2)\frac{1}{4}\right] + \left[f_3(4)\frac{1}{4} + f_3(3)\frac{1}{2} + f_3(2)\frac{1}{4}\right] \\ c(5-3) + \left[h(5)\frac{1}{4} + h(4)\frac{1}{2} + h(3)\frac{1}{4}\right] + \left[f_3(5)\frac{1}{4} + f_3(4)\frac{1}{2} + f_3(3)\frac{1}{4}\right] \\ c(6-3) + \left[h(6)\frac{1}{4} + h(5)\frac{1}{2} + h(4)\frac{1}{4}\right] + \left[f_3(6)\frac{1}{4} + f_3(5)\frac{1}{2} + f_3(4)\frac{1}{4}\right] \end{Bmatrix}$$

$$= \text{mín} \begin{Bmatrix} c(0) + \left[3 + 1 + \frac{1}{4}\right] + \left[-\frac{1}{2} - \frac{1}{2} + \frac{10}{2}\right] = 0 + \frac{17}{4} + 4 = \frac{32}{4} \\ c(1) + \left[1 + \frac{3}{2} + \frac{1}{2}\right] + \left[-\frac{3}{4} - \frac{2}{2} - \frac{1}{4}\right] = 3 + 3 - 2 = 4 \\ c(2) + \left[\frac{5}{4} + 2 + \frac{3}{4}\right] + \left[-1 - \frac{3}{2} - \frac{1}{2}\right] = 5 + 4 - 3 = 6 \\ c(3) + \left[\frac{3}{2} + \frac{5}{2} + 1\right] + \left[-\frac{5}{4} - 2 - \frac{3}{4}\right] = 6 + 5 - 4 = 7 \end{Bmatrix}$$

$$= \text{mín}\begin{Bmatrix} c(0)+\left[3+1+\frac{1}{4}\right]+\left[-\frac{1}{2}-\frac{1}{2}+\frac{10}{2}\right]=0+\frac{17}{4}+4=\frac{32}{4} \\ c(1)+\left[1+\frac{3}{2}+\frac{1}{2}\right]+\left[-\frac{3}{4}-\frac{2}{2}-\frac{1}{4}\right]=3+3-2=4 \\ c(2)+\left[\frac{5}{4}+2+\frac{3}{4}\right]+\left[-1-\frac{3}{2}-\frac{1}{2}\right]=5+4-3=\ 6 \\ c(3)+\left[\frac{3}{2}+\frac{5}{2}+1\right]+\left[-\frac{5}{4}-2-\frac{3}{4}\right]=6+5-4=\ 7 \end{Bmatrix}$$

$$= \text{mín}\left\{0+2-\tfrac{1}{2}, 3+3-22, 5+4-3, 6+5-4\right\} = \text{mín}\left\{1.5,\ 4, 6, 7\right\} = 1.5$$

$$f_2(3) = 1.5 \qquad y_2(3) = 3$$

De igual manera, se calculan los demás valores para esta etapa y se tiene:

$$\begin{aligned} f_2(2) &= 2.75 & y_2(2) &= 2 \\ f_2(1) &= 5.25 & y_2(1) &= 1 \\ f_2(0) &= 7.5 & y_2(0) &= 3 \\ f_2(-1) &= 8.75 & y_2(-1) &= 2 \\ f_2(-2) &= 11.25 & y_2(-2) &= 1 \end{aligned}$$

ETAPA 1

En esta etapa, el único valor posible es $x = 0$

$$f_1(0) = \min_{0\le y\le 3}\left\{c(y)+L^0(y)+\sum_{d=0}^{2} f_2(y-d)\,p(d)\right\} =$$

$$= \text{mín}\left\{0+5+9\tfrac{1}{16}, 3+1\tfrac{1}{2}+7\tfrac{1}{4}, 5+1+5\tfrac{3}{16}, 6+2+3\tfrac{1}{16}\right\} = \text{mín}\left\{14\tfrac{1}{16}, 11\tfrac{3}{4}, 11\tfrac{3}{16}, 11\tfrac{1}{16}\right\} = 11\tfrac{1}{16}$$

$$y_1(0) = 3$$

Entonces, el mínimo costo esperado es $11\frac{1}{16}$, es decir, 11.0625. Es óptimo ordenar tres artículos al inicio del periodo 1 y en el periodo 2, lo que es óptimo para ordenar depende de la demanda en el periodo 1.

La solución se puede visualizar en la figura siguiente:

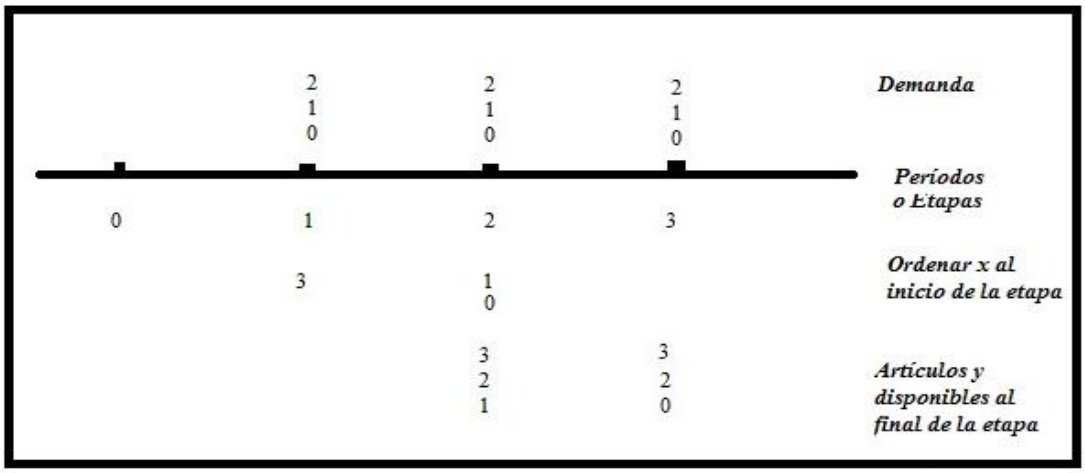


FIGURA 5.6. Diagrama de la solución para el inventario sin retraso de entrega

Al considerar el caso de las ventas perdidas, se tiene que, si hay un inventario no negativo de v artículos al final del periodo n, entonces, aún hay un costo de salvamento $s(v)$. Sin embargo, si hay un déficit al final del periodo n, entonces, no se cuenta con dicho costo al inicio del periodo $n + 1$, ya que, el exceso de la demanda se pierde.

$f_i(x)$ es como se definió en (5.9), entonces, si $D_i = d$, luego, $x_{i+1} = \text{máx}\ (y–d,\ 0)$ y $f_i(x)$ satisface la ecuación de recurrencia:

$$f_i(x) = \min_{y \geq x} \left\{ c_i(y-x) + L_i^0(y) + \alpha \sum_{d=0}^{\alpha} f_{i+1}[\text{máx}(y-d,0)] p_i(d) \right\} \quad i = 1,2,...,n$$

Para cada valor de i, $f_i(x)$ se calcula solo para valores no negativos de x, por lo que, en el caso de las ventas perdidas se requieren menos cálculos, que en el caso con déficit. La condición apropiada de frontera para este caso está dada por:

$$f_{n+1}(x) = -s(x) \quad \text{para } x \geq 0 \tag{5.12}$$

5.4.2. MODELOS CON RETRASO DE ENTREGA

En la sección anterior, se demostró cómo encontrar una política óptima de inventario para el caso de $\lambda = 0$, sin embargo, modificando un poco esta formulación se puede usar para $\lambda = 1$. Sin pérdida de generalidad, supondremos que $\lambda \geq 2$; se coloca una orden de z artículos al

inicio del periodo i, los cuales llegarán en el periodo $i + \lambda$, también, se supone que se incurre en el costo por ordenar cuando se tiene la entrega. Se supone que la compañía nunca ordena en los periodos $n - \lambda + 1$, $n - \lambda + 2,\ldots,n$ y que cualquier costo en el que se incurra al inicio del periodo $n+1$ está dado por (5.11) o (5.12) dependiendo del caso, si es con déficit o con ventas perdidas.

Para el caso con déficit, comenzamos con dos modelos de solución. El primero considera una etapa y λ variables de estado y se define como la función de valor esperado óptimo:

$f_i(w, z_{i-\lambda+1}, z_{i-\lambda+2},\ldots,z_{i-1}, z)\, p_i(d)$ = el costo descontado esperado total (descontado al periodo i) para los periodos i a n (otro costo que es debido a la entrega de las órdenes previas) dado que los w artículos estarán disponibles en el periodo i, después de $z_{i-\lambda}$ que se han entregado, se han ordenado z_j artículos en el periodo j ($j = i-\lambda + 1, i-\lambda + 2,\ldots,i - 1$) y se sigue una política óptima para ordenar. (5.13)

Si $L_i^0(w)$ está dado por (5.8), entonces, es fácil ver que f_i satisface la relación de recurrencia siguiente:

$$f_i\left(w, z_{i-\lambda+1},\ldots, z_{i-1}\right) = \min_{z\geq 0}\left\{\alpha^{\lambda} c_i\left(z\right) + L_i^0\left(w\right) + \alpha\sum_{d=0}^{\infty} f_{i+1}\left(w - d + z_{i-\lambda+1}, z_{i-\lambda+2}, z_{i-1}, z\right) p_i\left(d\right)\right\} \qquad (5.14)$$

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

donde z es la cantidad ordenada en el periodo i. La condición de frontera que, ahora, incluye los costos esperados por llevar el inventario y por déficit en los periodos $n - \lambda + 1$, $n - \lambda + 2,\ldots,n$, así como, el costo final está dado por:

$$f_{n-\lambda+1}\left(w, z_{n-2\lambda+2},\ldots, z_{n-\lambda}\right) = L_{n-\lambda+1}^0\left(w\right) + \sum_{i=n-\lambda+2}^{n} \alpha^{i-(n-\lambda+1)}\left\{\sum_{d=0}^{\infty} L_i^0\left(w + \sum_{j=n-2\lambda*2}^{i-\lambda} z_j - d\right) p_{n-\lambda+1,i-1}\left(d\right)\right\} +$$

$$+ \alpha^{\lambda}\sum_{d=0}^{\infty} t\left(w + \sum_{j=n-2\lambda+2}^{n-\lambda} z_j - d\right) p_{n-\lambda+1,n}\left(d\right)$$

(5.15)

donde $p_{i,j}(d)$ se define como al final de la sección 5.4. Suponiendo que no hay órdenes pendientes al inicio del periodo 1, entonces, $w_1 = x_1$ y el costo descontado esperado total para el problema de n periodos es $f_1(x_1,0,..,0)$.

La formulación de PD anterior requiere que para cada i $f_i(w, z_{i-\lambda+1}, z_{i-\lambda+2},\ldots,z_{i-1}, z)$, se calcule para todos los valores posibles de $w, z_{i-\lambda+1},\ldots,z_{i-1}$. Entonces, si λ es muy grande, esta formulación se hace computacionalmente intratable. Por esta razón, consideraremos otra formulación de PD que requiere una variable por etapa y una variable de estado.

Se tienen x artículos disponibles en el periodo i y se hacen las siguientes dos observaciones:

1) El número de artículos z ordenados en el periodo i no afecta en el costo incurrido en los periodos desde i hasta $i+\lambda-1$. Lo anterior, se debe a que el costo por ordenar $c_i(z)$, en el que se incurre, sucede en el periodo $i+\lambda$, más aún, la orden de z artículos que llega en el periodo $i+\lambda$ no afecta a los costos esperados por llevar inventario o por la escasez en los periodos i al $i+\lambda-1$.

   Por estas razones es posible omitir los costos en los periodos i hasta $i+\lambda-1$ en la definición de la función del valor esperado óptimo.

2) La segunda observación es que la cantidad de inventario disponible para satisfacer la demanda en el periodo $i+\lambda$, $w_{i+\lambda}$ depende, solamente, de x_i, cantidad de inventario disponible al inicio del periodo en z_i, orden o pedido en el periodo i (y no pedidos anteriores) y en $d_i, d_{i+1},\ldots,d_{i+\lambda-1}$, en particular $w_{i+\lambda}$, está dado por:

$$w_{i+\lambda} = y_i - \sum_{j=i}^{i+\lambda-1} d_j$$

Con estas dos observaciones en mente, definimos la función del valor esperado óptimo de la siguiente manera:

$f_i(x)$ = el costo descontado esperado total (descontado al periodo $i+\lambda$) para los periodos $i+\lambda$ a n, dado que hay x artículos disponibles para ordenar al inicio del periodo i, se sigue una política óptima para ordenar. (5.16)

Si se define $L_i(y)$ como el costo esperado para llevar un inventario y por déficit en el periodo $i+\lambda$, dado que $y_i = y$, entonces, $L_i(y)$ se calcula de la siguiente manera:

$$L_i(y) = \sum_{d=0}^{\infty}\left[\sum_{d'=0}^{\infty} L_{i+\lambda}^{0}(y-d,d')p_{i+\lambda}(d')\right]p_{i,i+\lambda-1}(d)$$

$$= \sum_{d=0}^{\infty} L_{i+\lambda}^{0}(y-d)p_{i,i+\lambda-1}(d)$$

La relación de recurrencia apropiada para esta formulación es la siguiente:

$$f_i(x) = \min_{y \geq x}\left\{c_i(y-x) + L_i(y) + \alpha\sum_{d=0}^{\infty} f_{i+1}(y-d)\,p_i(d)\right\} \quad i = 1,2,..,n-\lambda \tag{5.17}$$

y la condición de frontera es:

$$f_{n-\lambda+1}(x) = \sum_{d=0}^{\infty} t(x-d)p_{n-\lambda+1,n}(d) \tag{5.18}$$

Así, hay que notar que (5.17) es de la misma forma que (5.10), que es la relación de recurrencia para $\lambda = 0$. Si calculamos sucesivamente $f_{n-\lambda}$ hasta f_1, entonces, $f_1(x_1)$ es el costo descontado esperado total (descontado al periodo $\lambda + 1$) para los periodos $\lambda + 1$ hasta n. No se incluyen los costos esperados de inventarios y déficit para los periodos 1 a λ, ya que, no afectan las políticas para ordenar. El costo descontado esperado total (descontado al periodo 1) para los periodos n está dado por:

$$f_0 = L_1^0(x_1) + \sum_{i=2}^{\lambda}\left[\alpha^{i-1}\sum_{d=0}^{\infty} L_i^0(y-d)\,p_{1,i-1}(d)\right] + \alpha^{\lambda} f_1(x_1)$$

donde $L_i^0(w)$ está dado por (5.8).

Ejemplo 5.6. Encuentre la política óptima para ordenar y el costo mínimo esperado para un problema de inventario con los datos siguientes:

n = 4	λ = 2	$\alpha = 1$	$y = 0$
$p(0)$ = ¾	$p(1)$ =1/4		
$c(0)$ = 0	$c(1)$ = 3	$c(z) = \infty$	para $z \geq 2$
$h(v)$ = v	para $v \geq 0$	costo por inventario	
$\prod(v) = -3v$	para $v \geq 0$	penalización por no tener artículos	
$s(v)$ = $2v$	para $v \geq 0$	costo de salvamento	
$b(v)$ = $4v$	para $v \geq 0$	costo por comprar artículos	

Solución:

El problema se puede ver en la siguiente figura 5.7:

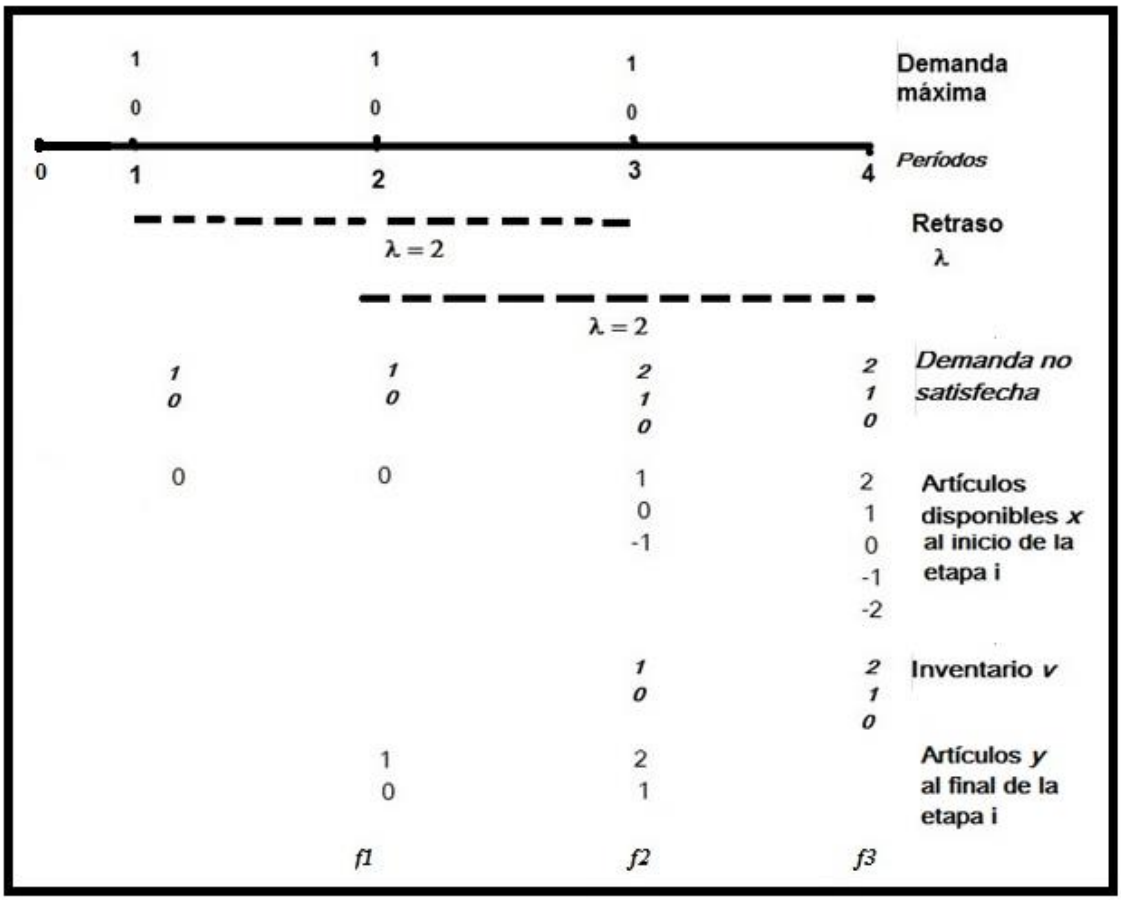


FIGURA 5.7. Inventario estocástico con retraso en la entrega

Si reflexionamos la definición de condición de frontera se tiene:

Condición de frontera

$$f_{n-\lambda+1}(x)=\sum_{d=0}^{\infty} t(x-d)p_{n-\lambda+1,n}(d)$$

Con *t* definida como:

$$t(v)=\begin{cases}-s(v), & \text{si } v\geq 0\\ b(-v), & \text{si } v<0\end{cases}$$

Para $x = -2, -1, 0, 1, 2$

donde la probabilidad está dada por la siguiente definición: la convolución de $p_i(d)$, $p_{i+1}(d),..,p_j(d)$ se denota por $p_{i,j}(d)$. Y se calcula recursivamente $p_{i,j}(d)$ (para $d = 0, 1, \ldots$) como sigue:

$$p_{i,i}(d) \equiv p_i(d)$$

$$p_{i,k}(d) = \sum_{l=0}^{d} p_{i,k-1}(l)\, p_k(d-l) \qquad k = i+1, i+2, .., j$$

Esta definición ya se había establecido antes de iniciar la sección 5.4.1, entonces, se tiene que la condición de frontera es la siguiente para $x = 2$:

$$f_{4-2+1}(2) = \sum_{d=0}^{2} t(2-d)\, p_{4-2+1,4}(d) = f_3(2) = t(2-0)\, p_{3,4}(0) + t(2-1)\, p_{3,4}(1) + t(2-2)\, p_{3,4}(2)$$

Aquí, cabe preguntarse ¿Por qué, si solo se demanda 0 o 1 artículo en cada etapa, se considera la demanda de 2? Recordemos que, si estamos en la etapa 3 puede haber una demanda acumulada de 2 artículos.

Pero, $p_{3,4}(0)$, $p_{3,4}(1)$ y $p_{3,4}(2)$ dada la definición de convolución de probabilidad son iguales a:

$P_{3,4}(0) = p_3(0)\, p_4(0) = (\,¾)\,(¾) = 9/16$

$P_{3,4}(1) = p_3(1)\, p_4(0) + p_3(0)\, p_4(1) = (¼)(¾) + (¾)(¼) = 6/16$

$P_{3,4}(2) = p_3(1)\, p_4(1) = (¼)\,(¼) = 1/16$

Al mismo tiempo, se calcula $t(v)$ donde v es el inventario disponible:

$$-s(v) \text{ para } v \geq 0 \text{ con } s(v) = 2v$$

$t(2) = -2(2) = -4$

$t(1) = -2(1) = -2$

$t(0) = -2(0) = 0$

Por lo tanto, sustituyendo en $f_3(2)$ se tiene:

$$f_3(2) = (9/16)\ t(2) + (6/16)\ t(1) + (1/16)\ t(0) = (9/16)(-4) + (6/16)(-2) + (1/16)(0)$$
$$= -9/4 - 3/4 - 0 = -3$$

Para $f_3(1)$ se tiene:

$$f_{4-2+1}(1) = \sum_{d=0}^{2} t(1-d)\, p_{4-2+1,4}(d) = f_3(1) = t(1-0)\, p_{3,4}(0) + t(1-1)\, p_{3,4}(1) + t(1-2)\, p_{3,4}(2)$$

$$f_3(1) = (9/16)\ t(1) + (6/16)\ t(0) + (1/16)\ t(-1) = -2(9/16) + 0(6/16) + 4\ (1/16) =$$
$$= -9/8 + 0 + 1/4 = -7/8$$

Observación 1: $t(-1) = 4\ (-v) = 4\ (-(-1)) = 4$

Con $f_3(0)$ se hacen los mismos cálculos:

$$f_{4-2+1}(0) = \sum_{d=0}^{2} t(0-d)\, p_{4-2+1,4}(d) = f_3(1) = t(0-0)\, p_{3,4}(0) + t(0-1)\, p_{3,4}(1) + t(0-2)\, p_{3,4}(2)$$

$$f_3(0) = (9/16)\ t(0) + (6/16)\ t(-1) + (1/16)\ t(-2) = 0\ (9/16) + 4\ (6/16) + 8\ (1/16) =$$
$$= 0 + 6/4 + 2/4 = 8/4 = 2$$

Observación 2: $t(-2) = 4\ (-v) = 4\ (-(-2)) = 8$

$$f_{4-2+1}(-1) = \sum_{d=0}^{2} t(-1-d)\, p_{4-2+1,4}(d) = f_3(-1) = t(-1-0)\, p_{3,4}(0) + t(-1-1)\, p_{3,4}(1) + t(-1-2)\, p_{3,4}(2)$$

$$f_3(-1) = (9/16)\ t(-1) + (6/16)\ t(-2) + (1/16)\ t(-3) = 4(9/16) + 8(6/16) + 12\ (1/16) =$$
$$= 9/4 + 6/2 + 3/4 = 6$$

Y, finalmente, $f_3(-2)$

$$f_3(-2) = (9/16)\ t(-2) + (6/16)\ t(-3) + (1/16)\ t(-4) = 8\ (9/16) + 12\ (6/16) + 16\ (1/16) =$$
$$= 9/2 + 9/2 + 1 = 10$$

ETAPA 3

Una vez hechos los cálculos para las condiciones de frontera, se usa la ecuación recursiva para esta etapa y la siguiente:

$$f_i(x)=\min_{y\geq x}\left\{c_i(y-x)+L_i(y)+\alpha\sum_{d=0}^{\infty}f_{i+1}(y-d)\,p_i(d)\right\}\qquad i=1,2,..,n-\lambda$$

Y, recuerde que $L_i\,(y)$ se calcula de acuerdo con (5.8).

Si se observa la gráfica, se puede ver que los valores posibles para x son 1, 0 y –1. Y para y son 1 y 2, ya que, son los artículos que se tienen al final de la etapa 2 y al inicio de la etapa 3:

$$f_2(1)=\min_{y=1,2}\left\{c_2(y-1)+L_2(y)+\sum_{d=0}^{2}f_3(y-d)\,p_2(d)\right\}$$

$$f_2(1)=\min_{y=1,2}\left\{c_2(1-1)+L_2(1)+\left[\tfrac{3}{4}f_3(1)+\tfrac{1}{4}f_3(0)\right],c_2(2-1)+L_2(2)+\left[\tfrac{3}{4}f_3(2)+\tfrac{1}{4}f_3(1)\right]\right\}=$$
$$=\min_{y=1,2}\left\{c_2(0)+L_2(1)+\left[\tfrac{3}{4}(-\tfrac{7}{8})+\tfrac{1}{4}(2)\right],c_2(1)+L_2(2)+\left[\tfrac{3}{4}(-3)+\tfrac{1}{4}(\tfrac{-7}{8})\right]\right\}$$

$$f_2(1)=\min\left\{0+L_2(1)+\left[\tfrac{3}{4}f_3(1)+\tfrac{1}{4}f_3(0)\right],3+L_2(2)+\left[-\tfrac{9}{4}-\tfrac{7}{32}\right]\right\}=$$
$$=\min\left\{0+\tfrac{15}{16}-\tfrac{5}{32},3+\tfrac{21}{16}-\tfrac{79}{32}\right\}=\min\left\{\tfrac{25}{32},1\tfrac{3}{16}\right\}=\frac{25}{32}\qquad y_2(1)=1$$

Observación 3: nuevamente, resulta útil ver lo que sucede en la figura 5.5, que se tiene de la definición (5.8):

$$L_i^0(w,d)=\begin{Bmatrix}h_i(w-d), & \text{si } w-d\geq 0\\ \prod_i(d-w)+h_i(0), & \text{si } w-d\leq 0\end{Bmatrix}$$

$$L_i^0(w)=\sum_{d=0}^{\infty}L_i^0(w,d)p_i(d)$$

además,

$$L_i(y)=\sum_{d=0}^{\infty}L_{i+\lambda}^0(y-d)p_{i,i+\lambda-1}(d)$$

Para $L_2(1)$ se hacen los cálculos del costo por llevar un artículo en inventario según la probabilidad de no haber tenido demanda en los periodos 2 y 3 y, al mismo tiempo, se calcula la penalización por no cubrir la demanda de dos artículos demandados en los periodos 2 y 3, esto se puede ver con el siguiente cálculo de probabilidades:

$$L_2(1) = L_4(0)\, p_{2,3}(0) + L_4(1)\, p_{2,3}(1) = h(v{=}1)\, p_2(0)\, p_3(0) + \prod(v = 2)\, p_2(1)\, p_3(1)$$

$$= 1(¾)(¾)+(3)(2)(¼)(¼) = 9/16 + 6/16 = 15/16$$

De la misma manera, se calcula para $L_2(2)$:

$$L_2(2) = h(v{=}2)\, p_2(0)\, p_1(0) + \prod(v{=}1)\, p_2(1)\, p_1(1) = 2(¾)(¾)+(3)(1)(¼)(¼) = 18/16 + 3/16$$
$$= 21/16$$

Para los valores de $x = 0$ y -1 se procede de manera similar y se obtienen los siguientes resultados:

$f_2(0) = 3(25/32)$ $\qquad$ $y_2(0) = 1$
$f_2(-1) = 8.25$ $\qquad$ $y_2(-1) = 0$

ETAPA 2

$$f_1(0) = \text{mín}\{0 + L_1(0) + [{}^3/_4 f_2(0) + ¼ f_2(-1)],\ 3 + L_1(1) + [{}^3/_4 f_2(1) + ¼ f_2(0)]\} =$$

$$= \text{mín}\ \{0 + 9/16 + 627/128,\ 3 + 15/16 + 196/128\} = \text{mín}\ \{5.46,\ 5.46\} \quad y_1(0) = 0,\ 1$$

Se calcula $L_1(0)$ considerando que no se tienen artículos y se tiene una demanda no satisfecha de un artículo para los periodos 1 y 2 por la probabilidad de dicha demanda y la penalización que conlleva. Y para $L_1(1)$ se hacen los cálculos del costo por llevar un artículo en inventario por la probabilidad de no haber tenido demanda en los periodos 1 y 2.

$$L_1(0) = L_3(0)\, p_{1,2}(0) = \prod(v{=}1)\, p_1(0)\, p_2(0) = 1\ (¾)(¾) = 9/16$$

$$L_1(1) = L_3(0)\, p_{1,2}(0) + L_3(0)\, p_{1,2}(1) = h(v{=}1)\, p_1(0)\, p_2(0) + \prod(v{=}2)\, p_1(1)\, p_2(1)$$

$$= 1(¾)(¾)+(3)(2)(¼)(¼) = 9/16 + 6/16 = 15/16$$

Por lo tanto, el mínimo costo esperado para los periodos 3 y 4 es 5.46 y el costo mínimo esperado para los periodos 1 al 4 es:

$$L_1^0(0)+\left[\tfrac{3}{4}L_2^0(0)+\tfrac{1}{4}L_2^0(-1)\right]+f_1(0)=\tfrac{9}{16}+\tfrac{3}{2}+5\tfrac{15}{32}=7.52$$

Se obtienen estos costos considerando que la demanda es aleatoria y que se tiene un retraso de dos periodos para la entrega de los artículos, luego, dependiendo de las demandas se encuentra la solución yendo hacia atrás y comenzando por la etapa 1 hasta la etapa 4.

La formulación anterior depende de $c_i(z)$ que incurre en el periodo $i+\lambda$, sin embargo, es posible dar una formulación similar cuando esto sucede en el periodo i.

También, existen los modelos para el caso de las ventas perdidas con $\lambda \geq 2$ y con un retraso en la entrega desconocida, sin embargo, el desarrollo es similar y no se abordará en este trabajo.

5.5. CONCLUSIONES

Los ejemplos que se han visto en este capítulo para los casos estocásticos de la PD son muy pequeños y, no obstante, se puede ver la complejidad que pueden tener, ya que, en general, como se sabe, entre más cercano sea un modelo a un problema real su complejidad va en aumento.

Para hacer todos los cálculos en problemas de mayor escala, se recomienda usar una hoja de cálculo o un método heurístico, en estos apuntes se han desarrollado de manera manual, pues es importante que se conozca a detalle la construcción y solución de los modelos.

Para el caso de los inventarios estocásticos existen más modelos, mas, estos se abordan directamente en las clases cuando es posible desarrollarlos. Sobre los modelos que se presentaron en este capítulo, se buscó la mayor claridad posible y la ayuda de diagramas para su mejor entendimiento.

5.6. NOTAS HISTÓRICAS

Programación estocástica. En el campo de la optimización matemática, la programación estocástica es un marco de referencia para la modelación de problemas de optimización que implican incertidumbre. Mientras que los problemas de optimización determinista se formulan

con parámetros conocidos, los problemas del mundo real, casi invariablemente, incluyen algunos parámetros desconocidos. Cuando los parámetros son conocidos sólo dentro de ciertos límites, un enfoque para abordar tales problemas se llama optimización robusta. Aquí, el objetivo es encontrar una solución que sea viable para todos estos datos y óptimo en algún sentido. Los modelos de programación estocástica son similares en estilo, pero se aprovechan del hecho de que las distribuciones de probabilidad que rigen los datos se conocen o se pueden estimar.

El objetivo aquí es encontrar alguna política que es factible para todos (o casi todos) los casos de datos posibles y maximizar la esperanza de alguna función de distribución asociada a las variables aleatorias. De manera más general, este tipo de modelos se formulan, resuelven analíticamente o numéricamente, y se analizan con el fin de proporcionar información útil para una toma de decisiones.

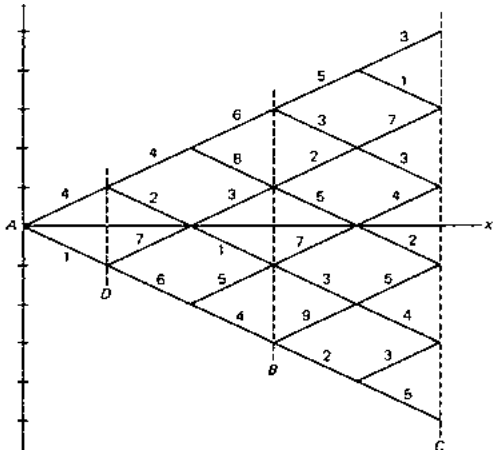
La programación dinámica estocástica se utiliza con frecuencia para modelar el comportamiento de los animales en campos como la ecología del comportamiento. Las pruebas empíricas de los modelos de forrajes óptimo, transiciones de historia de vida, tales como comportamiento incipiente en las aves y los huevos que ponen las avispas parasitoides han demostrado el valor de esta técnica de modelado en la explicación de la evolución del comportamiento y la toma de decisiones. Estos modelos suelen tener muchas etapas.

La programación dinámica estocástica es una herramienta útil en la comprensión de decisiones bajo incertidumbre. La acumulación de capital en condiciones de incertidumbre es un ejemplo; a menudo se utiliza por los administradores de recursos para analizar los problemas bioeconómicos donde la incertidumbre está presente, tales como el clima, etc.[29]

5.7. EJERCICIOS PROPUESTOS

1) Resuelva el problema de la ruta más corta del punto A a la línea C de la figura siguiente, donde $p_B = 1/3$ y $p_C = 2/3$.

[29] "Stochastic programming", trad. Idalia Flores de la Mota, 2015. Disponible en: http://en.wikipedia.org/wiki/Stochastic_programming



2) Encuentre la política óptima para ordenar y el costo mínimo esperado para un problema de inventario con los datos siguientes:

$n = 4$	$\lambda = 2$	$\alpha = 1$	$x_1 = 0$
$p(0) = ¾$	$p(1) = ¼$		
$c(0) = 0$	$c(1) = 3$	$c(z) = \infty$	para $z \geq 2$
$h(v) = 3$	para $v \geq 0$		
$\Pi(v) = 3v$	para $v \geq 0$		
$s(v) = 2v$	para $v \geq 0$		
$b(v) = 4v$	para $v \geq 0$		

3) Elabore un modelo para el problema de reemplazo del equipo estocástico.

CAPÍTULO 6

PROBLEMAS CON DINÁMICA LINEAL Y CRITERO CUADRÁTICO[30]

6.1. INTRODUCCIÓN

Hasta ahora, hemos considerado problemas específicos y cómo formularlos con PD (Programación Dinámica). En este capítulo, trataremos una clase de problemas más generales (Dreyfuss y Law, 1977) que tienen una estructura matemática en común, diferente a los problemas de PL (Programación Lineal), pero análoga a ellos. Para esta clase de problemas después de la formulación de PD, veremos un análisis matemático que nos brindará un método más simplificado de solución computacional. Pocos problemas de índole práctico caen en la clase de problemas que estudiaremos en este capítulo, pero, muchos de ellos se pueden aproximar, de manera bastante precisa, en los pasos sucesivos. Esto es similar al caso de la PL donde en la realidad muchos problemas no son realmente lineales, pero se pueden aproximar con esta estructura. Para ilustrarlo, de manera lo más sencillo posible, consideremos el problema:

¿Para cuál valor de x se minimiza $f(x)$? Cuando está dada por:

$$f(x) = x^4 - 4x^3 + 8x^2 - 4x + 1$$

Al sacar la derivada e igualarla a cero se tiene una ecuación cúbica que no es fácil de resolver, pero como se sabe que $f(x)$ es cuadrática, su derivada será lineal y, entonces, al igualarla a cero es más fácil de resolver. Por esta razón, es que se usan las funciones cuadráticas como funciones de aproximación.

Unos cuántos cálculos nos dicen que el mínimo de $f(x)$ ocurre en algún punto cerca de $x = 0$, entonces, tomamos $x = 0$ como una primera aproximación x_0, que minimiza x y se construye

[30] Dreyfus, Stuart E., and Averill M. Law, *Art and Theory of Dynamic Programming*, Orlando, Academic Press, 1977.

la función cuadrática que coincida con $f(x)$, en el valor de las dos primeras derivadas con x=0, llamando a esta función de aproximación $f_0(x)$ y obteniendo:

$$f_0(x) = f(x_0) + f'(x)\Big|_{x=x_0}(x-x_0) + \frac{1}{2}f''(x)\Big|_{x=x_0}(x-x_0)^2 = 1 - 4x + 8x^2$$

hacemos $f'_0(x)$ igual a cero y obtenemos:

$$0 = -4 + 16x \text{ y despejando } x \text{ se tiene } x = ¼$$

Tomamos ¼ en la inmediata aproximación llamada x_1 y se repiten los pasos anteriores. La función $f_1(x)$ que concuerda con $f(x)$ en valor y en las dos primeras derivadas en x = 1/4 es:

$$f_1(x) = \frac{113}{256} - \frac{11}{16}\left(x - \frac{1}{4}\right) + \frac{43}{8}\left(x - \frac{1}{4}\right)^2$$

entonces, se encuentra el valor de x que minimiza $f_1(x)$:

$$0 = \frac{11}{16} + \frac{43}{4}\left(x - \frac{1}{4}\right)$$

despejando x se obtiene x = 27/86 = 0.31395

Este esquema de aproximación sucesiva se conoce como el método de Newton-Raphson. El método tiene la desventaja de que no siempre converge o lo que se obtiene es un mínimo relativo y no el mínimo general, pero, lo que es importante, es que aproxima en cada paso al problema original por uno más sencillo de resolver y, si todo va bien, por uno que sea una buena aproximación del problema original, en cada paso, en una vecindad del mínimo.

Método de Newton-Raphson

En la figura siguiente, se puede ver una descripción gráfica del método. Si la estimación inicial para la raíz es x_i, se puede extender una tangente desde el punto [x_i, $f(x_i)$]. El punto donde la tangente cruza al eje de las x representa, usualmente, un estimado mejorado de la raíz.

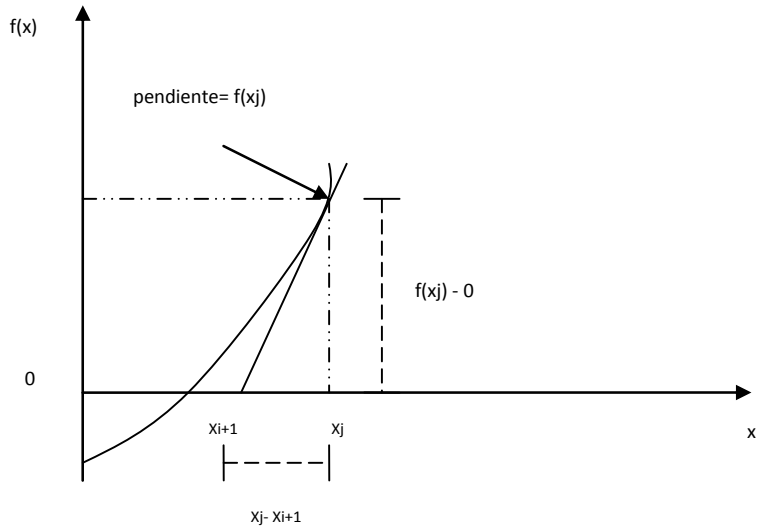


FIGURA 6.1. Método Newton Raphson

El método de Newton-Raphson se puede derivar con base en esta interpretación geométrica. Como en la figura 6.1, la primera derivada de x_i es equivalente a la pendiente:

$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}}$$

y reacomodando términos tenemos:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

que es conocida como la *fórmula Newton-Raphson.*

6.2. UN MODELO CON CRITERIO CUADRÁTICO Y DINÁMICA LINEAL

Se denota $x(i)$ el estado en la etapa i, entonces, sea $y(i)$ la decisión. En tal caso, suponemos que el estado en la etapa i +1 está dado por:

$$x(i+1) = g(i)x(i) + h(i)y(i) \qquad (6.1)$$

Donde $g(i)$ y $h(i)$ con $i = 0,\ldots,N–1$ son constantes conocidas. Esto se reconoce como un sistema lineal dinámico, ya que, la regla que da un nuevo estado es lineal en el viejo estado y en la decisión. Existen muchos ejemplos de sistemas dinámicos lineales y los no lineales son raros en el área de la investigación de operaciones, aunque, abundantes en otros como en los procesos químicos o en la aeronáutica. En la localización de recursos, donde $x(i)$ es el recurso total cuando consideramos actividades i a través de N y $y(i)$, que es la localización de la actividad i, se tiene la ecuación lineal dinámica siguiente:

$$x(i + 1) = x(i) – y(i)$$

En problemas de inventario, si $x(i)$ es el nivel de inventario empezando en la etapa i, $y(i)$ es la cantidad ordenada e inmediatamente entregada, $d(i)$ es la demanda durante la etapa i y es cuando se tiene la siguiente regla dinámica lineal:

$$x(i + 1) = x(i) + y(i) – d(i)$$

La cual es de la forma (6.1) excepto por la constante que se resta $d(i)$ y que se puede tratar por el mismo método. Independientemente, de todo, suponemos que $x(i)$ y $y(i)$ son variables continuas que pueden tomar valores reales ya sean positivos o negativos, por lo que, se puede hacer uso del cálculo.

El criterio es una suma de costos sobre N etapas más un costo terminal que depende de $x(N)$. Sin embargo, el costo de cada etapa es una función cuadrática de $x(i)$ y $y(i)$, por lo que, se designa *criterio cuadrático*. Suponemos en este ejemplo simple que el criterio J está dado por:

$$J = \sum_{i=0}^{N-1}\left[a(i)x^2(i) + c(i)y^2(i)\right] + lx^2(N)i = 0 \qquad (6.2)$$

Ya que, el estado en la etapa es cero $x(0)$, nuestro problema es escoger de las y correspondientes $y(0)$, $y(1),\ldots,y(N – 1)$ la que minimiza J, que está dada por (6.2), donde los estados $x(i)$ siguen la regla (6.1).

6.3. UN PROBLEMA PARTICULAR

Para hacer precisa la naturaleza de nuestro modelo, considere el siguiente problema. Puesto que $x(0) = 2$, seleccione $y(0)$, $y(1)$ y $y(2)$ para minimizar J que está dada por:

$$J = y^2(0) + 12x^2(1) + 2y^2(1) + 2x^2(2) + y^2(2) + \frac{1}{4}x^2(3) \qquad (6.3)$$

donde:

$$x(1) = ½\, x(0) + 1/6\, y(0)$$
$$x(2) = 3x(1) + ½\, y(1)$$
$$x(3) = 4x(2) + 2y(2) \qquad (6.4)$$

Este es un problema del tipo anterior con:

$a(0) = 0$	$c(0) = 1$	
$a(1) = 12$	$c(1) = 2$	
$a(2) = 2$	$c(2) = 1$	
$l = 1/4$		
$g(0) = ½$	$h(0) = 1/6$	
$g(1) = 3$	$h(1) = 1/2$	
$g(2) = 4$	$h(2) = 2$	$N = 3$

Una solución posible, pero no óptima es:

$y(0) = 6$	$y(1) = 2$	$y(2) = ½$ por (6.4) obtenemos:
$x(1) = 1 + 1 = 2$	$x(2) = 6 + 1 = 7$	$x(3) = 28 + 1 = 29$

y, entonces,

$$J = 36 + 48 + 8 + 98 + 1/4 + 841/4 = 400\ ½$$

Ejemplo 6.1. Para el problema anterior, una mejor solución consiste en hacer $x(1)$ igual a cero al seleccionar $y(0) = –6$ y, entonces, hacer todas las y subsecuentes igual a cero. Evalúe el costo de esta solución.

El problema lo podemos solucionar resolviendo primero para las *x´s* en términos de las *y´s* y obtener:

$$x(1) = 1 + \frac{1}{6}y(0)$$
$$x(2) = 3 + \frac{1}{2}y(0) + \frac{1}{2}y(1)$$
$$x(3) = 12 + 2y(0) + 2y(1) + 2y(2)$$

Solución:

El costo de esta solución es $J = 36$.

Entonces, se sustituyen las x en J obteniendo una función cuadrática en las *y´s*. Se pueden hacer $\delta J/\delta y(0) = 0$, $\delta J/\delta y(1) = 0$, $\delta J/\delta y(2) = 0$ y se resuelven las tres ecuaciones lineales resultantes para minimizar las *y´s*.

$$y(0) = -3 \; y(1) = -1 \; y(2) = -1$$

por lo tanto, $x(1) = 0.5, x(2) = 1, x(3) = 2$ y $J = 18$

6.4. Solución usando PD (Programación Dinámica)

De la manera usual, comenzamos definiendo el óptimo de la función $V_i(x)$ para el problema dado por (6.1) y (6.2) y expuesto así:

$$V_i(x) = \min_y \left[a(i)x^2 + c(i)y^2 + V_{i+1}\left(g(i)x + h(i)y\right) \right] \tag{6.5}$$

Condición de frontera:

$$V_N(x) = lx^2 \tag{6.6}$$

Se puede usar o no esta condición de frontera, dados los datos a, c, g, h y l para evaluar $V_N(x)$ en una disposición discreta de puntos tomados entre algunos límites superiores e inferiores, que sentimos que seguramente incluirán la solución óptima para la condición inicial dada.

Si $x(0) = 2$ podríamos usar una disposición consistente en:

$$-5, -4.9, -4.8,\ldots 0,\ldots 4.9, 5$$

De este modo, podemos determinar $V_{N\text{-}1}(x)$ en los mismos puntos al considerar solo aquellos valores de $y(N-1)$ dada $x(N-1)$, que lleva a los puntos de la disposición en donde $V_N(x)$ se ha calculado o usando una disposición de decisiones razonables (por ejemplo, $y = -5, -4.9,\ldots,0,\ldots,4.9, 5$) y cuando necesitamos conocer $V_N(x)$ en un punto que no se ha calculado, usamos una fórmula de interpolación. Esto es, podemos reemplazar el problema por algún tipo de aproximación discreta. En su lugar, usamos el cálculo para conocer las expresiones exactas para $V_i(x)$ basadas en (6.5) y (6.6) de donde se obtiene la siguiente fórmula:

$$V_{N-1}(x) = \min_y \left[a(N-1)x^2 + c(N-1)y^2 + V_N\left(g\left(N-1\right)x + h(N-1)y\right)\right] =$$

$$\min_y \left[a(N-1)x^2 + c(N-1)y^2 + lg^2(N-1)x^2 + 2lg(n-1)h(N-1)xy + lh^2(N-1)y^2 \right] \quad (6.7)$$

Ahora, usamos el cálculo para determinar el valor de y en términos de x, que minimiza la expresión entre los corchetes de (6.7), se iguala la segunda derivada a cero y se tiene:

$$0 = 2c(N-1)y + 2lg(N-1)h(N-1)x + 2lh^2(N-1)y$$

$$y = -\frac{lg(N-1)h(N-1)x}{c(N-1) + lh^2(N-1)} \quad (6.8)$$

este valor nos da el mínimo de y, si:

$$c(N-1) + lh^2(N-1) > 0 \quad (6.9)$$

Si el lado izquierdo de (6.9) es igual a cero y l, $g(N-1)$, $h(N-1)$ y x son diferentes de cero, entonces, $V_{N\text{-}1}(x)$ se puede hacer igual a $-\infty$ con una selección apropiada de y; y, si es negativa, y dada por (6.8) da el máximo y no existe el mínimo, es decir, seleccionando y, si es lo suficiente grande positivo o negativo el valor de la expresión en los corchetes de (6.7), se puede hacer tan pequeña como se desee. En estos casos, se dice que el problema no tiene solución. Entonces, suponemos que (6.9) se cumple y siempre que hagamos la derivada igual a cero, podemos resolver para y, y será mínima. No hay posibilidad de obtener un mínimo relativo ni uno absoluto, ya que, estamos tratando con funciones cuadráticas.

Y conociendo que la y que minimiza es una función lineal del estado x y es dada por (6.8), sustituimos y en (6.7) para obtener $V_{N\text{-}1}(x)$, lo que nos da:

$$V_{N\text{-}1}(x) = a(N-1)x^2 + c(N-1)\left[-\frac{lg(N-1)h(N-1)x}{c(N-1)+lh^2(N-1)}\right]^2 + lg^2(N-1)x^2 + 2lg(N-1)h(N-1)x$$

$$\times \left[-\frac{lg(N-1)h(N-1)x}{c(N-1)+lh^2(N-1)}\right] + lh^2(N-1)\left[-\frac{lg(N-1)h(N-1)x}{c(N-1)+lh^2(N-1)}\right]^2 =$$

$$\left[a(N-1)+lg^2(N-1)-\frac{l^2g^2(N-1)h^2(N-1)}{c(N-1)+lh^2(N-1)}\right]x^2 = p(N-1)x^2 \qquad (6.10)$$

Donde hemos denotado el coeficiente de x^2, que se puede calcular fácilmente de los datos dados por $p(N-1)$. Note que el valor óptimo de la función para el proceso comenzando en la etapa $N-1$ es una función cuadrática del estado x.

Ahora, usaremos (6.5) con $i = N-2$ para calcular $V_{N\text{-}2}(x)$. Si escribimos la fórmula usando (6.10) para $V_{N\text{-}1}(x)$ en la derecha, obtenemos exactamente (6.7) excepto que l en (6.7) es reemplazado por $p(N-1)$. Entonces, podemos escribir inmediatamente que y en la etapa $N-2$ está dada en términos de x en la etapa $N-2$ por:

$$y = -\frac{p(N-1)g(N-2)h(N-2)x}{c(N-2)+p(N-1)h^2(N-2)} \qquad (6.11)$$

Y de (6.10) se deduce que $V_{N\text{-}2}(x)$ está dada por:

$$V_{N\text{-}2}(x) = p(N-2)x^2 \qquad (6.12)$$

donde

$$p(N-2) = a(N-2) + p(N-1)g^2(N-2) - \frac{p^2(N-1)g^2(N-2)h^2(N-2)}{c(N-2)+p(N-1)h^2(N-2)} \qquad (6.13)$$

Este proceso se puede repetir muchas veces para obtener $V_{N\text{-}3}(x)$, $V_{N\text{-}4}(x),\ldots,V_0(x)$. Nuestro resultado general es que para $i = 0, 1,\ldots,N-1$, $V_i(x)$ está dada por:

$$V_i(x) = p(i)x^2 \qquad (6.14)$$

Donde $p(i)$ se determina recursivamente por $p(i+1)$ con la fórmula siguiente:

$$p(i) = a(i) + p(i+1)g^2(i) - \frac{p^2(i+1)g^2(i)h^2(i)}{c(i) + p(i+1)h^2(i)} \tag{6.15}$$

Y la condición de frontera:

$$P(N) = l \tag{6.16}$$

La y óptima en la etapa i para una $x(i)$ dada está determinada por:

$$y(i) = -\frac{p(i+1)g(i)h(i)x(i)}{c(i) + p(i+1)h^2(i)} \tag{6.17}$$

La manera elegante de probar este resultado es por inducción, notando que $V_N(x)$ es lx^2, suponemos que $V_{i+1}(x)$ tiene la forma:

$$V_{i+1}(x) = p(i+1)x^2 \tag{6.18}$$

Si usamos (6.5) como se hizo para obtener (6.7), (6.8) y (6.10), demostramos que $V_i(x)$ tiene la forma (6.14), cumpliendo las condiciones (6.15) y (6.16). Para asegurar que (6.17) da el mínimo de y, suponemos basados en (6.9) que:

$$c(i) + p(i+1)h^2(i) > 0 \tag{6.19}$$

Ejemplo 6.2. Usando las fórmulas anteriores, resuelva el mismo problema y verifique la secuencia óptima de $y(i)$:

$$\text{p(3)} = ¼ \qquad p(2) = 4 \qquad p(1) = 36 \qquad p(0) = 9/2$$

$$y(0) = -\frac{(36)\left(\frac{1}{2}\right)\left(\frac{1}{6}\right)(2)}{1 + 36\left(\frac{1}{36}\right)} = -3$$

$$x(1) = \frac{1}{2} \qquad y(1) = -\frac{(4)(3)\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)}{2 + (4)\left(\frac{1}{4}\right)} = -1$$

$$x(2) = 1,\ y(2) = -1,\ x(3) = 2,\ J = 18$$

$$V_0(2) = p(0)\, x^2(0) = (9/2)(4) = 18$$

*Ejemplo 6.3***.** Considere el problema de minimización donde el criterio J es la función cuadrática general:

$$J = \sum_{i=0}^{N-1}\left[a(i)x^2(i) + b(i)x(i)y(i) + c(i)y^2(i) + d(i)x(i) + e(i)y(i) + f(i)\right] + lx^2(N) + wx(N) + z$$

La función general recursiva de PD está dada por:

$$x(i+1) = g(i)x(i) + h(i)y(i) + k(i)$$

Pruebe por inducción que el óptimo de la función $V_i(x)$ tiene la forma:

$$V_i(x) = p(i)x^2 + q(i)x + r(i) \qquad i = 0,1,\ldots,N$$

Determine las fórmulas recursivas para $p(i)$, $q(i)$ y $r(i)$.

Solución:

$$V_i(x) = \min_y \left\{ a(i)x^2 + b(i)xy + c(i)y^2 + d(i)x + e(i)y + f + p(i+1)\left[g(i)x + h(i)y + k(i)\right]^2 + \right.$$
$$\left. + q(i+1)\left[g(i)x + h(i)y + k(i)\right] + r(i+1) \right\}$$

$$y = -\frac{\left[b(i) + 2p(i+1)g(i)h(i)x\right] + e(i) + 2p(i+1)h(i)k(i) + q(i+1)h(i)}{2\left[c(i) + p(i+1)h^2(i)\right]}$$

entonces,

$$V_i(x) = \left\{ a(i) + p(i+1)g^2(i) - \frac{\left[b(i) + 2p(i+1)g(i)h(i)\right]^2}{4\left[c(i) + p(i+1)h^2(i)\right]} \right\} x^2 +$$

$$+ \left\{ d(i) + 2p(i+1)k(i)g(i) + q(i+1)g(i) - \frac{\left[b(i) + 2p(i+1)g(i)h(i)\right]\left[e(i) + 2p(i+1)h(i)k(i) + q(i+1)h(i)\right]}{2\left[c(i) + p(i+1)h^2(i)\right]} \right\} x +$$

$$+ \left\{ f(i) + p(i+1)k^2(i) + q(i+1)k(i) + r(i+1) - \frac{\left[e(i) + 2p(i+1)h(i)k(i) + q(i+1)h(i)\right]^2}{4\left[c(i) + p(i+1)h^2(i)\right]} \right\} =$$

$$= p(i)x^2 + q(i)x + r(i)$$

con $p(N) = l$, $q(N) = w$, $r(N) = z$

Ejemplo 6.4. Para un problema similar al de la sección 6.3, $r(i)$ será igual a cero para toda i, si y solo si, cuando el estado inicial es cero y el costo óptimo del proceso restante es cero. ¿Por qué? $q(i)$ será cero para toda i, si y solo si, el costo óptimo es, tal que, $Vi\ (x) = Vi\ (-x)$ para toda x. ¿Por qué?

Mediante la inspección de (6.1) y (6.2), deduzca que $q(i)$ y $r(i)$ son cero para toda i como se muestra en (6.14).

Solución:

Dado que $V_i(x) = p(i)^2 + q(i)x + r(i)$, $V_i(0) = r(i)$. $V_i(x) - V_i(-x) = 2q(i)x$, por lo tanto, es el resultado.

Si $x_i(0)$ es igual a cero, tomando en cuenta las subsecuentes y'*s* igual a cero, todas las x'*s* subsecuentes son cero y el costo es 0. El problema no puede tener un mínimo finito menor a cero; si así fuese, multiplicando todas las y'*s* y las x'*s* en esa solución, por una constante K diferente de cero y aun satisficiera a (6.1) y multiplicando el valor crítico por K^2, da un valor crítico menor y, por tanto, una contradicción.

Ahora, considere la secuencia optima de y'*s* y x'*s* empezando desde $x_0(i)$ con un valor positivo dado de $x(i)$. Para un punto inicial $-x_0(i)$, revirtiendo los signos de las y'*s* en la solución previa, (6.1) dará la misma secuencia de x'*s* como la anterior, pero, con los signos al revés. (6.2) dará el mismo valor crítico para las dos soluciones. El problema que empieza en $-x_0(i)$ no puede tener una mejor solución, a menos que, se cambien los signos de las x'*s* y y'*s* y se

tendría una mejor solución que el del problema original, empezando en $x_0(i)$ que contradiría la suposición que la solución era la óptima.

6.5. CONDICIONES ESPECÍFICAS DE TERMINACIÓN

Para el problema definido en la sección 6.2 por (6.1) y (6.2), $x(N)$, el valor terminal para x, no estaba especificado, pero, sí se asignaba un costo $lx^2(N)$ al punto terminal. Ahora, veamos que sucede, si se requiere que las decisiones $y(i)$ se seleccionen, de tal manera que, $x(N) = t$, un número dado. No se asocia un costo al punto terminal, ya que, todas las políticas de decisión admisibles conducen a ello. Primero hay que hacer notar que $V_N(x)$ está, ahora, definido solo por $x = t$, ya que, si iniciamos en la etapa N, en cualquier otro punto, no hay forma de moverse hacia el punto final especificado en la misma etapa. Sin embargo, $V_{N\text{-}1}(x)$ queda definido para toda x en (6.1), dada $x(N–1)$, entonces, la $y(N–1)$ arroja $x(N) = t$. En términos de $x(N–1)$ se tiene:

$$t = g(N-1)x(N-1) + h(N-1)y(N-1) \tag{6.20}$$

Si suponemos que $h(N–1)$ es diferente de cero, se despeja $y(N–1)$ y se tiene:

$$y(N-1) = \frac{t - g(N-1)x(N-1)}{h(N-1)} \tag{6.21}$$

El costo para la etapa $N–1$ asociado a $V_{N\text{-}1}(x)$, si se inicia con el estado x y se usa (6.21) para determinar y, queda como sigue:

$$V_{N-1}(x) = a(n-1)x^2 + c(N-1)\left[\frac{t - g(N-1)x}{h(N-1)}\right]^2 = \left[a(N-1) + \frac{c(N-1)g^2(N-1)}{h^2(N-1)}\right]x^2 +$$

$$\left[-\frac{2c(N-1)tg(N-1)}{h^2(N-1)}\right]x + \left[\frac{c(N-1)t^2}{h^2(N-1)}\right] \tag{6.22}$$

De este modo, $V_{N\text{-}1}(x)$ tiene la forma $p(N–1)$, $q(N–1)$ y $r(N–1)$ dadas entre corchetes en la expresión (6.22).

Ejemplo 6.5. Resuelva el problema de la sección 6.3 ignorando el término $\frac{1}{4}x^2(3)$ del costo terminal, con la condición terminal $x(3) = 2$.

Solución:

Por (6.22) se tiene:

$$V_2(x) = 6x^2 - 4x + 1$$

$P(1) = 300/7$ y $q(1) = -48/7$ por el resultado del problema (6.4) y, por eso mismo, el resultado es $y(0) = -3$, por la PD:

$$x(1) = ½ \qquad y(1) = -1 \qquad x(2) = 1$$

Por (6.21) se tiene:

$$y(2) = -1 \qquad \text{y} \qquad x(3) = 2$$

6.6. VERSIÓN MATRICIAL DEL PROBLEMA[31]

El problema de control de un sistema lineal con una función objetivo cuadrática, también, se puede enunciar como sigue:

$$\underset{\{u(k)\}_{k=0}^{N-1}}{\text{maximizar}}\, J = \frac{1}{2}\sum_{k=0}^{N-1}\left[x(k)^T W_k x(k) + u(k)^T \Lambda_k u(k)\right] + \frac{1}{2}x(N)^T W_N x(N) \qquad (6.23)$$

sujeta a

$$x(k+1) = A_k\, x(k) + B_k\, u(k), \text{ para } k = 0,1,\ldots,N-1,$$
$$\text{con } x(0) = x_0$$

[31] Cerdá, Emilio, *Optimización dinámica*, Madrid, Prentice Hall, 2001.

En donde para una k dada, se tiene:

- $x(k)$ es el vector de estado (n-dimensional).
- $u(k)$ es el vector de control (m-dimensional).
- A_k, B_k, W_k, Λk son matrices dadas, siendo A_k de dimensiones nxn, B_k:nxm, W_k:nxn, simétrica y semidefinida negativa, Λ_k:mxm y definida negativa, v^T es el vector transpuesto de v.

Con la siguiente proposición, el problema se resuelve usando PD.

Proposición:

Para el problema (4.23), se obtiene la siguiente solución óptima: para k = 0, 1,…, N–1 el control óptimo y la función óptima son, respectivamente:

$$u^*(k) = G_k x(k) \qquad J^*_k\{x(k)\} = \tfrac{1}{2}\, x(k)^T K_k\, x(k)$$

en donde,

$$G_k = -\Theta_k^{-1}\Psi_k^T$$

y

$$K_k = \Phi_k - \Psi_k\Theta_k^{-1}\Psi_k^T \qquad \text{para } k = 0,1..,N-1 \qquad K_N = W_N$$

siendo

$$\Phi_k = W_k + A_k^T K_{k+1} A_k$$

$$\Theta_k = \Lambda_k + B_k^T K_{k+1} B_k$$

$$\Psi_k = A_k^T K_{k+1} B_k$$

además, la evolución del sistema controlado está dada por:

$$x^*(k+1) = \left[A_k + B_k G_k\right] x^*(k) \qquad \text{para } k = 0,1,\ldots,N-1 \qquad (6.24)$$

Demostración:

Cuando se usa la inducción sobre *k* y resolviendo el problema recursivamente hacia atrás, se tiene:

Final:
Sea $x(N)$ dado.

La aportación a la función objetivo por el hecho de que, el sistema finalice en dicho estado es:

$$J_N^*\{x(N)\} = \frac{1}{2}x(N)^T W_N x(N) = \frac{1}{2}x(N)^T K_N x(N) \tag{6.25}$$

En donde se ha definido la matriz $K_N=W_N$, la cual será útil en los próximos pasos.

ETAPA *N*
Sea $x(N-1)$ dada y la ecuación de Bellman para esta etapa es:

$$J_{N-1}^*\{x(N-1)\} = \max_{u(N-1)}\left\{\begin{array}{l}\frac{1}{2}x(N-1)^T W_{N-1}x(N-1)+\frac{1}{2}u(N-1)^T \Lambda_{N-1}u(N-1)+\\ +J_N^*\{A_{N-1}x(N-1)+B_{N-1}u(N-1)\}\end{array}\right\}$$

Al tomar en cuenta (4.25), se obtiene que la expresión anterior es igual a:

$$J_{N-1}^*\{x(N-1)\} = \max_{u(N-1)}\left\{\begin{array}{l}\frac{1}{2}x(N-1)^T W_{N-1}x(N-1)+\frac{1}{2}u(N-1)^T \Lambda_{N-1}u(N-1)+\\ +\frac{1}{2}\{A_{N-1}x(N-1)+B_{N-1}u(N-1)\}^T K_N\left[A_{N-1}x(N-1)+B_{N-1}u(N-1)\right]\end{array}\right\}$$

$$= \max_{u(N-1)}\left\{\begin{array}{l}\frac{1}{2}x(N-1)^T\left[W_{N-1}+A_{N-1}^T K_N A_{N-1}\right]x(N-1)+\\ +\frac{1}{2}u(N-1)^T\left[\Lambda_{N-1}+B_{N-1}^T K_N B_{N-1}\right]u(N-1)+x(N-1)^T A_{N-1}^T K_N B_{N-1}u(N-1)\end{array}\right\}$$

$$= \max_{u(N-1)} \left\{ \frac{1}{2} x(N-1)^T \Phi_{N-1} x(N-1) + \frac{1}{2} u(N-1)^T \Theta_{N-1} u(N-1) + x(N-1)^T \Psi_{N-1} u(N-1) \right\}$$

en donde:

$$\Phi_{N-1} = W_{N-1} + A_{N-1}^T K_N A_{N-1}$$

$$\Theta_{N-1} = \Lambda_{N-1} + B_{N-1}^T K_N B_{N-1}$$

$$\Psi_{N-1} = A_{N-1}^T K_N B_{N-1}$$

Imponiendo la condición necesaria para el mínimo (gradiente igual a cero), se obtiene que:

$$u(N-1)^T \Theta_{N-1} + x(N-1)^T \Psi_{N-1} = 0$$

por ser Θ_{N-1} una matriz simétrica, entonces, se tiene:

$$\Theta_{N-1} u(N-1) + \Psi_{N-1}^T x(N-1) = 0$$

de donde se llega a:

$$u(N-1) = -\Theta_{N-1}^{-1} \Psi_{N-1}^T x(N-1)$$

Así, es importante notar que la matriz Θ_{N-1} posee inversa, ya que, por hipótesis Λ_{N-1} es definida negativa y $K_N = W_N$ es semidefinida negativa, por lo que, Θ_{N-1} es definida negativa y, por lo tanto, no es singular y posee matriz inversa. También, la condición es suficiente para la optimalidad global, ya que, la función objetivo es cóncava y el máximo es un máximo global.

Se puede poner:

$$u^*(N-1) = G_{N-1} x(N-1)$$

en donde:

$$G_{N-1} = -\Theta_{N-1}^{-1} \Psi_{N-1}^T = -\left[\Lambda_{N-1} + B_{N-1}^T K_N B_{N-1} \right]^{-1} B_{N-1}^T K_N A_{N-1}$$

de este modo:

$$J_{N-1}^* \{x(N-1)\} = \frac{1}{2} x(N-1)^T \Phi_{N-1} x(N-1) + \frac{1}{2} x(N-1)^T G_{N-1}^T \Theta_{N-1} G_{N-1} x(N-1) + \\ + x(N-1)^T \Psi_{N-1} G_{N-1} x(N-1)$$

se puede poner:

$$J^*_{N-1}\{x(N-1)\} = \frac{1}{2}x(N-1)^T K_{N-1}x(N-1)$$

en donde:

$$K_{N-1} = \Phi_{N-1} + G^T_{N-1}\Theta_{N-1}G_{N-1} + 2\Psi_{N-1}G_{N-1} = \Phi_{N-1} - \Psi_{N-1}\Theta^{-1}_{N-1}\Psi^T_{N-1}$$

Hipótesis de inducción:

ETAPA $K+1$

Sea $x(k)$ dado. Supongamos que la proposición es cierta para k (según la hipótesis de inducción). Ahora, hay que demostrar que, también, se cumple para k–1 (que corresponde a la etapa k).

ETAPA K

Sea $x(k–1)$ dado. La ecuación de Bellman para esta etapa es:

$$J^*_{k-1}\{x(k-1)\} = \max_{u(k-1)}\left\{\begin{array}{l}\frac{1}{2}x(k-1)^T W_{k-1}x(k-1) + \frac{1}{2}u(k-1)^T \Lambda_{k-1}u(k-1) + \\ +J^*_k\{A_{k-1}x(k-1) + B_{k-1}u(k-1)\}\end{array}\right\}$$

Al tener en cuenta la expresión para J_k* de la hipótesis de inducción, se tiene que la expresión anterior es igual a:

$$J^*_{k-1}\{x(k-1)\} = \max_{u(k-1)}\left\{\begin{array}{l}\frac{1}{2}x(k-1)^T W_{k-1}x(k-1) + \frac{1}{2}u(k-1)^T \Lambda_{N-1}u(k-1) + \\ +\frac{1}{2}\{A_{k-1}x(k-1) + B_{k-1}u(k-1)\}^T K_k\left[A_{k-1}x(k-1) + B_{k-1}u(k-1)\right]\end{array}\right\}$$

$$= \max_{u(k-1)} \left\{ \begin{array}{l} \frac{1}{2} x(k-1)^T \left[W_{k-1} + A_{k-1}^T K_k A_{k-1} \right] x(k-1) + \\ + \frac{1}{2} u(k-1)^T \left[\Lambda_{k-1} + B_{k-1}^T K_k B_{k-1} \right] u(k-1) + x(k-1)^T A_{k-1}^T K_k B_{k-1} u(k-1) \end{array} \right\}$$

$$= \max_{u(k-1)} \left\{ \frac{1}{2} x(k-1)^T \Phi_{k-1} x(k-1) + \frac{1}{2} u(k-1)^T \Theta_{k-1} u(k-1) + x(k-1)^T \Psi_{k-1} u(k-1) \right\}$$

en donde:

$$\Phi_{k-1} = W_{k-1} + A_{k-1}^T K_k A_{k-1}$$

$$\Theta_{k-1} = \Lambda_{k-1} + B_{k-1}^T K_k B_{k-1}$$

$$\Psi_{k-1} = A_{k-1}^T K_k B_{k-1}$$

Imponiendo la condición necesaria del mínimo con el gradiente igual a cero, se tiene que:

$$u(k-1)^T \Theta_{k-1} + x(k-1)^T \Psi_{k-1} = 0$$

por ser $\Theta_{N\text{-}1}$ una matriz simétrica, entonces, se tiene:

$$\Theta_{k-1} u(k-1) + \Psi_{k-1}^T x(k-1) = 0$$

de donde se llega a que:

$$u(k-1) = -\Theta_{k-1}^{-1} \Psi_{k-1}^T x(k-1)$$

Así, es importante notar que la matriz $\Theta_{k\text{-}1}$ posee inversa, ya que, por hipótesis $\Lambda_{k\text{-}1}$ es definida negativa y K_{k+1} es semidefinida negativa, por lo que, $\Theta_{k\text{-}1}$ es definida negativa y, por lo tanto, no es singular y posee matriz inversa. También, la condición es suficiente para la optimalidad global, ya que, la función objetivo es cóncava y el máximo es un máximo global.

Se puede poner:

$$u^*(k-1) = G_{k-1} x(k-1)$$

en donde:

$$G_{k-1} = -\Theta_{k-1}^{-1}\Psi_{k-1}^{T} = -\left[\Lambda_{k-1} + B_{k-1}^{T}K_{k}B_{k-1}\right]^{-1}B_{k-1}^{T}K_{k}A_{k-1}$$

entonces,

$$J_{k-1}^{*}\left\{x(k-1)\right\} = \frac{1}{2}x(k-1)^{T}\Phi_{k-1}x(k-1) + \frac{1}{2}x(k-1)^{T}G_{k-1}^{T}\Theta_{k-1}G_{k-1}x(k-1) + \\ + x(k-1)^{T}\Psi_{k-1}G_{k-1}x(k-1)$$

se puede poner:

$$J_{k-1}^{*}\left\{x(k-1)\right\} = \frac{1}{2}x(k-1)^{T}K_{k-1}x(k-1)$$

en donde:

$$K_{k-1} = \Phi_{k-1} + G_{k-1}^{T}\Theta_{k-1}G_{k-1} + 2\Psi_{k-1}G_{k-1} = \Phi_{k-1} - \Psi_{k-1}\Theta_{k-1}^{-1}\Psi_{k-1}^{T}$$

El sistema controlado es lineal y se comprueba sustituyendo en la ecuación de estado el control óptimo obtenido.

6.7. PROBLEMAS ESTOCÁSTICOS CON DINÁMICA LINEAL Y CRITERIOS ESTOCÁSTICOS

Inicialmente, seguimos considerando el modelo visto en la sección 6.1, solamente que, ahora, tiene una versión estocástica donde hay un término aleatorio aditivo en la dinámica lineal; este modelo se va a resolver con una ligera modificación de los métodos usados para resolver el anterior, por ejemplo, la política de control óptimo es la misma que la obtenida para el caso determinístico, esto se conoce como *equivalencia de certidumbre*. El modelo es particularmente apropiado para el caso de problemas de inventario estocástico visto en el capítulo 4, donde el inventario al iniciar la etapa $i + 1$ es el inventario al iniciar la etapa i (el estado), más la cantidad de material adquirido (la decisión), menos la demanda (una variable aleatoria). Posteriormente, se considera un modelo con criterios cuadráticos y una dinámica lineal muy general, donde todos los datos son aleatorios y se encuentra que los modelos del caso determinístico aún se pueden aplicar.

6.7.1. EQUIVALENCIA DE CERTIDUMBRE

Se selecciona una política de retroalimentación $y(0), y(1), y(2),\ldots,y(N-1)$ donde $y(1),\ldots,y(N-1)$ depende de $x(1),\ldots,x(N-1)$, respectivamente, y minimiza el valor esperado de J dado por:

$$J = \sum_{i=0}^{N-1}\left[a(i)x^2(i) + c(i)y^2(i)\right] + lx^2N \tag{6.26}$$

Dada por la regla, dinámicamente, estocástica siguiente:

$$x\,(i+1) = g(i)\,x\,(i) + h(i)\,y\,(i) + z\,(i),\ x(0) \text{ conocido} \tag{6.27}$$

Donde $z(i)$ es una variable aleatoria independiente que se selecciona en cada etapa *i,* que tiene una distribución con media $\bar{z}(i)$ y varianza $\sigma^2(i)$. Se supondrá que se selecciona $y\,(i)$ antes que la variable aleatoria $z(i)$ sea observada, en el ejemplo 6.6 que se verá más adelante, se hace una suposición diferente.

Se llega a la solución con PD al definir la función del valor esperado óptimo de la siguiente manera:

$Vi(x)$ = el costo mínimo esperado del proceso remanente, si se empieza en la etapa i en el estado x y aún se desconoce el valor actual de $z(i)$. (6.28)

De esta forma, por el principio de optimalidad donde Ez es el valor esperado o la esperanza con respecto a la variable aleatoria z, se tiene:

$$V_i(x) = \min_y \left\{ E_{z(i)}\left[a(i)x^2 + c(i)y^2 + V_{i+1}\left(g(i+1)x + h(i)y + z(i)\right)\right]\right\} \tag{6.29}$$

Esto significa que por cada posible elección de y dadas i y x, se calcula el costo esperado del proceso remanente con respecto a la variable aleatoria $z(i)$ donde el estado siguiente es una variable aleatoria, que depende de $z(i)$ y se selecciona y que minimice el costo esperado remanente. La condición de frontera es:

$$V_N(X) = lx^2 \tag{6.30}$$

Ejemplo 6.6. Se supone que en cada etapa i se selecciona $y(i)$, después, de que se ha enunciado el valor de la variable aleatoria $z(i)$. El problema sigue siendo estocástico, ya que, no conocemos los valores para $z(i+1), \ldots, z(N-1)$ cuando se pide seleccionar $y(i)$, como se hizo en el caso de la ruta más corta estocástica del capítulo 4.

Solución:

Se define $V_i(x)$ para este proceso como en (6.28):

$$V_i(x) = \underset{z(i)}{E} \min_{y(x,z(i))} \left\{ a(i)x^2 + c(i)y^2 + V_{i+1}\left[g(i)x + h(i)y + z(i) \right] \right\}$$

En este caso, se puede observar que el mínimo y la esperanza están invertidos y que y depende de $z(i)$. La condición de frontera se mantiene la misma.

De tal manera, usando (6.29) y (6.30) para determinar la función $V_{N-1}(x)$, de manera analítica, se tiene:

$$V_{N-1}(x) = \min_{y} \left\{ \underset{z(N-1)}{E} \left[a(N-1)x^2 + c(N-1)y^2 + V_N\left(g(n-1)x + h(N-1)y + z(N-1) \right) \right] \right\}$$

$$= \min_{y} \left\{ \underset{z(N-1)}{E} \left[a(N-1)x^2 + c(N-1)y^2 + l\left(g(N-1)x + h(N-1)y + z(N-1) \right)^2 \right] \right\}$$

$$= \min_{y} \left\{ \underset{z(N-1)}{E} \left[\begin{array}{l} a(N-1)x^2 + c(N-1)y^2 + l\left(g^2(N-1)x^2 + h^2(N-1)y^2 + z^2(N-1) \right)^2 + \\ 2g(N-1)h(N-1)xy + 2g(N-1)xz(N-1) + 2h(N-1)yz(N-1) \end{array} \right] \right\}$$

$$= \min_{y} \left\{ \left[\begin{array}{l} a(N-1)x^2 + c(N-1)y^2 + lg^2(N-1)x^2 + lh^2(N-1)y^2 + l(\bar{z}^2(N-1) + \sigma^2(N-1) \\ 2lg(N-1)h(N-1)xy + 2lg(N-1)x\bar{z}(N-1) + 2lh(N-1)y\bar{z}(N-1) \end{array} \right] \right\} \tag{6.31}$$

En el último paso, se ha tomado explícitamente el valor esperado y se ha usado el hecho de que z es una variable aleatoria como media $\bar{z}$ y varianza σ^2.

$$E(z^2) = \bar{z}^2 + \sigma^2 \tag{6.32}$$

Para demostrar que esta ecuación es válida, se nota que por la definición de varianza:

$$\sigma^2 = E\,[(z-\bar{z}\,)^2] \tag{6.33}$$

Por tanto, reconociendo que $\bar{z}$ es un número y no una variable aleatoria, se tiene:

$$\sigma^2 = E(z^2-2z\,\bar{z}+\bar{z}^{\,2}) = E(z^2) -2\,\bar{z}\,E(z)+\bar{z}^{\,2} = E(z^2) -2\,\bar{z}^{\,2}+\bar{z}^{\,2} = E(z^2) -\bar{z}^{\,2} \tag{6.34}$$

de esta manera, se obtiene (6.32).

La y que minimiza la expresión entre los corchetes de la expresión (6.31), se encuentra a través de igualar a cero la derivada:

$$0 = 2c\,(N-1)\,y + 2lh^2(N-1)y + 2lg(N-1)h(N-1)x + 2lh(N-1)\;\bar{z}\,(N-1)$$

$$y = \frac{lg(N-1)h(N-1)x + lh(N-1)\,z(\bar{N}-1)}{c(N-1)+lh^2(N-1)} \tag{6.35}$$

En esta última ecuación, se supone que el denominador que es la segunda derivada con respecto a y es positiva. Se sustituye en (6.31) y, después, de algo de álgebra se obtiene la siguiente expresión cuadrática para $V_{N\text{-}1}(x)$.

$$V_{N-1}(x) = \left[a(N-1) + lg^2(N-1) - \frac{l^2h^2(N-1)}{c(N-1)+lh^2(N-1)} \right] x^2 +$$

$$+2\left[lg(N-1)\,z(N-1) - \frac{2l^2h^2(\bar{N}-1)g(N-1)\,z(\bar{N}-1)}{c(N-1)+lh^2(N-1)} \right] x$$

$$+\left[l\,\bar{z}^{2}(N-1) + l\sigma^2(N-1) - \frac{l^2h^2(N-1)\,z(\bar{N}^{2}-1)}{c(N-1)+lh^2(N-1)} \right] \tag{6.36}$$

De esta manera, la forma cuadrática lx^2 en la etapa N se convierte en la función cuadrática general de la forma $p(N-1)x^2 + q(N-1)\,x + r\,(N-1)$ en la etapa $N-1$. Se podría calcular $V_{N-2}\,(x)$ usando (6.29) con $i = N-2$ y el resultado de (6.36), si lo hiciéramos, encontraríamos que, también, es una forma cuadrática y lo que procede es una prueba por inducción misma que no desarrollamos en estos apuntes.

Lo anterior constituye la solución para el problema con término aleatorio aditivo en dinámica. Existe una conexión interesante entre esta solución y cierto problema cuadrático determinístico, donde se hace $b(i) = d(i) = e(i) = f(i) = 0$. Más aún, si se reemplaza $k(i)$ por $\bar{z}\,(i)$ se encuentran las fórmulas para $p(i)$, $q(i)$, $y(i)$ en términos de p y q.

6.8. CONCLUSIONES

Muchos problemas no lineales se pueden reducir a casos cuadráticos, lo cual reduce mucho el tiempo de solución. Las aplicaciones de estos modelos se usan mucho en la teoría de control, casos de optimización de recursos, producción y portafolios de inversión entre otros.

Se tienen modelos tanto determinísticos como estocásticos y se cuenta con una variedad de métodos de solución como son los algoritmos de punto interior, lagrangeanos, gradientes y algunas extensiones del método simplex. En los casos donde se puede dividir el problema en subproblemas o etapas, se hace uso de la PD reduciendo así, notablemente, los cálculos. Sin embargo, cabe aclarar que para problemas de tamaño mediano o grande no es aconsejable el uso de estos modelos.

6.9. NOTAS HISTÓRICAS

La **teoría moderna del portafolio** o **teoría moderna de selección de cartera** es una teoría de inversión que estudia como maximizar el retorno y minimizar el riesgo, mediante una adecuada elección de los componentes de una cartera de valores. Originada por Harry Markowitz, autor de un artículo sobre selección de cartera publicado en 1952, la teoría moderna de la selección de cartera (modern portfolio theory) propone que el inversor debe abordar la cartera como un todo, estudiando las características de riesgo y retorno global, en lugar de escoger valores individuales en virtud del retorno esperado de cada valor en particular.

La teoría de selección de cartera toma en consideración el retorno esperado a largo plazo y la volatilidad esperada en el corto plazo.

La volatilidad se trata como un factor de riesgo, y la cartera se conforma en virtud de la tolerancia al riesgo de cada inversor en particular, tras evaluar el máximo nivel de retorno disponible para el nivel de riesgo escogido.

En su modelo, Markowitz, establece que los inversionistas tienen una conducta racional a la hora de seleccionar su cartera de inversión y por lo tanto siempre buscan obtener la máxima rentabilidad sin tener que asumir un alto nivel de riesgo. Nos muestra también, como hacer una cartera óptima disminuyendo el riesgo de manera que el rendimiento no se vea afectado.

Para poder integrar una cartera de inversión equilibrada lo más importante es la diversificación ya que de esta forma se reduce la variación de los precios. La idea de la cartera es, entonces, diversificar las inversiones en diferentes mercados y plazos para así disminuir las fluctuaciones en la rentabilidad total de la cartera y por lo tanto también del riesgo.[32]

Harry Max Markowitz, (Chicago, 1927 -) recibió el premio Nobel de Economía en 1990. Markowitz es uno de los numerosos economistas laureados con el Nobel de Economía durante el siglo XX producidos por la prestigiosa Escuela de Economía de Chicago de la Universidad de Chicago.

Aunque el tema de las finanzas empresariales ya había sido tratado por otros dos laureados, James Tobin (1981) y Franco Modigliani (1985), el Nobel de 1990 sorprendió a los economistas estudiosos de la teoría económica *pura*. Aunque Markowitz, Miller y Sharpe —los tres premiados— eran ampliamente respetados en los ambientes académicos, no se esperaba que su trabajo —notoriamente puntual, dada la amplitud de la ciencia económica— fuese premiado; esto resulta evidente si se analizan los trabajos de los economistas previamente galardonados.

Habría una explicación concreta para que el premio haya sido concedido a la economía financiera (son ex presidentes de la Sociedad Estadounidense de Finanzas): el hecho de que los servicios financieros sean actualmente muy importantes en los países desarrollados y que los mercados bursátiles cubran amplios segmentos de ahorristas, lo que es indudablemente un fenómeno nuevo en la década de los años 1980.[33]

[32] "Teoría moderna del portafolio", 2014. Recuperado de: http://es.wikipedia.org/wiki/Teor%C3%ADa_moderna_del_portafolio
[33] "Harry Markowitz", 2015. Recuperado de: http://es.wikipedia.org/wiki/Harry_Markowitz

6.10. EJERCICIOS PROPUESTOS

1) Para el caso del ejemplo 6.3, suponga que $x(0)$ no se especifica, así como, la secuencia de las decisiones se seleccionan para minimizar el criterio. ¿Cómo usaría la solución del ejemplo para resolver este problema?

2) Considere el problema siguiente:

Escoja $y(0)$, $y(1)$ y $y(2)$ que minimizan J, dado por:

$$J = y^2(0) + x_1{}^2(1) + x_2{}^2(1) + y^2(1) + x_1{}^2(2) + x_2{}^2(2) + y^2(2)$$

sujeta a

$$x_1(0) = 7 \qquad x_2(0) = 1$$
$$x_1(3) = 2 \qquad x_2(3) = 1$$

donde:

$$x_1(i+1) = x_1(i) + x_2(i) + 2y(i)$$
$$x_2(i+1) = x_1(i) + x_2(i) + y(i)$$

Note que la z, como se define, está por encima de 2 para este ejemplo. Determine $V_1(x_1, x_2)$ usando el hecho de que las condiciones terminales determinan $y(1)$ y $y(2)$ en términos de $x_1(1)$ y $x_2(1)$. Use las fórmulas matriciales para encontrar $y(0)$ y V_0 (7,1).

3) Desarrolle las fórmulas recursivas para el ejemplo 6.6. ¿Se cumple la equivalencia de certidumbre?

ANEXOS

ANEXO A

FRACTALES Y RECURSIVIDAD[34]

A.1. INTRODUCCIÓN

En los últimos años, las matemáticas y las ciencias naturales han tenido una conexión a través del uso de *fractales* para explicar muchos fenómenos en diferentes áreas y niveles; por esta razón, nunca antes, los conocimientos matemáticos generaron tanta emoción en la mente del público. Los fractales han capturado la atención, literalmente, el entusiasmo y el interés de un público mundial; para el observador casual, el color de sus estructuras esenciales, su belleza y forma geométrica han cautivado los sentidos visuales como algunas otras cosas que nunca se han experimentado en las matemáticas. Para el científico de la computación, los fractales ofrecen un ambiente rico para explorar, crear y construir un nuevo mundo visual, y como artista, la configuración de una obra fresca.

Pero, ¿cuáles son las razones de esta fascinación? En primer lugar, el campo de investigación de los fractales ha creado tal poder y singularidad que una colección de ellos ha demostrado ser una de las series de más éxito en todo el mundo de las exposiciones, "Solamente en el Museo de Ciencia de Londres las exposiciones de las *Fronteras del Caos: Imágenes de Sistemas Dinámicos Complejos* por H. Jürgens, H. O. Peitgen, M. Prüfer, P. H. Richter y D. Saupe atrajo a más de 140,000 visitantes. Desde 1985 esta exposición ha viajado a más de 100 ciudades en más de 30 países en todos los continentes."[35] Sin embargo, más importante es el hecho de que la teoría del caos y la geometría fractal han cambiado la visión de la ciencia y la concepción del mundo.

---

[34] Shiffman, Daniel, "8. Fractals", *The Nature of Code*, 2012. Recuperado de: http://natureofcode.com/book/chapter-8-fractals/

[35] Peitgen, Heinz-Otto, *et al*., *Fractals for the Classroom*, New York, Springer Science & Business Media LLC, 1992.

Cuando se habla de geometría, se piensa en la geometría euclideana que debe su nombre al matemático griego Euclides, pero, cabe preguntarse: ¿se puede explicar el mundo a través de esta geometría de líneas, círculos y figuras perfectas? Ahora mismo, la pantalla de LCD (Liquid Crystal Display) que estoy mirando, seguro se parece a un rectángulo; y la ciruela que comí esta mañana es circular; pero, ¿si yo mirara más allá y considerara a los árboles que bordean la calle, a las hojas que cuelgan de los árboles, al relámpago de la última noche de tormenta, a la coliflor que comí en la cena, a los vasos sanguíneos de mi cuerpo, a las montañas y a las líneas costeras que cubren la tierra más allá de la ciudad de Nueva York? La mayor parte del material que se encuentra en la naturaleza no puede ser descrito por las formas geométricas idealizadas por la geometría euclidiana. Así que, si queremos empezar a construir diseños computacionales con los patrones más allá de la simple línea recta, es el momento de aprender acerca de los conceptos que hay detrás y simularlos con la ayuda de los fractales.

A.2. FRACTALES Y RECURSIVIDAD

Pero ¿cómo se define un fractal? En el capítulo 1 de estos apuntes, ya se ha descrito un fractal, retomaremos este tema y veremos cómo se conecta con la teoría de caos.

> El término fractal (del fractus latín, que significa "roto") fue acuñado por el matemático Benoit Mandelbrot en 1975. En su obra fundamental "La geometría fractal de la naturaleza", que define un fractal como "una forma geométrica áspera o fragmentada que puede ser dividido en partes, cada una de las cuales es (al menos aproximadamente) una copia del tamaño reducido de la totalidad.[36]

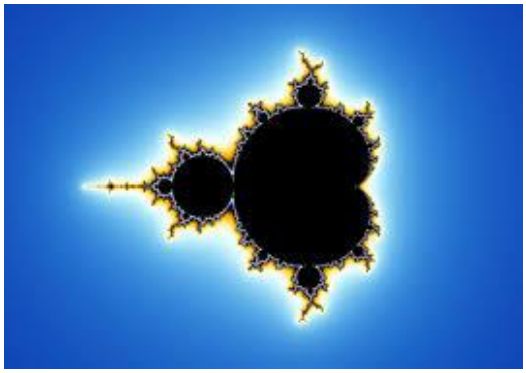


FIGURA A.1. Conjunto de Mandelbrot $z_{n+1} = z_n^2 + C$

[36] *Ídem,* trad. Idalia Flores de la Mota.

La definición de *fractal* se puede explicar con dos ejemplos sencillos. Para empezar, considere la estructura de un árbol como en la figura siguiente.

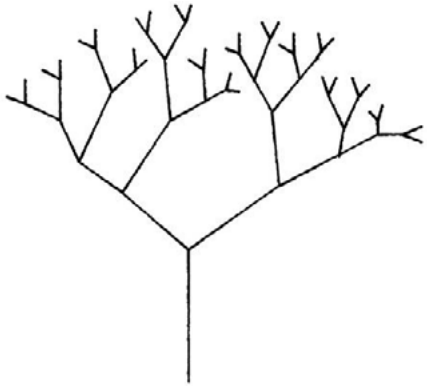


FIGURA A.2. Estructura de un árbol

En la figura A.2, observe cómo el árbol tiene una sola raíz con dos ramas conectadas en su extremo. Cada una de esas ramas tiene dos ramas en su extremo y las ramas tienen dos ramas y así, sucesivamente. ¿Qué pasaría, si tuviéramos que arrancar una rama del árbol y examinarla en sí misma?

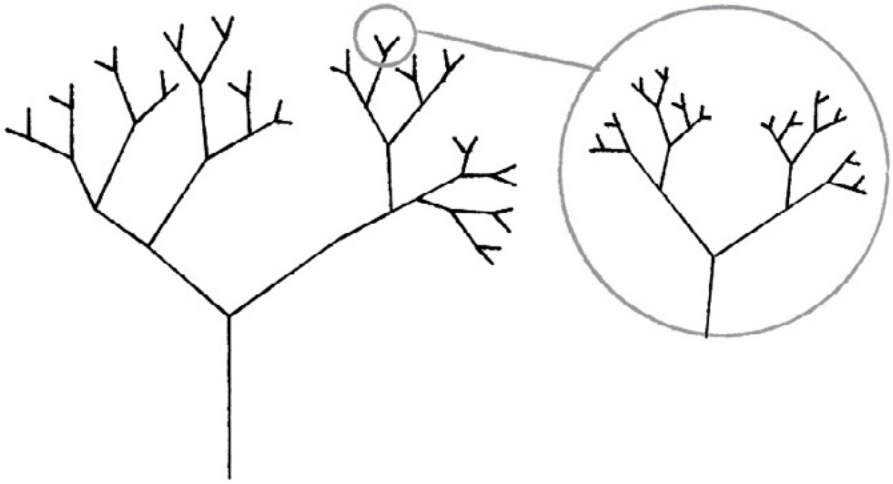


FIGURA A.3. Detalle del árbol

Si miramos de cerca una determinada sección del árbol, nos encontramos con que la forma de esta rama se parece al árbol mismo. Esto se conoce como *autosimilitud* como Mandelbrot declaró: cada parte es una "copia de tamaño reducido de la totalidad".

El árbol de arriba es, perfectamente, simétrico y las partes son, de hecho, réplicas exactas de la totalidad. Sin embargo, los fractales no tienen que ser perfectamente autosimilares. Echemos un vistazo a un gráfico del mercado de valores (adaptado desde el almacén de datos reales de Apple).

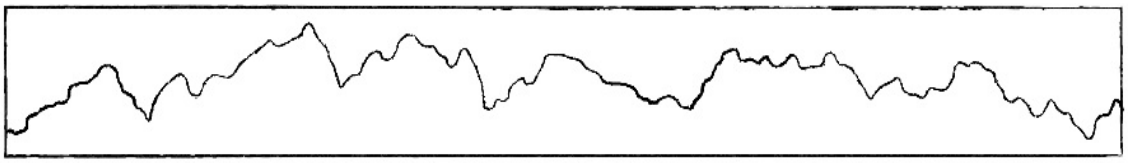


FIGURA A.4. Gráfica A del mercado de valores

Y otra más:

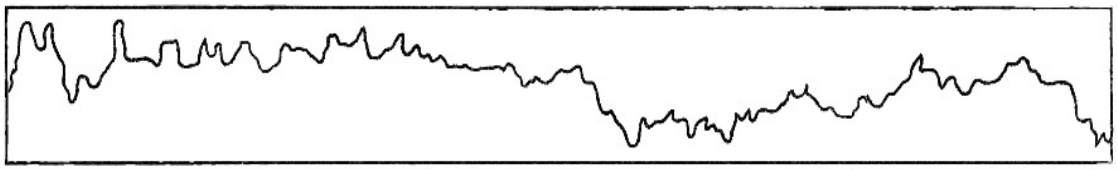


FIGURA A.5. Gráfica B del mercado de valores

En estas gráficas, el eje *x* es el tiempo y el eje *y* es el valor de la acción. No es un accidente que se hayan omitido las etiquetas, sin embargo, las gráficas de los datos del mercado de valores son ejemplos de fractales, porque tienen el mismo aspecto en cualquier escala. ¿Son estas las gráficas de las acciones de más de un año? ¿Un día? ¿Una hora? No hay manera de saberlo sin una etiqueta. (Por cierto, la gráfica A muestra seis meses de datos y la gráfica B se acerca a una parte pequeña de una gráfica, mostrando seis horas).

Las gráficas son ejemplos de fractales estocásticos, lo cual significa que se construyen fuera de las probabilidades y el azar. A diferencia de la estructura de la ramificación de un árbol determinista que es, estadísticamente, autosimilar.

Mientras que la autosimilitud es un rasgo clave de los fractales, es importante darse cuenta de que la autosimilitud, en sí misma, no hace un fractal. Después de todo, una línea es autosimilar; una línea tiene el mismo aspecto en cualquier escala y puede considerarse que comprende a un montón de líneas pequeñas, pero, no es un fractal. Los fractales se caracterizan por tener una estructura fina en pequeñas escalas (al mantener el *zoom* en la gráfica de la bolsa, se continuará encontrando fluctuaciones) y no pueden describirse con la geometría euclidiana. Si usted puede decir "¡es una línea!", entonces, no es un fractal.

Otro componente fundamental de la geometría fractal es la recursividad. Todos los fractales tienen una definición recursiva. Vamos a empezar con la repetición antes de desarrollar las técnicas y ejemplos del código para la construcción de patrones de fractales en procesamiento.

A.2.1. RECURSIVIDAD

Para iniciar, comencemos nuestra discusión de la recursividad mediante el examen de la primera aparición de los fractales en las matemáticas modernas. En 1883, el matemático alemán George Cantor desarrolló reglas simples para generar un conjunto infinito como se muestra en la figura A.6.

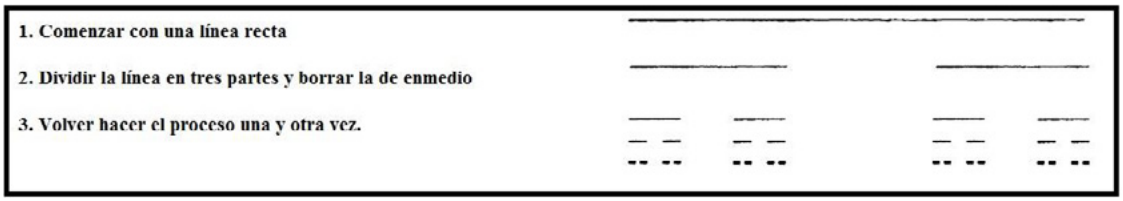


FIGURA A.6. Reglas simples de Cantor para generar un conjunto infinito

En este trabajo hay un circuito de retroalimentación: tomar una sola línea y dividirla en dos, luego, regresamos a esas dos líneas y aplicamos la misma regla, rompiendo cada línea en dos y, ahora, nos quedamos con cuatro. A continuación, regrese a esas cuatro líneas y aplique la regla, ahora tiene ocho. Este proceso se conoce como *recursión* que es la aplicación repetida de una regla para resultados sucesivos. Cantor estaba interesado en lo que sucede cuando se aplican estas reglas un número infinito de veces. Sin embargo, nosotros estamos trabajando en un espacio finito de píxeles y podemos ignorar la mayoría de las preguntas y paradojas que surgen de la recursividad infinita.

Ejemplo A.1. Función factorial.

Las funciones que se llaman *a sí mismas* son recursivas y buenas para resolver ciertos problemas. Por ejemplo, ciertos cálculos matemáticos se implementan de forma recursiva; el ejemplo más común es la factorial como se vio en el capítulo 1 de estos apuntes.

$$0! = 1$$

Para $n > 0$: $n! = n(n-1)!$

En seudocódigo es como sigue:

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

Pero, revisando la definición de factorial, podemos notar lo siguiente:
4! = 4 * 3 * 2 * 1
3! = 3 * 2 * 1

Y, entonces:
4! = 4 * 3!

En general, para cualquier número *n* positivo:
$n! = n * (n - 1)!$
1! = 1

Lo que nos lleva a lo siguiente, el factorial de *n* se define como *n* veces el factorial de $n - 1$. ¿La definición de factorial incluye factorial? Es algo así, como la definición de *cansado* y decir que es *la sensación que tienes cuando estás cansado*. Este concepto de autorreferencia en funciones es un ejemplo de recursividad y podemos utilizarlo para escribir una función factorial que se llama a sí misma.

```
int factorial(int n) {
if (n == 1) {
return 1;
} else {
return n * factorial(n-1);
}
}
```

Esto suena a locura, pero, funciona. En la figura siguiente vemos lo que sucede cuando factorial de 4 se llama a sí mismo.

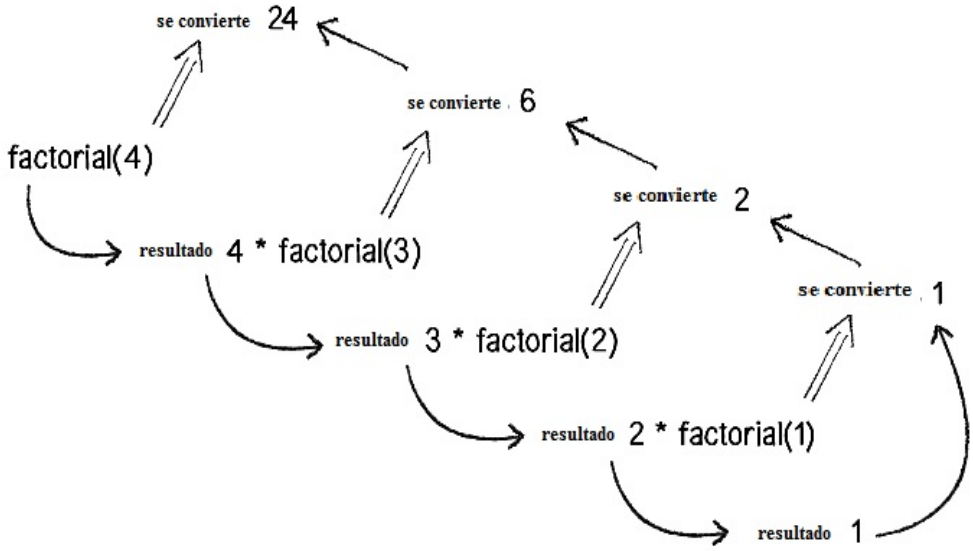


FIGURA A.7. Esquema de factorial de 4

Ejemplo A.2. Conjunto de Cantor.

Entonces, regresamos a analizar el conjunto de Cantor como una función recursiva. Se comienza con una línea y se muestra el seudocódigo para esta línea:

```
void cantor(float x, float y, float len) {
   line(x,y,x+len,y);
}
```

La función de Cantor dibuja una línea que comienza en el píxel de coordenadas (x, y) con una longitud "len". (La línea se dibuja horizontalmente aquí, pero, esto es una decisión arbitraria). Así que, si llamamos a esa función, diciendo:

```
cantor(10, 20, width-20);
```

Se obtiene lo siguiente:

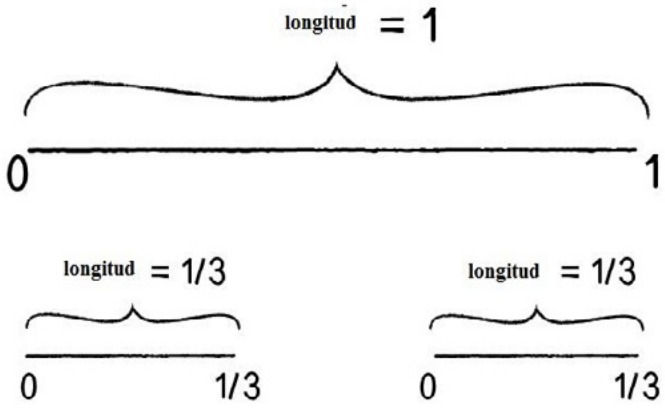


FIGURA A.8. Instrucciones para el conjunto de Cantor

Ahora, la regla de Cantor nos dice que debemos borrar el tercio medio de la línea, lo que nos deja con dos líneas, una desde el principio de la línea hasta la marca de una tercera parte y otra, de la marca de dos tercios hasta el final de la línea. Podemos añadir dos líneas más de código para dibujar el segundo par de líneas, mover la ubicación y abajo, un montón de píxeles para que podamos ver el resultado debajo de la línea original.

```
void cantor(float x, float y, float len) {
  line(x,y,x+len,y);

  y += 20;
From start to 1/3rd
  line(x,y,x+len/3,y);
From 2/3rd to end
  line(x+len*2/3,y,x+len,y);
}
```

Y, gráficamente, lo vemos en la figura A.9.

FIGURA A.9. Gráfica que muestra la línea dividida

Si bien, este es un buen comienzo, un enfoque de este tipo al llamar manualmente a la línea, para cada línea, no es lo que queremos. Será poco manejable, muy rápido y necesitaríamos cuatro, luego ocho y luego dieciséis llamadas "to line". Si un ciclo "for" es nuestra forma habitual para resolver estos problemas, pero, matemáticamente por cada iteración se demuestra, en forma rápida, que resulta excesivamente complicado, aquí, es donde viene la recursividad y nos rescata.

Ahora, es importante revisar lo que hicimos en la primera línea del código:

```
line(x,y,x+len/3,y);
```

En lugar de llamar a la función de línea, directamente, podemos llamar a la función Cantor en sí. Después de todo, ¿qué hace la función de Cantor? Se dibuja una línea en (*x*, *y*) con una longitud dada y, entonces:

```
line(x,y,x+len/3,y);        becomes ------->  cantor(x,y,len/3);
```

Y para la segunda línea:

```
line(x+len*2/3,y,x+len,y);   becomes ------->  cantor(x+len*2/3,y,len/3);
```

Y nos queda:

```
void cantor(float x, float y, float len) {
  line(x,y,x+len,y);

  y += 20;

  cantor(x,y,len/3);
  cantor(x+len*2/3,y,len/3);
}
```

Y puesto que la función de Cantor se llama a sí misma de forma recursiva, la misma regla se aplicará a las siguientes líneas y a la siguiente y a la siguiente como Cantor, una y otra vez. Este código solo requiere una condición de salida para poder ejecutarlo, ya que, la longitud de la línea es cada vez menor.

Ejemplo A.3. Los números de Fibonacci y el problema de los conejos.

Leonardo Pisano, también conocido como Fibonacci (por *filius Bonacci*, hijo de Bonacci) fue una de las figuras destacadas en las matemáticas occidentales medievales. Viajó ampliamente en el mundo mediterráneo antes de establecerse en su Pisa natal. En 1202, publicó su libro *Liber abaci* con las cifras árabe indias 0, 1, 2, ... También, su libro contenía el siguiente problema que ha inspirado desde entonces. Hay un par de conejos que han nacido en el tiempo 0; después de un mes, el par, ya maduros, se aparean y un mes más tarde, la pareja da a luz a una nueva pareja de conejos y continúan haciéndolo (es decir, todos los meses un nueva pareja nace de la pareja original). Por ejemplo:

TABLA A.1. Crecimiento de la población de conejos por mes

Tiempo	**0**	**1**	**2**	**3**	**4**
Conejos	**1**	**1**	**1**	**1**	**1**
			1	1	1
				1	1
					1
					1

Sean J_n y A_n el número de pares de jóvenes y adultos después de n meses. Inicialmente, en el tiempo $n = 0$ solo hay un par de jóvenes ($J_0 = 1$, $A_0 = 0$). Después de un mes, la joven pareja se ha convertido en adultos de un mes ($J_1 = 0$, $A_1 = 1$); después de dos meses, la pareja de adultos da a luz a un par joven ($J_2 = 1$, $A_2 = 1$) y otra vez, después del próximo mes. Cabe agregar que la joven pareja se convierte en un adulto ($J_3 = 1$, $A_3 = 2$).

Por supuesto, la regla general es que el número de pares de recién nacidos $J_n + 1$ es igual a la población adulta anterior A_n. La población adulta crece por el número de pares de inmaduros J_n, respecto al mes anterior; de este modo, las dos fórmulas siguientes describen completamente la dinámica poblacional.

$$J_{n+1} = A_n$$
$$A_{n+1} = A_n + J_n$$

Como valores iniciales, tomamos $J_0 = 1$ y $A_0 = 0$. A partir de la primera de las ecuaciones anteriores, se deduce que $J_n = A_{n-1}$ e insertando esto en la segunda ecuación obtenemos:

$$A_{n+1} = A_n + A_{n-1}$$

Con $A_0 = 0$ y $A_1 = 1$ son una sola ecuación para la población total de conejos y usando esta ecuación se calcula, fácilmente, el número de pares en generaciones sucesivas:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233,...

Cada número de esta sucesión es la suma de sus dos predecesores, esta sucesión recibe el nombre de *sucesión de Fibonacci*.

Al parecer, la sucesión de Fibonacci puede crecer más allá de todos los límites. Nuestros conejos presentan una especie de población que pasa de generación en generación. Para ello, nos fijamos de nuevo en los números de Fibonacci y calculamos las proporciones de las sucesivas generaciones (redondeado a seis decimales).

TABLA A.2. Lista de las proporciones sucesivas

n	A_n	A_{n+1}/A_n	**En decimales**
0	1	1/1	1.0
1	1	2/1	2.0
2	2	3/2	1.5
3	3	5/3	1.6666
4	5	8/5	1.6
5	8	13/8	1.625
6	13	21/13	1.61538
7	21	34/21	1.61904
8	34	55/34	1.61764
9	55	89/55	1.61818
10	89	144/89	1.61797
11	144	233/144	1.61805
12	233	377/233	1.61702

En apariencia, nos estamos aproximando, si no rápidamente, sí en forma consistente a un número en particular, este misterioso número es:

1.618033988749894848820

¿Han visto este número con anterioridad?

$$1.61803398... = \frac{1+\sqrt{5}}{2}$$

Pues, es la famosa proporción áurea o *divina proportione* (proporción divina) como lo llamaban en la Edad Media. Este número ha inspirado a los matemáticos, astrónomos y filósofos como ningún otro número en la historia de las matemáticas.

A.3. TEORÍA DEL CAOS[37]

Cuando examinamos el desarrollo de un proceso a lo largo de un periodo de tiempo, se habla en términos que se utilizan en la teoría del caos. Cuando estamos más interesados en las formas estructurales que deja un proceso caótico a su paso, entonces, se utiliza la terminología de la geometría fractal que es, realmente, la geometría, cuyas estructuras son las que dan orden al caos.

La correlación del caos y la geometría es cualquier cosa menos casual. Más bien, es el testimonio de su profundo parentesco; parentesco que se puede ver mejor en el *conjunto de Mandelbrot*, un objeto matemático descubierto por Benoît Mandelbrot en 1980. El conjunto de Mandelbrot ha sido descrito por algunos científicos como el más complejo y, posiblemente, el objeto más precioso, nunca visto en las matemáticas. Sin embargo, la característica más fascinante del conjunto de Mandelbrot apenas ha sido recientemente descubierta, la cual se puede interpretar como una enciclopedia ilustrada de un número infinito de algoritmos. Este objeto está fantásticamente organizado como un almacén de imágenes eficientes y es el ejemplo por excelencia del orden en el caos.

El caos es la ciencia de las sorpresas, de la no linealidad y lo impredecible; nos enseña a esperar lo inesperado. Mientras que la mayoría de la ciencia tradicional trata con fenómenos supuestamente predecibles como la gravedad, la electricidad o las reacciones químicas, la *teoría del caos* se ocupa de las cosas no lineales que son efectivamente imposibles de predecir o controlar, como la turbulencia, el tiempo, el mercado de valores, nuestros estados cerebrales y, así, sucesivamente. Estos fenómenos se describen a menudo por las matemáticas fractales que captura la complejidad infinita de la naturaleza. Muchos objetos naturales exhiben propiedades fractales, incluyendo los paisajes, nubes, árboles, órganos, ríos, etc., y muchos de los sistemas en los que vivimos exhiben un comportamiento caótico complejo. Reconociendo la naturaleza caótica fractal de nuestro mundo, podemos tener una nueva visión: el poder y la sabiduría. Por ejemplo, mediante la comprensión de la dinámica compleja caótica de la atmósfera, un piloto de globo puede *dirigir* un globo a una ubicación deseada. Al comprender que nuestros ecosistemas, nuestros sistemas sociales y nuestros sistemas económicos están interconectados, lo que

[37] Fractal Foundation, "What is Chaos Theory?". Recuperado de: http://fractalfoundation.org/resources/what-is-chaos-theory/

podemos hacer es entenderlos para evitar acciones que puedan llegar a ser perjudiciales para nuestro bienestar a largo plazo, como lo ha sido el cambio climático.

Henri Poincaré, eminente científico francés, estaba al tanto de este fenómeno cuando escribió respecto a su estudio del movimiento de los planetas en 1903: "…puede suceder que pequeñas diferencias en las condiciones iniciales den lugar a otras muy grandes en los fenómenos finales…". Un pequeño error en el primero producirá un error enorme en el segundo. La predicción se vuelve imposible.

A.3.1. PRINCIPIOS DEL CAOS

El efecto mariposa: Este efecto otorga el poder de causar un huracán en China, si una mariposa bate sus alas en Nuevo México. Se puede tomar un tiempo muy largo, pero la conexión es real. Si la mariposa no hubiera batido sus alas justo en el punto correcto en el espacio/tiempo, el huracán no habría sucedido. Una forma más rigurosa de expresar esto es que los pequeños cambios en las condiciones iniciales conducen a cambios drásticos en los resultados. Nuestras vidas son una demostración permanente de este principio. ¿Quién sabe lo que serán los efectos a largo plazo en la enseñanza de millones de niños acerca del caos y los fractales?

La imprevisibilidad: Porque nunca podemos conocer todas las condiciones iniciales de un sistema complejo en suficiente detalle (es decir, perfecto), no podemos esperar predecir el destino final de un sistema complejo. Incluso, leves errores en la medición del estado de un sistema se amplificarán de forma espectacular, lo que hace inútil cualquier predicción. Dado que es imposible medir los efectos de todas las mariposas en el mundo, entonces, la predicción meteorológica precisa de largo alcance y siempre seguirá siendo imposible.

Orden/desorden: El caos no es simplemente el desorden. El caos explora las transiciones entre orden y desorden, que a menudo se producen de manera sorprendente.

Mezcla: La turbulencia asegura que dos puntos adyacentes en un sistema complejo eventualmente terminan en posiciones muy diferentes después de transcurrido algún tiempo. Ejemplos: Dos moléculas de agua vecinas pueden terminar en diferentes partes del océano o incluso en diferentes océanos. Un grupo de globos de helio que se lanzan juntos, finalmente, aterrizarán en lugares completamente diferentes. La mezcla es a fondo porque la turbulencia se produce en todas las escalas. También son no lineal: los fluidos que no pueden estar sin mezclar.

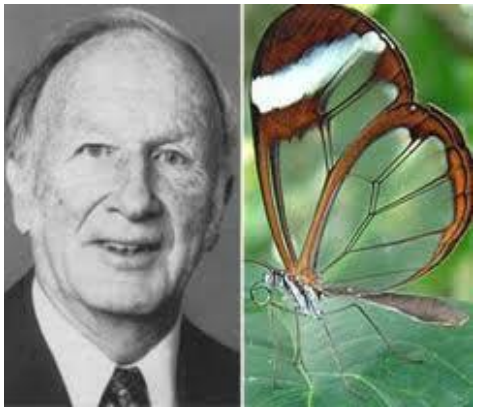
Retroalimentación: Los sistemas a menudo se convierten en caóticos cuando hay retroalimentación presente. Un buen ejemplo es el comportamiento del mercado de valores. A medida que el valor de una acción sube o baja, la gente tiende a comprarla o venderla. Esto a su vez afecta aún más el precio de las acciones, haciendo que suba o baje de forma caótica.

Fractales: Un fractal es un patrón sin fin. Los fractales son patrones infinitamente complejos y son auto-similares a través de diferentes escalas. Se crean mediante la repetición de un proceso simple una y otra vez en un bucle de retroalimentación en curso. Impulsados por la recursividad, los fractales son imágenes de los sistemas dinámicos - las imágenes del Caos. Geométricamente existen entre nuestras dimensiones conocidas, son muy familiares, ya que la naturaleza está llena de fractales. Por ejemplo: los árboles, ríos, costas, montañas, nubes, conchas de mar, huracanes, etc.[38]

A.3.2. ¿CÓMO COMENZÓ LA TEORÍA DEL CAOS?

Todo comenzó en 1960, cuando un hombre llamado Edward Lorenz creó un modelo del clima en computadora en el Instituto de Tecnología de Massachusetts. El modelo del clima de Lorenz consistió en una amplia gama de fórmulas complejas que se llenaban de números para analizar el clima. Sus colegas y estudiantes se maravillaron con el modelo, ya que nunca parecía repetir una secuencia; era absolutamente como el tiempo real. Algunos incluso esperaban que Lorenz por fin hubiera construido el último predictor del clima, si los parámetros de entrada fueran elegidos idénticos a los del tiempo real.

Pero, un día Lorenz decidió engañar un poco. Un rato antes había dejado el programa con determinados parámetros para generar un patrón determinado de tiempo y quería tener una mejor vista de los resultados.



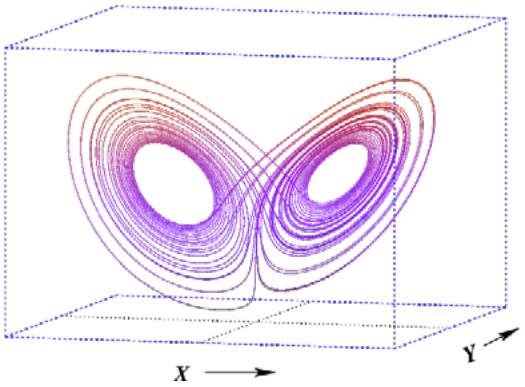
Pero, en lugar de dejar el programa con la configuración inicial y calcular el resultado, Lorenz decidió comenzar a mitad del camino con los valores de la última corrida. La computadora con la que Lorenz estaba trabajando usaba los parámetros calculados con una precisión de seis decimales. Pero, con los cambios de inicio que Lorenz hizo, dio como resultado números con una precisión de tres decimales. Así que, en lugar de introducir ciertos números (como el viento, la temperatura y cosas por el estilo) tan precisos como el equipo los tenía, Lorenz se conformó con aproximaciones; 5.123456 se convirtieron en 5,123 (por ejemplo). Y esa pequeña e insignificante inexactitud parecía amplificarse y hacer que todo el sistema estuviera fuera de control.

Exactamente, ¿por qué es importante todo esto? Pues bien, en el caso de los sistemas meteorológicos es muy importante. El tiempo es el comportamiento total de todas las moléculas

[38] *Ídem*, trad. Idalia Flores de la Mota.

que componen la atmósfera de la tierra. Y una diminuta partícula no puede ser exactamente acentuada, debido al principio de incertidumbre. Y esta es la única razón por la que los pronósticos del tiempo comienzan a ser falsos en torno a un día o dos en el futuro. No podemos obtener una solución exacta sobre la situación actual, solo una mera aproximación, por lo que nuestras ideas sobre el clima están condenadas a caer en mala alineación en cuestión de horas y completamente en las nebulosas de la fantasía en cuestión de días. La naturaleza no se deja predecir.

El Principio de Incertidumbre prohíbe exactitud. Por lo tanto, la situación inicial de un sistema complejo no se puede determinar con precisión y la evolución de un sistema complejo, por lo tanto, no puede predecirse con precisión.



Atractores

Los sistemas complejos a menudo parecen demasiado caóticos para reconocer un patrón a simple vista. Pero, mediante el uso de ciertas técnicas, grandes conjuntos de parámetros se pueden abreviar en un punto, en una gráfica. En la pequeña gráfica de la lluvia o el sol, cada punto representa una condición completa con la velocidad del viento, caída de la lluvia, la temperatura del aire, etc., pero mediante el procesamiento de estos números, en cierta forma, se pueden representar por un punto.

Los primeros teóricos del caos descubrieron que los sistemas complejos a menudo parecen correr a través de algún tipo de ciclo, a pesar de que las situaciones rara vez son exactamente duplicadas y repetidas. El trazo de muchos sistemas en gráficas simples reveló que a menudo parece que hay algún tipo de situación que el sistema intenta lograr, un equilibrio de algún tipo. Por ejemplo: imagine una ciudad de 10.000 personas, con el fin de dar cabida a estas personas, la ciudad va a generar un supermercado, dos piscinas, una biblioteca y tres iglesias. Y vamos a suponer que esta configuración agrada a todo el mundo y se logra un equilibrio. Pero luego, una empresa decide abrir una planta de helados, en las afueras de la ciudad, que es la apertura de puestos de trabajo para 10.000 personas más. La ciudad se expande rápidamente para dar cabida a 20.000 personas; se añade un supermercado, dos piscinas, una biblioteca, tres iglesias y el equilibrio se mantiene. En este caso, el equilibrio se llama un *atractor*.

Atractor de Edward Lorenz:
Ahora, imaginemos que en lugar de añadir 10.000 personas a el original 10.000, 3.000 personas dejan la ciudad y 7.000 permanecen. Los jefes de la cadena de supermercados calculan que un supermercado solo puede existir cuando tiene 8.000 clientes regulares. Así que, después de un tiempo la tienda cierra y la gente de la ciudad se queda sin víveres. La demanda sube y alguna otra empresa decide construir un supermercado con la esperanza de que un nuevo supermercado atraiga a gente nueva y lo hace. Pero, muchos ya estaban en el proceso de mudanza y un nuevo supermercado no va a cambiar sus planes.

La compañía mantiene la tienda funcionando por un año y luego se llega a la conclusión de que no hay suficientes clientes y cierra de nuevo. Las personas se alejan. La demanda aumenta. Alguien abre un supermercado. La gente se muda, pero no lo suficiente, se cae en un círculo vicioso.

Esta terrible situación es también una especie de equilibrio, pero dinámico. Un equilibrio de tipo dinámico se llama un *atractor extraño*. La diferencia entre un atractor y un atractor extraño es que un atractor representa un estado en que un sistema finalmente se instala, mientras que uno extraño representa algún tipo de trayectoria en la cual un sistema funciona de una situación a otra sin tener que asentarse.

El descubrimiento de los atractores fue emocionante y explicó muchas cosas, pero el fenómeno más impresionante que la teoría del caos descubrió, fue una pequeña cosa llamada auto-similitud. El desvelamiento de la auto-similitud permitió a la gente una idea de los mecanismos mágicos que dan forma a nuestro mundo y, tal vez, incluso a nosotros mismos.

Veamos un sencillo ejemplo: Un copo de nieve es un objeto compuesto por moléculas de agua. Estas moléculas no tienen un sistema nervioso común, ADN o una molécula principal que lleva la voz cantante. ¿Cómo saben estas moléculas a dónde ir y pasar el rato con el fin de formar una estrella de seis puntas? Y ¿de dónde sacan la audacia para formar una diferente cada vez? ¿De qué manera una molécula en una parte del copo sabe cuál es el diseño a seguir en el resto del copo, cuando está en otras partes de un copo a una distancia de miles de kilómetros?

¿Cómo funcionan las moléculas en la naturaleza en forma de copos de nieve o cristales o cualquier otra forma regular? La teoría del caos tiene una respuesta: *Auto Similitud*, un principio fundamental que permite a los bloques de construcción imitar su propia forma en el edificio que hacen. Y como ya pueden adivinar esto, se explica a través de la geometría fractal.

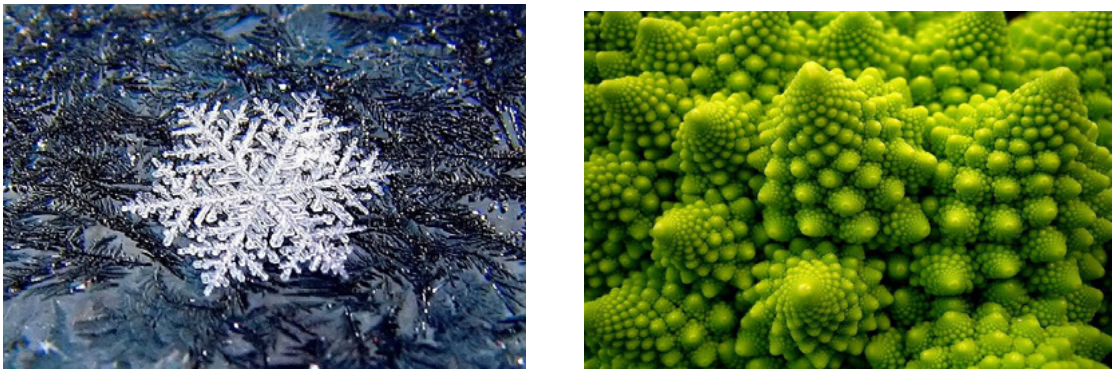


FIGURA A.10. Un copo de nieve a la izquierda y a la derecha un brócoli

Un gran número de partículas mostrará un patrón que es casi igual a las posibilidades iniciales de una sola partícula.

Algo similar sucede aquí: Un gran número de elementos pueden formar un perfil que se deriva del perfil de un elemento. Y debido a que no se puede obligar a ningún elemento a seguir un cierto camino, no hay un gran número de elementos que mostrarán el mismo patrón exacto como el que tiene el otro grupo. Los patrones pueden seguirse por un gran número de elementos que los hace similares, pero no idénticos iguales. De ahí, que todos los copos de nieve se parecen, pero no hay dos que sean exactamente idénticos.

La auto-similitud es realmente algo grande, ocurre sobre toda la naturaleza y muchos han argumentado que la auto-similitud es uno de los principios naturales clave que dan forma a nuestro mundo tal como es. La Auto-similitud se ha observado en todos los campos de investigación no solo en la física, sino también en biología e incluso psicología y sociología.

También, ocurre en todo como en las escrituras y ha sido ampliamente estudiado, más a menudo referido como en la Teología. En el libro del Éxodo, por ejemplo, Moisés construye el Tabernáculo de acuerdo a los patrones celestiales que observó (Éxodo 25:40, Hebreos 8:5), por lo que, el tabernáculo tiene una auto-similitud de algo que existe en un nivel diferente de complejidad, a saber, el cielo.

No es de extrañar que en un determinado momento, Jesús dijo: "Abriré en parábolas mi boca; voy a decir cosas escondidas desde la fundación del mundo" (Mateo 13:35).

La auto-similitud es la repetición de un patrón, comportamiento o forma a diferentes niveles de complejidad. No es una copia idéntica, pero sí una variación del mismo patrón.

Una de las primeras obras de arte fractal fue hecha en 1904 por el matemático sueco Helge von

Koch y su obra fue el llamado copo de nieve de Koch. Tomó un triángulo y se añadió un triángulo similar, pero más pequeño, a cada uno de los lados del primero. Luego, añadió triángulos más pequeños a los lados de los segundos, de nuevo, a los lados de los terceros y así, sucesivamente, hasta el infinito.

HELGE VON KOCH

Cada libro de texto que habla sobre el copo de nieve de Koch explica que, si realmente seguimos este proceso hasta el infinito, vamos a agregar longitud de forma infinita, por lo tanto, se tendría una línea infinitamente larga, lo que teóricamente es cierto. Pero con cada paso del triángulo añadido, la figura se hace más pequeña y en el mundo real no hay tal cosa tan infinitamente pequeño.

Esto es más importante de lo que parece, la gran diferencia entre los copos de nieve en la naturaleza y los copos de nieve de Koch es que los naturales son todos diferentes, mientras que los de Koch son todos iguales. Los copos de nieve de Koch son estériles, mientras que con los naturales, su variedad es lo que hace toda la diferencia. Es por ello que no hay dos árboles, dos montañas ni dos seres humanos iguales, pero hay más.

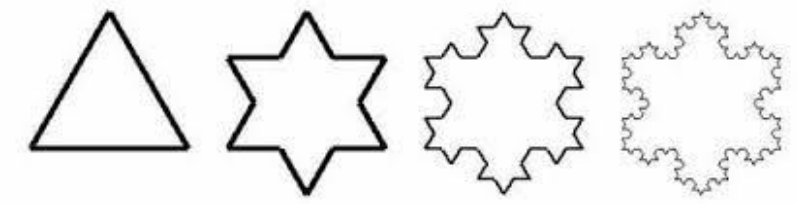


FIGURA A.11. El copo de nieve de Koch

Los triángulos más pequeños de Koch son idénticos al gran triángulo, pero, siguiendo por este camino *ad infinitum*, resulta ser una estructura imposible. Es un nofractal dado ya que los triángulos más pequeños van a perder su forma y nitidez. Un triángulo elegante como ese solo ocurrirá en la mente de Koch y quizás Platón, pero, no en la naturaleza. La naturaleza produce copos de nieve que nunca son los mismos, porque, el fenómeno a gran escala no imita la forma del fenómeno a pequeña escala, debido a la imprevisibilidad, la aleatoriedad y la soberanía. Y las matemáticas no tienen manera de generar aleatoriedad.

Aleatoriedad

La generación de números aleatorios es como un deporte entre los matemáticos. Porque, ¿cómo

diseñar un programa para generar aleatoriedad? ¡No hay manera! Se puede hablar de números que tienen una expansión decimal infinita (como el número pi) y se le dice a una computadora que seleccione decimales a ciertos intervalos (como cada cuatro dígitos o el séptimo o el que sea). La expansión decimal de pi es un número, es aleatoria e infinita (por lo que sabemos hasta ahora), construir números aleatorios a partir de la secuencia de pi no es la aleatoriedad real, porque, cualquier otra computadora podría analizar el resultado y hacer sonar el silbato: Esto no es aleatorio al azar, esto es pi azar. Y como el secreto está fuera, el resto de la secuencia pseudo-aleatoria se puede predecir. Si el segundo equipo divide todos los resultados de la primera y por la forma en que se genera la secuencia pseudo-aleatoria, entonces, es predecible. Eso viola la definición principal de la aleatoriedad y la secuencia no es aleatoria. La verdadera aleatoriedad no puede ser dividida por algo distinto de sí mismo.

Lo mismo sucede con todos los números irracionales como *e* o raíz de 2. Por lo tanto, como la realidad es difusa, la precisión es aproximarla.[39]

El juego del caos[40]

Uno de los fractales más interesantes surge de lo que Michael Barnsley llamó "El juego del caos" y se juega de la siguiente manera.

Primero, vamos a describir las reglas del juego. Bueno, en realidad, no hay un solo partido, hay un número infinito de ellos. Pero, todos siguen el mismo esquema. Tenemos un dado y algunas reglas simples para elegir. Aquí, está uno de los juegos:

Preparación: tome una hoja de papel y un lápiz y marque tres puntos en una hoja, etiquetarlos como 1, 2, 3 y se llamarán bases. Tener una matriz que le permita elegir los números 1, 2 y 3 al azar. Es obvio cómo fabricar una matriz así; luego, tome un dado ordinario y solo identifique las caras 6 con 1, las 5 con 2 y las 4 con 3, también, se puede hacer con colores.

Regla de inicio: a continuación, tomar un dado y colorear dos de las caras rojas, dos azules y dos verdes. Ahora, comenzar con cualquier punto del triángulo. Este punto es la *semilla* para el juego. (En realidad, la semilla puede estar en cualquier lugar en el plano, incluso a kilómetros de distancia del triángulo). Después, tire el dado. Dependiendo de qué color salga, mueva la semilla a la mitad de la distancia hacia el vértice del color que corresponda. Es decir, si sale rojo, moverse al punto medio de la distancia al vértice rojo.

[39] "7. Chaos Theory for Beginners", trad. Idalia Flores de la Mota. Recuperado de: http://www.abarim-publications.com/ChaosTheoryIntroduction.html#.VGt4VvmG-So

[40] Devaney, Robert L., "The Chaos Game", 1995. Recuperado de: http://math.bu.edu/DYSYS/chaos-game/node1.html

Ahora, borre el punto original y empiece de nuevo, usando el resultado de la tirada anterior como la semilla para el siguiente. Es decir, tire el dado de nuevo y mueva el nuevo punto de la mitad de la distancia al vértice del color apropiado y luego borre el punto de partida, como se puede ver en la figura A.12.

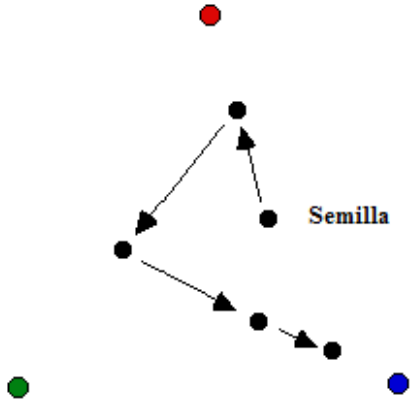


FIGURA A.12. El juego del caos en colores rojo, verde y azul

El objetivo del juego del caos es tirar el dado muchos cientos de veces y predecir cuál será el modelo resultante de puntos. Para los estudiantes que no están familiarizados con el juego de adivinar una imagen, les resultará ser una prueba aleatoria de puntos. Algunos predicen que los puntos, eventualmente, llenarán todo el triángulo. Ambas conjeturas son bastante naturales, dado el carácter aleatorio del juego del caos. Pero, ambas suposiciones están completamente equivocadas. La imagen resultante es cualquier cosa menos una mancha aleatoria; con probabilidad uno, los puntos forman lo que los matemáticos llaman el *triángulo de Sierpinski* como se puede ver en la figura A.13.

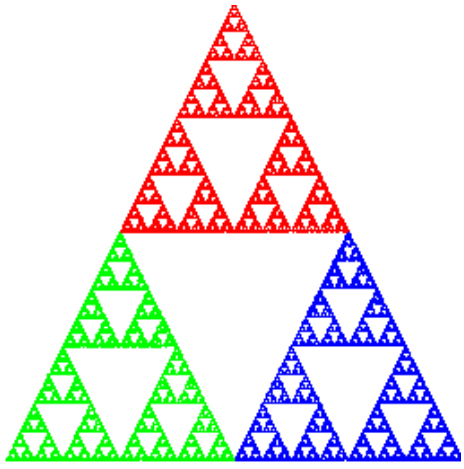


FIGURA A.13. El triángulo de Sierpinski

El color simplemente lo hemos utilizado para indicar la proximidad del vértice con el color dado. Por ejemplo, la porción del triángulo más cercano al vértice verde es de color verde y así, sucesivamente.

Hay una cierta terminología asociada con el juego del caos que es importante. La secuencia de puntos generados por el juego del caos se llama la *órbita* de la semilla. El proceso de repetición de la matriz y el rastreo de la órbita resultante se llama *iteración*. La iteración es importante en muchas áreas de las matemáticas. De hecho, la rama de las matemáticas conocida como *teoría de los sistemas dinámicos discretos* es el estudio de tales procesos iterativos.

Hay dos facetas notables en el juego del caos: la primera es la complejidad geométrica de la figura resultante; el triángulo de Sierpinski **S** es uno de los tipos más básicos de *fractales*. El segundo es el hecho de que no importa la semilla que se utiliza para comenzar el juego: con probabilidad uno, la órbita de cualquier semilla, finalmente, se rellena conforme a **S**.

ANEXO B

COMPLEJIDAD COMPUTACIONAL

B.1. INTRODUCCIÓN

Para que una computadora lleve a cabo una tarea es preciso decirle qué operaciones debe realizar, es decir, debemos describir cómo debe realizar la tarea; dicha descripción se llama *algoritmo*.

Un algoritmo describe el método mediante el cual se realiza una tarea. Un algoritmo consiste en una secuencia de instrucciones, las cuales cuando están realizadas adecuadamente dan lugar al resultado deseado.

La noción de algoritmo no es exclusiva de la computación o de las matemáticas. Existen algoritmos que describen toda clase de procesos de la vida real, por ejemplo, las recetas de cocina, las partituras musicales, etc. En todos los casos anteriores, el ejecutor o procesador de las instrucciones que realiza la tarea correspondiente es el hombre. Sin embargo, el agente que interpreta y realiza las instrucciones de un algoritmo se llama *procesador*.

Un procesador puede ser una persona, una computadora o cualquier otro sistema electrónico o mecánico. Un procesador realiza un proceso siguiendo o ejecutando el algoritmo correspondiente. La ejecución de un algoritmo requiere la realización de cada uno de los pasos o instrucciones que lo constituyen. De aquí, se desprende que una computadora no es más que un tipo particular de procesador.

Como ya se ha dicho, para que un procesador lleve a cabo un proceso debe estar previamente provisto de un algoritmo adecuado. Por ejemplo, el cocinero debe conocer la receta, el pianista, la partitura, etc.

Si el procesador de un algoritmo es una computadora, el algoritmo se debe expresar en forma de programa. Un programa se escribe en un lenguaje de programación y a la actividad consistente para expresar un algoritmo en un lenguaje de programación se le llama *programar*.

Cada paso del algoritmo se expresa mediante una instrucción o sentencia en el programa. Por tanto, un programa consiste en una secuencia de instrucciones, cada una de las cuales especifica ciertas operaciones por realizar en la computadora.

Por tanto, podría afirmarse que son más importantes los algoritmos que los lenguajes de programación e incluso las mismas computadoras, considerando que un lenguaje de programación es simplemente un medio conveniente para expresar un algoritmo y una computadora es simplemente un procesador para ejecutarlo; es decir, tanto los lenguajes de programación como las computadoras son medios para lograr un fin: ejecutar un algoritmo.

Con esto no se pretende restar importancia a las computadoras y a los lenguajes de programación. Los avances en la tecnología informática permiten día a día ejecutar algoritmos, rápidamente, con mayor precisión, y a mejor precio.

Un aspecto importante es el *diseño de algoritmos*. ¿Cómo diseñar buenos algoritmos? El diseño de buenos algoritmos requiere creatividad e ingenio y no existen, en general, reglas para diseñar algoritmos; en otras palabras, no existe un algoritmo para diseñar algoritmos.

Sí existen varios algoritmos para resolver un problema ¿Cuál de ellos es el *mejor*, en el sentido de que necesita menos recursos informáticos? ¿Cuáles son los mínimos recursos informáticos necesarios para llevar a cabo una tarea determinada? (es decir, ¿Qué recursos utilizará el mejor algoritmo posible para realizar dicha tarea?) ¿Se puede averiguar, cuál es el mejor algoritmo? ¿Existen problemas para los cuales el mejor algoritmo posible requerirá tantos recursos que hará inviable su ejecución, incluso, con la computadora más grande y rápida existente?

Todas estas cuestiones se engloban para su estudio bajo el título: *complejidad de algoritmos.*

B.2. COMPLEJIDAD DE ALGORITMOS: TIEMPO Y ESPACIO

Como sabemos, un programa es la representación de un algoritmo en un lenguaje de programación, el cual puede interpretar y ejecutarse en una computadora.

La manera de representar un algoritmo en forma de programa no es, en general, única. Asimismo, para resolver un problema para el cual existe un algoritmo, no es único el algoritmo que lo resuelve. Cabe, por tanto, preguntarse ¿cuál es el mejor algoritmo de entre dos algoritmos dados que resuelven el mismo problema?

Una posible alternativa para contestar dicha cuestión consiste en representar los dos algoritmos

mediante un lenguaje de programación, a continuación, ejecutarlos en una computadora y medir el tiempo requerido para cada uno de ellos y obtener la solución de un mismo problema particular.

El *tiempo* de ejecución requerido por un algoritmo para resolver un problema es uno de los parámetros importantes en la práctica para medir la bondad de un algoritmo, pues, entre otros factores, el tiempo de ejecución equivale al tiempo de utilización de la computadora y, en consecuencia, al coste económico.

Incluso, si el tiempo de ejecución es demasiado grande, puede suceder que el algoritmo sea en la práctica inútil, pues, el tiempo necesario para su ejecución puede sobrepasar al tiempo disponible de la computadora.

Resulta evidente que el tiempo real requerido por una computadora para ejecutar un algoritmo es directamente proporcional al número de operaciones básicas elementales, que la misma debe realizar en su ejecución, medir, por lo tanto, el tiempo real de ejecución equivale a medir el número de operaciones elementales realizadas. (Nosotros supondremos, desde ahora, que todas las operaciones básicas se ejecutan en una unidad de tiempo. Para una mayor precisión habría que distinguir los tiempos de ejecución de cada una de las distintas operaciones elementales). Por esta razón, se suele llamar *tiempo de ejecución* no al tiempo real físico, sino al número de operaciones elementales realizadas.

Otro de los factores importantes, en ocasiones el decisivo, para comparar algoritmos es la cantidad de memoria de la máquina requerida para almacenar los datos durante el proceso.

La cantidad de memoria utilizada por un algoritmo durante el proceso se suele llamar *espacio* requerido por el algoritmo.

Para entender la diferencia del espacio requerido por dos algoritmos que resuelven el mismo problema, consideremos lo siguiente:

Ejemplo B.1

Dados *n* números naturales en forma secuencial, es decir, dados de uno en uno con un intervalo de tiempo entre uno y otro, encuentre ¿cuál es el máximo de todos ellos? Entonces, veamos dos algoritmos que resuelven el mismo problema.

Algoritmo 1

PASO 1: almacenar los n números utilizando n variables $a_1,...,a_n$
PASO 2: asignar $m = a_1$, $i = 2$
PASO 3: si $m>a_i$, entonces, $i = i + 1$, en caso contrario, $m = a_i$, $i = i + 1$
PASO 4: si $i>n$ el máximo es m FIN. En caso contrario, regresar al paso 3

Algoritmo 2

PASO 1: asignar el primer elemento a la variable m
PASO 2: asignar el siguiente elemento a la variable a
PASO 3: si $a>m$, entonces, $m = a$
PASO 4: mientras queden elementos, volver al paso 2
PASO 5: el máximo es m FIN

Los dos algoritmos anteriores resuelven el problema del máximo; sin embargo, el espacio requerido por el algoritmo 1 depende del valor de n. Cuanto mayor sea n, mayor es el número de variables a las que hay que asignar valores en el paso 1 y, en consecuencia, mayor será la cantidad del espacio necesario.

El algoritmo 2, por el contrario, utiliza solo las variables m y a, independientemente, de la cantidad de números, pues este segundo algoritmo, antes de recibir un nuevo número, calcula el máximo de los números anteriores, manteniéndolo en la variable m.

Para que el segundo algoritmo pueda ejecutarse, está claro que el intervalo de tiempo que transcurre entre la entrada de dos números debe ser mayor o, a lo sumo, igual al tiempo requerido para efectuar las operaciones del paso 3.

En adelante, nos ocuparemos solamente del *tiempo* requerido por un algoritmo para su ejecución.

Si observamos la fórmula propuesta para medir el tiempo requerido por un algoritmo, advertimos que, al no ser la única manera de representarlo mediante un programa y al no ser, tampoco, la única computadora en donde se ejecuta, resulta que dicha medida será variable dependiendo, fundamentalmente, de los siguientes factores:

1) El lenguaje de programación elegido
2) El programa que representa el algoritmo
3) La computadora que lo ejecuta

En definitiva, al existir gran cantidad de lenguajes, técnicas de programación y computadoras diferentes, resulta que la forma propuesta de medir el tiempo y, en consecuencia, la bondad de un algoritmo, no resulta adecuada.

Por esta razón, surge la necesidad de medir el tiempo requerido por un algoritmo, independientemente, de su representación y del procesador que lo ejecute.

Por otra parte, si la cantidad de datos del problema por resolver es pequeña, prácticamente, cualquier algoritmo que lo resuelva lo hará en un espacio corto de tiempo, siendo, en este caso, verdaderamente indiferente elegir uno u otro algoritmo para resolverlo.

Cuando aparece la dificultad en cuanto al tiempo requerido por un algoritmo, es cuando el volumen de datos del problema por resolver aumenta.

Cuando el volumen de datos es suficientemente grande, es cuando un algoritmo puede realmente aventajar a otro para resolver el mismo problema. Veamos lo siguiente:

Ejemplo B.2

Dada una lista $\{a_1, a_2,...,a_n\}$ de números ordenados de menor a mayor, verifique, si hay algún número repetido. Los siguientes algoritmos resuelven el problema:

Algoritmo 3

PASO 1: $i = 1, j = 2$
PASO 2: si $a_i = a_j$, entonces, la respuesta es SÍ. FIN
PASO 3: si $j < n$, entonces, $i = i + 1$; $j = j + 1$ y volver al paso 2
PASO 4: la respuesta es NO. FIN

Algoritmo 4

PASO 1: $i = 1$; $j = 2$
PASO 2: si $a_i = a_j$, entonces, la respuesta es SÍ. FIN
PASO 3: si $j < n$, entonces, $j = j + 1$ y volver al paso 2
PASO 4: si $i < n - 1$, entonces, $i = i + 1$; $j = i + 1$ y volver al paso 2
PASO 5: la respuesta es NO. FIN

Si contamos solo las comparaciones que se efectúan en el paso 2 de ambos algoritmos, observamos que el primero compara cada número de la lista con el inmediatamente posterior, por lo que, efectúa $n - 1$ comparaciones como máximo (el peor caso será cuando no haya repeticiones o bien, cuando estén repetidos los dos últimos números solamente); mientras que, en el segundo algoritmo puede suceder el caso que se compare cada número con todos los que le siguen, por lo que, el número de comparaciones puede llegar a ser:

$$(n - 1) + (n - 2) + (n - 3) +...+ 2 + 1 = n^2 /2 - n/2$$

Si se tiene en cuenta la tabla B.1, en la cual se observan los valores que toman n–1 y $n^2/2$–$n/2$ para distintos valores de n, se deduce de inmediato que el algoritmo 3 es claramente más ventajoso que el 4, especialmente, cuando el valor n es grande.

TABLA B.1. Valores de n para diferentes funciones

n	$n - 1$	$n^2/2 - n/2$
2	1	1
10	9	45
100	99	4 950
1 000	999	499 500
100 000	99 999	49 995 000
1 000 000	999 999	4 999 950 000

B.3. PEOR CASO Y CASO PROBABILÍSTICO

En el estudio anterior del tiempo de ejecución de los algoritmos, hemos analizado el número de comparaciones que podrían llegar a realizarse en algún caso extremo. A estos casos en que el tiempo es el mayor posible de entre todos los casos que se pueden presentar, se les llama *el peor caso.*

En ocasiones, el peor caso se presenta a menudo al resolver un problema y en otras se presenta con poca frecuencia. Por esta razón, tiene interés el estudio del tiempo de ejecución de un algoritmo en el *caso medio* o *caso probabilístico*. Este estudio entra dentro del campo del cálculo de probabilidades y de la estadística y ya que es de gran interés en la evaluación de los algoritmos, resulta en ocasiones sumamente complejo.

En adelante, el análisis que hagamos de los algoritmos, siempre, será estudiando el comportamiento del mismo en el peor caso.

A pesar de la ventaja que esto supone en ocasiones, un algoritmo cuyo comportamiento en el peor caso es desastroso, puede ser útil en una gran cantidad de casos, si se presenta el peor caso con una frecuencia muy pequeña.

El análisis de algoritmos se encarga del estudio del *tiempo* y *espacio* requeridos por un algoritmo para su ejecución. Ambos parámetros, como hemos visto, pueden ser estudiados respecto del *peor caso* (o caso general) o respecto del *caso probabilístico* (o caso esperado).

En la práctica, casi siempre es más difícil determinar el tiempo de ejecución promedio que el del peor caso, pues el análisis se hace intratable en matemáticas. Así pues, se utilizará el tiempo de ejecución del peor caso como medida principal de la complejidad del tiempo, aunque, se mencionará la complejidad del caso promedio cuando pueda hacerse en forma significativa.

Tanto el tiempo como el espacio de un algoritmo son parámetros que, en general, dependen del tamaño n de los datos de entrada del algoritmo. En consecuencia, son funciones $T(n)$ y $E(n)$ enteras de la variable entera.

En general, no resulta sencillo determinar el valor exacto de la función tiempo $T(n)$ de un algoritmo. Para poder hacerlo es preciso conocer con exactitud cuáles y cuántas veces se realizan las operaciones básicas, cuya ejecución requiere un tiempo constante conocido.

En definitiva, lo que aparece al analizar un algoritmo es un problema de combinatoria. No obstante, como hemos dicho, no siempre resulta fácil hacer el cálculo exacto del número de operaciones requeridas por un algoritmo. Por esta razón, en muchas ocasiones el análisis de los algoritmos se reduce a estudiar el *comportamiento* de la función tiempo $T(n)$ y no cuál es su valor exacto.

B.4. ANÁLISIS ASINTÓTICO DE FUNCIONES

En esta sección, el objeto de estudio serán las funciones reales de la variable natural f: $N \to R$. La idea fundamental consiste en comparar funciones de este tipo para poder decir cuál tiene mejor el comportamiento asintótico, es decir, cuál es menor cuando la variable independiente es suficientemente grande.

Si sabemos hacer esto con dos funciones, podremos utilizar las funciones de tiempo de dos algoritmos y, así, determinar cuál de ellos tiene mejor comportamiento asintótico.

El problema que en ocasiones resulta complicado es el de hallar, explícitamente, la función tiempo de un algoritmo. El análisis asintótico que haremos permitirá, en ocasiones, conocer cómo se comporta una función aún sin conocer esta.

Definición:

Si f y g son dos funciones de N en R, se dice que g domina en forma asintótico a f (o simplemente que g domina a f), si existen enteros $k \geq 0$ y $m \geq 0$, tales que, se verifica la desigualdad de: $|f(n)| \leq k\,|g(n)|$ para todo entero $n \geq m$.

Si g domina a f y $g(n) \neq 0$, entonces, $|f(n)/g(n)| \leq k$ para casi todos los enteros n (es decir, para todos salvo una cantidad finita).

En concreto, para todos los valores $n \geq m$ se verifica dicha desigualdad.

En esta situación, si f y g son funciones de tiempo de dos algoritmos F y G, respectivamente, resulta que el algoritmo F nunca tardará más de k veces el tiempo que tarda el algoritmo G en resolver un problema del mismo tamaño.

Por ejemplo, si $f(n) = n$ y $g(n) = n^3$, se verifica que g domine a f: en efecto, si $m = 0$ y $k = 1$, se verifica que para todo $n \geq m$ se cumple la desigualdad:

$$|n| \leq k\,|n^3|$$

En este caso, $|n| \leq |n^3|$, ya que, $k = 1$, pues, el cubo de un número natural es siempre mayor o igual que dicho número.

La definición de dominación establece una relación binaria en el conjunto de las funciones de N en R. El siguiente teorema da propiedades de dicha relación.

Teorema B.1

La relación de dominación $<$ definida por $f < g \leftrightarrow g$ domina a f es una relación reflexiva y transitiva. La demostración de este teorema queda como ejercicio; así mismo, proponemos como ejercicio el comprobar que dicha relación no es simétrica, es decir, si g domina a f no se verifica necesariamente que f domine a g. En consecuencia, la relación de dominación no es una relación de equivalencia.

Por otra parte, esta relación, tampoco, es relación de orden, pues, no verifica la propiedad antisimétrica (también, compruébelo como ejercicio).

Si f es una función de N en R, denotaremos por $O(f)$ al conjunto de todas las funciones dominadas por f.

Con notación matemática:

$$O(f) = \{ g \mid g : N \to R \text{ es aplicación y } g < f\}$$

Si una función g pertenece al conjunto $O(f)$, se suele decir que g es una o mayúscula de F o que g *es de orden f.*

Por ejemplo, decir que el tiempo de ejecución de un programa $T(n)$ es $O(n^2)$ significa que existen constantes enteras positivas m y k, tales que, para $n \geq m$ se tiene $T(n) \leq kn^2$.

Ejemplo B.3

Si suponemos que $T(0) = 1$, $T(1) = 4$ y, en general, $T(n) = (n + 1)^2$. Entonces, se observa que $T(n)$ es $O(n^2)$ cuando $m = 1$ y $k = 4$, es decir, para $n \geq 1$ se tiene que $(n + 1)^2 \leq 4n^2$ que es fácil de demostrar. Observe que no se puede hacer $m = 0$, pues, $T(0) = 1$ no es menor que $k0^2 = 0$ para ninguna constante k.

Se dice que $T(n)$ es $O(f(n))$, si existen m y k, tales que, $T(n) \leq kf(n)$ cuando $n \geq m$. Cuando el tiempo de ejecución de un programa es $O(f(n))$, se dice que tiene velocidad de crecimiento $f(n)$.

Teorema B.2

Se verifican las siguientes propiedades de los conjuntos $O(f)$:

1) $f \in O(f)$
2) Si $f \in O(g)$, entonces, $cf \in O(g)$ para todo $c \in R^+$
3) Si $f \in O(g)$ y $h \in O(g)$, entonces, $f + g \in O(g)$
4) $f \in O(g)$, si y solo si $O(f) \subset O(g)$
5) Si $f \in O(g)$ y $g \in O(f)$, entonces, $O(f) = O(g)$
6) Si $f \in O(g)$ y $g \in O(h)$, entonces, $f \in O(h)$

Algunos conjuntos $O(f)$ que aparecen con frecuencia al analizar algoritmos son:

$$O(1),\ O(\log n),\ O(n),\ O(n \log n),\ O(n^2),...,O(n^p),...,O(c^n),\ O(n!)$$

Si la función tiempo de un algoritmo es de orden 1, se dice que dicho algoritmo tiene complejidad *constante*; si es de orden log n, se dice que es *logarítmica*; si es de orden n, se dice que es *lineal*; si es de orden n^p, siendo p un número natural, se dice que es *polinómica*; si es de orden c^n con $c > 1$, se dice que es *exponencial*; si es de orden $n!$, se dice que es *factorial.*

Observación: mientras no se diga lo contrario, se entenderá que los logaritmos que aparecen son en base 2.

Ejemplo B.4

La función $T(n) = 3n^3 + 2n^2$ es $O(n^3)$. Para comprobar esto, sean $m = 0$ y $k = 5$, entonces, para $n \geq 0$, $3n^3 + 2n^2 \leq 5n^3$. También, se podría decir que $T(n)$ es $O(n^4)$, pero, sería una afirmación más débil que decir que $T(n)$ es $O(n^3)$.

Ejemplo B.5

La función 3^n no es $O(2^n)$.

Demostración:
Se supone que existen m y k, tales que, para toda $n \geq m$ se tiene $3^n \leq k\ 2^n$, entonces, $k \geq (3/2)^n$ para cualquier $n \geq m$, pero, $(3/2)^n$ se hace arbitrariamente grande conforme n crece y, por lo tanto, ninguna constante k puede ser mayor que $(3/2)^n$ para toda n.

Cuando se dice que $T(n)$ es $O(f(n))$ se sabe que $f(n)$ es una cota superior para la velocidad de crecimiento de $T(n)$. Para especificar una cota inferior para la velocidad de crecimiento de $T(n)$, se usa la notación $\Omega(g(n))$ que se lee "$T(n)$ es omega de $g(n)$", lo cual significa que existe una constante c, tal que, $T(n) \geq cg(n)$ para un número infinito de valores de n.

Ejemplo B.6

Para verificar que la función $T(n) = n^3+2n^2$ es $\Omega(n^3)$, sea c=1, entonces, $T(n) \geq cn^3$ para n=0, 1, ...

Teorema B.3

Dadas las funciones de tiempo, se verifican las siguientes contenciones:

$$O(1) \subset O(\log n) \subset O(n) \subset O(n\log n) \subset O(n^2) \subset O(c^n) \subset O(n!)$$

Siendo $c > 1$.

Además, dichas contenciones son propias, en ninguna de ellas se da la igualdad.

Demostración:

1) $O(1) \subset O(\log n)$: en efecto, si $m = 2$ y $k = 1$, se verifica que para todo $n > 2$ es $1 \leq \log n$, por lo cual, la función constante 1 pertenece a $O(\log n)$, por lo tanto, por el apartado 4 del teorema 2.2, $O(1) \subset O(\log n)$.

   Además, la contención es propia, si $O(\log n)$ estuviera contenido en $O(1)$, se verificaría que $\log n$ pertenecería a $O(1)$, y por lo tanto, $\log n$ estaría dominada por la función constante 1.

   Esto significa que existen $m \geq 0$ y $k \geq 0$, tales que, para todo $n > m$ sería:

$$|\log n| \leq k\,|1|$$

   es decir,

$$\log n \leq k$$

   Lo cual es una contradicción, pues como:

$$lim_{\,n\to\infty} \log n = \infty$$

   La función $\log n$ no puede estar acotada por una constante k.

2) $O(\log n) \subset O(n)$: considerando $m = 0$ y $k = 1$, se verifica que para todo $n > m$ se cumple:

$$|\log n| \leq k|n|$$

   Ya que, para todo $n > 0$ es $\log n < n$. Por lo tanto, $\log n \in O(n)$ y por la misma razón de antes:

$$O(\log n) \subset O(n)$$

3) $O(n) \subset O(n \log n)$: si $n > 2$, se verifica que $\log n > 1$ y, por tanto, $n \log n > 1$. $n = n$, por lo que, tomando $m = 2$ y $k = 1$, se verifica que para todo $n > m$ se tiene:

$$|n| \leq k\,|n \log n|$$

Luego, $n \in O\,(n \log n)$ y, por lo tanto, $O(n) \subset O(n \log n)$.

4) $O(n \log n) \subset O(n^2)$: como $\log n < n$, si $n > 0$, resulta que $n \log n < n$. $n = n^2$. Por lo tanto, considerando $m = 0$ y $k = 1$, se verifica que para todo $n > m$ es:

$$|n \log n| < k\,|n^2|$$

Luego, $n \log n \in O(n^2)$ y, por lo tanto, $O(n \log n) \subset O(n^2)$.

5) $O(n^2) \subset O(c^n)$ (si $c > 1$): como en casos anteriores, es suficiente probar que para una n suficientemente grande se verifica la desigualdad:

$$n^2 < c^n$$

Lo que equivale, cuando se toman logaritmos (tenga en cuenta que como $c > 1$ es $\log c > 0$) a la igualdad:

$$\log n^2 < \log c^n$$

o bien,

$$(\log n) / n < (\log c)/2$$

Como $(\log c)/2$ es una constante positiva, resulta que existe un valor m, tal que, para todo valor $n > m$.

Es decir:

$$\lim_{n \to \infty} \frac{\log n}{n} = 0$$

$$(\log n)/n < (\log c)/2$$

Con lo cual queda demostrado.

6) $O(c^n) \subset O(n!)$: sean $k = [c]$ y m = máx $\{[c^k], k\}$ donde $[c]$ representa al menor entero mayor que c.

Entonces, si $n > m$, se verifica:

$$n! = n\,(n-1)\,(n-2)\,...(k+1)\,k(k-1)...2\,1$$

Y como $k \geq [c]$ y $[c] \geq c$, se verifica:

$$(n-1)(n-2)...(k+1)\,k \geq c^{n-k}$$

Por otra parte, como $n > c^k$, resulta:

$$n! > c^k\,c^{n-k} = c^n$$

En consecuencia $n! > c^n$, si $n > m$ y, por lo tanto, $O(c^n) \subset O(n!)$ como antes.

Teorema B.4

Si c es una constante, se verifica:

$$\text{a) } \Sigma\, c \in O(n)$$

$$\text{b) } \Sigma\, i \in O(n^2)$$

$$\text{c) } \Sigma\, i^2 \in O(n^3)$$

La demostración se deja como ejercicio.

B.5. VELOCIDAD DE CRECIMIENTO Y CÁLCULO DEL TIEMPO DE EJECUCIÓN DE UN PROGRAMA

Para comenzar, partiremos del supuesto de que es posible evaluar programas comparando sus funciones de tiempo de ejecución, sin considerar las constantes de proporcionalidad. Es decir, un programa con tiempo de ejecución $O(n^2)$ es mejor que uno con tiempo de ejecución $O(n^3)$, sin embargo, además de los factores constantes debidos al compilador y a la máquina, existe un

factor constante debido a la naturaleza del programa mismo. Es posible, que con una combinación determinada del compilador y máquina, el primer programa tarde $100n^2$ milisegundos, mientras que el segundo tarda $5n^3$ milisegundos; en este caso, ¿no es preferible el segundo programa que el primero? La respuesta a esto depende del tamaño de las entradas, las cuales se esperan sean procesadas por los programas.

Para entradas de tamaño $n < 20$, el programa con tiempo de ejecución $5n^3$ será más rápido que el del tiempo de ejecución $100n^2$. Así, si el programa se va a ejecutar con entradas pequeñas, será preferible el programa cuyo tiempo de ejecución es $O(n^3)$. No obstante, conforme n crece, la razón de los tiempos de ejecución que es $5n^3/100n^2 = n/20$ se hace arbitrariamente grande. Así, a medida que crece el tamaño de la entrada, el programa $O(n^3)$ requiere un tiempo, significativamente, mayor que $O(n^2)$. Pero, si hay algunas entradas grandes en los problemas para cuya solución se están diseñando estos dos programas, será mejor optar por el programa cuyo tiempo de ejecución tiene la menor velocidad de crecimiento.

Ejemplo B.7

En la figura B.1, se muestran, gráficamente, los tiempos de ejecución de cuatro programas de distintas complejidades de tiempo con medidas en segundos, para una combinación determinada de compilador y máquina.

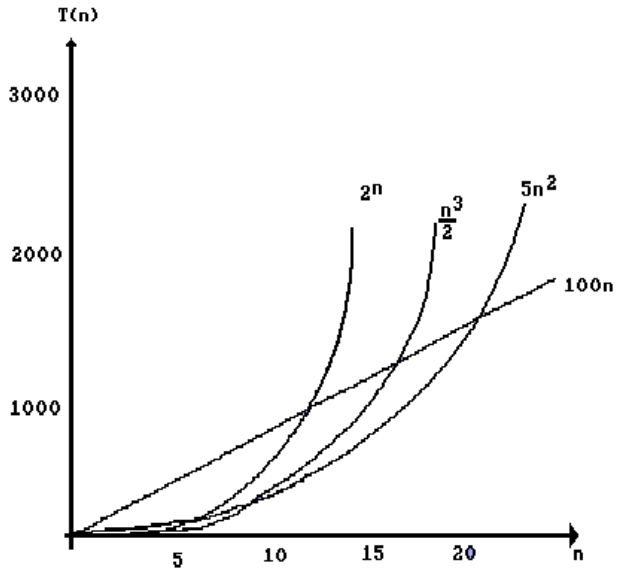


FIGURA B.1. Velocidad de crecimiento de diferentes funciones

A continuación, estime que se dispone de 1 000 segundos o alrededor de 17 minutos para resolver un problema determinado ¿qué tamaño de problema se puede resolver? Considerando estos tiempos, en la segunda columna de la tabla B.2, se muestra que los cuatro algoritmos pueden resolver problemas de un tamaño similar en 10^3 segundos. Si se adquiere una máquina que funciona diez veces más rápido sin costo adicional, entonces, es posible dedicar 10^4 segundos a la solución de problemas que antes requerían 10^3 segundos. Ahora, el tamaño máximo de un problema que es posible resolver con los cuatro algoritmos se muestra en la tercera columna de esta tabla y la razón entre los valores de la segunda y tercera columna se muestran en la cuarta.

Se observa que un aumento de 1 000% en la velocidad de la computadora origina apenas un incremento del 30% en el tamaño del problema que se puede resolver con el programa $O(2^n)$. Los aumentos adicionales de un factor de diez en la rapidez de la computadora, a partir de este punto, originan aumentos porcentuales aún menores en el tamaño de los problemas. De hecho, el programa $O(2^n)$ solo puede resolver problemas pequeños, independientemente, de la rapidez de la computadora.

TABLA B. 2. Tiempo de ejecución y tamaño máximo del problema

Tiempo de ejecución $T(n)$	**Tamaño máximo del problema para 10^3 segundos**	**Tamaño máximo del problema para 10^4 segundos**	**Incremento en el tamaño máximo del problema**
$100\,n$	10	100	10.0
$5n^2$	14	45	3.2
$n^3/2$	12	27	2.3
2^n	10	13	1.3

En la tercera columna de la tabla se puede apreciar una superioridad evidente del programa $O(n)$, este permite un aumento del 1 000% en la rapidez de la computadora. Y se observa que los programas $O(n^3)$ y $O(n^2)$ permiten aumentos del 230% y 320%, respectivamente, en el tamaño del problema para un incremento del 1 000% en la rapidez de la computadora. Estas razones se mantendrán vigentes para los incrementos adicionales en la velocidad de la computadora.

A medida que las computadoras aumenten su rapidez y disminuyan su precio, también, el deseo de resolver problemas más grandes y complejos seguirá creciendo. Así, la importancia del descubrimiento y el empleo de algoritmos eficientes (cuyas velocidades de crecimiento sean pequeñas) irán en aumento.

Observaciones:

1) La velocidad de crecimiento del tiempo de ejecución, del peor caso, no es el único criterio ni necesariamente el más importante para evaluar un algoritmo.

2) De acuerdo con el punto anterior, si un programa solo se va a usar algunas veces, el costo de su escritura y depuración es el dominante, de manera que, el tiempo de ejecución raramente influirá en el costo total. En tal caso, debe elegirse el algoritmo que sea más fácil de aplicar correctamente.

3) Un algoritmo eficiente, pero complicado, puede no ser apropiado porque posteriormente puede suceder que tenga que darle mantenimiento otra persona distinta del escritor. Se espera que al difundir el conocimiento de las principales técnicas de diseño de algoritmos eficientes, se podrán utilizar, libremente, algoritmos más complejos, pero, debe considerarse la posibilidad de que un programa resulte inútil debido a que nadie entiende sus sutiles y eficientes algoritmos.

4) Existen ejemplos de algoritmos eficientes que ocupan mucho espacio para poder aplicarlos sin un almacenamiento secundario lento, lo cual puede anular la eficiencia.

5) En los algoritmos numéricos, la precisión y la estabilidad son tan importantes como la eficiencia.

Tiempo de ejecución de un programa

Primero, comenzaremos con algunas reglas básicas de suma y producto en notación asintótica:

Sean $T_1(n)$ y $T_2(n)$ los tiempos de ejecución de dos fragmentos P_1 y P_2 de un programa y que $T_1(n)$ es $O(f(n))$ y $T_2(n)$ es $O(g(n))$, entonces:

$$T_1(n) + T_2(n)$$

El tiempo de ejecución de P_1 seguido de P_2 es $O(\text{máx } [f(n), g(n)])$; esto se puede verificar observando que para algunas constantes, c_1, c_2, n_1, n_2, si $n \geq n_1$, por lo tanto:

$$T_1(n) \leq c_1 f(n)$$

Y, si $n \geq n_2$, entonces:

$$T_2(n) \leq c_2 g(n)$$

Sea m = máx (n_1, n_2), si $n \geq m$, en tal caso:

$$T_1(n) + T_2(n) \leq c_1 f(n) + c_2 g(n)$$

De aquí, se concluye, si $n \geq m$, siendo así:

$$T_1(n) + T_2(n) \leq (c_1 + c_2) \text{ máx } [f(n), g(n)]$$

por lo tanto: $T_1(n) + T_2(n)$ es $O(\text{máx } [f(n), g(n)])$

Ejemplo B.8

La regla de la suma anterior se puede usar para calcular el tiempo de ejecución de una secuencia de pasos del programa, donde cada paso puede ser fragmento de un programa arbitrario con ciclos y ramificaciones. Considere que se tienen tres pasos cuyos tiempos de ejecución son, respectivamente, $O(n^2)$, $O(n^3)$ y $O(n \log n)$; entonces, el tiempo de ejecución de los dos primeros pasos ejecutados en secuencia es $O(\text{máx } (n^2, n^3))$ es $O(n^3)$ y el tiempo de ejecución de los tres juntos es $O(\text{máx}(n^3, n \log n))$ es $O(n^3)$.

En general, el tiempo de ejecución de una secuencia fija de pasos, dentro de un factor constante, es igual al tiempo de ejecución del paso con mayor tiempo de ejecución. En raras ocasiones, dos pasos pueden tener tiempos de ejecución inconmensurables (ninguno es mayor que el otro ni son iguales). Por ejemplo, pueden haber pasos con tiempo de ejecución $O(f(n))$ y $O(g(n))$ donde:

$$f(n) = \begin{cases} n^4, & \text{si } n \text{ es par} \\ n^2, & \text{si } n \text{ es impar} \end{cases} \qquad g(n) = \begin{cases} n^2, & \text{si } n \text{ es par} \\ n^3, & \text{si } n \text{ es impar} \end{cases}$$

En tales casos, la regla de la suma debe aplicarse directamente; en el ejemplo, el tiempo de ejecución es $O(\text{máx } [f(n), g(n)])$, esto es n^4, si n es par y n^3, si n es impar.

Otra observación útil sobre la regla de la suma es que, si $g(n) \leq f(n)$ para toda $n \geq m$, entonces, $O(f(n) + g(n))$ es lo mismo que $O(f(n))$. Por ejemplo, $O(n^2 + n)$ es lo mismo que $O(n^2)$.

La regla del producto es la siguiente: si $T_1(n)$ y $T_2(n)$ son $O(f(n))$ y $O(g(n)$, respectivamente, entonces, $T_1(n)T_2(n)$ es $O(f(n)g(n))$, su demostración se deja como ejercicio. Según la regla del producto, $O(cf(n))$ significa lo mismo que $O(f(n))$, si c es una constante positiva cualquiera. Por ejemplo, $O(n^2/2)$ es lo mismo que $O(n^2)$.

Una vez que se han estudiado las herramientas necesarias para analizar el comportamiento de la función tiempo de un algoritmo, se hará el análisis de algunos algoritmos.

Ejemplo B.9

Encuentre el máximo de un conjunto de n números naturales $[a_1, a_2,...,a_n]$. Resolviendo se tiene:

Algoritmo máximo secuencial

Entrada: la lista $L = \{ a_1,...,a_n\}$
PASO 1: $m = a_1 \quad i = 2$
PASO 2: si $m < a_i$, entonces, $m = a_i$
PASO 3: $i = i + 1$
PASO 4: si $i > n$ FIN. En caso contrario, volver al paso 2
Salida: el máximo es m

Análisis del algoritmo:

En el paso 1, se realizan dos operaciones; en el paso 2, en el peor caso, otras dos; en el paso 3 una y en el paso 4, en el peor caso dos. Por lo tanto, cada vez que del paso 4 volvemos al paso 2 serán necesarias cinco operaciones; después, de aplicar el paso 1, el ciclo 2 – 4 se ejecuta $n - 1$ veces (hasta que i toma el valor $n + 1$), el número total de operaciones que requiere este algoritmo, en el peor caso, será:

$$T(n) = 2 + 5(n - 1)$$

El comportamiento asintótico queda dado por la expresión:

$$T(n) \in O(n)$$

Posteriormente, comparamos este algoritmo con el siguiente algoritmo recursivo.

Algoritmo máximo binario

Entrada: la lista $L = \{ a_1,...,a_i, a_{i+1},...,a_j\}$ con n elementos, $i = 1$
PASO 0: si $i = j$, entonces, $m = a_1$ e ir al paso 5
PASO 1: $k = [i+j/2]$ (donde [ ] indica la parte entera de $i+j/2$)
PASO 2: $L_1 = \{ a_i,...,a_k\}$, $L_2 = \{a_{k+1},...,a_j\}$
PASO 3: m_1 = MÁXIMO BINARIO (L_1), m_2 = MÁXIMO BINARIO (L_2)
PASO 4: si $m_1 > m_2$, entonces, $m = m_1$. En caso contrario, $m = m_2$
PASO 5: FIN
Salida: el máximo es m

Análisis del algoritmo:

Si llamamos $T(n)$ al tiempo que requiere el algoritmo para hallar el máximo de una lista de n números, se verifica que el tiempo requerido para ejecutar el paso 3 será 2 $T(n/2)$ (cuando n sea potencia de 2). Como el número de operaciones que requieren los pasos 0, 1, 2, 4 y 5 es constante se verificará que:

$$T(n) \leq 2\ T(n/2) + c$$

Donde c es una constante. Como t (1) es constante, podemos suponer que $T(1) \leq c$, por lo que, se verifica que $T(n) \in O(n)$.

De esto último, se concluye que el comportamiento asintótico de ambos algoritmos es el mismo. Observe que en el análisis asintótico de este segundo algoritmo no fue necesario obtener la función de tiempo del mismo.

ANEXO C

SOLUCIONES PARA EL PROBLEMA DEL AGENTE VIAJERO

C.1. INTRODUCCIÓN

Como vimos en el capítulo 3, el PAV (Problema del Agente Viajero) o TSP (Traveling Salesman Problem) es un clásico de optimización debido a la amplia cantidad de aplicaciones, por esta razón hay una amplia variedad de métodos de solución que se pueden clasificar en exactos y heurísticos; estos últimos, se han desarrollado por la necesidad de contar con soluciones rápidas para el problema. Dentro de los métodos exactos se encuentran los de la PD (Programación Dinámica) como ya hemos visto y, en este anexo, se muestra otro método de solución exacto, desarrollado en el área de la PE (Programación Entera).

C.2. ALGORITMO DE R-A (RAMIFICACIÓN Y ACOTAMIENTO)

En PLE (Programación Lineal Entera) existen métodos para resolver el PAV tanto para el simétrico como para el asimétrico y uno de los métodos es el de ramificación y acotamiento. En esta sección se presenta uno de los primeros algoritmos de R-A.

Un método de ramificación y acotamiento, para un problema en particular, queda definido al especificar sus operaciones de R-A. En el caso del PAV existen tres métodos de ramificación y acotamiento:

1) Construcción del circuito según la matriz reducida.

2) Eliminación de subcircuitos a partir de la solución de problemas de asignación.

3) Construir la trayectoria basándose en el árbol de expansión mínima.

Para resolver el problema, usaremos el primer método. El funcionamiento de este método se basa en la construcción del circuito hamiltoniano según la matriz reducida.

Algoritmo Little-Murty

El propósito del algoritmo Little-Murty es determinar el circuito hamiltoniano de costo mínimo dada una matriz de costos. El método utiliza la propiedad de la matriz de costos reducida para probar la inclusión o exclusión de un arco en el circuito.

Descripción:

PASO 1: dada la matriz de costos D, se efectúan sustracciones en los renglones y en las columnas de la matriz D sin permitir que aparezcan valores negativos. Con esto, obtenemos una cota inferior del PAV al sumar los elementos que se restaron a los renglones y a las columnas. La matriz D' es la matriz reducida de D.

PASO 2: sea S un nodo del árbol y $ev(S)$ la cota inferior de este nodo S, con cada arco (i, j) con $d_{ij}' = 0$, debemos usar algún arco comenzando en i y penalizarlo por α_i. También, debemos usar algún arco en j y podríamos penalizarlo por β_j, con lo cual $\Theta_{ij} = \alpha_i + \beta_j$ que es la penalización de no escoger (i, j). Luego, se selecciona el arco que tiene el máximo de los Θ_{ij}.

PASO 3: si el arco (i, j) no se selecciona, $ev(S) + \Theta_{ij}$ es una cota inferior. Si el arco (i, j) se seleccionó, entonces, la matriz se reduce omitiendo el renglón i y la columna j. Buscar la condición adicional sobre D' para excluir subcircuitos posibles.

PASO 4: seleccione el vértice que tiene el menor costo y vaya al paso 1.

En el ejemplo siguiente se ilustra este algoritmo.

Ejemplo C.1

Sea la matriz de tiempos D que se presenta en la tabla C.1, entonces, teniendo el tiempo de duración de cada uno de los viajes entre las distintas paradas, se desea encontrar el circuito hamiltoniano de menor duración.

TABLA C.1. Distancias entre las ciudades *A* a *F*

	A	***B***	***C***	***D***	***E***	***F***
A	∞	27	43	16	30	26
B	7	∞	16	1	30	25
C	20	13	∞	35	5	0
D	21	16	25	∞	18	18
E	12	46	27	28	∞	5
F	23	5	5	9	5	∞

PASO 1

Si *T* es la duración de un recorrido hamiltoniano asociado a la matriz de tiempos de la tabla C.1, entonces, la duración de ese mismo circuito con la matriz de tiempos obtenida de restar un escalar h_i al renglón *i* está dado por $T–h_i$, porque, cada circuito contiene uno y solo un elemento de este renglón. Lo mismo sucede, si restamos un escalar h^j de una columna *j* (*j = A, B, C, D, E, F*). Una cota inferior del circuito hamiltoniano óptimo es sencilla de obtener, si restamos de cada renglón de la matriz de tiempos *D*, el mínimo elemento correspondiente como se muestra en la tabla C.2.

$$h_i = \text{mín}\ d_{ij} \geq 0$$

TABLA C.2. Con reducción en los renglones

	A	***B***	***C***	***D***	***E***	***F***
A	∞	11	27	0	14	10
B	6	∞	15	0	29	24
C	20	13	∞	35	5	0
D	5	0	9	∞	2	2
E	7	41	22	23	∞	0
F	18	0	0	4	0	∞

Entonces, en la matriz *D*', así reducida, restamos de cada columna la constante *j*:

$$h^j = \text{mín}_j\ d_{ij}^{\,1} \geq 0$$

La nueva matriz D^1 tiene elementos no-negativos y contiene, al menos, un elemento cero en cada renglón y en cada columna como se muestra en la tabla C.3.

TABLA C.3. Con distancias reducidas por renglón y columna

	A	***B***	***C***	***D***	***E***	***F***
A	∞	11	27	0	14	10
B	1	∞	15	0	29	24
C	15	13	∞	35	5	0
D	0	0	9	∞	2	2
E	2	41	22	23	∞	0
F	13	0	0	4	0	∞

Tomamos:

$$h = \sum_i h_i + \sum_j h_j$$

Entonces, la duración $z(t)$ de un circuito hamiltoniano t de la matriz D es igual a $z(t) = h + z'(t)$, donde $z'(t)$ es la duración del circuito t de la matriz D'. Ya que $z'(t)$ (porque $dij^1 \geq 0$), entonces, tenemos que $z(t) \geq h$, donde h es una evaluación por cota inferior del conjunto Ω de circuitos hamiltonianos y se denota $ev(\Omega)$.

De la matriz dada D^1, obtenemos:

$$h_1 = 16, h_2 = 1, h_3 = 0, h_4 = 16, h_5 = 5, h_6 = 5$$
$$h^1 = 5, h^2 = h^3 = h^4 = h^5 = h^6 = 0$$

Un aspecto importante es que cada circuito hamiltoniano asociado a la tabla C.3 tiene una duración no negativa y que difiere en tiempo del circuito original en 48 unidades de tiempo. Donde, una cota inferior de la duración del circuito mínimo es 48 o bien:

$$ev\,(\Omega) = \Sigma\, h_i + \Sigma\, h^j = 16 + 1 + 16 + 5 + 5 + 5 = 48$$

PASO 2

Considerando todas las entradas de la matriz de la tabla C.3 iguales a cero, calculamos los retardos o penalizaciones por no usar alguno de esos arcos que prometen un circuito de menor costo, ya que, fueron los que al restar los mínimos en los renglones y las columnas quedaron igual a cero.

Se calcula el retardo Θ_{ij} correspondiente a $d_{ij}' = 0$.

$$\Theta_{AD} = \alpha_A + \beta_D = 10 + 0 = 10$$
$$\Theta_{BD} = \alpha_B + \beta_D = 1 + 0 = 1$$
$$\Theta_{CF} = \alpha_C + \beta_F = 5 + 0 = 5$$
$$\Theta_{DA} = \alpha_D + \beta_A = 0 + 1 = 1$$
$$\Theta_{DB} = \alpha_D + \beta_B = 0 + 0 = 0$$
$$\Theta_{EF} = \alpha_E + \beta_F = 2 + 0 = 2$$
$$\Theta_{FB} = \alpha_F + \beta_B = 0 + 0 = 0$$
$$\Theta_{FC} = \alpha_F + \beta_C = 0 + 9 = 9$$
$$\Theta_{FE} = \alpha_F + \beta_E = 0 + 2 = 2$$

En general, un arco (i,j) con $d_{ij}' = 0$ no se escoge, ya que, debemos dejar el punto i, usar algún arco comenzando en i y penalizarlo por α_i. También, debemos usar algún arco en j y podríamos penalizarlo por β_j, con lo cual $\Theta_{ij} = \alpha_i + \beta_j$ que es la penalización de no escoger (i, j). Dicha penalización Θ_{ij} se puede interpretar, en este caso, como un retardo.

Si $ev\ (S)$ es la evaluación obtenida reduciendo la matriz por el nodo S, entonces, $ev\ (S) + \Theta_{ij}$ es una primera evaluación por cota inferior del nodo ij, el cual es obtenido de S por no escoger el arco (i, j).

PASO 3

Se separa la pareja (i, j) que maximiza el retardo Θ_{ij} para garantizar una mayor cota inferior de la duración de los circuitos hamiltonianos $\Theta_{AD} = 10$.

Entonces, el nodo AD tiene una evaluación por cota inferior igual a $48 + 10 = 58$.

Una manera de proceder a la determinación del circuito hamiltoniano de mínima duración consiste en dividir el conjunto de circuitos hamiltonianos como sigue:

a) Circuitos hamiltonianos que usan el arco AD
b) Circuitos hamiltonianos que no usan el arco AD

En el primer caso, la matriz de tiempos entre localidades se reduce a una nueva matriz en donde se elimina el renglón A y la columna D. Asimismo, el tiempo entre la localidad D a la A se hace igual a infinito o a un número muy grande para evitar usar el arco DA; pues, sabemos que no

forma parte del circuito hamiltoniano mínimo. Si no hacemos este tiempo infinito existe la posibilidad de la aparición de subcircuitos. La tabla C.4 muestra la matriz reducida donde se han eliminado el renglón *A* y la columna *D,* y el arco *DA* se hace igual a infinito.

TABLA C.4. Está reducida sin renglón *A* y sin columna *D*

	A	***B***	***C***	***E***	***F***
B	1	∞	15	29	24
C	15	13	∞	5	0
D	∞	0	9	2	2
E	2	41	22	∞	0
F	13	0	0	0	∞

Una cota inferior de la duración de los circuitos hamiltonianos asociados con esta tabla es sencilla de obtener, si restamos una unidad a cada elemento del renglón uno como se muestra en la tabla C.5.

TABLA C.5. Se reduce en una unidad la tabla C.4

	A	***B***	***C***	***E***	***F***
B	0	∞	14	28	23
C	15	13	∞	5	0
D	∞	0	9	2	2
E	2	41	22	∞	0
F	13	0	0	0	∞

PASO 4

Cabe hacer notar que como resultado de las manipulaciones anteriores, podemos decir que los circuitos hamiltonianos asociados a la tabla C.1, que usen el arco *AD,* tienen una duración no menor de 49 unidades de tiempo, 48 acumuladas hasta la obtención de la tabla C.3 y una unidad de tiempo al pasar de la tabla C.4 a la tabla C.5, por lo que, 49 unidades representan una cota inferior a la duración de los circuitos hamiltonianos que usan el arco *AD*.

PASO 2

El siguiente paso consiste en calcular los retardos correspondientes a la tabla C.5.

$$\Theta_{BA} = \alpha_B + \beta_A = 14 + 2 = 16$$
$$\Theta_{CF} = \alpha_C + \beta_F = 5 + 0 = 5$$
$$\Theta_{DB} = \alpha_D + \beta_B = 2 + 0 = 2$$
$$\Theta_{EF} = \alpha_E + \beta_F = 2 + 0 = 2$$
$$\Theta_{FB} = \alpha_F + \beta_B = 0 + 0 = 0$$
$$\Theta_{FC} = \alpha_F + \beta_C = 0 + 9 = 9$$
$$\Theta_{FE} = \alpha_F + \beta_E = 0 + 2 = 2$$

Se escoge el arco *BA* para efectuar la ramificación como sigue:

a) Circuitos hamiltonianos que usan el arco *BA*
b) Circuitos hamiltonianos que no usan el arco *BA*

PASO 3

En el primer caso, partimos de la tabla C.5 eliminando el renglón *B* y la columna *A* y hacemos el tiempo de *A* a *B* igual a infinito para evitar circuitos innecesarios. En este momento, como *AD* y *BA* se han seleccionado para formar el circuito hamiltoniano, hacemos que el tiempo de *DB* sea infinito. La tabla C.6 exhibe esta situación:

TABLA C.6. Está reducida sin renglón *B* y sin columna *A*

	B	***C***	***E***	***F***
C	13	∞	5	0
D	∞	9	2	2
E	41	22	∞	0
F	0	0	0	∞

Con el propósito de tener un elemento cero en cada renglón y columna, así como, mejorar la cota inferior de los circuitos hamiltonianos, procedemos a restar dos unidades del renglón dos en la tabla C.6 para obtener:

TABLA C.7. Con ceros en renglones y columnas

	B	***C***	***E***	***F***
C	13	∞	5	0
D	0	7	0	0
E	41	22	∞	0
F	0	0	0	∞

Y se puede observar que una cota inferior de los circuitos hamiltonianos que usan *BA* son 51 unidades de tiempo.

PASO 2

Nuevamente, calculamos las penalizaciones para saber qué arco es el que se va a usar:

$$\Theta_{CF} = \alpha_C + \beta_F = 5 + 0 = 5$$
$$\Theta_{DE} = \alpha_D + \beta_E = 0 + 0 = 0$$
$$\Theta_{DF} = \alpha_D + \beta_F = 0 + 0 = 0$$
$$\Theta_{EF} = \alpha_E + \beta_F = 22 + 0 = 22$$
$$\Theta_{FB} = \alpha_F + \beta_B = 0 + 13 = 13$$
$$\Theta_{FC} = \alpha_F + \beta_C = 0 + 7 = 7$$
$$\Theta_{FE} = \alpha_F + \beta_E = 0 + 0 = 0$$

PASO 3

Ya que se tienen 22 para *EF*, entonces, se incluye al arco *EF* y, si no se incluye, se tiene una penalización de 51 + 22 = 73. La tabla C.8 exhibe el resultado:

TABLA C.8. No incluye el arco *EF*

	B	***C***	***E***
C	13	∞	5
D	∞	7	0
F	0	0	∞

Nos fijamos en el primer renglón y restamos 5 para tener 0 quedando:

TABLA C.9. Con ceros en reglones y columnas

	B	***C***	***E***
C	8	∞	0
D	∞	7	0
F	0	0	∞

PASO 2

Lo que nos da un costo de 56 unidades, nuevamente, calculamos las penalizaciones:

$$\Theta_{CE} = \alpha_C + \beta_E = 8 + 0 = 8$$
$$\Theta_{DE} = \alpha_D + \beta_E = 7 + 0 = 7$$
$$\Theta_{FB} = \alpha_F + \beta_B = 0 + 8 = 8$$
$$\Theta_{FC} = \alpha_F + \beta_C = 0 + 7 = 7$$

PASO 3

De acuerdo con esto, podemos elegir como arco a usar el *CE* o el *FB*; escogemos el *FB* y obtenemos la siguiente tabla:

TABLA C.10. Se incluye el arco *FB*

	C	***E***
C	∞	0
D	7	0

Posteriormente, restamos 7 en el renglón de *D,* lo que nos da un costo de 63 unidades por usar el arco *FB* y obtenemos:

TABLA C.11. Con ceros en renglones y columnas

	C	***E***
C	∞	0
D	0	0

Donde los únicos arcos que quedan y que se pueden usar son *CE*, *DC* y *DE* de los cuales, escogemos *CE* y *DC* y asignamos infinito a *DE* para evitar posibles subcircuitos y nos queda la siguiente tabla:

TABLA C.12. Final del problema

	C	***E***
C	∞	0
D	0	∞

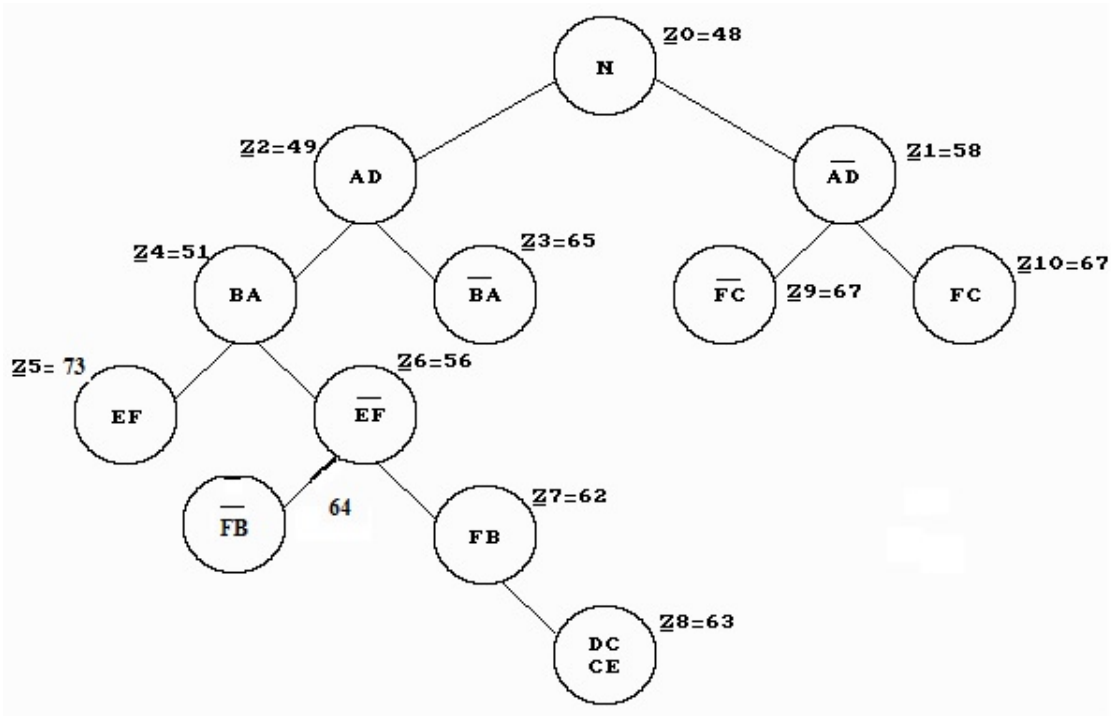


FIGURA C.1. Árbol de solución usando R-A para el PAV

De donde se concluye que el circuito hamiltoniano óptimo es *A-D-C-E-F-B-A* con una duración de 63 unidades de tiempo.

Para verificar que esta solución es óptima, debemos examinar el vértice *AD* cuya evaluación por cota inferior es 58 (< 63). La separación del vértice *AD* produce, usando *FC,* una cota de 63 y sin usar *FC,* una cota de 67, por lo cual el circuito propuesto es óptimo.

C.3. ALGORITMO BASADO EN EL PROBLEMA DE ASIGNACIÓN

El PAV se puede escribir como un modelo de asignación de la siguiente manera:

$$\text{mín } z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

sujeta a

$$\sum_{j=1}^{n} xij = 1 \quad i = 1,\ldots,n$$

$$\sum_{i=1}^{n} xij = 1 \quad j = 1,\ldots,n$$

$$xij = 0 \text{ o } 1$$

Donde $c_{ij} = \infty$ para $i = j$ y $x_{ij} = 1$, si el agente viajero va de la ciudad i a la ciudad j.

Excepto por el requerimiento de que la solución sea un circuito hamiltoniano, la formulación es un modelo de asignación. Desafortunadamente, no hay una garantía de que la solución óptima del modelo de asignación será un circuito hamiltoniano; más bien, la solución nos presentará una serie de subcircuitos a partir de los cuales podremos construir dicho circuito. Esto se logra a través de un algoritmo de R-A, aunque, note que el problema se convierte para resolver un problema de asignación en cada nodo del árbol de búsqueda.

A continuación, el algoritmo que se describe está basado en el principio general de R-A, los únicos detalles para completar el algoritmo son los siguientes:

1) Se determinan cotas superiores e inferiores en la función objetiva óptima del PAV, es decir, dado z^* el valor óptimo asociado al circuito, entonces, z_L y z_U se determinan, tales que, $z_L \leq z^* \leq z_U$.

2) Se especifica el procedimiento exacto para ramificar en cada nodo.
   Inicialmente, se hace $z_U = \infty$; sin embargo, si el circuito es factible con $c_{12} + c_{23} + \ldots + c_{n1} < \infty$, entonces, z_U es igual a este valor. El valor inicial de z_U se determina resolviendo el problema de asignación del problema original. Si z^0 es el valor óptimo de la función objetiva, entonces, $z_L = z^0$ es una cota inferior inicial.

Naturalmente, si la solución asociada a z^0 es un circuito, el algoritmo termina. De otra manera, la solución dada consiste en al menos de dos subcircuitos; se selecciona el subcircuito con el menor número de ciudades, si ese subcircuito incluye las ciudades $i_1, i_2,...,i_k$, entonces:

$$x_{i1,i2} = x_{i2i3} = ... = x_{iki1} = 1$$

La ramificación se diseña, de tal forma que, el problema de asignación asociado con los nodos subsecuentes que emanen del nodo actual se eliminarán del subcircuito; esto se puede lograr haciendo una de las variables igual a cero. El algoritmo se describe a continuación.

Algoritmo asignación

El propósito del algoritmo asignación consiste en resolver el PAV con base en el problema de asignación.

Descripción:

En la iteración r-ésima, se tiene que z^r es la función objetiva óptima, pero, como la cota inferior z^* cambia con el nodo, z^r se definirá automáticamente como z_L:

PASO 0: determine z^0. Si la solución asociada es el circuito completo, pare. De otra manera, guarde $z_U = c_{12} + c_{23} +...+ c_{n1}$ y $(1, 2,...,n,1)$ como el circuito asociado. Haga $r = 0$ y vaya al paso 1.

PASO 1: seleccione un subcircuito asociado a z^r con el menor número de ciudades e inicialice tantas ramas como número de variables x_{ij} que son igual a uno en el subcircuito. Para la rama (i,j), defina una nueva matriz de costos que difiere de la primera en que $c_{ij} = \infty$. Haga $r = r + 1$ y vaya al paso 2.

PASO 2: seleccione uno de los nodos que no se han ramificado, si no queda ninguno, deténgase el recorrido asociado con zU que es el óptimo. De otra manera, vaya al paso 3.

PASO 3: resuelva el problema de asignación asociado con el nodo seleccionado. De aquí, pueden resultar tres casos:

i) Si $z^r \geq z_U$, entonces, el nodo actual se dice que se ha examinado, ya que, no puede dar un mejor circuito que el asociado con z_U. Ahora, haga $r = r + 1$ y vaya al paso 2.

ii) Si $z^r < z_U$ y la solución asociada es un circuito, entonces, haga $z_U = z^r$ y guarde el recorrido o circuito asociado como el mejor disponible hasta este momento.

iii) Si $z^r < z_U$, pero, la solución asociada no es un recorrido, entonces, haga $r = r + 1$ y vaya al paso 1.

Ejemplo C.2

Para iniciar, considere el PAV asociado a la siguiente matriz de costos:

TABLA C.13. Matriz de costos del PAV

	A	***B***	***C***	***D***	***E***
A	∞	2	0	6	1
B	1	∞	4	4	2
C	5	3	∞	1	5
D	4	7	2	∞	1
E	2	6	3	6	∞

El valor inicial es $z_U = c_{AB} + c_{BC} + c_{CD} + c_{DE} + c_{EA} = 10$ y el circuito asociado (A,B,C,D,E,A), como se puede ver en la figura C.2.

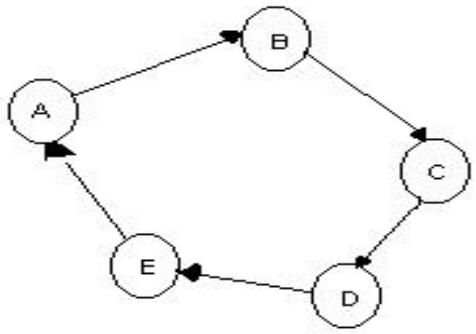


FIGURA C.2. Circuito con un valor inicial de 10

Si se resuelve el problema de asignación usando LINDO, se tiene:

```
MIN  2XAB + 6XAD + XAE +XBA + 4XBC + 4XBD + 2XBE + 5XCA + 3XCB + XCD + 5XCE + 4XDA
+ 7XDB + 2XDC + XDE + 2XEA + 6XEB + 3XEC + 6XED
ST
   XAB + XAC + XAD + XAE = 1
   XBA + XBC + XBD + XBE = 1
   XCA + XCB + XCD + XCE = 1
   XDA + XDB + XDC + XDE = 1
   XEA + XEB + XEC + XED = 1
   XBA + XCA + XDA + XEA = 1
   XAB + XCB + XDB + XEB = 1
   XAC + XCB + XDC + XEC = 1
   XAD + XBD + XCD + XED = 1
   XAE + XBE + XCE + XDE = 1
END
INT 20
```

Cuya solución está dada en el reporte siguiente:

OBJECTIVE FUNCTION VALUE

1) 8.000000

VARIABLE	VALUE	REDUCED COST
XAB	1.000000	2.000000
XAD	0.000000	6.000000
XAE	0.000000	1.000000
XBA	1.000000	1.000000
XBC	0.000000	4.000000
XBD	0.000000	4.000000
XBE	0.000000	2.000000
XCA	0.000000	5.000000
XCB	0.000000	3.000000
XCD	1.000000	1.000000
XCE	0.000000	5.000000
XDA	0.000000	4.000000
XDB	0.000000	7.000000
XDC	0.000000	2.000000
XDE	1.000000	1.000000
XEA	0.000000	2.000000
XEB	0.000000	6.000000
XEC	1.000000	3.000000
XED	0.000000	6.000000
XAC	0.000000	0.000000

Lo que da $z^0 = 8$ y la solución $x_{AB} = x_{BA} = 1$, $x_{CD} = x_{DE} = x_{EC} = 1$ que consiste en 2 subrecorridos como se puede ver en la figura C.3.

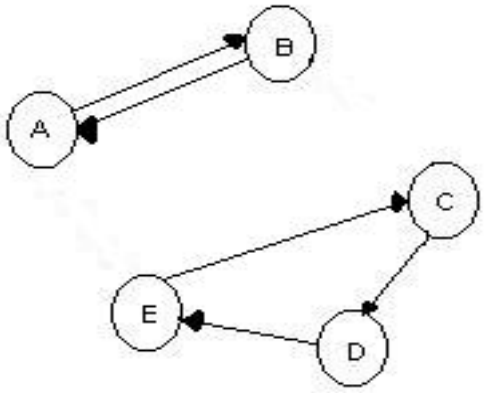


FIGURA C.3. Subcircuitos generados al resolver el PAV como problema de asignación

Ya que el recorrido (*A*, *B*, *A*) tiene el menor número de ciudades, este se usa como ramificación. Así, se crean los dos nodos correspondientes a $x_{AB} = 0$ (o el equivalentemente $c_{AB} = \infty$ en la matriz C^0) y $x_{BA} = 0$ (o el equivalentemente $c_{BA} = \infty$ en la matriz C^0).

El nodo asociado con $x_{AB} = 0$ lo seleccionamos, la asignación nos da $z^1 = 9$ con un circuito (*B*, *A*, *C*, *D*, *E*, *B*). Así, $z^U = 9$ y se guarda el circuito, esto se puede ver en la figura C.4.

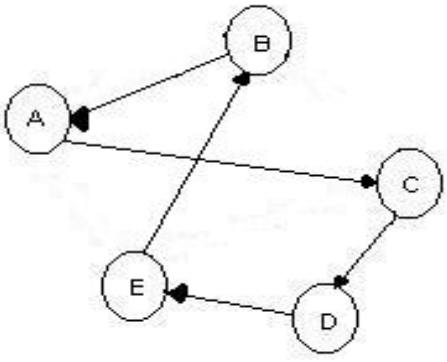


FIGURA C.4. Circuito que se forma al hacer *BA* y *AB* igual a cero

Si se resuelve el problema de asignación usando LINDO, obtenemos:

```
MIN  2XAB + 6XAD + XAE + XBA + 4XBC + 4XBD + 2XBE + 5XCA + 3XCB + XCD + 5XCE + 4XDA
+ 7XDB + 2XDC + XDE + 2XEA + 6XEB + 3XEC + 6XED
ST
   XAB + XAC + XAD + XAE = 1
   XBA + XBC + XBD + XBE = 1
   XCA + XCB + XCD + XCE = 1
   XDA + XDB + XDC + XDE = 1
   XEA + XEB + XEC + XED = 1
   XBA + XCA + XDA + XEA = 1
   XAB + XCB + XDB + XEB = 1
   XAC + XCB + XDC + XEC = 1
   XAD + XBD + XCD + XED = 1
   XAE + XBE + XCE + XDE = 1
   XAB = 0
END
INT 20
```

Y la solución:

OBJECTIVE FUNCTION VALUE

1) 9.000000

VARIABLE	VALUE	REDUCED COST
XAB	0.000000	2.000000
XAD	0.000000	6.000000
XAE	0.000000	1.000000
XBA	1.000000	1.000000
XBC	0.000000	4.000000
XBD	0.000000	4.000000
XBE	0.000000	2.000000
XCA	0.000000	5.000000
XCB	0.000000	3.000000
XCD	1.000000	1.000000
XCE	0.000000	5.000000
XDA	0.000000	4.000000
XDB	0.000000	7.000000
XDC	0.000000	2.000000
XDE	1.000000	1.000000
XEA	0.000000	2.000000
XEB	1.000000	6.000000
XEC	0.000000	3.000000
XED	0.000000	6.000000
XAC	1.000000	0.000000

El nodo remanente $x_{BA} = 0$ donde se obtiene C^2 de C^0 haciendo $c^{21} = \infty$, nos da $z^2 = 9$ como se puede ver de LINDO:

```
MIN  2XAB + 6XAD + XAE + XBA + 4XBC + 4X24 + 2XBE + 5XCA + 3XCB + XCD + 5XCE + 4XDA +
7XDB + 2XDC + XDE + 2XEA + 6XEB + 3XEC + 6XED
ST
   XAB + XAC + XAD + XAE = 1
   XBA + XBC + XBD + XBE = 1
   XCA + XCB + XCD + XCE = 1
   XDA + XDB + XDC + XDE = 1
   XEA + XEB + XEC + XED = 1
   XBA + XCA + XDA + XEA = 1
   XAB + XCB + XDB + XEB = 1
   XAC + XCB + XDC + XEC = 1
   XAD + XBD + XCD + XED = 1
   XAE + XBE + XCE + XDE = 1
   XBA = 0
END
INT 20
```

Y la solución:

OBJECTIVE FUNCTION VALUE

1) 9.000000

VARIABLE	VALUE	REDUCED COST
XAB	1.000000	2.000000
XAD	0.000000	6.000000
XAE	0.000000	1.000000
XBA	0.000000	1.000000
XBC	0.000000	4.000000
XBD	0.000000	4.000000
XBE	1.000000	2.000000
XCA	0.000000	5.000000
XCB	0.000000	3.000000
XCD	1.000000	1.000000
XCE	0.000000	5.000000
XDA	0.000000	4.000000
XDB	0.000000	7.000000
XDC	1.000000	2.000000
XDE	0.000000	1.000000
XEA	1.000000	2.000000
XEB	0.000000	6.000000
XEC	0.000000	3.000000
XED	0.000000	6.000000
XAC	0.000000	0.000000

Con dos subcircuitos (*A*, *B*, *E*, *A*) y (*C*, *D*, *C*): como $z_2 = z^U$, el nodo dos no puede producir una mejor solución y la búsqueda termina con la solución óptima (*B*, *A*, *C*, *D*, *E*, *B*) con $z^* = 9$ como se puede ver en la C.5.

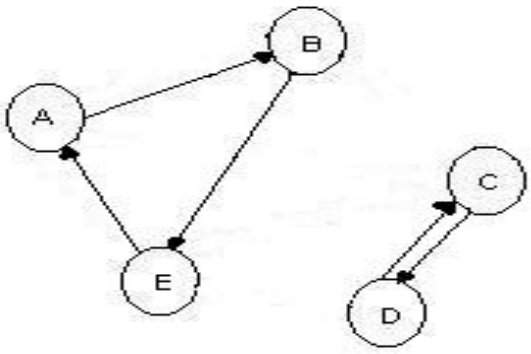


FIGURA C.5. Subcircuitos generados al usar el nodo dos

El árbol de soluciones de R-A se muestra en la figura C.6.

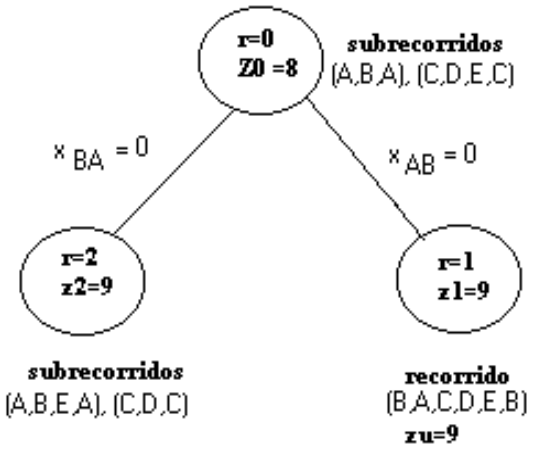


FIGURA C.6. Árbol de solución de R-A para el PAV como un problema de asignación

ANEXO D

PLANTEAMIENTO DE MODELOS DE PROGRAMACIÓN LINEAL ENTERA

D. 1. INTRODUCCIÓN

Varios de los problemas que se han resuelto usando PD (Programación Dinámica) es posible modelarlos como enteros puros o mixtos dada su estructura y restricciones. Estos modelos enteros, cuando tienen las opciones de decisión, sí o no, son enteros binarios. Las variables enteras binarias están vinculadas a los operadores lógicos de condicionamiento, de independencia, de conjunción o de disyunción y se deben cumplir bajo ciertas condiciones. Este apéndice muestra, a través de ejemplos diferentes, los casos de este tipo de modelos que servirán para tener una idea más amplia de la modelación con PLE (Programación Lineal Entera).

D.2. PROBLEMAS RESUELTOS CON PD TOMADOS DE LA PLE

Ejemplo D.1. (Presupuesto de capital)

Se están evaluando cinco proyectos durante un horizonte de planeación de tres años. La tabla D.1 muestra los ingresos esperados para cada uno y los gastos anuales correspondientes.

TABLA D.1. Ingresos y egresos anuales del problema

	Egresos ($ millones/año)			**Ingresos ($ millones/año)**
Proyecto	**1**	**2**	**3**	
1	5	1	8	20
2	4	7	10	40
3	3	9	2	20
4	7	4	1	15
5	8	6	10	30
Fondos disponibles:	25	25	25	

¿Cuáles proyectos se deben seleccionar para el horizonte de tres años?

El problema se reduce a tomar una decisión *sí* o *no* para cada proyecto. Se define la variable binaria X_j como sigue:

$$x_j = \begin{cases} 1, & \text{si se selecciona el proyecto } j \\ 0, & \text{si no se selecciona el proyecto } j \end{cases}$$

entonces, el modelo lineal entero es:

$$\text{maximizar } z = 20x_1 + 40x_2 + 20x_3 + 15x_4 + 30x_5$$

sujeta a

$$\begin{aligned} 5x_1 + 4x_2 + 3x_3 + 7x_4 + 8x_5 &\leq 25 \\ x_1 + 7x_2 + 9x_3 + 4x_4 + 6x_5 &\leq 25 \\ 8x_1 + 10x_2 + 2x_3 + x_4 + 10x_5 &\leq 25 \\ x_1, x_2, x_3, x_4, x_5 &= (0,1) \end{aligned}$$

La solución lineal entera es $x_1 = x_2 = x_3 = x_4 = 1$, $x_5 = 0$ con $z = 95$ (millones de $). Esta solución indica que se deben de seleccionar todos los proyectos, menos el 5.

La solución obtenida es interesante compararla con PLC (Programación Lineal Continua) y con PLE.

El programa lineal óptimo obtenido que reemplaza $x_j = (0,1)$ con $0 \leq x_j \leq 1$ para toda j da como resultado $x_1 = 0.5789$, $x_2 = x_3 = x_4 = 1$, $x_5 = 0.7368$ y $z = 108.66$ (millones de $). Esta solución no tiene sentido, porque, dos de las variables asumen valores fraccionarios. Se puede *redondear* la solución a los valores enteros más cercanos con lo que se obtiene $x_1 = x_5 = 1$. Sin embargo, la solución resultante no es factible, porque, se violan las restricciones. Lo más importante es que no se debe aplicar el concepto de *redondeo*, ya que, x_j representa una decisión *sí-no* para la cual no tienen sentido los valores fraccionarios.

Ejemplo D.2. (Problema de cargo fijo)

Tres empresas telefónicas pidieron que me suscribiera a su servicio de larga distancia dentro del país. MaBell cobra \$16.00 fijos por mes, más \$0.25 por minuto. PaBell cobra \$25.00 por mes, pero el costo por minuto se reduce a \$0.21. Y con BabyBell, la tarifa fija es \$18.00 mensual y la proporcional es \$0.22 por minuto. Suelo hacer un promedio de 200 minutos de llamadas de larga distancia al mes. Suponiendo que no pago el cargo fijo, si no hago llamadas y que puedo repartir a voluntad mis llamadas entre las tres empresas. ¿Cómo debo repartir las llamadas entre las tres empresas para minimizar mi recibo telefónico mensual?

Este problema se puede resolver con facilidad sin PLE. Sin embargo, es ilustrativo formularlo como programa entero.

Se definen:

x_1 = minutos de larga distancia por mes con MaBell
x_2 = minutos de larga distancia por mes con PaBell
x_3 = minutos de larga distancia por mes con BabyBell
$y_1 = 1$, si $x_1 > 0$ y 0, si $x_1 = 0$
$y_2 = 1$, si $x_2 > 0$ y 0, si $x_2 = 0$
$y_3 = 1$, si $x_3 > 0$ y 0, si $x_3 = 0$

Se puede asegurar que y_j sea igual a 1, si x_j es positiva usando la restricción:

$$X_j \leq My_j, j = 1, 2, 3$$

Se debe seleccionar el valor de M lo suficientemente grande como para no restringir en forma artificial a las variables x_j. Como yo hago aproximadamente 200 minutos de llamadas por mes, entonces, $x_j \leq 200$ para toda j y se puede seleccionar $M = 200$ con seguridad.

El modelo completo es:

$$\text{minimizar } z = 0.25x_1 + 0.21x_2 + 0.22x_3 + 16y_1 + 25y_2 + 18y_3$$

sujeta a

$$x_1 + x_2 + x_3 \geq 200$$
$$x_1 \leq 200\, y_1$$
$$x_2 \leq 200\, y_2$$
$$x_3 \leq 200\, y_3$$
$$y_1, y_2, y_3 = (0,1)$$

Esta formulación indica que la tarifa del j-ésimo mes será parte de la función objetiva z solo, si $y_1 = 1$, lo cual solo puede suceder, si $x_j > 0$ (de acuerdo con las últimas tres restricciones del modelo). Si $x_j = 0$ en el óptimo, entonces, la minimización de z, junto con el hecho de que el coeficiente objetivo de y_j es estrictamente positivo y obligará a que y_j sea igual a cero, lo cual es lo que se quería.

La solución óptima es $x_3 = 200$, $y_3 = 1$ y todas las demás variables iguales a cero y eso requiere que debo seleccionar a BabyBell para mi servicio de larga distancia. Nótese que la información que aporta $y_3 = 1$ es redundante, porque, el mismo resultado se implica en $x_3 > 0$ (= 200). En realidad, la razón principal de usar y_1, y_2 y y_3 es para tener en cuenta la tarifa fija mensual. De hecho, las tres variables binarias convierten un modelo de mal comportamiento (no lineal) en una formulación manejable, analíticamente. La conversión ha causado la introducción de las variables (binarias) enteras en un problema que hubiera sido continuo.

Ejemplo D.3. (Problema tipo mochila)

Se deben cargar cinco artículos en un barco. A continuación, se muestran en la tabla D.2 el peso w_i, el volumen v_i y el valor unitario r_i de cada artículo i.

TABLA D.2. Datos para el problema de la mochila

Artículo i	**Peso unitario** w_i **(tonelada)**	**Volumen unitario** v_i **(yd^3)**	**Valor unitario** r_i **($100)**
1	5	1	4
2	8	8	7
3	3	6	6
4	2	5	5
5	7	4	4

El peso w y el volumen máximo de la carga v son 112 toneladas y 109 yardas cúbicas, respectivamente. Formule el modelo de PLE.

Sea x_i = número de unidades del artículo i, i = 1, 2,…,5, entonces, se tiene:

$$\text{máx } z = 4x_1 + 7x_2 + 6x_3 + 5x_4 + 4x_5$$

sujeta a

$$5x_1 + 8x_2 + 3x_3 + 2x_4 + 7x_5 \leq 112$$
$$x_1 + 8x_2 + 6x_3 + 5x_4 + 4x_5 \leq 109$$
$$x_i \geq 0 \text{ enteras}$$

Ejemplo D.4. (Optimización de costos)

Jacobo planea producir al menos 2 000 piezas en tres máquinas. El tamaño mínimo del lote en cualquier máquina es de 500 piezas. La siguiente tabla contiene los datos pertinentes del caso.

TABLA D.3. Datos del problema de costos

Máquina	**Costo de preparación**	**Costo de producción/unidad**	**Capacidad (unidades)**
1	300	2	600
2	100	10	800
3	200	5	1 200

Jacobo quiere que dicha producción sea al mínimo costo, considerando las capacidades, así como, la producción mínima total.

Sea x_j = número de piezas producidas en la máquina j, j = 1, 2, 3. Y sean y_j variables binarias asociadas a la decisión de usar o no la máquina, entonces, se tiene:

$$\text{mín } z = 2\,x_1 + 10\,x_2 + 5\,x_3 + 300\,y_1 + 100\,y_2 + 200\,y_3$$

sujeta a

$$\begin{aligned} & x_1 + x_2 + x_3 \geq 2\,000 \\ & x_1 - 600\,y_1 \leq 0 \\ & x_2 - 800\,y_2 \leq 0 \\ & x_3 - 1\,200\,y_3 \leq 0 \\ & x_1, x_2, x_3 \geq 500 \qquad \text{enteras} \\ & y_i = (0, 1) \end{aligned}$$

Ejemplo D.5. (Producción de películas)

Un importante estudio de cine planea producir cinco películas durante los próximos tres años. Se define una variable y_{it} donde el subíndice i se refiere a la película en particular (i = 1, 2, 3, 4, 5) y el subíndice t al año (t = 1, 2, 3). y_{it} es una variable cero/uno que tiene el valor de 1, si la i-ésima película se produce en el t-ésimo año y tiene un valor 0 de otro modo. Considere

cada una de las siguientes situaciones de manera independiente y formule una o más restricciones lineales con variables enteras que satisfagan la condición establecida.

a) No puede producirse más que una película en el primer año.
b) La película 2 no puede producirse antes que la película 3; sin embargo, pueden producirse en el mismo año.
c) Se debe producir, por lo menos, una película cada año.
d) La película 4 debe producirse a más tardar en el segundo año.
e) Las películas 1 y 5 no pueden producirse en el mismo año.

Solución:

a) $y_{11} + y_{21} + y_{31} + y_{41} + y_{51} \leq 1$
b) $y_{21} \leq y_{31}$; $y_{22} \leq y_{31} + y_{32}$; $y_{23} \leq y_{31} + y_{32} + y_{33}$
c) Se debe cumplir que:

$$\begin{aligned} y_{11} + y_{21} + y_{31} + y_{41} + y_{51} &\geq 1 \\ y_{12} + y_{22} + y_{32} + y_{42} + y_{52} &\geq 1 \\ y_{13} + y_{23} + y_{33} + y_{43} + y_{53} &\geq 1 \end{aligned}$$

d) Equivale a: $y_{41} + y_{42} = 1$
e) Equivale a: $y_{11} + y_{51} \leq 1$; $y_{12} + y_{52} \leq 1$; $y_{13} + y_{53} \leq 1$

Ejemplo D.6. (Un problema de inversión)

La gerente de investigación del Consejo de Ciencia y Tecnología está tratando de decidir cuáles proyectos se deberán financiar para el próximo año; ella recibió ocho propuestas que se presentan en la tabla D.4. Después de un estudio minucioso, hizo un cálculo estimado del valor de cada proyecto en una escala de 0 a 100. La gerente de investigación desea encontrar una combinación de los proyectos que tenga el valor total más alto; sin embargo, existen varias limitaciones. Primera, cuenta únicamente con un presupuesto de 320 000.00 UM (Unidad Monetaria). Segunda, ella debe aceptar o descartar un proyecto (es decir, no hay inversión parcial). Tercera, hay ciertos proyectos relacionados, específicamente, los proyectos *G* y *H* en los cuales, como máximo, se invertiría en uno. El proyecto *D* no debe recibir recursos, a menos que, el proyecto *A* también lo haga (no obstante, el proyecto *A* puede ser apoyado sin el proyecto *D*).

Ahora, formule el problema de la gerente de investigación como un problema de PLE.

TABLA D.4. Propuestas para inversión

Propuestas	**Proyectos**	**Costo**	**Valor**
1	*A*	80	40
2	*B*	15	10
3	*C*	120	80
4	*D*	65	50
5	*E*	20	20
6	*F*	10	5
7	*G*	60	80
8	*H*	100	100

Solución:

Se definen las variables de decisión binaria ($i = 1, \ldots, 8$):

$$x_i = \begin{cases} 1, & \text{si se realiza el proyecto } i \\ 0, & \text{otro caso} \end{cases}$$

Entonces, el problema consiste en:

$$\text{máx } z = 40x_1 + 10x_2 + 80x_3 + 50x_4 + 20x_5 + 5x_6 + 80x_7 + 10x_8$$

sujeta a

Presupuesto: $80x_1 + 15x_2 + 120x_3 + 65x_4 + 20x_5 + 10x_6 + 60x_7 + 100x_8 \leq 320$

G y *H* no simultáneos: $x_7 + x_8 \leq 1$

D, a menos que, se incluya *A*: $x_4 \leq x_1$

Ejemplo D.7. (Mezcla de producto)

La empresa Bendigo fabrica tres productos que son ANZA, BOZO y KARMA. Bendigo formuló, parcialmente, su problema semanal de mezcla del producto como un problema de PLE de la siguiente manera:

$$\text{máx } z = 100A + 120B$$

sujeta a

$3A + 7B + 2K \leq 1\,000$	Tiempo de máquina disponible
$A + 1.5B + 2K \leq 300$	Horas de trabajo disponibles
$23A + 18B + 25K = \text{MPU}$	(Materia Prima Utilizada)

Donde *A, B* y *K* se refieren a la cantidad de unidades ANZA, BOZO y KARMA producidas por semana, respectivamente; en este punto, la empresa Bendigo tuvo problemas y pide ayuda para completar la formulación.

Para cada una de las siguientes preguntas, adicione o modifique el planteamiento para incorporar la situación descrita. Formule el problema como uno de PLE, utilizando variables con valores enteros (incluyendo binarias), según se requiera. Defina, en cada caso, las nuevas variables, las restricciones o las nuevas modificaciones y las adiciones o modificaciones que se hagan a la función objetiva. Cada parte debe tratarse de manera independiente.

a) Los productos KARMA se estampan en una máquina especial que tiene en arriendo el fabricante; pueden estamparse hasta 200 unidades en una semana. El contrato de arrendamiento especifica un costo fijo de 800.00 UM por semana, si se produce alguna unidad de KARMA; sin embargo, no existe cargo alguno, si la máquina no se utiliza durante alguna semana.

b) El costo de la materia prima es 0.25 UM por unidad, si se compran menos de 1 000 unidades. Si se compra esa cantidad o más, el costo es 0.20 UM por unidad para todas las unidades adquiridas. Semanalmente, se dispone de una cantidad ilimitada de materia prima que no puede almacenarse de una semana a la siguiente. Modifique la correspondiente restricción.

c) El ingreso por ventas para una unidad de KARMA depende de la cantidad comprada. Bendigo recibe 100.00 UM por cada una de las primeras 20 unidades producidas, 150.00 UM por cada una de las siguientes 50 unidades producidas y 120.00 UM por cada unidad adicional hasta un máximo de 200 unidades producidas en total.

d) El departamento de ventas calcula que las ventas máximas son 100 unidades para los ANZA y 120 unidades para los BOZO. Sin embargo, podría iniciarse una campaña de ventas a un costo de 5 000.00 UM para cada producto. La campaña incrementaría la demanda en 50 unidades para el producto seleccionado. Debido a la limitación en las ventas, la campaña podría realizarse para el producto ANZA o BOZO, pero, no para ambos.

Solución:

a) Sea y_1 variable binaria con valor 1, si se usa la máquina de estampado y 0 de otro modo. En las restricciones, se añade $K \leq 200 y_1$. En la función objetiva se agrega $-800 y_1$.

b) Sea y_2 variable binaria con valor 1, si se compran 1 000 unidades o más y 0 de otro modo. Sea R_1 la cantidad de materia prima comprada a un precio de 0.25 UM. Sea R_2 la cantidad de materia prima comprada a 0.20 UM. En las restricciones se añaden:

$$\text{MPU} \leq R_1 + R_2 \quad ; \quad R_2 \leq M y_2$$

Donde M es un escalar positivo y grande. En la función objetiva se agrega:

$$-0.25 R_1 - 0.20 R_2$$

Nota: la restricción usual $R_1 \leq 1\,000\,(1 - y_2)$ puede añadirse, pero, no es necesaria.

c) Sea K_1 el número de unidades KARMA vendidas en 100.00 UM, K_2 el número vendido en 150.00 UM y K_3 el número vendido en 120.00 UM. Sea z_1 una variable binaria con valor de 1, si se producen 20 unidades o menos; z_2, también, es una variable binaria con valor de 1, si la producción está entre 20 y 70 unidades; y z_3 es una variable binaria con valor de 1, si la producción supera las 70 unidades.

Se agregan las siguientes restricciones:

$$K = K_1 + K_2 + K_3; \qquad K_1 \leq 20 z_1; \qquad K_2 \leq 50 z_2; \qquad K_3 \leq 130 z_3$$
$$K_1 \geq 20 z_2; \qquad K_2 \geq 50 z_3; \qquad z_1 \geq z_2; \qquad z_2 \geq z_3$$

Finalmente, se añade el término 100 K_1 + 150 K_2 + 120 K_3 a la función objetiva.

d) Sea w_1 una variable binaria que toma el valor de 1, si se realiza la campaña para los ANZA y 0 de otro modo; de manera similar, w_2 = 1, si se realiza la campaña para los BOZO; entonces, se adicionan las siguientes restricciones:

$$A \leq 100 + 50w_1$$
$$B \leq 120 + 50w_2$$
$$w_1 + w_2 \leq 1$$

Se añade a la función objetiva: $-5\,000w_1 - 5\,000w_2$.

Ejemplo D.8. (Asignación de personal)

El jefe del departamento de policía de Jilotepec prepara el presupuesto para el año siguiente. Un elemento clave es el número de policías que necesitará para labores de patrullaje y, justamente, acaba de recibir un estudio que se muestra en la tabla D.5 que calcula el número de patrullas que debe haber en las calles de Jilotepec durante periodos de cuatro horas.

TABLA D.5. Requisitos estimados de patrullas

Periodo (horas)	**Patrullas necesarias**
22:00 – 2:00	13
2:00 – 6:00	1
6:00 – 10:00	7
10:00 – 14:00	6
14:00 – 18:00	6
18:00 – 22:00	17
Total:	50

En parte, estos cálculos se basaron en experiencias anteriores respecto a las solicitudes de ayuda y a la investigación de los crímenes y, hasta cierto punto, en requisitos para un nuevo programa de prevención del crimen. El nuevo programa enfatiza en la visibilidad de las patrullas en las calles durante horas de alto potencial de acciones criminales.

El problema del jefe surge porque tiene que programar al personal de la policía en turnos de ocho horas y no solo por segmentos de cuatro horas como se muestra en la tabla D.5. El

departamento opera con seis turnos escalonados de ocho horas que se muestran en la tabla D.6. El jefe desea mantener esta política.

TABLA D.6. Turnos escalonados

Turno de inicio	Hora de inicio	Hora de terminación
1	22:00	6:00
2	2:00	10:00
3	6:00	14:00
4	10:00	18:00
5	14:00	10:00
6	18:00	2:00

El dilema del jefe es decidir el número mínimo del personal necesario para cumplir los requisitos. Cada patrulla cuenta con un solo patrullero. Únicamente, considere el problema para los días laborables. Formule un problema de PLE para resolver el problema.

Solución:

Sea x_t el número de personas programadas para el *t*-ésimo cambio (*t* = 1, 2,..., 6) variables de decisión con valores enteros. Las restricciones del problema son:

Periodo	Necesidades que se deben cumplir
22:00 – 2:00	$x_6 + x_1 \geq 13$
2:00 – 6:00	$x_1 + x_2 \geq 1$
6:00 – 10:00	$x_2 + x_3 \geq 7$
10:00 – 14:00	$x_3 + x_4 \geq 6$
14:00 – 18:00	$x_4 + x_5 \geq 6$
18:00 – 22:00	$x_5 + x_6 \geq 17$

Y la función objetiva es:

$$\text{mín } z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6$$

Ejemplo D.9. (Problema de distribución)

La empresa Arena y Grava debe repartir cierta cantidad de materiales entre seis clientes como se muestra en la tabla D.7.

TABLA D.7. Datos del problema

Cliente	**Cantidad (toneladas)**
A	¼
B	½
C	1½
D	½
E	¾
F	1

Para hacer estas entregas, hay cinco camiones que pueden utilizarse. Con el uso de divisiones, un camión puede entregar una mezcla de cargas hasta llenar su capacidad. Sin embargo, la orden de un cliente no puede distribuirse en distintos camiones. Los camiones disponibles y sus capacidades se presentan en la tabla D.8.

TABLA D.8. Camiones disponibles y sus capacidades

Camión	**Cantidad (toneladas)**
1	1
2	2
3	1
4	2
5	1½

Los costos asociados de cada camión que haga la entrega de pedidos a los clientes, se muestran en la tabla D.9.

TABLA D.9. Costos de entrega de pedidos a clientes (UM)

Camión	**Cliente**					
	A	***B***	***C***	***D***	***E***	***F***
1	17	19	21	20	20	21
2	15	18	20	18	19	23
3	18	19	22	22	21	22
4	15	16	19	18	18	20
5	16	15	20	22	19	20

Además de estos costos, existe un cargo fijo de 10.00 UM en los camiones 1, 3 y 5 y de 15.00 UM en los camiones 2 y 4. Se incurre en estos costos, si el camión se usa con carga completa. Finalmente, existe la restricción de que en el mismo camión no pueden entregarse los pedidos a los clientes *A* y *B*. Formule este problema como uno de PLE.

Solución:

Sea y_{ij} (variable cero/uno) con valor uno, si el i-ésimo camión (i = 1,2,..,5) se utiliza para hacer entregas al j-ésimo cliente (j = 1(*A*), 2(*B*),…,6(*F*)) y cero en caso contrario. Sea u_i **(**variable cero/uno) con valor uno, si el i-ésimo camión usa parte o toda su capacidad y tiene valor cero, si no es usado.

Sea c_{ij} el costo de entrega de pedidos a clientes. Sea c_i el costo fijo por utilizar el camión i y k_i su capacidad. Por último, sea a_j el tamaño de la carga del j-ésimo cliente (en toneladas). Los valores numéricos para los c_{ij}, c_i, k_i y a_j se dan en el problema. El problema consiste en:

$$\text{minimizar } \sum_{i=1}^{5}\sum_{j=1}^{6} c_{ij}y_{ij} + \sum_{i=1}^{5} c_i u_i$$

sujeta a

$\sum_{i=1}^{5} y_{ij} = 1$; para toda j (j = 1, 2,…,6)	entregas
$\sum_{j=1}^{6} y_{ij}a_j \le k_i$; para toda i (i = 1, 2,…,5)	capacidades de los camiones
$\sum_{j=1}^{6} y_{ij} \le Mu_i$; para toda i (i = 1, 2,…,5) (M = 6)	uso de camiones
$y_{i1} + y_{i2} \le 1$; para toda i (i = 1, 2,…,5)	no entrega común a *A* y *B*

Ejemplo D.10. (Otro problema de asignación)

Pizzas León opera dos cadenas de tiendas de alimentos listos para consumir: la cadena Piazo Pizza y la cadena Fisher Kola. La empresa está expandiendo sus operaciones en el condado de Santa Clara y busca sitios para abrir nuevas tiendas. Ya se identificaron diez sitios potenciales y se calculó el ingreso neto (en términos del valor presente) de cada uno para utilizarlos como una sede de Piazo Pizza o como un restaurante de Fisher Kola. Sea P_j (j = 1,2,...,10) el

ingreso neto para utilizar el sitio *j* para la pizzería y F_j (j = 1, 2, ...,10) el ingreso neto del mismo sitio, si fuera utilizado como *p*.

La empresa opera ambas cadenas de restaurantes mediante franquicias en una región determinada. De acuerdo con los términos del contrato, con cualquier franquicia de Pizza, los demás restaurantes tienen derechos exclusivos en un radio de 2 millas a la redonda. Esto significa que cualquier sitio seleccionado para las pizzerías debe quedar, por lo menos, a 4 millas de distancia. Existe un acuerdo similar en los contratos con las franquicias de Fisher Kola, pero, los derechos exclusivos están garantizados dentro de 2.5 millas, es decir, que los restaurantes deben quedar, por lo menos, a 5 millas de distancia. No existe restricción sobre las localidades entre las dos cadenas (un local de Pizza podría quedar al lado de un restaurante Fisher). La tabla D.10 muestra las distancias entre los sitios.

TABLA D.10. Distancia entre los sitios potenciales para los restaurantes (en millas)

Desde	**Hasta**								
	2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	3.7	2.2	3.2	6.4	7.3	8.6	8.8	13.0	10.2
2		2.6	5.7	3.3	6.6	9.6	10.3	13.1	7.5
3			3.2	4.5	5.3	7.4	7.7	11.4	8.2
4				7.5	5.7	5.6	5.7	10.4	10.3
5					5.2	9.4	10.1	11.5	4.4
6						4.8	5.6	6.6	5.5
7							1.2	5.2	10.2
8								5.9	11.4
9									10.2

Cada uno de los 10 sitios potenciales puede constituirse como pizzería o restaurante, pero, no ambos. Se desea maximizar el rendimiento neto de Pizzas León.

Solución:

Primero, empezaremos por definir las variables binarias (j = 1,..., 10):

$$x_j = \begin{cases} 1, & \text{si el sitio } j \text{ se usa como una pizzería} \\ 0, & \text{si no se usa} \end{cases}$$

$$y_j = \begin{cases} 1, & \text{si el sitio } j \text{ se usa como una pescadería} \\ 0, & \text{si no se usa} \end{cases}$$

Si en la tabla de distancias entre dos sitios i, j, que se denotan por d_{ij} se tiene $d_{ij} \leq 4$, entonces, $x_i + x_j \leq 1$ (dos pizzerías deben de quedar por lo menos a 4 millas de distancia). En términos de los datos proporcionados:

$$x_1 + x_2 \leq 1$$
$$x_2 + x_3 \leq 1$$
$$x_7 + x_8 \leq 1$$
$$x_1 + x_3 \leq 1$$
$$x_2 + x_5 \leq 1$$
$$x_1 + x_4 \leq 1$$
$$x_3 + x_4 \leq 1$$

Un análisis semejante para la cadena Fisher Kola establece que, si $d_{ij} \leq 5$, se tiene $y_i + y_j \leq 1$ (dos negocios deben de quedar, por lo menos, a 5 millas de distancia):

$$y_1 + y_2 \leq 1$$
$$y_2 + y_3 \leq 1$$
$$y_5 + y_{10} \leq 1$$
$$y_3 + y_5 \leq 1$$
$$y_1 + y_3 \leq 1$$
$$y_2 + y_5 \leq 1$$
$$y_6 + y_7 \leq 1$$
$$y_1 + y_4 \leq 1$$
$$y_3 + y_4 \leq 1$$
$$y_7 + y_8 \leq 1$$

Por otra parte, para toda localización j (j = 1,..., 10) se cumple que:

$$x_j + y_j \leq 1$$

Finalmente, la función objetiva es:

$$\text{máx } Z = \sum_{j=1}^{10} P_j x_j + \sum_{j=1}^{10} F_j y_j$$

Ejemplo D.11. (Planeación de la producción)

Cierta línea de producción fabrica dos productos. Los datos pertinentes para la planeación de la producción aparecen en la tabla D.11.

TABLA D.11. Datos para la planeación de la producción

Concepto	**Producto**	
	A	***B***
Tiempo de arranque	5 horas	10 horas
Tiempo de producción por unidad	0.5 horas	0.75 horas
Costo de arranque	200.00 UM	400.00 UM
Costo de producción por unidad	10.00 UM	15.00 UM
Precio de venta	20.00 UM	30.00 UM

El tiempo total disponible (para la producción y la puesta en marcha) cada semana es de 80 horas. La empresa no tiene inventario de producto alguno al principio de la semana 1 y no se permite que lo tenga al final de la semana 4. El costo por almacenar una unidad de una semana a la siguiente es de 4.00 UM por producto. Una unidad de demanda no satisfecha cuesta 10.00 UM para el producto *A* y 15.00 UM para el producto *B*. Los datos sobre la demanda aparecen en la tabla D.12.

Tabla D.12. Datos sobre la demanda de los productos

Producto	**Semana**			
	1	**2**	**3**	**4**
A	80	100	75	80
B	15	20	50	30

La línea se cierra cada fin de semana para realizar las operaciones de limpieza. Por tanto, si un producto es fabricado en una semana, tendría que pagarse el costo y tiempo de arranque del

equipo. Solo un tipo de producto puede fabricarse durante la semana. No puede haber producción durante el tiempo en que se pone en marcha la línea. Formule y resuelva este modelo de planeación de 4 semanas como una PLEM (Programación Lineal Entera Mixta). El objetivo es maximizar las utilidades obtenidas durante estas 4 semanas.

Solución:

Primero, comenzaremos por entender la problemática. Se observa que existe una máquina que durante cada semana trabaja en la producción de un artículo *A* o de un artículo *B*, pero, no ambos. Existen tiempos y costos de arranque conocidos, así como, tiempos y costos unitarios de producción. Una manera de visualizar la problemática se muestra en la figura D.1 anexa, usando el tradicional formato producción inventario/déficit. Note que la demanda no satisfecha en un periodo se pierde. Con base en estos diagramas, definiremos las variables de decisión:

$yA_i\,(yB_i)=$ variable binaria 0/1. yA_i (yB_i) es uno, si la máquina es usada en la producción del artículo *A* en la semana *i*; es cero, si no es usada.

$xA_i\,(xB_i)=$ es la producción del *A*(*B*) en la semana *i*.

$IA_i\,(IB_i)=$ es el número de artículos *A*(*B*) en inventario durante la semana *i*.

$DFA_i\,(DFB_i)=$ es el número de artículos *A*(*B*) que no se vendieron o en déficit durante la semana *i*.

$DEMA_i\,(DEMB_i)=$ es la demanda del artículo *A*(*B*) en la semana *i*.

El modelo de PLEM consiste en maximizar la utilidad neta expresada como:

$$\text{U.N.}=\sum_{i=1}^{4}\left[(20-10)xA_i+(30-15)xB_i\right]-\sum_{i=1}^{4}\left[200yA_i+400yB_i\right]-\sum_{i=1}^{4}\left[4IA_i+4IB_i\right]-\sum_{i=1}^{4}\left[10DFA_i+15DFB_i\right]$$

sujeta a

$yA_i+yB_i=1$	uso excluyente de la máquina
$5yA_i+6.5xA_i\le 80yA_i$	producción *A*
$10yB_i+0.75xB_i\le 80yB_i$	producción *B*

Ecuaciones de balance:

$$xA_i + IA_{i-1} - IA_i + DFA_i = DEMA_i$$
$$IA_0 = IB_0 = 0$$
$$xB_i + IB_{i-1} - IB_i + DFB_i = DEMB_i \qquad (i = 1, 2, 3, 4)$$

donde xA_i, xB_i, IA_i, IB_i, DFA_i y DFB_i son variables no negativas, mientras que, yA_i y yB_i son variables enteras 0/1.

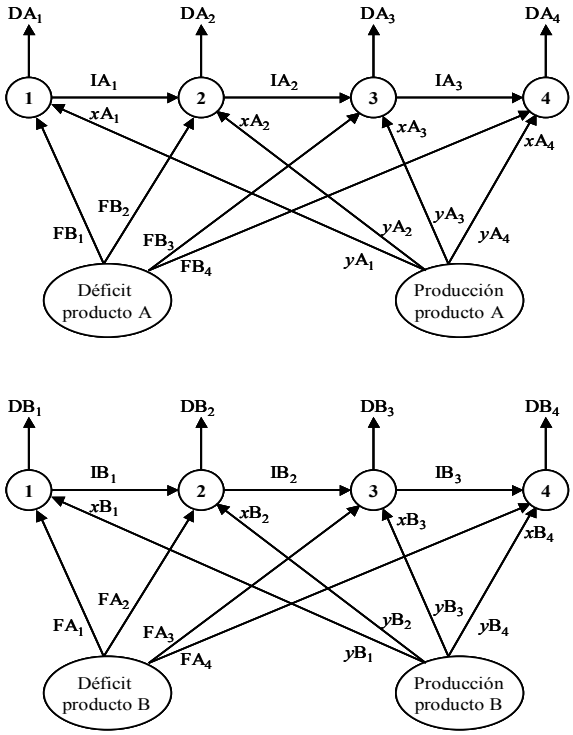


FIGURA D.1. Plan de producción

Ejemplo D.12. (Expansión de capacidad)

Una compañía de servicio eléctrico está planeando ampliar su capacidad de generación durante los próximos cinco años. La capacidad actual es 800 MW (*megawatts*), pero, de acuerdo con sus propios pronósticos de demanda va a requerir la capacidad adicional como se muestra en la tabla D.13.

TABLA D.13. Capacidad de generación por año

Año	Capacidad mínima MW
1	880
2	960
3	1 050
4	1 160
5	1 280

La compañía de servicio eléctrico podrá aumentar su capacidad de generación con la instalación de unidades generadoras de 10, 50 o 100 MW. El costo de instalación de un generador depende de su tamaño y del año en que entre en servicio. Los datos relevantes están en la tabla D.14.

TABLA D.14. Costos de instalación para el servicio eléctrico

Capacidad del generador MW	Año				
	1	2	3	4	5
10	300	250	208	173	145
50	670	558	465	387	322
100	950	791	659	549	458

Una vez que el generador entra en servicio, su capacidad está disponible para satisfacer la demanda en los años subsiguientes. Formule y resuelva una PLE que minimice el costo de poner en servicio a los generadores, satisfaciendo, al mismo tiempo, los requisitos de la capacidad mínima. Sugerencia: sean x_t, y_t y z_t las cantidades respectivas de los generadores de 10, 50 y 100 MW puestos en servicio en el año t y sea c_t la capacidad total en el año t cuando esos generadores hayan entrado en servicio.

Solución:

El problema consiste en:

$$\text{mín } Z = 300x_1 + 670y_1 + 950z_1 + \ldots + 145x_5 + 322y_5 + 458z_5$$

sujeta a

$$C_1 = 800 + 10x_1 + 50y_1 + 100z_1$$
$$C_t = C_{t\text{-}1} + 10x_t + 50y_t + 100z_t \quad ; \qquad t = 2, \ldots, 5$$
$$C_1 \geq 880 \; ; \; C_2 \geq 960 \; ; \ldots ; \; C_5 \geq 1\,280$$

En donde x_t, y_t y z_t son variables enteras no negativas (t = 1, 2, . . . , 5).

La solución óptima al problema es Z^* = 3 407

$$x_t = y_t = 0 \; ; \; z_t = 1, \; t = 1, 2, \ldots, 5.$$

BIBLIOGRAFÍA

Aho, Alfred V., *et al*., *Estructura de datos y algoritmos*, México, Addison-Wesley Iberoamericana, 1988.

Ahuja, Ravindra K., *et al*., *Network Flows*: *Theory, Algorithms and Applications*, Englewood Cliffs, Prentice Hall, 1993.

Bellman, Richard E., *Dynamic Programming*, Princeton, Princeton University Press, 1957.

Bertsekas, D. P., *Dynamic Programming*: *Deterministic and Stochastic Models*, Englewood Cliffs, Prentice Hall, 1987.

Cerdá, Emilio, *Optimización dinámica*, Madrid, Prentice Hall, 2001.

Cooper, L., and Mary W. Cooper, *Introduction to Dynamic Programming*, New York, Pergamon Press, 1981.

Denardo, E. V., *Dynamic Programming*: *Models and Applications*, Englewood Cliffs, Prentice Hall, 1982.

Dreyfus, Stuart E., and Averill M. Law, *Art and Theory of Dynamic Programming*, vol. 130, Orlando, Academic Press, 1977.

Gondran, M., and Michel Minoux, *Graphs and Algorithms*, Chichester, John Wiley & Sons, 1984.

Grossman, S. I., *Aplicaciones de algebra lineal*, México, Iberoamérica, 1988.

Horowitz, E., and Sartaj Sahni, *Fundamentals of Computer Algorithms*, Rockville, Computer Science Press, 1984.

Lawler, E. L., *et al*., *The Traveling Salesman Problem*. *A Guided Tour of Combinatorial Optimization*, vol. 34, John Wiley & Sons, 1985.

Martello, S., and P. Toth, "Algorithm 37. Algorithm for the Solution of 0-1 Single Knapsack Problem", *Computing*, vol. 21, Springer-Verlag, 1978.

McGuire, M., *An eye for Fractals*, Redwood City, Addison-Wesley, 1991.

Peitgen, Heinz-Otto, *et al*., *Fractals for the Classroom*, New York, Springer Science & Business Media LLC, 1992.

Roberts, E., *Thinking Recursively*, New York, John Wiley & Sons, 1986.

Salkin, H. M., *Integer Programming,* Reading, Addison-Wesley, 1974.

Taha, Hamdy A., *Integer Programming*: *Theory, Applications and Computations*, New York, Academic Press, 1975.

——*Investigación de operaciones,* 7.ª ed., México, Pearson, 2004.

——*Investigación de operaciones*, trad. José de la Cera Alonso, 5.ª ed., Mexico, Alfaomega, 1995.

Torres, C., y Howard Anto, *Aplicaciones de álgebra lineal*, México, Limusa, 1979.

Winston, Wayne L., *Investigación de operaciones*. *Aplicaciones y algoritmos*, 4.ª ed., México, Thomson, 2005.

Wirth, N., *Algoritmos y estructura de datos*, México, Prentice Hall, 1987.

MESOGRAFÍA

http://www.stephen.com/mondrimat/

http:http://dubrieldice.blogspot.mx/2010/10/fractales.html

http://www.ite.educacion.es/formacion/enred/web_espiral/general_1/naturaleza_1/vegetal_1/vegetal.htm

http://moirajohnson.blog.com/2013/06/24/nautilus/

http://wordaligned.org/articles/recursive-pictures

http://en.wikipedia.org/wiki/Tower_of_Hanoi

http://es.wikipedia.org/wiki/N%C3%BAmero_primo_de_Mersenne

http://es.wikipedia.org/wiki/Funci%C3%B3n_de_Ackermann

http://www.buenastareas.com/ensayos/Zen%C3%B3n-De-Elea-y-Meliso-De/3219782.html

http://es.wikipedia.org/wiki/Piet_Mondrian

http://es.wikipedia.org/wiki/M._C._Escher

http://es.wikipedia.org/wiki/%C3%89douard_Lucas

http://es.wikipedia.org/wiki/Marin_Mersenne

http://es.wikipedia.org/wiki/Leonardo_de_Pisa

http://es.wikipedia.org/wiki/Wilhelm_Ackermann

http://es.wikipedia.org/wiki/Richard_Bellman

http://es.wikipedia.org/wiki/Edsger_Dijkstra

http://searches.uninstallmaster.com/search/web?channel=unknown&type=other&q=Example+of+Dynamic+Programming+Algorithm+for+the+TSP

http://www.mafy.lut.fi/study/DiscreteOpt/tspdp.pdf

http://es.wikipedia.org/wiki/Problema_de_la_mochila

http://es.wikipedia.org/wiki/Richard_Karp

http://es.wikipedia.org/wiki/Andr%C3%A9i_M%C3%A1rkov

http://es.wikipedia.org/wiki/Cantidad_Econ%C3%B3mica_de_Pedido

http://money.howstuffworks.com/how-inventory-management-systems-work1.htm

http://www.gestiondeoperaciones.net/programacion_lineal/problema-de-produccion-e-inventario-resuelto-con-solver-de-excel/

http://en.wikipedia.org/wiki/Stochastic_programming

http://es.wikipedia.org/wiki/Teor%C3%ADa_moderna_del_portafolio

http://es.wikipedia.org/wiki/Harry_Markowitz

http://natureofcode.com/book/chapter-8-fractals/

http://fractalfoundation.org/resources/what-is-chaos-theory/

http://www.abarim-publications.com/ChaosTheoryIntroduction.html#.VGt4VvmG-So

http://math.bu.edu/DYSYS/chaos-game/node1.html

Programación dinámica, se publicó de manera
digital el 14 de febrero de 2022 en la página
de la Facultad de Ingeniería, Ciudad Universitaria,
México, Ciudad de México. C.P. 04510