ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

Grado en Ingeniería Informática en Tecnologías de la Información

ALGORITMIA

Tema 6:

Esquema Voraz
(Problemas de Planificación de Tareas)

Curso 2021-2022

Un único servidor, como por ejemplo, un procesador, un surtidor de gasolina o un cajero de un banco, tiene que atender n clientes que llegan todos juntos al sistema. El tiempo de servicio para cada cliente se conoce de antemano: el cliente requerirá un tiempo t_i con $1 \leq i \leq n$.

El problema de minimización del tiempo de espera consiste en minimizar el tiempo medio invertido por cada cliente en el sistema.

SOLUCIÓN.-

a) El problema de minimización del tiempo de espera es un problema de optimización, consiste en minimizar el tiempo medio invertido por cada cliente del sistema. Dado que el número de clientes está predeterminado, esto equivale a minimizar el tiempo total invertido en el sistema por todos los clientes, es decir, deseamos minimizar

$$\boldsymbol{T} = \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} (tiempo\ total\ que\ el\ cliente\ i\ está\ en\ el\ sistema)$$

b) La solución se puede expresar como una secuencia de decisiones $<x_1,x_2,\ldots,x_n>$ donde x_1 es el cliente que se atenderá en primer lugar, x_2 es el cliente que se servirá en segundo lugar, … y x_n recogerá el cliente que será atendido en último lugar. Como se puede apreciar, todos los candidatos del problema (los clientes) pasan a formar parte de la solución, lo relevante aquí es el orden que ocupan en la solución.

c) Criterio de selección: Atender en cada paso el cliente no atendido con menor tiempo de servicio.

d) Analizar si el criterio adoptado conduce a la solución:

Veámoslo a través del siguiente ejemplo

Sea n=3 y t[1..3]={5,10,3}.-

etapa	Cliente seleccionado	<x_1,x_2,x_3>	T
inicial	**---**	**< -, -, - >**	**0**
1	**3**	**< 3, -, - >**	**3**
2	**1**	**< 3, 1, - >**	**3+(3+5)=11**
3	**2**	**< 3, 1, 2 >**	**3+(3+5)+(3+5+10)=29**

Existen n!=6 posibles ordenaciones. Estas ordenaciones y sus respectivos valores de T se muestran a continuación:

ordenación	T
1,2,3	5+(5+10)+(5+10+3)=38
1,3,2	5+(5+3)+(5+3+10)=31
2,1,3	10+(10+5)+(10+5+3)=43
2,3,1	10+(10+3)+(10+3+5)=41
3,1,2	3+(3+5)+(3+5+10)=29
3,2,1	3+(3+10)+(3+10+5)=34

La ordenación óptima es 3,1,2.

La planificación óptima se obtiene cuando se sirve a los tres clientes por orden creciente de tiempos de servicio (criterio voraz expuesto previamente): el cliente 3, que necesita el menor tiempo, es servido en primer lugar. Seguidamente, de los clientes que quedan por atender, se seleccionará al cliente 1 dado que tiene un tiempo de servicio menor. Por último, se atenderá al cliente 2 porque es el que necesita mayor tiempo.

El criterio voraz construye la planificación óptima elemento a elemento. Supongamos que después de planificar el servicio para los clientes $x_1x_2 \ldots x_k$ se añade al cliente j. El incremento del tiempo T en esta fase es igual a la suma de los tiempos de servicio para los clientes desde el x_1 hasta el x_k, dado que esto es lo que tiene que esperar el cliente j antes de recibir el servicio, más t_j que es el tiempo necesario para servir al cliente j. Para minimizar esto, dado que un algoritmo voraz nunca reconsidera sus decisiones anteriores, lo único que podemos hacer es minimizar t_j.

El algoritmo voraz es muy sencillo: en cada paso se añade al final de la planificación al cliente que requiera menor tiempo de servicio de entre los que quedar por atender.

Este algoritmo voraz siempre proporciona la solución óptima para el problema de minimización del tiempo de espera.

Sea un conjunto de n tareas que hay que realizar, cada una de las cuales requiere un tiempo unitario. En cualquier instante t podemos ejecutar únicamente una tarea. La tarea i, $1 \leq i \leq n$, nos produce un beneficio b_i (b_i>0) sólo en el caso de que sea ejecutada en un instante anterior a d_i.

El problema de planificación de tareas a plazo fijo consiste en encontrar una secuencia de realización de las tareas que maximice el beneficio total.

SOLUCIÓN.-

a) El problema de planificación de tareas a plazo fijo es un problema de optimización que consiste en maximizar la suma de los beneficios de aquellas tareas que se realizarán dentro de su plazo fijado.

b) La solución se puede expresar como una secuencia de decisiones $<x_1,x_2,\ldots,x_k>$ donde k indica el número de tareas que se realizan dentro de su plazo siendo x_1 la tarea que se realiza en primer lugar (del instante 0 al instante 1), x_2 es por tanto la tarea que se realiza en segundo lugar y así sucesivamente.

c) Criterio de selección: Atender en cada paso aquella tarea con mayor beneficio siempre que la solución parcial ampliada con esa nueva tarea y reordenada con ese orden creciente de plazo, sea una solución parcial ampliada factible, es decir, que ninguna tarea se ejecute fuera de su plazo.

d) Analizar si el criterio adoptado conduce a la solución:

Veámoslo a través del siguiente ejemplo

i	1	2	3	4
b_i	50	30	15	10
d_i	2	1	2	1

etapa	Actividad seleccionada	Secuencia de decisiones	Aceptada(A)/ Rechazada(R)	Beneficio total
inicial	---		---	0
1	1	<1>	A	0+50=50
2	2	<2,1>	A	50+30=80
3	3	<2,1>	R	80
4	4	<2,1>	R	80

En el ejemplo anterior, seleccionamos en primer lugar la tarea 1. Después tomamos la tarea 2; digamos que el conjunto {1,2} es factible porque se puede ejecutar en el orden <2,1>. A continuación probamos el conjunto {1,2,3} que resulta no ser factible; por tanto se rechaza la tarea 3. Por último probamos el conjunto {1,2,4}, que tampoco es factible, así que se rechaza la tarea 4. Nuestra solución, que es la óptima, consiste en ejecutar el conjunto de tareas {1,2}, que sólo se puede efectuar en el orden <2,1>.

El algoritmo voraz esbozado anteriormente siempre encuentra la planificación óptima.

Podemos utilizar una técnica diferente para calcular si un conjunto de tareas es factible: cuando es el turno de considerar la tarea i, se la incorpora a la secuencia lo más tarde posible siempre que sea antes de su vencimiento. Puede ocurrir que algunos instantes de tiempo queden vacíos. Esto no afectará a la factibilidad u optimalidad y siempre se podrá, si se desea, y cuando todas las tareas hayan sido consideradas, comprimir la secuencia para eliminar huecos.

Veamos el funcionamiento con el siguiente ejemplo:

i	1	2	3	4	5	6
b_i	20	15	10	7	5	3
d_i	3	1	1	3	1	3

Obtendremos un conjunto de 3 tareas que corresponde a

min(n,max{d[i]})=min(6,3)=3

Inicialización

	1	2	3
X			

1ª etapa: Se considera la primera tarea y se incluye en la solución ocupando la posición correspondiente al instante más tardío en el que se pueda realizar, esto es, en la posición 3 debido a que d[1] es 3.

	1	2	3
X			1

2ª etapa: Se considera la segunda tarea y se incluye en la solución ocupando la posición correspondiente al instante más tardío en el que se pueda realizar, esto es, en la posición 1 debido a que d[2] es 1.

	1	2	3
X	2		1

3ª etapa: Se considera la tercera tarea y no tiene cabida en la solución debido a que el instante 1 está ocupado y no hay instantes anteriores.

4ª etapa: Se considera la cuarta tarea y se incluye en la solución. Dado que la correspondiente al instante más tardío en el que se pueda realizar, esto es, la posición 3 está ocupada, se busca si hay instantes vacíos anteriores y ese hueco existe y corresponde con la posición 2.

	1	2	3
X	2	4	1

5ª etapa: Se considera la quinta tarea y no tiene cabida en la solución debido a que el instante 1 está ocupado y no hay instantes anteriores.

6ª etapa: Se considera la sexta tarea y no tiene cabida en la solución debido a que el instante 3 está ocupado y tampoco hay hueco en instantes anteriores.

Por tanto, la secuencia obtenida es: 2, 4, 1. Ésta es la secuencia óptima y su beneficio asociado es 42.