TIEMPO EN EL SISTEMA MÍNIMO

Es un planteamiento de una solución voraz al problema de planificación de tareas

Consiste en minimizar el tiempo medio que pasa cada tarea en el sistema. Este es la suma del tiempo que está esperando a que se ejecuten las planificadas antes más el que tarda en ejecutarse.

ALBERTO VERDEJO

UNIVERSIDAD COMPLUTENSE MADRID

Tiempo en el sistema mínimo

- Tenemos *n* tareas, cada una de las cuales requiere un tiempo de ejecución t_i.
- Todas están disponibles para ser ejecutadas, por un único procesador, en secuencia.
- Queremos minimizar el *tiempo medio de estancia de una tarea en el sistema*, esto es, el tiempo transcurrido desde el comienzo de todo el proceso hasta que la tarea termina de ejecutarse.

Supongamos que un peluquero tiene varios clientes esperando diferentes servicios. Por ejemplo, un corte de pelo sencillo un lavado, una permanente... No todos los servicios requieren la misma cantidad de tiempo, pero peluquero sabe bien cuanto le lleva cada uno. Atender a los clientes de manera que el tiempo que tardemos es mínimo. Esto incluye el tiempo que espera un paciente hasta que le hacen lo que sea que le tengan que hacer.

Tiempo en el sistema mínimo

- T_i es el *tiempo en el sistema* de la tarea *i*, que es la suma de su tiempo de ejecución t_i, más los tiempos de ejecución de las tareas que se ejecutan antes que ella.

  El cliente llega. Se sienta. Antes tenemos que cortarle el pelo a dos clientes. El Tiempo en el sistema es el tiempo de atender a esos dos clientes más el tiempo de atender al propio cliente.

- El problema consiste en minimizar el tiempo medio en el sistema

$$TM = \frac{1}{n} \sum_{i=1}^{n} T_i$$

- Pero como *n* está fijo, esto es equivalente a minimizar el tiempo total

$$T = \sum_{i=1}^{n} T_i$$

Siendo n el número de tareas

Ejemplo

$$n = 3, t_1 = 5, t_2 = 10, t_3 = 3$$

- Si el orden de ejecución es 1, 2, 3:

$$T = 5 + (5 + 10) + (5 + 10 + 3) = 38$$

- Si el orden de ejecución es 1, 3, 2:

$$T = 5 + (5 + 3) + (5 + 3 + 10) = 31$$

- Si el orden de ejecución es 3, 1, 2:

$$T = 3 + (3 + 5) + (3 + 5 + 10) = 29$$

Planificación óptima

- La planificación óptima consiste en atender a las tareas por *orden creciente de tiempo de ejecución*.

lo óptimo es ordenar las tareas de menor a mayor tiempo de ejecución.

- Sea $I = i_1, i_2, \ldots, i_n$ una permutación cualquiera de los enteros del 1 al n.

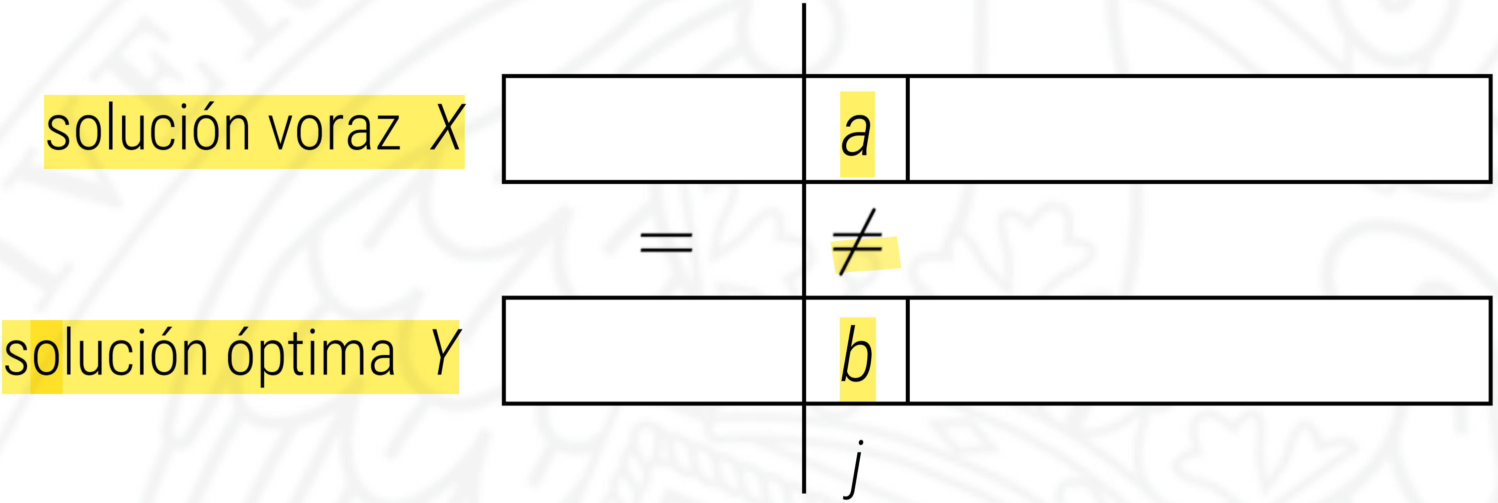
índices de las tareas.

primero se ejecuta i1 luego i2,...

$$
\boxed{
\begin{aligned}
T(I) &= t_{i_1} + (t_{i_1} + t_{i_2}) + (t_{i_1} + t_{i_2} + t_{i_3}) + \cdots + (t_{i_1} + t_{i_2} + \cdots + t_{i_n}) \\
&= n t_{i_1} + (n-1) t_{i_2} + \cdots + 2 t_{i_{n-1}} + t_{i_n} \\
&= \sum_{k=1}^{n} (n-k+1) t_{i_k}
\end{aligned}
}
$$

Depende de las tareas que tenga delante o detrás. Y no de cuales sean.

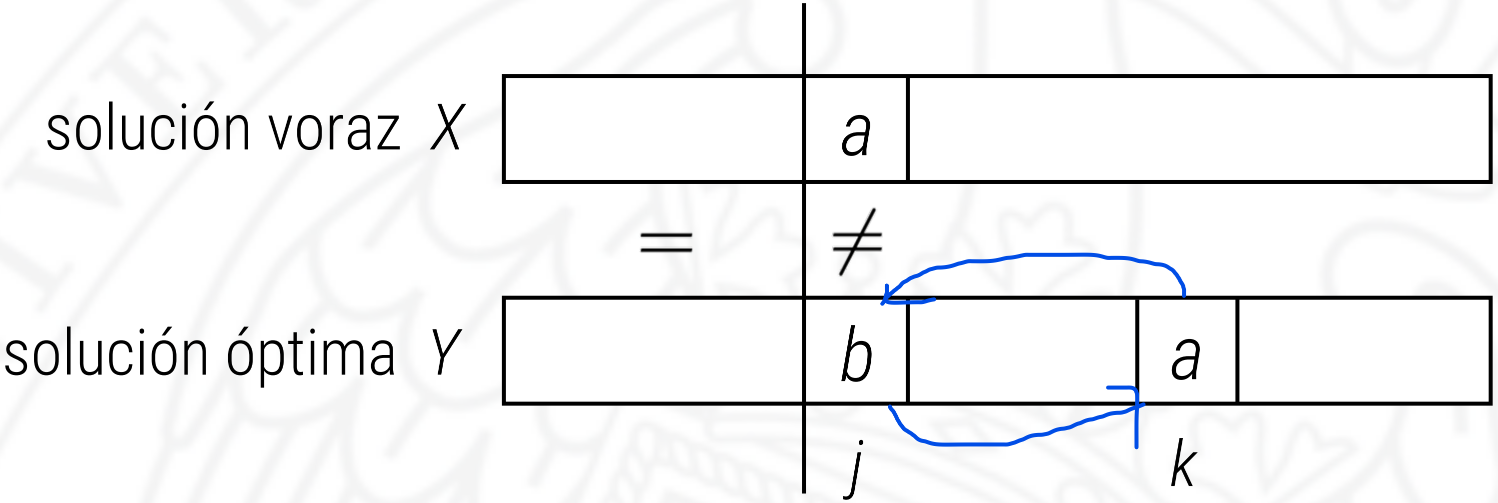
Demostración de optimalidad 1



- Sabemos que $t_a \leq t_b$

La solución voraz plantea las tareas en orden creciente de ejecución, luego la tarea t1 es menor o igual que t2

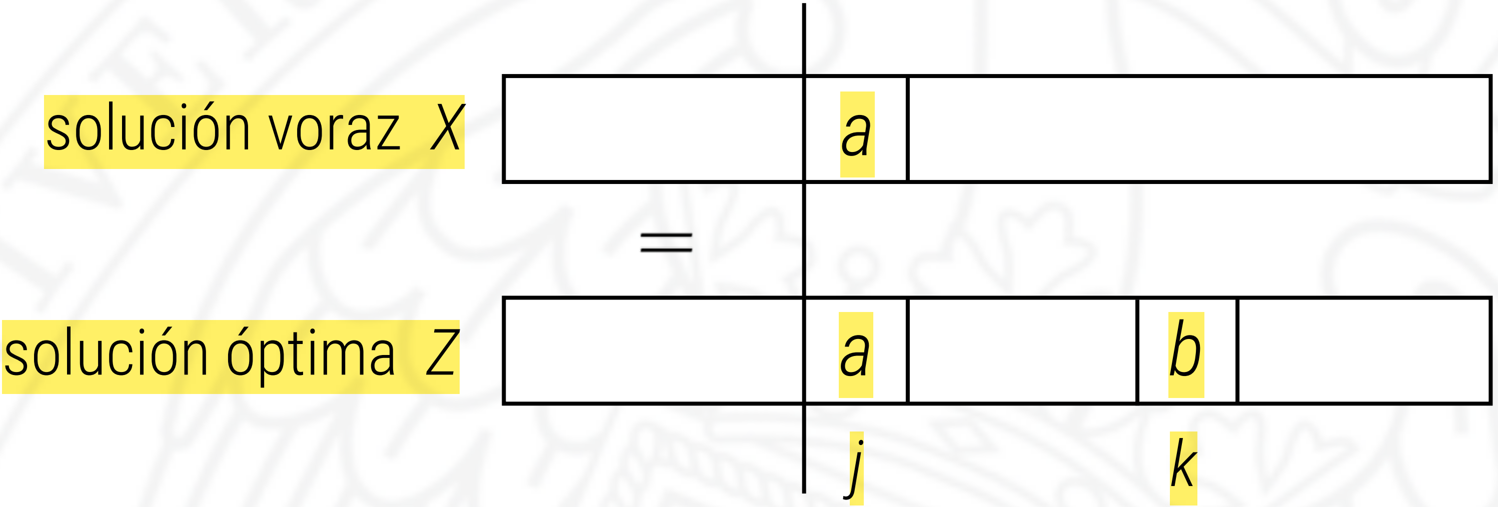
Demostración de optimalidad 1



Ambas soluciones manejan todas las tareas. Existirá una posición K mayor que j, donde la solución óptima i planifique la tarea a.

- Proponemos intercambiar en Y las tareas a y b, para hacer Y más parecida a X.

Demostración de optimalidad 1



Para comprobar que es la solución óptima tenemos que comparar los tiempos totales de Y y de Z

Demostración de optimalidad 1

$$T(Y) = \left(\sum_{\substack{i=1 \\ i \neq j,k}}^{n} (n-i+1)t_{y_i} \right) + (n-j+1)t_b + (n-k+1)t_a$$

Separamos de los sumatorios lo que aportan al tiempo total las tareas a y b en las posiciones j y k. Son las únicas que han cambiado de posición. Además, las soluciones coinciden en todas las posiciones distintas de la j y de la k. Por lo que los sumatorios son iguales

$$T(Z) = \left(\sum_{\substack{i=1 \\ i \neq j,k}}^{n} (n-i+1)t_{z_i} \right) + (n-j+1)t_a + (n-k+1)t_b$$

Demostración de optimalidad 1

$$\begin{aligned} T(Y) - T(Z) &= (n-j+1)t_b + (n-k+1)t_a - (n-j+1)t_a - (n-k+1)t_b \\ &= -jt_b + kt_b - kt_a + jt_a \\ &= t_b(k-j) - t_a(k-j) \\ &= (k-j)(t_b - t_a) \\ &\geq 0 \end{aligned}$$

Sumatorios se anulan

$k > j \quad t_b \geq t_a$

$T(Y) \geq T(Z)$

Como hemos conseguido una mejor solución en Z entonces sabemos que puede ser mayor o igual que la mejor solución en Y, porque hasta ese punto, hemos cogido las mismas tareas, pero las siguientes no sabemos.

Repetir este proceso hasta que la solución óptima sea igual a la voraz.

Demostración de optimalidad 2

- Cualquier solución $Y = (y_1, y_2, \ldots, y_n)$ que no siga la estrategia voraz se puede mejorar.
- Si Y no está ordenada de forma no decreciente, tiene que existir una posición j tal que $t_{y_j} > t_{y_{j+1}}$.
- Si las intercambiamos, la tarea y_{j+1} no tendrá que esperar a y_j, pero y_j tendrá que esperar a y_{j+1}.
- Tras el intercambio, el tiempo será $T(Y) - t_{y_j} + t_{y_{j+1}} < T(Y)$.

Habrá mejorado.