



Departamento de Ciencias de la Computación
Sistemas Operativos

Tarea N°1: Investigación Algoritmo del Banquero

Docente:

AVILA ORTIZ ROMULO DANIEL

NRC:

29048

Nombre:

Madellyn Tasipanta

Fecha:

25-11-2025

Periodo:

PREGRADO S-I OCT 25 - MAR 26

Definición

El Algoritmo del Banquero propuesto por primera vez por Edsger Dijkstra, es una forma de evitar el interbloqueo en sistemas operativos al mantener el sistema en un estado seguro mediante la asignación cuidadosa de recursos a los procesos, lo cual, generalmente no puede ser satisfecho en la práctica. El sistema opera analizando cada solicitud de recursos para verificar que su asignación no lleve a un estado inseguro.

Este algoritmo puede ser explicado a través de una analogía con el funcionamiento de un banco, donde los Clientes representan los procesos, estos tienen un crédito límite y el dinero representaría a los recursos, mientras que el banco es el sistema operativo.

Un sistema se encuentra en un estado seguro si existe un orden en que pueden concederse peticiones de recursos a todos los procesos, el algoritmo funciona encontrando estados de este tipo, es otras palabras, debe existir una secuencia segura como $\langle P_1, \dots, P_n \rangle$, donde para un proceso P_i el pedido de recursos puede ser satisfecho con los recursos disponibles sumados los recursos que están siendo utilizados por P_j , donde $j < i$. Si no hay suficientes recursos para el proceso P_i , debe esperar hasta que algún proceso P_j termine su ejecución y libere sus recursos. Es cuando P_i recién podrá tomar los recursos necesarios, utilizarlos y terminar su ejecución. (Adrian, s. f.)

Componentes

- *Disponible*

Es una matriz 1-D de tamaño 'n' que indica la cantidad de recursos disponibles de cada tipo

- *Máximo*

Matriz 2-D de tamaño 'm*n' que define la demanda máxima de cada proceso en un sistema

- *Asignación*

Matriz 2-D de tamaño 'm*n' que define la cantidad de recursos de cada tipo asignados a cada proceso

- *Necesidad*

Matriz 2-D de tamaño 'm*n' que indica la necesidad de los recursos restantes de cada proceso

Algoritmo para determinar si un estado es seguro (Adrian, s. f.)

1. Se busca una línea "R" en la que las necesidades de recursos insatisfechas sean iguales o inferiores a "A":
 - El sistema se bloqueará en algún momento y ningún proceso podrá finalizar si no hay tal línea.
2. Asumamos que el procedimiento de la línea seleccionada requiere de todos los recursos necesarios y termina:
 - Se indica que el proceso ha finalizado y se incorporan sus recursos al vector "A".
3. Los pasos 1 y 2 se repiten:
 - Hasta que todos los procedimientos se marquen como finalizados, lo que indicaría que el estado inicial era seguro, o
 - Hasta que se produzca un bloqueo, en cuyo caso no lo era.

Desventajas

- Se necesita que exista un número fijo de recursos asignables, pero generalmente no se puede manejar que la cantidad de recursos se mantenga constante
- Requiere que la población de usuarios se mantenga constante
- Se requiere que los procesos reintegren los recursos en un tiempo finito
- Generalmente no es utilizado en S.O. reales

Ventajas

- Es menos restrictivo que la prevención
- No es necesario expulsar y hacer retroceder procesos

Ejemplo de un interbloqueo

Consideremos dos procesos P1 y P2 y dos recursos R1 y R2:

1. P1 adquiere R1 y solicita R2.
2. P2 adquiere R2 y solicita R1.

3. Ambos procesos ahora están esperando recursos retenidos por el otro, lo que provoca un bloqueo.

Ejemplo

Proceso	Máx (R1, R2)	Asignado	Necesita
P1	(7, 5)	(3,1)	(4,4)
P2	(3,2)	(1,1)	(2,1)
P3	(9,0)	(2,0)	(7,0)

Disponibles

$$R1 = 10 - (3 + 1 + 2) = 4$$

$$R2 = 5 - (1 + 1 + 0) = 3$$

P2 pide (1,1)

1. Realizar una comparación con lo disponible:

$$(1 \leq 4) \text{ y } (1 \leq 3)$$

2. Comprobar estado seguro:

Si se otorgan, disponibles (3,2)

P1: requiere (4,4)

P2: requerirá (1,0)

P3: requiere (7,0)

P2 puede completar con (3,2).

Cuando P2 finaliza, libera (2,2):

Disponibles (5,4)

P1: necesita (4,4) entonces puede concluir.

Después P1 libera (3,1): queda disponible (8,5).

P3: requiere (7,0) entonces puede concluir.

Como todos tienen la capacidad de terminar esta en un Estado seguro.

Conclusiones

Al simular con cuidado la distribución de los recursos y asegurar que el sistema permanezca en un estado seguro previene la aparición de interbloqueos, por lo que El Algoritmo del Banquero sigue siendo uno de los enfoques fundamentales para evitar interbloqueos en sistemas operativos y programación concurrente.

En sistemas con exigencias de recursos muy difíciles de predecir, estrategias distintas como romper las esperas circulares, fijar periodos de espera o recurrir a la prelación de recursos pueden resultar más eficaces.

Bibliografías:

- GeeksforGeeks. (2025, 4 septiembre). Banker's algorithm. GeeksforGeeks.
<https://www.geeksforgeeks.org/operating-systems/bankers-algorithm-in-operating-system-2/>
- Adrian, J. (s. f.). 38491448 algoritmo del banquero. Scribd.
<https://es.scribd.com/doc/164763361/38491448-Algoritmo-Del-Banquero>