Investigación 03: Algoritmos de administración de memoria

Maestro:

Javier Rosales Martinez

Materia:

Seminario de Solución de Problemas de Sistemas Operativos

Sección:

D06

Alumno:

Alejandro Covarrubias Sánchez

Código:

221350192

Administración de memoria

La administración de memoria se encarga de gestionar el uso y la asignación de la memoria principal (RAM) de un sistema informático. Su objetivo es optimizar el uso de la memoria de manera eficiente, permitiendo que múltiples programas se ejecuten simultáneamente sin que interfieran entre sí y asegurando que el sistema funcione de manera estable y rápida. Cumple varias funciones como asignar memoria a los procesos cuando lo necesitan y liberarla cuando ya no la están utilizando, evitar que un proceso acceda a la memoria asignada a otro proceso, protegiendo así la integridad de los datos, administrar las áreas de la memoria que no están siendo utilizadas para que puedan ser asignadas a nuevos procesos de manera eficiente e implementar mecanismos que permiten que la memoria física (RAM) se use de manera más eficiente mediante el uso de almacenamiento secundario (disco duro) cuando la RAM no es suficiente.

Algoritmos de administración de memoria

Los sistemas operativos utilizan diferentes algoritmos de administración de memoria para gestionar la asignación y liberación de bloques de memoria a los procesos que lo necesitan. En particular, estos algoritmos buscan optimizar la colocación de procesos en las áreas disponibles de la memoria, reduciendo la fragmentación y mejorando el rendimiento.

Primer ajuste

Busca el primer bloque de memoria libre que sea lo suficientemente grande para acomodar el proceso o la solicitud de memoria. Una vez que encuentra un bloque adecuado, asigna la memoria solicitada a ese bloque y deja el resto (si sobra memoria) como un bloque libre.

Ventajas:

- Es simple y rápido de implementar porque no se necesita explorar toda la lista de bloques libres.

- En muchos casos, la memoria se asigna rápidamente ya que no es necesario realizar una búsqueda exhaustiva.

Desventajas:

- Genera fragmentación externa: al asignar el primer bloque disponible, se pueden dejar pequeños fragmentos de memoria dispersos que no se pueden reutilizar eficientemente.
- Si los primeros bloques disponibles son pequeños, el sistema puede terminar revisando varias veces una gran cantidad de bloques, lo que puede afectar el rendimiento a largo plazo.

Mejor ajuste

Busca el bloque de memoria libre que sea lo más cercano posible en tamaño a la solicitud de memoria. Es decir, intenta encontrar el bloque que deje la menor cantidad de espacio desperdiciado.

Ventajas:

- Al reducir el espacio libre restante, disminuye la probabilidad de generar fragmentos pequeños e inútiles.
- Puede ser eficiente en términos de uso de la memoria, ya que maximiza el uso de los bloques más pequeños.

Desventajas:

- Mayor tiempo de búsqueda: como el algoritmo revisa toda la lista de bloques libres para encontrar el mejor ajuste, el tiempo de búsqueda puede ser más largo, especialmente si hay muchos bloques.
- Aunque minimiza la fragmentación externa, a largo plazo puede hacer que la memoria se llene de pequeños fragmentos que son difíciles de usar.

Peor ajuste

Selecciona el bloque de memoria libre más grande disponible para acomodar la solicitud de memoria. El objetivo es dejar bloques de memoria más grandes y útiles después de la asignación, en lugar de dividir en fragmentos pequeños.

Ventajas:

- Reduce la probabilidad de que queden bloques de memoria muy pequeños después de cada asignación, ya que utiliza los bloques más grandes.
- Puede ser útil cuando las solicitudes de memoria varían mucho en tamaño.

Desventajas:

- Fragmentación interna: asignar un bloque grande para una solicitud pequeña puede dejar una gran cantidad de memoria desaprovechada.
- Mayor tiempo de búsqueda, ya que se deben revisar todos los bloques para encontrar el más grande disponible.

Siguiente ajuste

Es una variante del primer ajuste, pero con una diferencia clave: en lugar de buscar desde el principio de la lista de bloques libres cada vez que se necesita memoria, este algoritmo continúa la búsqueda desde el último lugar donde se realizó la asignación. Si llega al final de la lista de bloques libres y no encuentra uno adecuado, vuelve al principio de la lista.

Ventajas:

- A menudo tiene un rendimiento más rápido que el primer ajuste, ya que evita buscar desde el principio cada vez.
- Puede distribuir las asignaciones más uniformemente por la memoria, lo que puede reducir la concentración de fragmentos en un área específica.

Desventajas:

- Todavía puede generar fragmentación externa, similar al primer ajuste.
- Puede ser menos eficiente que otros algoritmos en términos de minimizar el espacio libre restante, ya que la búsqueda no siempre se hace de manera óptima.

Bibliografía

Corral Liria, A. L. (2008). *Tema 3. Gestión de Memoria*. Diseño de Sistemas Operativos. https://w3.ual.es/~acorral/DSO/Tema_3.pdf

Greyrat, R. (2022, julio 05). *Gestión de memoria en el sistema operativo*. Barcelona Geeks. https://barcelonageeks.com/gestion-de-memoria-en-el-sistema-operativo/

Wolf, G. (2016). *Administración de memoria*. Sistemas Operativos - Gunnar Wolf. https://sistop.gwolf.org/html/04_administracion_de_memoria.html