UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMAS OPERATIVOS

Proyecto 2

Planificación por Retroalimentación Multinivel y por Lotería

PROFESOR: ING. GUNNAR EYAL WOLF ISZAEVICH

Grupo: 06

Integrantes
• Sierra Ruiz Nayeli Selene • Sotelo Ramirez Axel Rodrigo

Fecha de Entrega: 09 de mayo del 2023

Explicación de ambos algoritmos

Planificación por multinivel

La forma en la que se implementó este algoritmo fue la siguiente: Lo que hicimos fue poner hasta arriba las listas que se fueron utilizando durante la creación del algoritmo, después viene una función de la que hablaremos más adelante… Se utiliza un ciclo for en donde se itera el número de procesos que se estarán utilizando (dicho valor se genera aleatoriamente) y dentro del ciclo lo que se hace es crear un diccionario con los valores de cada proceso generados aleatoriamente, es importante mencionar que creamos una lista llamada llegadas que nos ayudará a tomar un control de qué proceso ha llegado y necesita ejecutarse. Después dicho proceso es agregado a una lista de diccionarios llamada procesos y dentro del mismo ciclo el proceso es agregado a 1 de 3 listas disponibles para poder trabajar la prioridad. Con este ciclo podemos crear una lista de diccionarios donde cada valor es un proceso, además ya tenemos listas hechas con procesos dentro para poder trabajar las prioridades con ellas.

Posteriormente se muestran en pantalla los procesos creados junto con sus valores, después se ordena la lista de llegadas para saber qué proceso es el que iniciará. Después a esto se inician los procesos. Para ello se crea un ciclo for each que recorre la lista de procesos ordenados según su propiedad llegada. Para cada iteración se recorren los tickets asignados, se van recorriendo para que podamos saber cuando otro proceso llegue, es decir, si por ejemplo un proceso llega en 80 y tiene una duración de 5, entonces se itera 5 veces a partir de 80, esto lo hacemos así porque dentro de cada iteración vamos haciendo una condición que valida si alguno de los valores entre 80 y 85 son alguna llegada de otro proceso, en caso de ser así se comprueba con otra condición cuál de los dos procesos tiene mayor prioridad (entre más grande la prioridad, más importancia tiene) de esta manera la maquina podrá saber qué proceso ejecutar. En caso de que sea más importante el nuevo proceso, entonces el algoritmo indica que se cambiara de proceso e indica por cuál. Al momento de implementar esta idea tuvimos un pequeño gran problema; lo explico, resulta que cuando nosotros empezamos a iterar sobre 80 (tomando el ejemplo antes mencionado) el algoritmo piensa que es un nuevo proceso que llega porque lo comparamos contra la lista de llegadas, esto genera un error de razonamiento ya que el algoritmo sigue las instrucciones para verificar la prioridad de cada algoritmo. El error se da en la línea 88 del código y quiero pensar que es porque se implemento mal la variable temp… decidimos dejarlo así ya que el tiempo nos ganó… Finalmente usamos recursividad para saltar de un proceso a otro y mostrar los datos. 😊

Planificador por loteria:

Para este algoritmo hicimos lo siguiente… Creamos listas para ir guardando datos importantes durante el desarrollo del algoritmo las cuales hablaremos más adelante, también creamos 3 funciones que más adelante hablaremos para qué sirven.

Lo primero fue utilizar un ciclo for que itera sobre el valor de numero de procesos (el cual fue obtenido de manera aleatoria en un intervalo ya predefinido) y dentro de este ciclo fuimos creando cada proceso utilizando un diccionario, al final se agrega el proceso a una lista de procesos. Después se muestra en terminal los datos de los procesos. Utilizamos la misma estrategia para saber en qué momento se encuentra un nuevo proceso en espera utilizando una lista llamada "llegadas". Posteriormente se recorre cada proceso con un ciclo for each, dentro del cual se imprime el proceso con el que se trabajará y después se itera cada duración del mismo. En el caso de que llegue un nuevo proceso (eso lo sabemos comparando la lista de llegadas con nuestra iteración) se obtiene el proceso entrante (sabemos que es el siguiente en la iteración) y utilizamos nuestra función de generarBoletos donde le damos como parámetro una lista de procesos en ejecución y con ella generará boletos solo para esos procesos, después llamamos a traer a nuestra función loteria que nos permitirá saber qué proceso es el que va a continuar, en caso de que sea uno diferente, entonces llamamos a la función de ejecutarProceso y ejecutamos el proceso completo. Y después regresamos al proceso anterior a repetir los mismo.

Preguntas a resolver

- ¿Cómo se compararía este método con los otros abordados?
- ¿Para qué tipo de carga es más apto y menos apto? El método de planificación por lotería es más apto para cargas de trabajo en las que se desea una distribución equitativa de los recursos y evitar la inanición de los procesos. Al asignar probabilidades a los procesos en función de sus boletos, se les da a todos los procesos la oportunidad de ser seleccionados y ejecutados, lo que ayuda a evitar que algunos procesos acaparen los recursos y dejen a otros en espera indefinida.
- ¿Qué tan susceptible resulta a producir inanición? El método de planificación por lotería es menos susceptible a producir inanición en comparación con otros métodos de planificación, como la planificación por prioridad. En la planificación por lotería, todos los procesos tienen una posibilidad de ser seleccionados para su ejecución, aunque la probabilidad puede variar según la cantidad de boletos asignados a cada proceso. Esto evita que algunos procesos acaparen continuamente los recursos y permite que todos los procesos tengan oportunidades justas de ejecución.

- ¿Qué tan justa sería su ejecución? La ejecución en el método de planificación por lotería se puede considerar justo en términos de asignación de recursos. Cada proceso tiene asignado un número de boletos proporcional a su prioridad, lo que determina su probabilidad de ser seleccionado para su ejecución. Si se asignan boletos de manera equitativa en función de las prioridades, todos los procesos tendrán una oportunidad justa de ser ejecutados.
- ¿Qué modificaciones requeriría para planificar procesos con necesidades de tiempo real? (aunque sea *tiempo real suave*): modificaríamos la forma en la que se asignan los boletos y también la forma en la que se manejan los procesos críticos, dándoles un poco más de importancia y no dejándolo al azar
- ¿Alguna otra reflexión o inquietud que les provoque? Ninguna es muy interesante como se implementa