Introduccion al HPC

Examen 4: Problema 5 Filosofos y productor-consumidor

*Merino Ortega Angel Nahum

***Abstract*— To choose the most suitable graph, you should consider the type of data you are dealing with (categorical or numerical), your objectives (comparison, trends, distribution, relationships), the amount of data to represent, and your audience. For categorical data, bar charts and pie charts are useful, while numerical data can be displayed using line or scatter plots. Boxplots and violin plots summarize distributions, while Sankey diagrams and heatmaps show flows and connections. The choice should prioritize clarity and effectiveness of communication. Experimenting and seeking feedback can help you determine the most appropriate graph.**

Palabras Clave- problemas, algoritmo, deadlock, paralelo, programación.

I. INTRODUCCION

En este documento analizaremos dos problemas el primero es el problema de los 5 filósofos propuesto por Edsger Dijkstra. Este es un problema de sincronización de procesos donde se tienen 5 filósofos en una mesa, los filósofos pueden hacer dos acciones pensar o comer, para comer hay un único plato de espagueti, para tomar el espagueti se requieren tomar 2 tenedores. Entre cada plato hay un tenedor, por lo que si uno decide comer no pueden comer los filósofos sentados a lado de este.

Para el segundo problema se plantea lo siguiente, se tiene un productor que produce una cierta cantidad de productos que debe colocar en un contenedor, si el contenedor se llena el productor debe esperar a que el consumidor haga espacio para nuevos productos. Cada que se procesa un producto se quita del contenedor, si el contenedor esta vacío el consumidor deberá esperar a que haya productos para procesar.

II. PANTEAMIENTO Y SOLUCIONES AL PROBLEMA 5 FILOSOFOS

En el problema de los 5 filósofos se emplea una parte fundamental de los sistemas distribuidos, qué es la concurrencia. Podemos definir las características importantes de la concurrencia en un sistema operativo con lo siguiente: Los mecanismos de concurrencia para asegurar el ordenamiento en la ejecución de procesos: sincronización, exclusión mutua y distribución de recursos en un entorno de memoria compartida asincrónica son las herramientas que se están considerando en el trabajo.

Existen varias soluciones posibles para este problema aquí veremos 5 opciones viables para solucionar este problema:

Para la primera solución se propone que un filósofo empiece a comer y una vez termine le preste su tenedor al siguiente, haciendo que se turnen. Solo uno come mientras los demás esperan su turno, esto, aunque permite que los filósofos coman hace que si son muchos los filósofos esto causa que se aumente el tiempo de espera causando problemas.

Para la segunda opción se proponen diferentes turnos los cuales y con ayuda de N personas que sirven la comida ayudan a repartir los diferentes turnos entre los filósofos que existan repartiendo la comida mediante diferentes fichas. Las cuales se van pasando a los filósofos que están cerca entre sí, para esto se cambia de turno cada cierto tiempo. Esto, aunque efectivo si el tiempo de cada turno es corto con respecto al tiempo en que les da hambre nuevamente. Ya que si este es mayor podría causar que los resultados sean peores incluso que la solución anterior.

En la tercera opción se propone una cola de tenedores donde cada filosofo que quiere comer se pone en una cola de tenedores, cuando un tenedor esta libre lo toma y cuando tiene libre dos tenedores come y después los libera para que los usen los demás filósofos. Cada tenedor solo puede tener dos filósofos en cola esperándolos. Esto igual que las soluciones anteriores puede crear que si todos quieren comer al mismo tiempo ninguno pueda comer ya que se bloquea el sistema.

Para la solución de este conflicto se plantea una cuarta opción la cual en realidad es una modificación en el método anterior. Esta vez cada que un filósofo tiene un tenedor espera un tiempo variable para conseguir el segundo tenedor. Si en este tiempo no queda libre el otro tenedor esta suelta el que tiene y vuelve a ponerse en cola para nuevamente obtener los dos tenedores. Así si un filosofo esta esperando mucho tiempo mejor libera el tenedor para que alguien mas coma y este vuelve a su espera. Para evitar el bloqueo el tiempo de espera debe ser aleatorio y no definido.

La última solución encontrada prone lo siguiente, se indica a cada filosofo que abandonen la mesa cuando no tengan hambre y no regresen a ella hasta que vuelvan a tener hambre para esto se agregara a alguien que supervise el número de filósofos comiendo, limitando el numero a n-1 en este caso al ser 5 el valor de n permite un máximo de 4 filósofos comiendo, logrando así que al menos uno coma.

III. PANTEAMIENTO Y SOLUCION PROBLEMA PRODUCTOR CONSUMIDOR

Este es un ejemplo muy popular de la programación en párelo en el problema se plantea lo siguiente: se tienen 3 objetos el consumidor, un contenedor y el productor. El productor se encarga de producir objetos, una vez producido el objeto se coloca dentro del contenedor, cuando el contenedor se llena se el productor espera a que el consumidor haga espacio para nuevos productos. El consumidor consume los productos, una vez consumidos los productos se quitan del contenedor, cuando

el contenedor está vacío el consumidor debe esperar a que se produzcan los productos.

Para la solución de este problema se propone lo siguiente: el productor pone en lo que produce en una cola para el contenedor. En esta cola el consumidor empieza consumir. Cuando no hay objetos este avisa que no hay objetos. Si el contenedor se llena el productor revisa cada cierto tiempo si hay espacio para más objetos mientras alerta al consumidor de que ya hay productos. Si el contenedor se vacía el consumidor revisa cada cierto tiempo si el productor ya puso nuevos objetos para esto este le avisa que no hay productos. Aumentar la velocidad del consumidor hará que se vacíe mas veces el contenedor, aumentar la velocidad del productor hará que el contenedor se llene más rápido. En esta solución no se plantea un descanso ya que mientras uno consume o produce el otro revisa el estado del contenedor.

Conclusion

En este examen vimos los diferentes métodos que pueden ser usados para la solución de los problemas esto es muy importante y aunque durante el desarrollo de las soluciones no se dijo explícitamente, todos los problemas planteados aquí se son el ejemplo de diferentes técnicas en la computación.

Para el primer problema la primera solución planteada es básicamente la programación secuencial que conocemos tradicionalmente, esta nos dice que solo se hace una tarea para después esperar a las demás, en este caso cada filosofo come y después pasa el turno al siguiente.

La segunda solución planteada es similar solo que esta vez se ejecutan de forma paralela según la cantidad de meseros que para términos de sistemas distribuidos se pueden ver como los núcleos en la programación paralela. En este caso los N ayudantes representarían los núcleos que ejecutan diferentes tareas al mismo tiempo.

La tercera y cuarta opción plantean un mismo método que es mas parecido a lo visto en administración de la memoria ya que aquí se propone que los filósofos no abarquen los tenedores esperando el segundo y provocando un deadlock, ya que se propone un cierto tiempo de espera que ayuda a que se liberen los tenedores para que alguien mas los ocupe y este filosofo lo vuelva a intentar otra vez.

La última opción propone que se separen los filósofos que están comiendo y los que no, para evitar el bloqueo se propone que entren un numero N-1 de filósofos a comer por lo que si el problema plantea 5 solo se dejen entrar a 4 para que uno siempre pueda comer y no exista bloqueo. Esto es interesante ya que propone que no se use al 100% todos los recursos, sino que se tenga suficientes recursos libres para que al menos un proceso se complete y pueda así provocar que todos puedan completarse de forma correcta.

Para el segundo problema costo encontrar una solución ya que es algo complejo, para esto se planteó una sola solución la cual no se descansa o duerme esperando a que se llene el buffer si no que se revisa constantemente el estado en el que está el contenedor. Solucionando así que se cause un deadlock.

Referencias

Colaboradores de Wikipedia. (2023). Problema de la cena de los filósofos. Wikipedia, La Enciclopedia Libre. https://es.wikipedia.org/wiki/Problema_de_la_cena_de_los_fil%C3%B3sofos.

Colaboradores de Wikipedia. (2022). Problema productor-consumidor. Wikipedia, La Enciclopedia Libre. https://es.wikipedia.org/wiki/Problema_productor-consumidor