Análisis de Round Robin para sistemas de multiprocesadores

Cóndor Torres Jesús Ángel.[1], Lazon Vera, Erick.[2]
jcondort@uni.pe [1], elazon@uni.pe [2]
Universidad Nacional de Ingeniería, Rimac, Lima, Perú

***Resumen*—Si bien en sus inicios los computadores contaban con una arquitectura de solo con un procesador, en la actualidad es muy común ver sistemas de multiprocesadores y su tendencia a este tipo de sistema es a extenderse y evolucionar, por ejemplo, ya podemos observarlas en nuestras máquinas de escritorio, laptops e incluso en nuestros smartphones (que incluso cuentan con unidades multinúcleo dedicadas a la inteligencia artificial). El rendimiento comparado a los procesadores mononúcleo ha hecho que proliferen en distintos sistemas, sin embargo junto con los beneficios han aparecido dificultades entre algunas de ellas complejas, la primera es adaptar software que ya funcionaba en un solo núcleo para aprovechar todos los núcleos del sistema empleando hilos (programación paralela). Por otro lado surge la necesidad que dichos sistemas multiprocesadores gestionen de manera adecuada los recursos y los tiempos de cada procesador. En esta ocasión estudiaremos el comportamiento de un planificador Round Robin para un sistema multiprocesador.**

I. Introducción

En un sistema informático tipico existen muchos procesos concurrentes que compiten por recursos, para esto el sistema operativo debe manejar la asignación de recursos de forma eficiente y con respectivo cuidado. El mayor desafío de estos sistemas es establecer planificadores adecuados, un planificador impacta directamente en el rendimiento de un sistema pues es quien se encarga de asignar tiempos de uso del procesador y recursos para la ejecución. Primero debemos familiarizarnos con algunos términos:

I-A. Multiprogramación

Planteamos la siguiente situación, en una oficina de correos un usuario X se une a una cola de atención para recibir su pedido, luego de un tiempo transcurrido finalmente llega su turno, pero su envío aún no está preparado para su despacho entonces lo pasan a un lado a la espera que su envío sea empacado mientras tanto otro cliente es atendido, esta idea aunque simple nos permite mostrar la idea básica de la multiprogración que se busca que la atención no se detenga sino que se fluya a pesar de estos tipos de incovenientes.

Análogamente al ejemplo anterior, algo muy similar ocurre con el concepto de multiprogramación, por ejemplo, en un sistema operativo cuando hay uno o más programas listos para ejecutarse en la memoria principal, como la CPU sólo puede hacer la ejecución de un programa, la idea es tener un sistema multiprogramado para que se pueda optimizar el tiempo del CPU. El sistema operativo puede tener varios tipos de interrupciones como la espera de IO, el transferimiento de control a otro cliente entre otros, por eso uno de los objetivos de la multiprogramación es que los procesos que esperan IO no deban bloquear otros procesos que a su vez generan más desperdicio de tiempo haciéndolo ineficiente.

Hay que tener en cuenta también los problemas que puede abordar el cargar varios programas en la memoria principal como lo es fragmentación o también cuando un programa sea más grande que la memoria(lo cual se puede solucionar a través del uso de la memoria virtual). En resumen la multiprogramación permite que múltiples procesos residan en la memoria principal y tiene como objetivo principal optimizar la utilización de la CPU reduciendo el tiempo de inactividad de esta.

I-B. Multiprocesamiento

Se refiere a las unidades de CPU que el sistema puede contar, si el hardware proporciona más de un procesador,entonces usamos el término **Multiprocesamiento**, puede haber variaciones en el esquema básico como tener múltiples núcleos, dados o paquetes en el sistema.

I-C. Multitarea

Este término es usado en los sistemas operativos modernos en el cual se refiere cuando varias tareas comparten un recurso de procesamiento común, también se refiere a la ilusión de paralelismo cuando la CPU reasigna a otra tarea (cambio de contexto), hay que tener en cuenta que una tarea en un sistema multitarea no es un programa de aplicación completo ya que los sistemas operativos modernos cuentan con su división en páginas lógicas. Se puede decir que la multitarea y la multiprogramación tienen un concepto similar la cual es el compartimiento del tiempo del CPU.

I-D. Multi Threading

Para este término primero veamos el concepto de subprocesamiento la cual es un modelo de ejecución que permite que un solo proceso tenga múltiples subprocesos. El subprocesamiento múltiple es una forma de escribir software concurrente aunque también hay que tener en cuenta la sincronización de subprocesos.

I-E. Tiempo compartido

Los sistemas operativos de programación múltiple y multitarea son estructuras de sistemas de tiempo compartido. En la programación múltiple, aunque la CPU se comparte entre los programas, no es una estructura eficiente para compartir el tiempo de la CPU debido a que el programa sigue ejecutándose hasta que se bloquea, por otro lado, en una estructura de multitarea, el tiempo compartido tiene una mejor eficiencia porque cada proceso de ejecución solo toma un tiempo justo cantidad del tiempo de CPU llamado tiempo cuántico. Incluso en un sistema de multiprocesamiento cuando tenemos más de un procesador, cada tiempo de procesador se comparte entre los procesos en ejecución. Por eso podemos ver que todos los términos están relacionados de alguna manera u otra, sin embargo, no usar el término correcto en el contexto correcto es lo que hace que genere la confusión de términos.

I-F. Sistema de tiempo real

En un sistema de tiempo compartido, los procesos del sistema operativo se programan para que el tiempo de CPU se comparta entre todo el grupo. Dependiendo del algoritmo de programación, cada proceso obtiene su parte del tiempo de CPU, pero no hay garantía de que un proceso obtenga la CPU cuando lo desee. También hay que tener en cuenta que en un sistema en tiempo real, un proceso garantiza la atención de la CPU cuando ocurre un evento específico, por lo tanto debe haber un tiempo límite operativo desde el momento en que se activa el evento hasta el momento en que el sistema responde, es por eso que se dice que los procesos de un sistema en tiempo real son de misión crítica. Un ejemplo práctico es el caso de los robots industriales en una línea de ensamblaje, donde se espera que en cada etapa tenga lugar una determinada operación.

II. Organización del informe (Secciones)

Para esta sección presentaremos los algoritmos o soluciones (secciones) que tendrá nuestro informe final, por ahora sólo mencionaremos los métodos que podemos proponer. Primeramente nuestro proyecto consistirá de un distribuidor de cargas acíclicas, osea un servidor llamado ”Maestro”que distribuirá entre los otros procesadores (esclavos). Las formas de distribuir pueden ser de distintas formas, algunas de ellas pueden ser:

II-A. Método Round Robin

Es un método que selecciona un grupo de manera equitativa y en orden racional, que recorre desde el primero hasta el último y si no se finalizaron los trabajos, se vuelve a repetir el ciclo hasta que terminen todos los trabajos, una característica principal de este método es el uso del **quantum**, el cual es un número fijo de pulsos o ciclos de reloj, debido a esto el quantum debe fijarse en un cierto tamaño de modo que la mayoría de peticiones o trabajos requieran menos tiempo que la duración del cuanto. Otras características que puede presentar son:

- Cuando se genera una interrupción, el proceso que está en ejecución se traslada a una cola de lista y se selecciona el siguiente trabajo.
- Está diseñado especialmente para sistemas de tiempo compartido.
- Para un parámetro crítico, su efectividad depende del tamaño del cuanto pero hay que tener en cuenta el tiempo que se dedica al cambio de contexto.

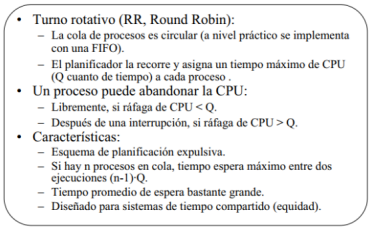

Figura 1. Funcionamiento de Round Robin.

Los conceptos a tener en cuenta en el algoritmo de este método son:

1. **Quantum**: Número máximo de intervalos de tiempo que un proceso puede usar la CPU.
2. **Tiempo de llegada**: Intervalo de tiempo en que comienza el proceso.
3. **Tiempo de ejecucion o Rafaga**: Intervalo de tiempo que demora un proceso en ejecutarse.
4. **Tiempo de finalizacion**: Intervalo de tiempo en que termina el proceso.
5. **Tiempo de retorno**: Suma de intervalos que mide desde que comienza el proceso hasta que finaliza su ejecución.

II-B. Tablas Hash

Otra proposición que podríamos hacer, pero no asegurar es la utilización de tablas hash, la cual es un contenedor asociativo que permite el almacenamiento de entradas con una clave única para luego ser recuperadas posteriormente de manera eficiente, debido a que la clave es única para cada entrada entonces se puede usar a esta clave como un identificador.

Su estructura hace que la recuperación de información sea eficiente con un tiempo constante, sin embargo, esta tabla hash frecuentemente se producen colisiones, esto se pasa generalmente cuando una clave se genera para dos entradas, sin embargo esto tiene solución gracias a que ya cuenta con una función de resoluciones de colisiones, por eso existen dos tipos de tablas hash en función a cómo se resuelven sus colisiones:

- Encadenamiento separado: Las colisiones se resuelven insertándolas en una lista.
- Direccionamiento abierto: Se usa un vector como representación y cuando ocurre la colisión, le volvemos a reasignar otra clave no repetida.

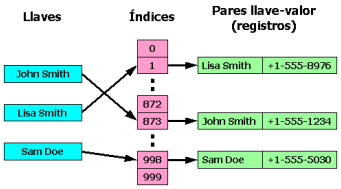

Figura 2. Caption

Las operaciones que se pueden utilizar en una tabla hash son:

- Insertar: Puede ser un valor entero o una cadena de caracteres.
- Borrar: Si uno quiere borrar un nodo debe seleccionar dicho nodo.
- Vaciado de lista: Elimina todos los elementos de la tabla.

II-C. Teoría de Colas

Una de las herramientas que se utilizan en los sistemas multiprocesadores es el uso de colas, la cual se define como un fenómeno matemático y tiene una gran cantidad de aplicaciones en diferentes ramas como Ciencias Médicas, Ciencias de la Gestión, Sistemas informáticos y econometría. Según el modelo de teoría de colas, esta surge con el propósito de predecir algunas medidas de rendimiento en las computadoras, ya que esta se enmarca en estimar los tiempos de espera y la longitud de la cola. Según la teoría de colas que pertenece a la rama de Investigación de operaciones, estas utilizan los recursos para ofrecer el servicio dependiendo de la decisión comercial que pueda tener el modelo. Para estos modelos existen **Sistemas Multiservidor de cola múltiple** y el **Sistema de cola de Servidor único de cola múltiple**, ambas cuentan con las siguientes características:

- Promedio de la tasa de llegada.
- Promedio de la tarifa de servicio.
- Eficiencia del sistema.
- Número de clientes existentes en la cola.
- Número de clientes en el sistema.
- Tiempo de espera del cliente en la cola.
- Tiempo de espera del cliente en el sistema.

III. Estado del Arte

En el planteamiento de este proyecto, además de las herramientas que tenemos a nuestra disposición, debemos usar métodos que nos ayuden incrementar la eficiencia del sistema, entre estos tenemos los algoritmos de planificación los cuales sobresalen los siguientes:

1. **Algoritmo de planificación basado en timestamp**.
2. **Algoritmo de planificación basado en Round Robin**.
3. **Algoritmos híbridos**

Debido a que nos enfocamos en el segundo tipo de algoritmo de planificación veamos como funciona, para empezar este algoritmo asigna una ranura de tiempo a cada flujo y hace que los envíos de paquetes sea de forma alternada, Por ejemplo, si se tuvieran 3 colas (Q1, Q2 y Q3) a cada una de estas colas se le asigna una ranura de tiempo de duración tiempo t1, entonces el algoritmo comenzará a enviar los paquetes de Q1 durante el tiempo t1, luego de eso comienza a enviar paquetes de la cola Q2 durante el mismo tiempo y finamente envía paquetes de la cola Q3 durante un mismo tiempo y si aun no termina los paquetes vuelve de nuevo a la cola Q1 y así hasta que se acabe los paquetes de las tres colas. A pesar que Round Robin tiene una mejor distribución de los labores, tiene una respuesta muy pobre con respecto al retraso de cada paquete obtenido.

III-A. Weight Round Robin (WRR)

Este algoritmo es una modificacion del Round Robin en el cual solo se envia un paquete por cola en cada turno, este funcionamiento se da por cada flujo en el cual se le da un peso en parte de ancho de banda, entonces, el número de paquetes a enviar en cada turno se calcula con base en el peso del flujo y en la capacidad del enlace.

WRR funciona bien en el envío de paquetes de tamaño fijo, sin embargo, presenta problemas de desempeño en los paquetes que son de longitud variable, por ejemplo, en el caso de paquetes de tamaño mayor, este utiliza más ancho de banda de lo que debería asignarse, lo cual hace que el desempeño en el envío de paquetes de las otras colas disminuya, un ejemplo claro de este tipo de algoritmo encontramos en los dispositivos de interconectividad Huawei en los cuales se manejan ocho colas en el que se les asigna un determinado porcentaje de ancho de banda.

III-B. Déficit Round Robin

Este algoritmo mejora las dificultades de WRR y ahora permite enviar paquetes de diferentes longitudes, tiene dos variables, una es el **quantum** que indica la cantidad de bits a enviar en cada turno y la otra variable es el **contador de déficit** que se encarga de almacenar el valor del quantum obtenido.

El funcionamiento de este algoritmo comienza enviando paquetes de colas que tengan por mandar, si el tamaño de este paquete es menor o igual al valor del quantum, el paquete es enviado sino el paquete debe esperar a ser transmitido a otro turno, es ahí donde entra el déficit counter, lo que hace iniciar su cola e incrementa su valor por cada paquete que no es enviado debido a la condicional del quantum, cuando el déficit counter llega a su tope o límite de paquetes estas son enviadas a procesarse y el déficit counter vuelve a cero.

De acuerdo con algunos artículos científicos el método Round Robin es utilizado en modelos de metodología empírica, la cual propone un modelo matemático en una comparación de un evento con su numero de participantes (ejemplo de una liga de fútbol con su número de participantes).

Considerando el ejemplo anterior donde se utiliza el método doble Round Robin, el análisis de este ejemplo considera que en todo torneo o liga hay un porcentaje de desequilibrio en el balance competitivo. Algunos investigadores destacan la posibilidad de utilizar métodos estadísticas para buscar diferencias significativas entre las medias de los distintos torneos.

Tabla 1. **Máximo desequilibrio en una Liga *Round-Robin* duplo con veinte participantes**

Puesto	Club	V	E	D	Puntos	Diferencia
1	A	38	0	0	114	
2	B	36	0	2	108	6
3	C	34	0	4	102	6
4	D	32	0	6	96	6
5	E	30	0	8	90	6
6	F	28	0	10	84	6
7	G	26	0	12	78	6
8	H	24	0	14	72	6
9	I	22	0	16	66	6
10	J	20	0	18	60	6
11	K	18	0	20	54	6
12	L	16	0	22	48	6
13	M	14	0	24	42	6
14	N	12	0	26	36	6
15	O	10	0	28	30	6
16	P	8	0	30	24	6
17	Q	6	0	32	18	6
18	R	4	0	34	12	6
19	S	2	0	36	6	6
20	T	0	0	38	0	6
						114

Note: V = victoria, E = empate, D = derrota.

Figura 3. Resultados de un método doble Round Robin

IV. Metodología o diseño del proyecto

El proyecto consiste en la distribución de cargas a través de un proceso Servidor (maestro) y los otros procesos serán aquellos que trabajen las cargas (esclavos), para plasmar esta idea en código se propone hacerlo en lenguaje java donde se compondrá de dos partes:

- **Servidor**: Se encargará de la distribución de tareas o cargas, además comprobará que los todos los procesos estén ejecutando correctamente.
- **Cliente**: Se encargará de la ejecución de las tareas que se den y una vez resuelta esta devolverá al servidor.

El programa Servidor - cliente estará codificado en lenguaje java utilizando la herramienta de los sockets para la comunicación, ya que esto nos permite mandar mensajes a otras máquinas a través de la conexión TCP, eso quiere decir que el servidor esperará que los .[es]clavosrespondan para que puedan procesar sus tareas, dependiendo de lo que el Servidor les encargue realizar.

V. Conclusiones

Se propuso la estructura y forma del proyecto, los cuales estarán en constantes pruebas para obtener una buena distribución de cargas.

Se propuso los métodos de Round Robin y Tabla Hash para que los procesos puedan tener una mejor eficiencia en la distribución y ejecución de tareas.

Se mostró un modelo ejemplo de cómo se puede utilizar el método Round Robin para un caso real con probabilidades.

Referencias

[1] Franck, E. (2014), *Financial fair play in European football clubs: What is it all about? International Journal of Sports Finance, 9(3), 193-217.*

[2] Haan, M., Koning, R. H., Witteloostuijn, A. (2007), *Competitive balance in national European soccer competitions. In J. Albert, R. H. Koning (Eds.). Statistical thinking in sports (Chap. 4, pp. 63-76). London: Taylor and Francis.*

[3] Levin, R., Mitchell, G., Volcker, P. Will, G. (2000), *The report of the independent members of the Comissioners Blue Ribbon Panel on baseball economics*, Recuperado en: http://www.mlb.com/mlb/downloads/blue_ribbon.pdf

[4] Zhiwen Chen, *student at the College of Computer Science and Electronic Engineering, Hunan University, China. His research interests are in parallel computing and multi-core systems.*

[5] Jianhua Sun, *Associate Professor at the College of Computer Science and Electronic Engineering, Hunan University, China. She received the Ph.D. degree in Computer Science from Huazhong University of Science and Technology, China in 2005. Her research interests are in security and operating systems. She has published more than 70 papers in journals and conferences, such as IEEE Transactions on Parallel and Distributed Systems*, IEEE Transactions on Computers.

[6] Víctor M. Alfaro Ruíz, *Métodos de sintonización de controladores PID que operan como reguladores*, San José, Costa Rica, 2002. Disponible en: http://eie.ucr.ac.cr/uploads/file/documentos/pub_inv/articulos/valfaro02B.pdf

[7] Knightson, K. Morita, N., Towle T. NGN Architecture: Generic *Principles, Funcional Architecture, and Implementation. IEEE Communications Magazine*. Octubre 2005.