TECNOLÓGICO DE MONTERREY, CAMPUS GUADALAJARA

Escuela de Graduados en Ingeniería y Arquitectura

Cómputo Paralelo y Distribuido

Reporte sobre Arquitecturas Multiprocesador

Adrián García Betancourt

Álvaro Izaguirre Serrano

29 de agosto de 2014

El procesamiento múltiple en computación, según la Enciclopedia Británica, se refiere a un modo de operación en el cual dos o más procesadores en una computadora procesan dos o más porciones distintas de un mismo programa de manera simultánea. El multiprocesamiento se realiza normalmente por dos o más micro-procesadores, cada uno siendo en efecto una unidad de procesamiento central (CPU). La ventaja principal de una computadora multiprocesador es la velocidad, y por ende la habilidad de manejar cantidades mayores de información.

Las computadoras han dependido de incrementos en la velocidad del procesador, medida en megahertz (MHz) o gigahertz (GHz), lo cual se relaciona con el número de operaciones que el CPU puede realizar por segundo. Sin embargo los aumentos en la velocidad del procesador traen consigo sobrecalentamiento en los circuitos y otras dificultades que lo hacen insostenible. Otro enfoque fue desarrollado en el cual procesadores especializados son utilizados en tareas como la reproducción de video. Estos procesadores comúnmente vienen en unidades conocidas como tarjetas de video, o tarjetas aceleradoras de gráficos. Las demandas por mejores tarjetas gráficas ha llevado a la industria a desarrollar microchips multiprocesador (multiprocessing, 2014).

Entre las computadoras multiprocesador, existen diversas arquitecturas que dan soporte al procesamiento paralelo. “A los sistemas que aplican una misma instrucción a múltiples datos se les llama computadores SIMD y, en el caso que apliquen diferentes instrucciones a los datos, reciben el nombre de MIMD (González).” Ambas arquitecturas se utilizan en la actualidad para diferentes fines.

En el caso de las arquitecturas SIMD, se utilizan principalmente en aplicaciones científicas y de ingeniería, ya que son capaces de realizar una misma operación sobre estructuras que contienen diferentes datos, como lo son los arreglos o los vectores. Los ejemplos más conocidos de ordenadores con arquitectura SIMD son los computadores paralelos, o procesadores de arreglos, y los procesadores de vectores. La arquitectura SIMD se destaca de las demás en que aplica una instrucción sobre una secuencia de datos; esto se lleva a cabo ya sea mediante una serie de procesadores que operan a la vez (procesadores de arreglos), o de manera secuencial sobre una secuencia de datos (procesadores de vectores).

Las arquitecturas MIMD son las más utilizadas en la actualidad, en las computadoras personales. Combinan múltiples procesadores, capaces de realizar distintas instrucciones sobre los datos. González menciona dos tipos de clasificación para estas arquitecturas, según si la memoria es compartida o distribuida. Los sistemas de memoria compartida se llaman multiprocesadores, y los de memoria distribuida, multicomputadores o clusters. Los clusters son débilmente acoplados, mientras que los multiprocesadores son fuertemente acoplados.

Los sistemas multiprocesador más conocidos son llamados multiprocesadores simétricos (SMP); todos los procesadores son del mismo tipo y acceden de la misma forma a los sistemas de memoria y de entrada/salida. Los multiprocesadores cuentan con un espacio virtual de direcciones de memoria en común. De esta forma se comunican a través de variables compartidas, facilitando la programación para dichas arquitecturas. Las memorias cachés representan un problema de coherencia entre las posibles copias de la información, destaca González, para el caso de arquitecturas con múltiples memorias caché. La memoria caché se introdujo originalmente para reducir la contención de acceso a la memoria compartida.

Es común en los multiprocesadores que existan un único sistema operativo con una tabla de páginas y una de procesos. Puede darse el caso que se deba trabajar con más de una tabla de páginas. Esto ocurre con los multiprocesadores de memoria compartida distribuida (DSM); cada nodo de la computadora tiene su propia memoria virtual, y su tabla de páginas. Cada página es localizada en una memoria asociada a un nodo. Si se llega a dar un fallo de página, el sistema operativo solicita la página al procesador que la tiene localmente, para ser enviada. El sistema operativo se encarga de buscar la página en el disco (González).

En las arquitecturas multicomputadores o clusters, la memoria es distribuida. Cada procesador cuenta con un espacio físico y virtual distinto, por lo tanto, cada procesador puede acceder a su memoria local, pero no a la memoria de otro procesador. Para llevar a cabo esta acción, es necesario comunicarse mediante el paso de mensajes a través de la red de interconexión, lo cual representa una dificultad al momento de programar para estos sistemas. A pesar de esto, los sistemas multicomputadores son mucho más fáciles y baratos de montar con respecto a los multiprocesadores. Es posible construir un sistema multicomputador mediante la interconexión de procesadores de carácter general; además, estos sistemas son muy escalables.

Finalmente, los sistemas MIMD híbridos sin "un sistema multicomputador con memoria distribuida entre los diferentes nodos que lo forman, conectados a través de una red de interconexión, y dentro de cada nodo, tener un sistema multiprocesador con memoria compartida… (González)." Aquí se combinan las ventajas de escalabilidad, y facilidad de programación.

Los procesadores multicore son muy comunes en la actualidad. Estos chips cuentan con más de un CPU, golpearon el mercado hace poco más de una década, y hoy en día prácticamente cualquier computadora cuenta con un CPU de doble núcleo. En el año 2006, la consola PlayStation 3 de Sony ya contaba con un chip multicore de nueve procesadores, para una experiencia de juego más realista y fluida. Sin embargo, los sistemas multicore no solo traen ventajas a los video-jugadores, también son muy útiles para la investigación (Gorder).

Desde hace tiempo se conocía la ley de Moore, y se sabía que se alcanzaría un límite en la velocidad de los procesadores. Anteriormente el software se volvía cada vez más rápido debido a que los fabricantes de chips seguían añadiendo transistores a la clásica arquitectura de un solo procesador. En la actualidad, la velocidad de los procesadores está limitada alrededor de los 4 GHz; si se intenta ir más allá, es necesario adoptar la concurrencia y hacer uso de los procesadores múltiples simultáneos. Al final, lo que llevó a la industria hacia las tecnologías multicore no fue la necesidad de más velocidad de procesamiento, sino la necesidad de menos calentamiento y más eficiencia en el consumo de energía de los procesadores. Los chips más veloces se calentaban más allá de lo que cualquier ventilador pudiera enfriar, y los chips de un solo procesador dependen de muchos transistores que consumen mucha energía para funcionar.

Un procesador multicore se enfría más fácilmente que un chip de un solo procesador, porque los CPUs son más sencillos y contienen menos transistores. Esto también indica que utilizan menos energía y disipan menos el calor. En cuanto al rendimiento, cada CPU multicore es capaz de trabajar en distintas tareas en paralelo. Normalmente el procesamiento paralelo requiere más de un procesador, o requiere algún algoritmo inteligente para simular el procesamiento paralelo de lado del software. Por el otro lado, en un procesador multicore, el procesamiento paralelo se da por sí solo (Gorder).

Aunque los procesadores multicore ayudan en el procesamiento paralelo de tareas distintas de forma automática, los científicos trabajan en desarrollar herramientas que sean capaces de procesar en forma paralela los algoritmos que se utilizan en la investigación en la actualidad. Cada uno de estos proyectos han recibido fondos del *US National Science Foundation* para su desarrollo: procesar de forma paralela las grandes cantidades de datos irregulares que comúnmente manejan los científicos, como lo son los árboles y los grafos; ayudar con el cálculo cuántico mediante un solucionador de sistemas lineales; por último, Gorder menciona un proyecto en el cual se pretende desarrollar una herramienta que recibe como entrada código secuencial, después el usuario dibuja un bosquejo de cómo quiere que sea la implementación paralela de dicho código, y un compilador se encarga del resto.

Cada vez se vuelve más popular el uso de sistemas multiprocesador para aplicaciones de ejecución en tiempo real. Una razón es la caída en los precios de estos sistemas. Otra, los incrementos en tiempos de respuesta y las características de tolerancia a fallos presentes en tales sistemas. En comparación con la tarea de planificación de tareas de tiempo real en un sistema de procesador único, la planificación en multiprocesador y en sistemas distribuidos es un problema mucho más difícil. Sin embargo, determinar una planificación óptima para un conjunto de tareas en tiempo real en un multiprocesador o sistema distribuido es un problema de dificultad NP (Scheduling Real-Time Tasks in Multiprocessor and Distributed Systems).

La planificación de tareas de tiempo real consiste en dos sub-problemas: alojamiento de tareas en procesadores, y planificación de tareas en cada procesador individual. El problema de la asignación de tareas concierne el cómo hacer particiones a un conjunto de tareas, y posteriormente cómo asignarlas a cada procesador. Esto puede ser de forma estática o dinámica. En el caso del alojamiento estático, el alojamiento de tareas a nodos se hace de forma permanente, y no cambia con el tiempo. Mientras que en la asignación dinámica, tareas son asignadas a nodos conforme van surgiendo. Por ejemplo, diferentes instancias de una misma tarea pueden ser alojadas en diferentes procesadores en el alojamiento dinámico. Posterior a un alojamiento exitoso de tareas en procesadores, el problema puede ahora considerarse como planificación en un sistema de procesador único.

Para los sistemas multiprocesador, existen diversos algoritmos que intentan resolver la planificación y distribución de tareas entre los nodos. Ejemplos de estos algoritmos son: balanceo de utilización, ordena las tareas en orden creciente de su utilización, remueve las tareas una por una de la cabeza de la fila, y las aloja en el procesador que esté siendo menos utilizado en ese momento; algoritmo *Next-Fit*, un conjunto de tareas se divide de manera que cada parte es planificada en un procesador, el algoritmo intenta utilizar la menor cantidad de procesadores posible; el algoritmo *Bin Packing*, intenta alojar las tareas de forma que la utilización en cada procesador no exceda cierto valor (Scheduling Real-Time Tasks in Multiprocessor and Distributed Systems).

Referencias

González, D. J. (s.f.). *Multiprocesadores y Multicomputadores.* Universitat Oberta de Catalunya.

Gorder, P. F. (s.f.). Multicore Processors for Science and Engineering. *IEEE Xplore*.

multiprocessing. (2014). Obtenido de Enciclopaedia Britannica:
http://www.britannica.com/EBchecked/topic/397249/multiprocessing

Scheduling Real-Time Tasks in Multiprocessor and Distributed Systems. (s.f.). *IIT Kharagpur*.