

Arquitectura del Computador y Sistemas Operativos

Vigesimooctava Clase



Multiprocesamiento (1/10)

Aumentando la capacidad de cómputo

Hemos visto muchas técnicas para aumentar la capacidad de cómputo de un CPU, pero siempre hay un límite.

Para ir aún más lejos, la única alternativa es usar muchos CPUs conectados juntos.

Para algunas tareas, como graficar una perspectiva o simular escenarios, es sencillo, porque el grado de interacción entre las distintas sub-tareas a realizar es bajo, y los CPUs trabajan cada uno en un caso con zonas de memoria distintas.

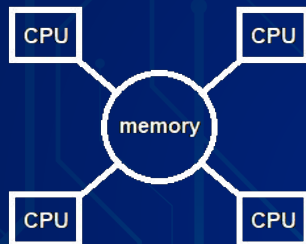
Poca interacción entre tareas → n CPUs → n veces más rápido

Pero para sub-tareas con mucha interacción mutua, se debe compartir memoria y el proceso se hace menos óptimo y más complicado.

Multiprocesamiento (2/10)

Escenarios

Shared Memory - Multiprocessors

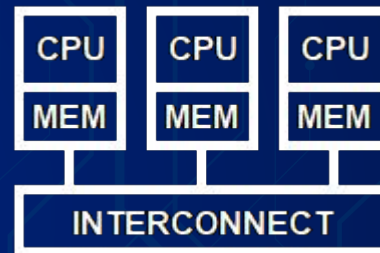


Todas las CPUs comparten una única memoria a la que acceden en igualdad de condiciones.

Se requiere mucho sincronismo dado que todos los accesos deben sincronizarse con las otras CPUs, pero son fáciles de programar porque el sincronismo es transparente a los programas

Aquí entran los multiCore y multiCPU.

Shared Interconnect - Multicomputers



Cada CPU tiene su propia memoria, y comparten una zona de memoria con las otras CPUs.

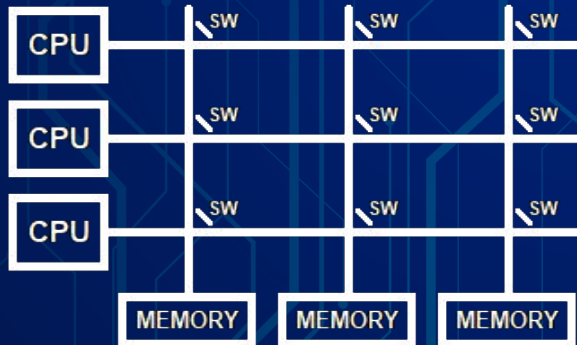
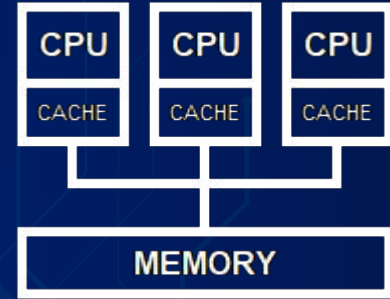
Son más fáciles de construir, pero más difíciles de programar

Multiprocesamiento (3/10)

Shared Memory - Multiprocessors / UMA (Uniform Memory Access) (1/2)

Todas las CPUs acceden a la memoria de la misma forma y a la misma velocidad. Hay varias alternativas:

- Single Bus: Todas las CPUs comparten la misma memoria a través de un BUS. Las contenciones son muchas, y aparece la necesidad del caché. Aún así la eficiencia se degrada para valores de entre 10 y 20 CPUs. Es necesario tener un protocolo de coherencia de caché

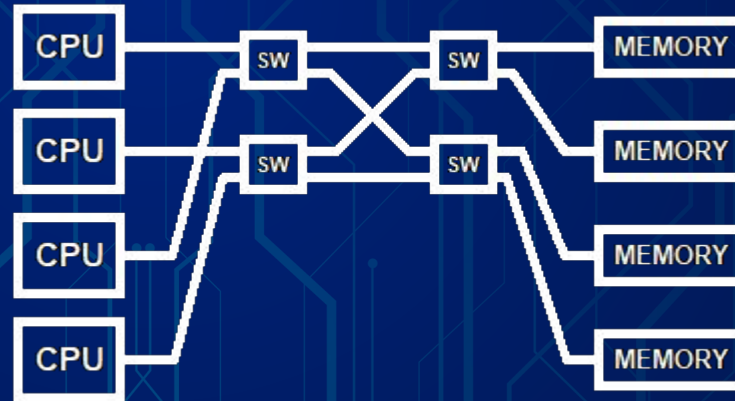


- Cross-switch: Cada CPU tiene su propio camino a memoria. Salvo que quieran acceder a la misma posición, no hay contenciones. El problema es que la complejidad del HW crece con el cuadrado de la cantidad de pares CPU-Memoria.

Multiprocesamiento (4/10)

Shared Memory - Multiprocessors / UMA (Uniform Memory Access) (2/2)

- Red Omega: Es una buena solución de compromiso. Si bien hay más contención porque cada CPU no tiene su propio camino a memoria, se mantiene en valores aceptables aún con cientos de CPUs con un costo de HW aceptable. La cantidad de llaves necesarias para n pares CPU-Memoria se reduce a $(n/2) \log_2 n$.



Multiprocesamiento (5/10)



SO

Shared Memory - Multiprocessors / NUMA (Non Uniform Memory Access) (1/2)

Ahora todas las CPUs tiene una memoria local (no propia) de acceso rápido y una común a la que se demora más en acceder. Sin embargo TODAS las CPUs ven el mismo espacio de memoria.

Se puede optar por no usar caché (NC-NUMA), o usarlo (CC-NUMA), en cuyo caso también hay que tener un protocolo de coherencia de caché.

Shared Interconnect - Multicomputers

El éxito de este modelo radica en dos cosas:

- Por un lado en la velocidad de intercambio de mensajes vía el bloque de interconexión.
- Por otro el desarrollo de aplicaciones que utilicen en forma eficiente este hardware.

El primer problema se enfrenta con hardware dedicado, que dado un mensaje se encarga de entregarlo al destinatario casi sin intervención de la CPU de destino que sigue haciendo otra cosa, por ejemplo vía DMA.

El segundo está apoyado en características especiales ofrecidas por el SO, que exponen funciones para:

- Dado un mensaje y el destinatario se encarga de entregarlo.
- Una cola de mensajes entrantes que cada tarea puede consultar.
- Remote procedure calls (RPCs).
- Distributed Shared Memory.





Multiprocesamiento (7/10)

Sistemas Distribuidos (1/2)

Son una forma de multicomputadora pero hay muy poco vínculo entre los nodos. No comparten memoria ni tienen un sistema operativo que trabaja en forma coordinada. A veces son miles de nodos, pero intercambiar un mensaje puede llevar milisegundos.

El vínculo entre los nodos se usa más para comunicación que para llevar a cabo una tarea en común. Por ejemplo en un 737 hay varias computadoras individuales formando un sistema distribuido:

- Air Data Inertial Reference Unit: tiene como finalidad calcular la actitud y velocidad del avión.
- Flight Management Computer: Usa la posición del ADIRU y la compara con el plan de vuelo, para generar la información para el Flight Director y Piloto Automático.
- Common Display System toma los valores del ADIRU y se los presenta a los pilotos en los monitores
- Los motores tienen computadores que dosifican el combustible y monitorean los parámetros del motor, usando datos como la velocidad y altitud.



Multiprocesamiento (8/10)

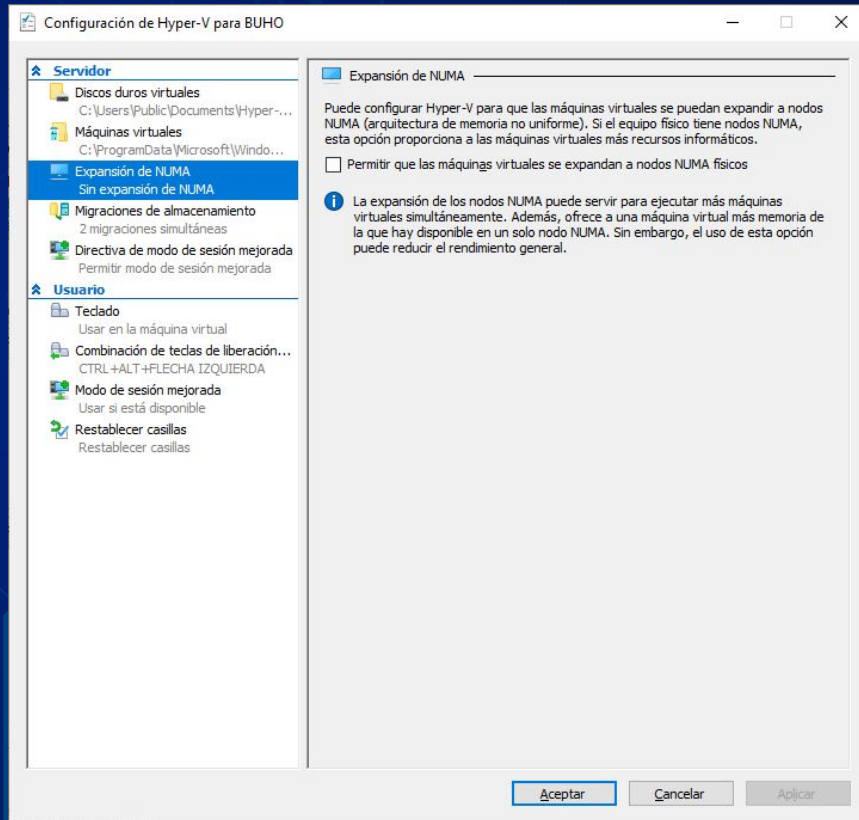
Sistemas Distribuidos (2/2)

Los sistemas operativos que se utilizan en sistemas distribuidos pueden ser:

- Client-Server: Hay nodos que simplemente esperan consultas. Al recibir las las analizan, procesan y responden.
- Peer-to-Peer: Los nodos pueden enviarse mensajes (preguntas y/o respuestas) en forma indistinta.
- Middleware: Un SO se encarga de que las tareas compartan cualquier dato, sin necesidad de mandar o recibir mensajes.
- Three-Tier: Divide las aplicaciones en 3 capas: Interfaz con el usuario, procesamiento y datos. Cada parte se distribuye entre los nodos.
- N-Tier: Va más allá de la anterior, separando cada capa en más capas que son ejecutadas por hardware específico. Mejora la velocidad por la especificidad, pero puede aumentar la latencia por el agregado de capas.

Muchos de los hipervisores soportan la distribución de sus máquinas como NUMA nodes.

Permite ejecutar máquinas muy grandes, que superan la capacidad de un nodo entre varios nodos. Podemos ver un ejemplo en Microsoft Hyper-V:

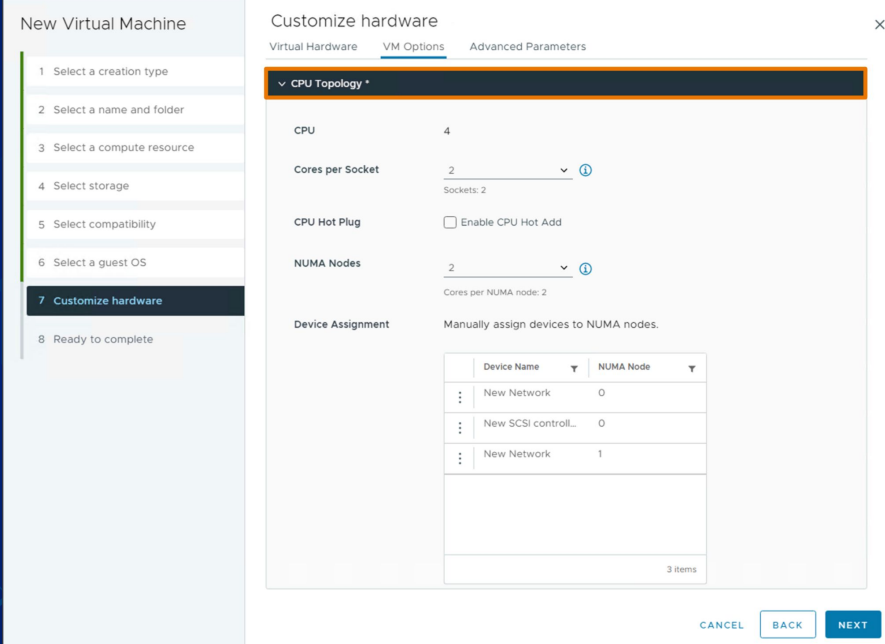


SO

Multiprocesamiento (10/10)

Máquinas virtuales (2/2)

VMWare también lo soporta. Divide los recursos disponibles en nodos NUMA, que luego pueden asignarse a distintas máquinas.



New Virtual Machine

- 1 Select a creation type
- 2 Select a name and folder
- 3 Select a compute resource
- 4 Select storage
- 5 Select compatibility
- 6 Select a guest OS
- 7 **Customize hardware**
- 8 Ready to complete

Customize hardware

Virtual Hardware VM Options Advanced Parameters

▼ CPU Topology *

CPU 4

Cores per Socket 2 Sockets: 2

CPU Hot Plug ☐ Enable CPU Hot Add

NUMA Nodes 2 Cores per NUMA node: 2

Device Assignment Manually assign devices to NUMA nodes.

Device Name	NUMA Node
New Network	0
New SCSI controll...	0
New Network	1

3 items

CANCEL BACK NEXT

An abstract pattern of light blue lines and dots on a dark blue background, resembling a circuit board or a network diagram, located on the left side of the slide.

Fin
¿Preguntas?