

Sistemas Paralelos

Sistemas Distribuidos y Paralelos

Clase 1



Facultad de Informática
UNLP

Datos de la Materia

- **Sistemas Paralelos:** materia obligatorio para Licenciatura en Informática, y optativa para Licenciatura en Sistemas.
- **Sistemas Distribuidos y Paralelos:** materia obligatorio para Ingeniería en Computación.
- **Curso válido para el Doctorado en Ciencias Informáticas.**
- **Profesores:**
 - Armando E. De Giusti
 - Franco Chichizola.

Modalidad del curso

- Teorías: miércoles de 8 a 11 hs.
- Práctica: lunes de 15 a 18 hs. Se realizarán trabajos en máquina con entregas y coloquios en el momento.
- Evaluación de la cursada: se aprueba la cursada a partir de la aprobación de los trabajos a entregar y posterior coloquio.
- Evaluación final de la materia:
 - Rendir un final escrito al inscribirse en la mesa de final.
 - Aprobar un curso de posgrado en la facultad dentro del área de HPC + realización de un trabajo dado por la cátedra y su posterior coloquio (al inscribirse en la mesa de final, hasta la última mesa de 2014).
 - Aprobar un parcial teórico + realización de un trabajo dado por la cátedra y su posterior coloquio (al inscribirse en la mesa de final, hasta la última mesa de 2014).

Bibliografía Básica

- **Introduction to Parallel Computing.** Grama, Gupta, Karypis, Kumar. Addison Wesley. 2003.
- **Foundations of Multithreaded, Parallel and Distributed Programming.** Andrews. Addison Wesley. 2000.
- **Parallel Programming.** Wilkinson, Allen. Prentice Hall 2005.
- **Sourcebook of Parallel Computing.** Dongarra, Foster, Fox, Gropp, Kennedy, Torczon, White. Morgan Kauffman. 2003.
- **Parallel and Distributed Computing. A Survey of Models, Paradigms and Approaches.** Leopold. Wiley. 2001.



¿Por qué Paralelismo?

¿Cómo ejecutar aplicaciones en menos tiempo?

➤ Hay 3 modos de mejorar la performance:

- Trabajar más duramente.
- Trabajar más inteligentemente.
- Conseguir ayuda.

➤ Analogía con los procesadores

- Usar hardware más rápido.
- Usar algoritmos y técnicas optimizadas para resolver las tareas computacionales.
- Múltiples computadoras para resolver tareas particulares.

¿Por qué Paralelismo?

- Incrementar la velocidad de cómputo y/o los recursos.
- Aprovechar las características del hardware actual.
- Incrementar los puntos de acceso a los datos, disminuyendo los tiempos de disponibilidad de los mismos (de gran impacto en las aplicaciones comerciales).
- El desarrollo de los sistemas distribuidos.

¿Por qué Paralelismo?

- Aunque los procesadores individuales crecen en performance, es muy difícil asegurar la sustentabilidad de las aplicaciones de procesamiento masivo de datos, en tiempos y costos factibles.
- El incremento de rendimiento de los procesadores individuales está condicionado por la disipación de calor:

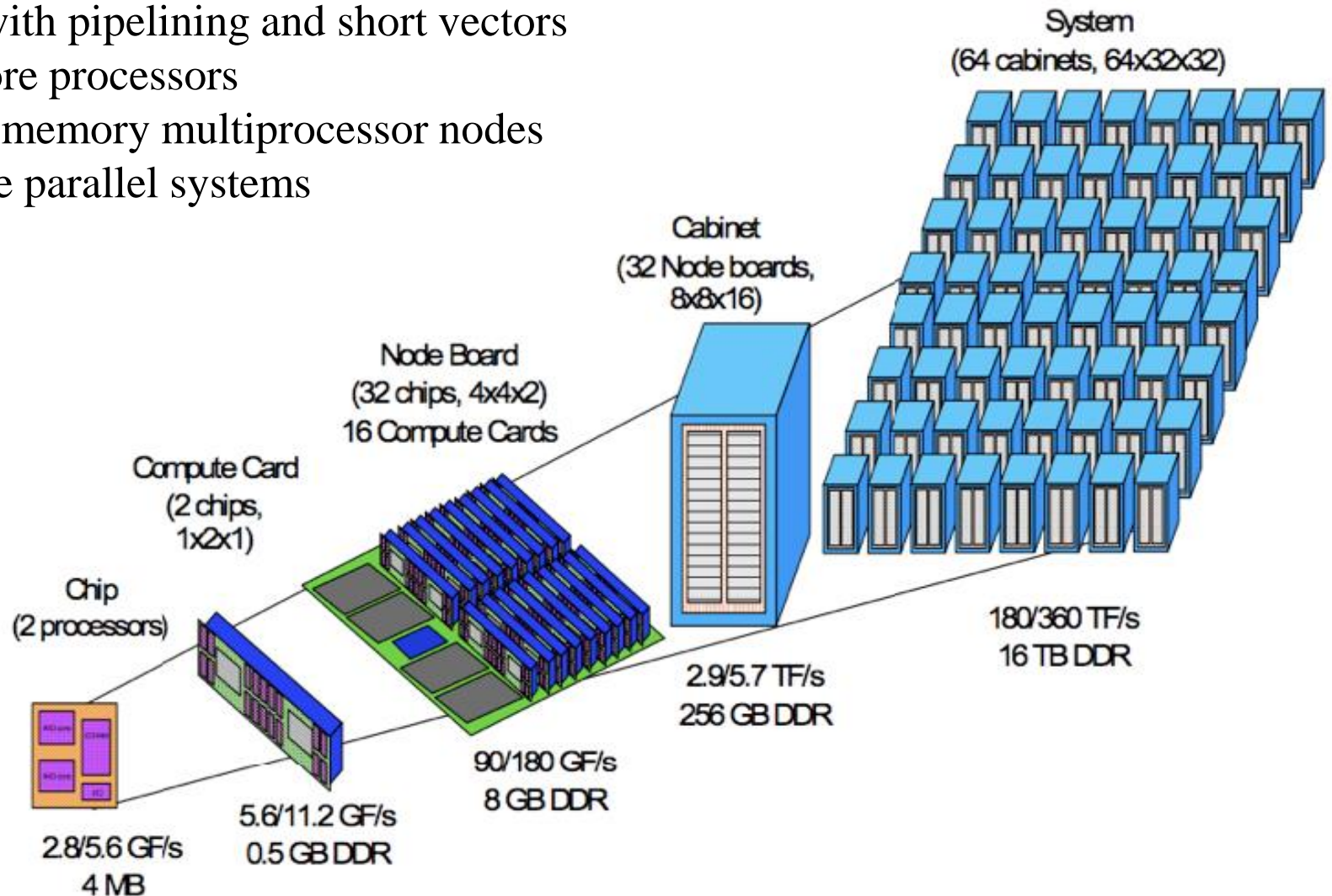


¿Por qué Paralelismo?

- El desarrollo de hardware y software paralelo ha sido tradicionalmente dificultoso (en tiempo y capacidad requerida). La tendencia es desarrollar entornos/bibliotecas y metodologías para reducir el tiempo y asegurar la confiabilidad de las aplicaciones paralelas.
- Todos los sistemas distribuidos (en particular los WEB) son inherentemente paralelos (como el mundo real...) → Sistemas Distribuidos como Maquinas Paralelas
- Los procesadores son básicamente multiprocesadores, con lo cual todo el software se basa en concurrencia/paralelismo.

Paralelismo Jerárquico en Supercomputadoras

- Cores with pipelining and short vectors
- Multicore processors
- Shared-memory multiprocessor nodes
- Scalable parallel systems



El argumento de la Potencia de Cómputo

- ¿COMO trasladar el incremento del número de transistores por circuito integrado en MAYOR número de Operaciones (de cálculo o no) útiles por segundo?
- La respuesta es PARALELISMO, implícito o explícito.
- En la mayoría de los procesadores “seriales” se aprovecha el paralelismo “implícito”. En este curso nos enfocaremos prioritariamente en el paralelismo “explícito” con múltiples procesadores físicos trabajando al mismo tiempo.

El argumento de la relación de velocidad Memoria/Disco

- Mientras que el clock de los procesadores ha venido incrementando en un porcentaje mucho mayor al tiempo de acceso de las memorias DRAM → *Pérdida de Performance en el acceso a memoria.*
- Jerarquía de memorias disminuye la brecha.
- Las plataformas multiprocesador para procesamiento paralelo incrementan el ancho de banda para el sistema de memoria y también aumentan el tamaño de caché disponible en el sistema global → *Mayor rendimiento y mayor complejidad en la Memoria distribuida.*

El argumento de la relación de velocidad Memoria/Disco

- En los algoritmos paralelos se trata de explotar el principio de localidad de los datos, optimizando los tiempos de acceso a memoria.

Esto vale para los Multiprocesadores distribuidos y también para los Multicores en un procesador

- Algunas de las aplicaciones de mayor importancia en Sistemas Paralelos se basan hoy en la eficiencia en el acceso a datos distribuidos, más que en la potencia de cálculo.

El argumento de la evolución de las Comunicaciones

- Mejor Tecnología de Comunicaciones → *Mejores posibilidades de interconectar múltiples procesadores sin pérdida importante de performance.*
- El crecimiento de Internet ha producido un impacto en el desarrollo de aplicaciones que se basan en configurar una máquina virtual paralela sobre la Red → *Clusters, Multiclusters, GRID y ahora Cloud.*
- Cuando los datos están físicamente distribuidos y su manejo debe ser local, se impone la utilización de técnicas paralelas sobre una configuración de red que requiere optimizar las comunicaciones (en velocidad y confiabilidad).



Ejemplo de aplicaciones para Cómputo Paralelo

Aplicaciones de Sistemas Paralelos

- Diseño de vehículos (automóviles, aeronaves, motores).
- Control de sistemas de tiempo real que manejan múltiples señales. (ej. Radares).
- Diseño y simulación de sistemas desde nano y micro escala a macro sistemas.
- Procesamiento de imágenes y reconstrucción 3D.
- Modelización del clima, estudio de fenómenos naturales como incendios, inundaciones, rotura de diques.
- Sistemas Embebidos. Sistemas de control distribuido. Robótica. Visión por computadora.

Aplicaciones de Sistemas Paralelos

- Aplicaciones en astrofísica en temas como la evolución de las galaxias, los procesos termonucleares y el análisis de grandes volúmenes de datos (por ejemplo de telescopios como el Hubble).
- Caracterización funcional y estructural de genes y proteínas.
- En general los problemas de Bioinformática que manejan conjuntos de datos muy importantes y distribuidos físicamente.
- Modelización de sistemas económicos. Aplicaciones en las Bolsas de todo el mundo.
- Data mining para analizar y optimizar la toma de decisiones en empresas y organizaciones.

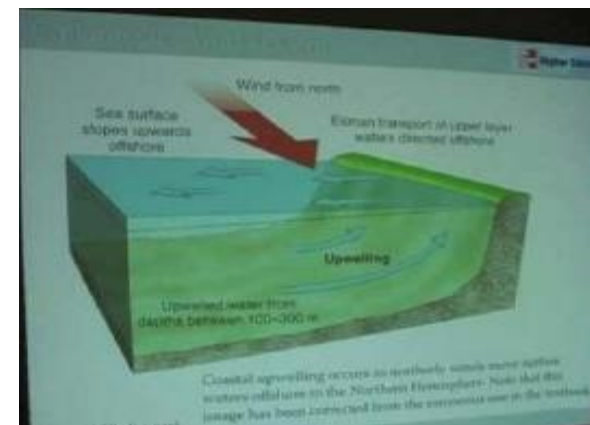
Aplicaciones de Sistemas Paralelos


- Servidores de altas prestaciones (como los de Web o Mail servers) que se implementan en plataformas paralelas.
- Análisis de procesos físicos y químicos complejos (por ejemplo estudios de nuevos materiales) estudiando en paralelo diferentes contextos y su comportamiento.
- Criptografía, control de intrusiones en redes, seguridad.
- Máquinas de uso diario (ej. un automóvil) tienen decenas de procesadores que se comunican entre sí y con un servidor externo.
- Algoritmos paralelos para el manejo de sistemas de comunicaciones móviles.

Necesidad de Cómputo Paralelo

Ej: *simulación de circulación oceánica.*

- División del océano en 4096×1024 regiones, y cada una en 12 niveles. Aproximadamente 50 millones de celdas 3D.
- Una iteración del modelo simula la circulación por 10 minutos y requiere alrededor de 30 billones de cálculos de punto flotante.
- Se intenta usar el modelo para simular la circulación en un período de años...





Conceptos y diferencias de procesamiento paralelo y distribuido

Concurrencia / Paralelismo / Cómputo Distribuido

- La Concurrencia es un concepto de software.
- La Programación Paralela se asocia con la ejecución concurrente en múltiples procesadores (que pueden o no tener memoria compartida).
- El Cómputo Distribuido es un “caso” de concurrencia con múltiples procesadores y sin memoria compartida.
- Especificar la concurrencia implica *especificar los procesos concurrentes, su comunicación y su sincronización.*

Procesamiento Paralelo y Distribuido:

Áreas tradicionales de Procesamiento Paralelo

El Procesamiento Paralelo significa *reducir el tiempo de ejecución* de una tarea utilizando múltiples procesadores al mismo tiempo (o problemas más grandes).

El hardware brinda respuestas, pero lo fundamental es el Software → resolver los problemas de *concurrency entre múltiples procesadores*.

Un resultado inevitable ha sido obtener performance con múltiples procesadores *configurados según UNA arquitectura particular*.

Al mismo tiempo el hardware evoluciona → de las *supercomputadoras* a los *clusters, multiclusters, grid y cloud*.

Procesamiento Paralelo y Distribuido:

Áreas tradicionales de Procesamiento Distribuido

Un sistema distribuido es un conjunto de computadoras autónomas interconectadas que *cooperan compartiendo recursos* (físicos y datos).

Se pueden ejecutar múltiples aplicaciones de múltiples usuarios.

La *granularidad* de los nodos y el *grado de acoplamiento* de los procesadores determina las características y aplicaciones de un sistema distribuido.

Muchas veces los sistemas distribuidos no se fabrican como tales, sino que *evolucionan* a partir de redes LAN y WAN → *heterogeneidad*.

Problemas (entre otros): no hay reloj único, se requiere scheduling, la escalabilidad depende de las comunicaciones, heterogeneidad, seguridad de los datos...

Procesamiento Paralelo y Distribuido: Breve Comparación

Características comunes:

- Se usan múltiples procesadores.
- Los procesadores se interconectan por algún tipo de red.
- Múltiples tareas evolucionan al mismo tiempo y cooperan / compiten.

A veces se usan ambos términos con el mismo significado e incluso hay autores que consideran al Paralelismo una subárea del procesamiento distribuido.

Diferencias básicas:

- En procesamiento paralelo la aplicación se descompone en tareas que se ejecutan al mismo tiempo.
- En procesamiento distribuido la aplicación se descompone en tareas que se ejecutan físicamente en diferentes lugares con diferentes recursos locales.

Procesamiento Paralelo y Distribuido: Breve Comparación

En procesamiento paralelo:

- A menudo las tareas están fuertemente acopladas.
- Se trabaja sobre UNA aplicación y el objetivo es minimizar el tiempo de ejecución de la misma.
- Es común utilizar arquitecturas homogéneas y puede existir memoria compartida (centralizada o distribuida).

En procesamiento distribuido:

- El cómputo requiere recursos físicamente distribuidos (procesadores, memorias, discos, impresoras, bases de datos).
- Pueden ejecutarse múltiples aplicaciones de múltiples usuarios. El objetivo es optimizar la performance global.
- No existe memoria compartida.
- Los sistemas distribuidos son heterogéneos, abiertos y dinámicos.



Sistemas Paralelos

Trasparencia en Procesamiento Paralelo y Distribuido

Trasparencia → Ocultar al usuario/programador los detalles que dan funcionalidad a un sistema → En procesamiento paralelo o distribuido suele ser equivalente a SSI (single system image)

Trasparencia de Ubicación → El usuario utiliza recursos locales o remotos sin distinguirlos.

Trasparencia a la Concurrencia → El usuario no “ve” la competencia y sincronización por los recursos.

Trasparencia a las fallas → en las comunicaciones, en los procesadores → mayor complejidad.

Trasparencia al Paralelismo y su implementación → Partir de la especificación del problema y no detenerse en el modo de descomponerlo en tareas o de ejecutarlo en múltiples procesadores → mayor abstracción → muy difícil obtener alta eficiencia.

Obviamente la transparencia es deseable, pero el precio SIEMPRE es algún grado de **overhead**: acelera el proceso de producción y administración de software paralelo pero disminuye su eficiencia.

Por esto aparece el concepto de “**Sistema Paralelo**”

¿Qué es un Sistema Paralelo?

Es la combinación de un *algoritmo* paralelo y una *computadora* paralela

No debe separarse el algoritmo paralelo de la máquina en que se ejecuta

Modelos de Sistemas Paralelos

En los sistemas paralelos la capa de Hardware puede resolverse de múltiples modos: Supercomputadoras, Clusters, Hipercubos, Multicores, etc.

En la capa de la aplicación tenemos también múltiples herramientas: ADA, JAVA, C, Pthreads, LINDA, OpenMP, MPI, PVM, etc.

En la capa media de estos modelos tenemos básicamente dos esquemas:

Memoria compartida y Memoria Distribuida.

También existen soluciones “mixtas” con Memoria Compartida Distribuida.

Otra característica importante: *Portabilidad*

La *portabilidad de código* se obtiene a través de estándares que permiten que el desarrollo de programas paralelos/distribuidos se ejecuten sobre una variedad de arquitecturas (ejemplos: ADA, MPI, PVM, JAVA).

La *portabilidad de performance* es mucho más difícil de lograr → significa que en diferentes arquitecturas el programa logre un rendimiento o eficiencia similar. Esto NO se consigue al mismo tiempo que la portabilidad de código... Por qué??

Comentarios

- Si bien existen “moldes” es necesario pensar en formas novedosas de resolver los problemas que pueden llevar a reescribir **todo** el código secuencial.
- Las técnicas de *debugging* y *tuning seriales* no siempre son utilizables en el mundo paralelo.
- Necesidad de referirse al *modelo de computación paralela* para el que se construye el algoritmo por la falta de un modelo unificador.
- Problemas paralelizables a distintos grados, o no paralelizables.

Comentarios

- Los procesos no son completamente independientes y comparten recursos. La necesidad de comunicar y sincronizar agrega complejidad a los programas.
- Puede haber *no determinismo* en el interleaving de procesos → 2 ejecuciones de un programa no necesariamente son idénticas → dificultades en la interpretación y el debugging.
- La comunicación y sincronización produce un overhead inútil para el procesamiento, lo que puede desvirtuar la mejora de performance buscada.
- Para obtener una real mejora de performance es necesario adaptar el *soft concurrente* al *hard paralelo*.

Comentarios

- El desarrollo de las aplicaciones paralelas depende mucho de la disponibilidad de ambientes y lenguajes de software adecuados a Programación Paralela.
- Los desarrolladores de Software Paralelo tienen que manejar problemas como:
 - No-determinismo
 - Comunicación
 - Sincronización
 - División y distribución de los datos.
 - Balance de carga.
 - Tolerancia a fallas.
 - Heterogeneidad.
 - Manejo de memoria compartida o distribuida.
 - Deadlocks.