ALGORITMOS PARALELOS
Tema 1: Introducción a la Computación Paralela

- Necesidad de la computación paralela
- ¿Qué es la programación paralela?
- Modelos de computadores
- Evaluación de los computadores paralelos

Necesidad de la Computación Paralela

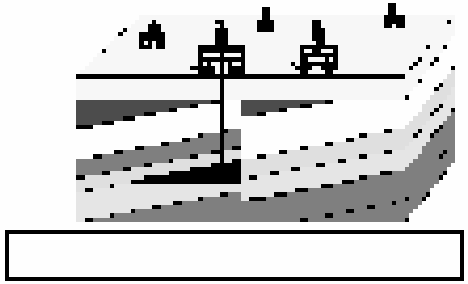
- La velocidad de los computadores secuenciales convencionales se ha incrementado continuamente para adaptarse a las necesidades de las aplicaciones

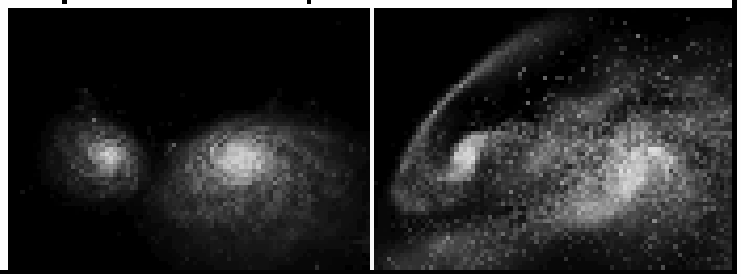
- Simultáneamente hay una demanda continua de un poder computacional superior
  - Modelado y simulación numérica de problemas en ciencias e ingeniería
  - Costosos cálculos iterativos sobre grandes cantidades de datos y fuertes restricciones temporales:
  - Ejemplos: predicción meteorológica, biocomputación, astrofísica
  - Son sistemas cada vez más complejos que requieren mayor tiempo de cómputo

Necesidad de la Computación Paralela

Why Turn to Simulation?

- Climate / Weather Modeling
- Data intensive problems (data-mining, oil reservoir simulation)
- Problems with large length and time scales (cosmology)





Necesidad de la Computación Paralela

Applications: Unlimited Opportunities

- Generic mathematics; visualization & virtual reality
- Computational chemistry, e.g. biochemistry and materials
- Mechanical engineering without prototypes
- Image/signal processing: medicine, maps, surveillance.

Necesidad de la Computación Paralela

Global Weather Forecasting Example

- *Suppose whole global atmosphere divided into cells of size 1 mile $\times$ 1 mile $\times$ 1 mile to a height of 10 miles (10 cells high) - about 5 $\times$ 10^8 cells.*

- *Suppose each calculation requires 200 floating point operations. In one time step, 10^{11} floating point operations necessary.*

- *To forecast the weather over 7 days using 1-minute intervals, a computer operating at 1Gflops (10^9 floating point operations/s) takes 10^6 seconds or over 10 days.*

- *To perform calculation in 5 minutes requires computer operating at 3.4 Tflops (3.4 $\times$ 10^{12} floating point operations/sec).*

Necesidad de la Computación Paralela

Modeling Motion of Astronomical Bodies

- *Each body attracted to each other body by gravitational forces. Movement of each body predicted by calculating total force on each body.*
- *With N bodies, N - 1 forces to calculate for each body, or approx. N2 calculations. (N log2 N for an efficient approx. algorithm.)*
- *After determining new positions of bodies, calculations repeated.*
- *A galaxy might have, say, 10^{11} stars.*
- *Even if each calculation done in 1 ms (extremely optimistic figure), it takes 10^9 years for one iteration using N^2 algorithm and almost a year for one iteration using an efficient N $\log_2$ N approximate algorithm.*

Necesidad de la Computación Paralela

- Siempre habrá aplicaciones que requieren más poder computacional
- La relación coste/prestaciones se hace desfavorable si se pretende incrementar más aún la potencia de los computadores secuenciales.
- Además, el rendimiento de los computadores secuenciales está comenzando a saturarse.
- En todo caso hay límites para una única CPU
  - Memoria disponible
  - Prestaciones

- **Solución**: Usar varios procesadores. Sistemas paralelos

  Con la tecnología VLSI, el costo de los procesadores es menor.

Necesidad de la Computación Paralela

- **Solución**: Usar varios procesadores. Sistemas paralelos

  Con la tecnología VLSI, el costo de los procesadores es menor.

- Muchas posibilidades:
  - Pipeline
  - Cachés
  - Paralelismo a nivel de instrucción
  - Ejecución fuera de orden
  - Especulación
  - Varios procesadores en un chip
  - LAN de altas prestaciones

¿Qué es la programación paralela?

- Uso de varios procesadores trabajando juntos para resolver una tarea común

- El modo de uso de los procesadores puede ser diseñado por el programador:
  - Cada procesador trabaja en una porción del problema.
  - Los procesos pueden intercambiar datos, a través de la memoria o por una red de interconexión.

Ventajas de la Computación Paralela

- La programación paralela permite:
  - Resolver problemas que no caben en una CPU
  - Resolver problemas que no se resuelven en un tiempo razonable

- Se pueden ejecutar
  - Problemas mayores
  - Más rápidamente (aceleración)
  - Más problemas

Conceptos relacionados pero no iguales

- Programación **concurrente**:

  Varios procesos trabajando en la solución de un problema, puede ser paralela (varios procesadores)
- Computación **heterogénea**:

  Varios procesadores con características distintas
- Programación **adaptativa**:

  Durante la ejecución el programa se adapta a las características del sistema
- Programación **distribuida**:

  Varios procesadores geográficamente distribuidos. Hay paso de mensajes pero se necesita infraestructura especial
- Computación **en la web**:

  Necesidad de herramientas que permitan la utilización de sistemas de computación en la web
- Computación **cuántica** o **biológica**

Aspectos a considerar

Aspectos a tener en cuenta en la computación paralela son:

- **Diseño de computadores paralelos**. Escalabilidad y Comunicaciones.
- **Diseño de algoritmos eficientes**. No hay ganancia si los algoritmos no se diseñan adecuadamente.
- Métodos para **evaluar los algoritmos paralelos**: ¿Cómo de rápido se puede resolver un problema usando una máquina paralela? ¿Con qué eficiencia se usan esos procesadores?
- **Lenguajes** para computadores paralelos, flexibles para permitir una implementación eficiente y que sean fáciles de programar.
- **Herramientas** para la programación paralela.
- Programas paralelos **portables**.
- **Compiladores** paralelizantes.

Tipos de computadores

Tipos de computadores según la taxonomía de Flynn

- **SISD**: Computador secuencial. Un procesador y una memoria. **Computador secuencial**.
- **SIMD**: Máquina con varios procesadores pero una única Unidad de Control. **Computador vectorial**.
- **MIMD**: Máquina con varios procesadores "completos". **Computador paralelo**.

Computador secuencial: SISD

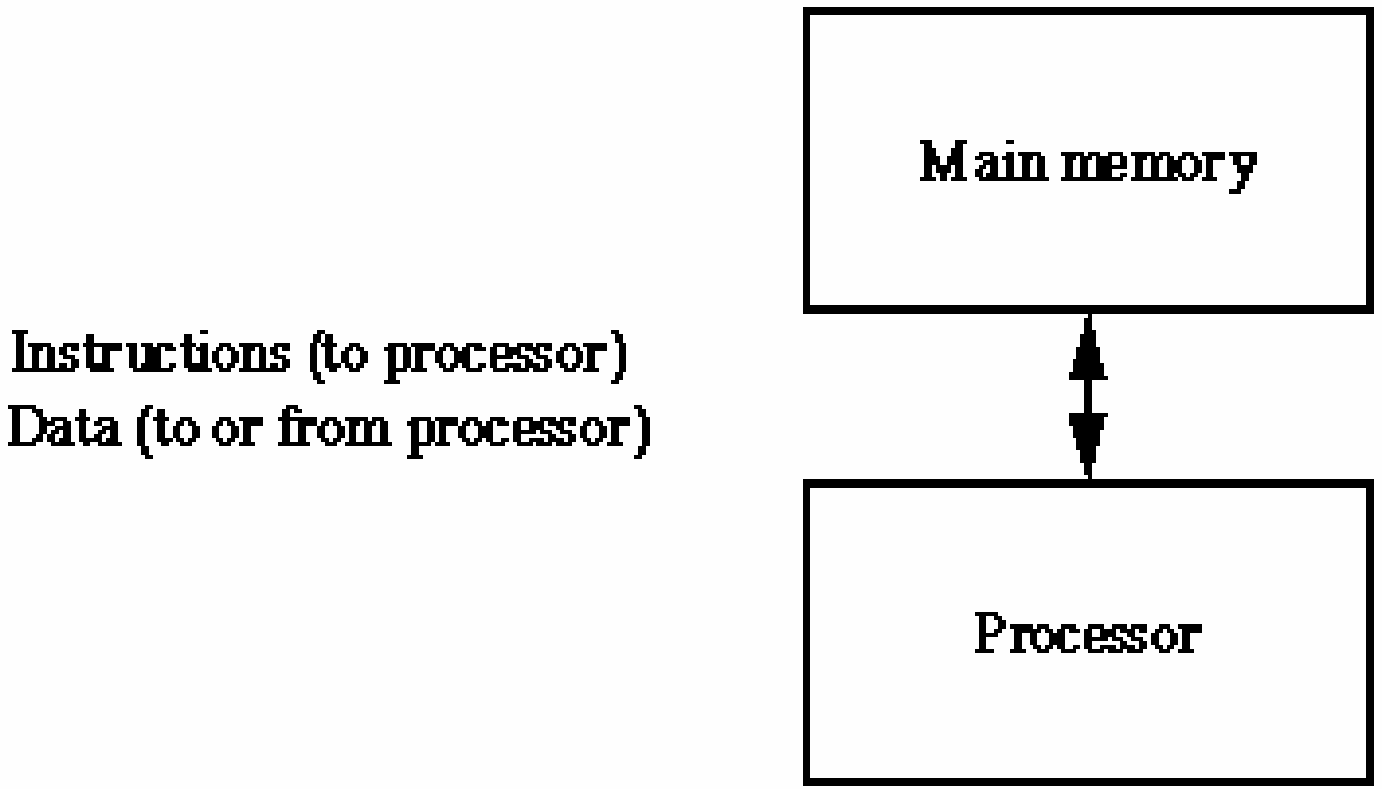


Figure 1.1 Computer having a single processor and memory

Computador “paralelo” : SIMD

- SIMD. Una única Unidad de Control. La misma instrucción se ejecuta síncronamente por todas las unidades de procesamiento. Sincronización automática. Requiere menos hardware porque sólo necesita una U.C global y menos memoria porque tiene una sola copia del programa.

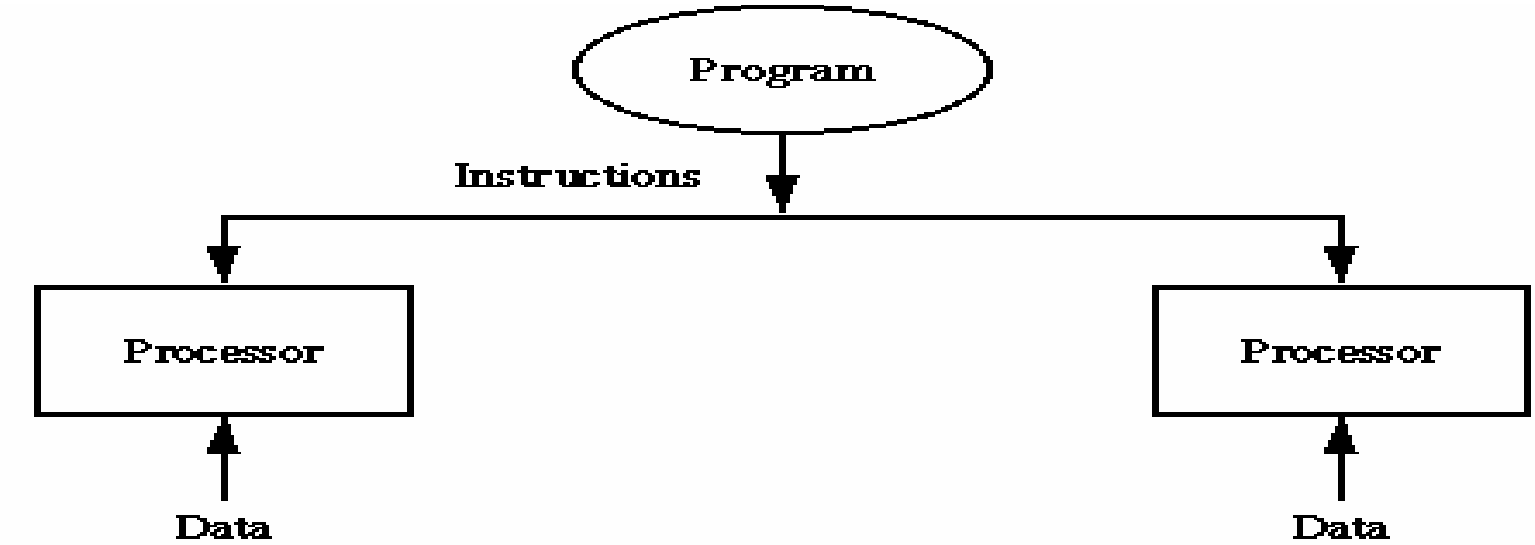


Figure 1.5 SIMD model

Computador paralelo: MIMD

- MIMD. Cada procesador ejecuta un programa diferente independientemente de los otros procesadores.

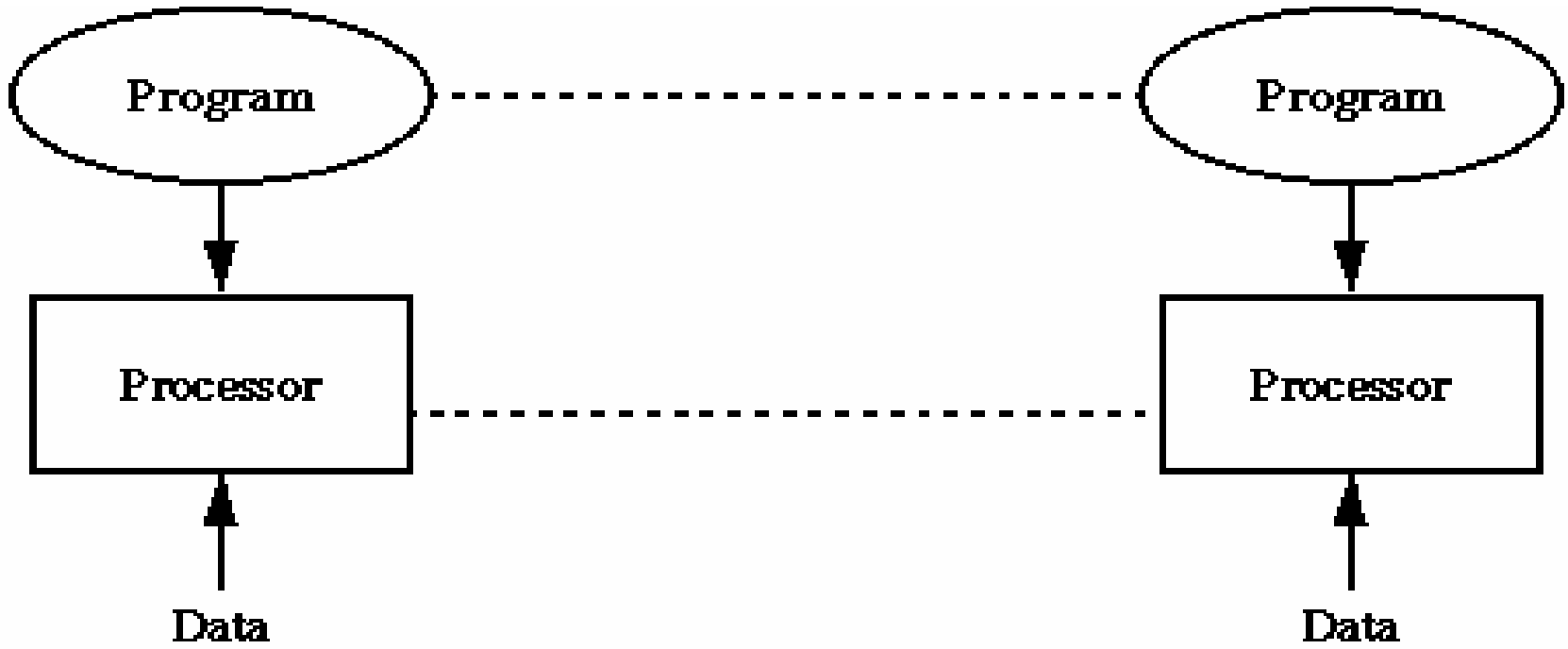
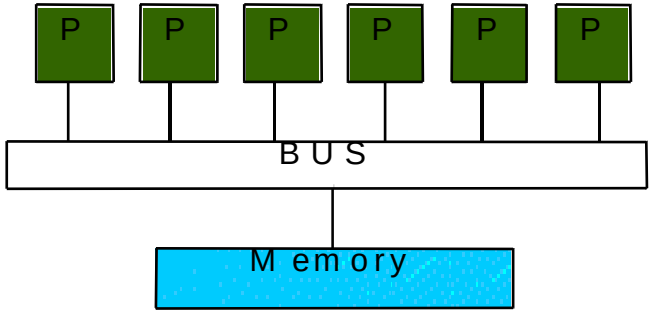


Figure 1.4 MIMD model

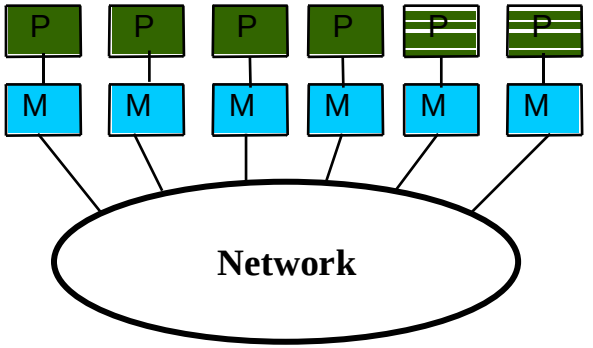
Tipos de computadores paralelos



Memoria compartida – un único espacio de memoria. Todos los procesadores tienen acceso a la memoria a través de una red de conexión:

- Bus
- Red de barras cruzadas
- Red multietapa

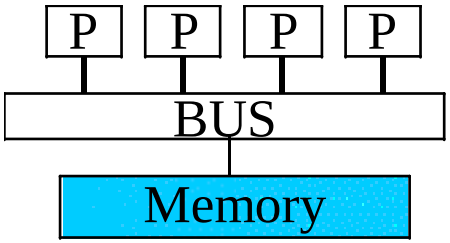
MULTIPROCESADOR



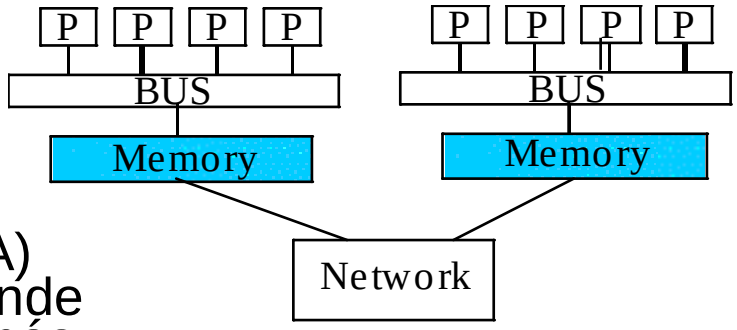
Memoria distribuida – cada procesador tiene su propia memoria local. Se utiliza paso de mensajes para intercambiar datos.

MULTICOMPUTADOR

Comp.Paralelos de memoria compartida



Uniform memory access (UMA)
Cada procesador tiene acceso uniforme a memoria. También se llaman symmetric multiprocessors (SMPs)



Non-uniform memory access (NUMA)
El tiempo de acceso depende de dónde están los datos. El acceso local es más rápido. Más fácil y barato de escalar que SMPs

Sistemas de memoria compartida

Redes basadas en buses.

Cuando un procesador necesita acceso global a memoria, genera una solicitud al bus.

Esta red es atractiva, dada su simplicidad y capacidad para proporcionar acceso uniforme a la memoria compartida.

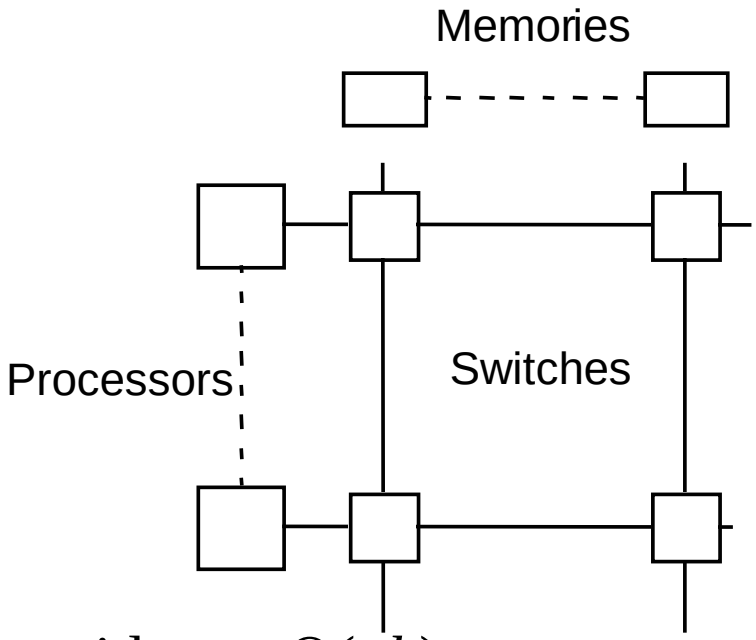
Pero el bus sólo puede llevar una determinada cantidad de datos entre la memoria y los procesadores.

El rendimiento se satura para un número pequeño de procesadores.

Si los procesadores disponen de memorias locales caché se puede solventar el problema.

Sistemas de memoria compartida

Redes de barras cruzadas.



Utiliza una red de conmutadores.

Conecta p procesadores
con b módulos de memoria.

El número total de conmutadores requeridos es $\Theta(pb)$.

Conforme crece p, la complejidad de la red aumenta según $\Omega(p^2)$.

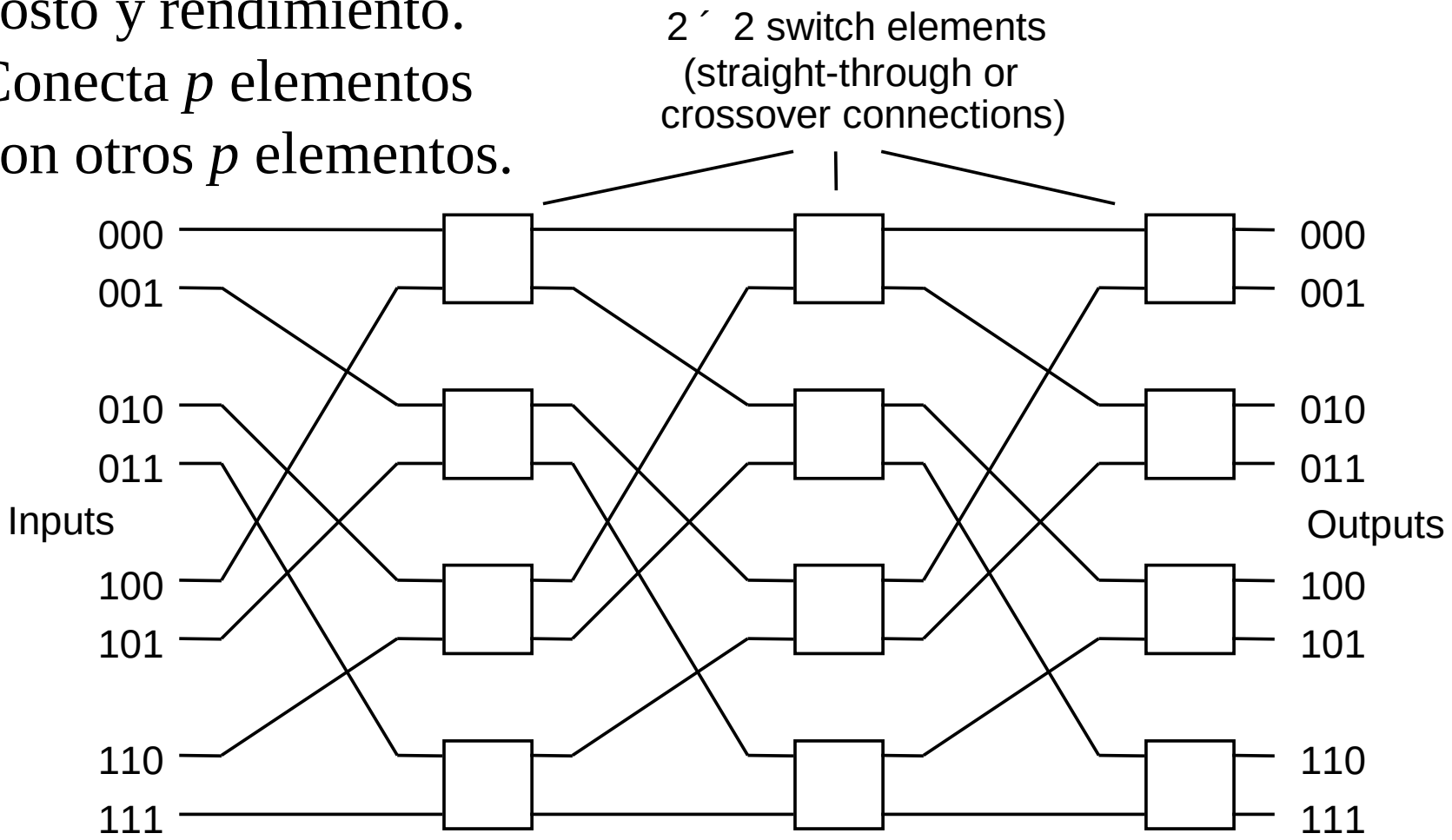
Por tanto no son muy escalables en términos de coste.

Sistemas de memoria compartida

Redes de interconexión multietapa.

Es un tipo de red intermedia en términos de escalabilidad en costo y rendimiento.

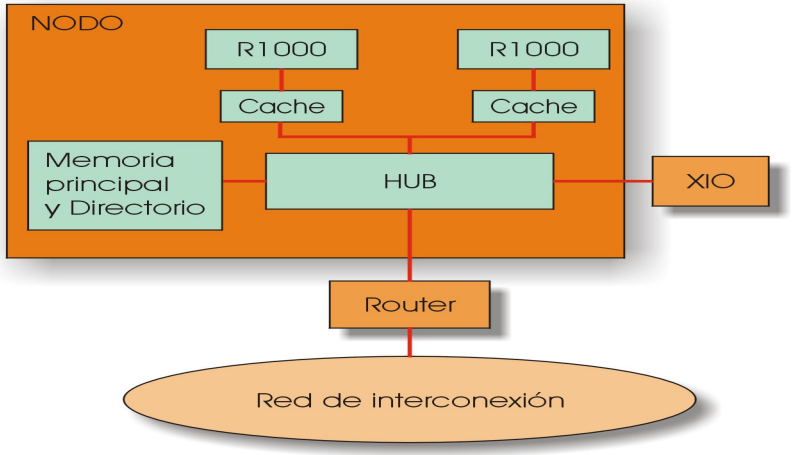
Conecta *p* elementos con otros *p* elementos.

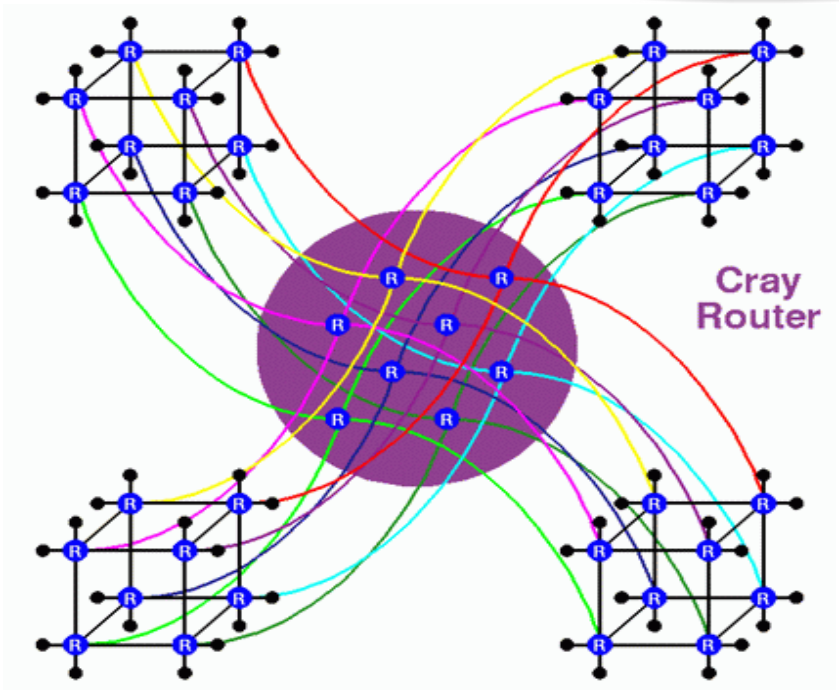


Problemas de los sistemas UMA

- Los sistemas UMA no escalan bien
  - Los sistemas basados en bus se pueden saturar.
  - Una red de barras cruzadas grande puede ser muy cara.
- Problema de la coherencia de caché
  - Puede haber copia de una variable en varias cachés
  - Cuando un procesador escribe puede no ser visible al resto
  - Es necesario asegurar la visibilidad o la coherencia de caché

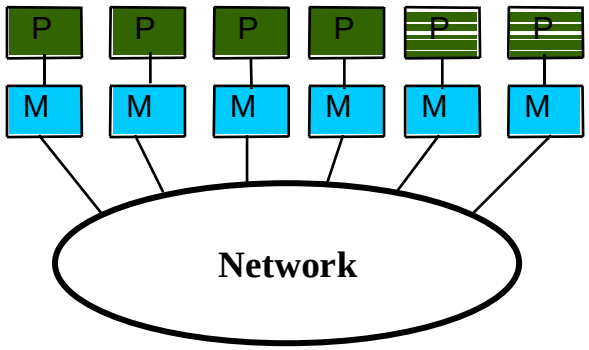
Ejemplo NUMA: SGI Origin 2000





Comp.Paralelos con memoria distribuida

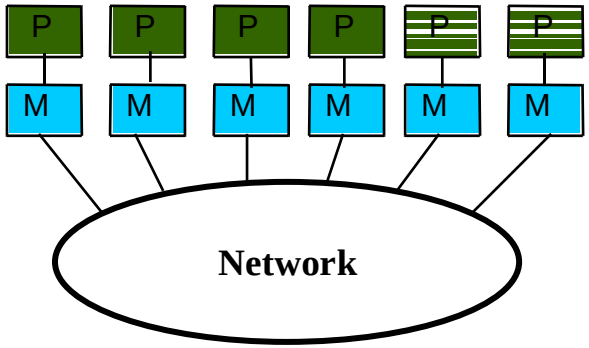
- En esta arquitectura, el computador paralelo es esencialmente una colección de procesadores secuenciales, cada uno con su propia memoria local, que pueden trabajar conjuntamente.
- Cada nodo tiene rápido acceso a su propia memoria y acceso a la memoria de otros nodos mediante una red de comunicaciones, habitualmente una red de comunicaciones de alta velocidad.
- Los datos son intercambiados entre los nodos como mensajes a través de la red.



Comp.Paralelos con memoria distribuida

Redes de ordenadores

- Una red de ordenadores, especialmente si disponen de una interconexión de alta velocidad, puede ser vista como un multicomputador de memoria distribuida y como tal ser utilizada para resolver problemas mediante computación paralela.



Topologías de interconexión

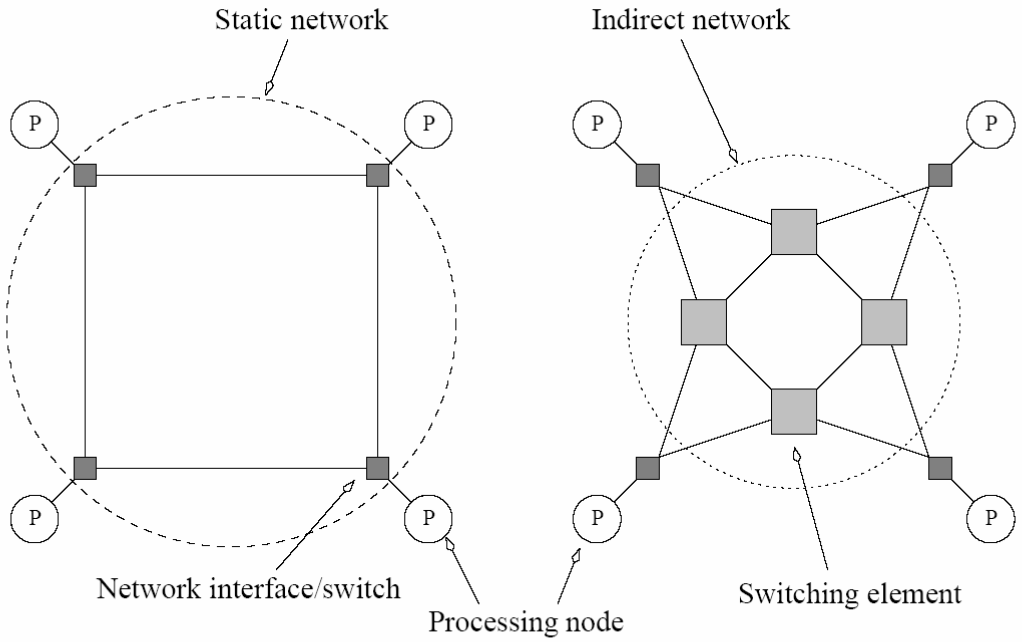


Figure 2.6 Classification of interconnection networks: (a) a static network; and (b) a dynamic network.

Topologías de interconexión estáticas

- Línea / Anillo
- Matriz 2-D (malla) / Toro
- Hipercubo
- Conexión completa
- Estrella
- Árbol

Topologías de interconexión

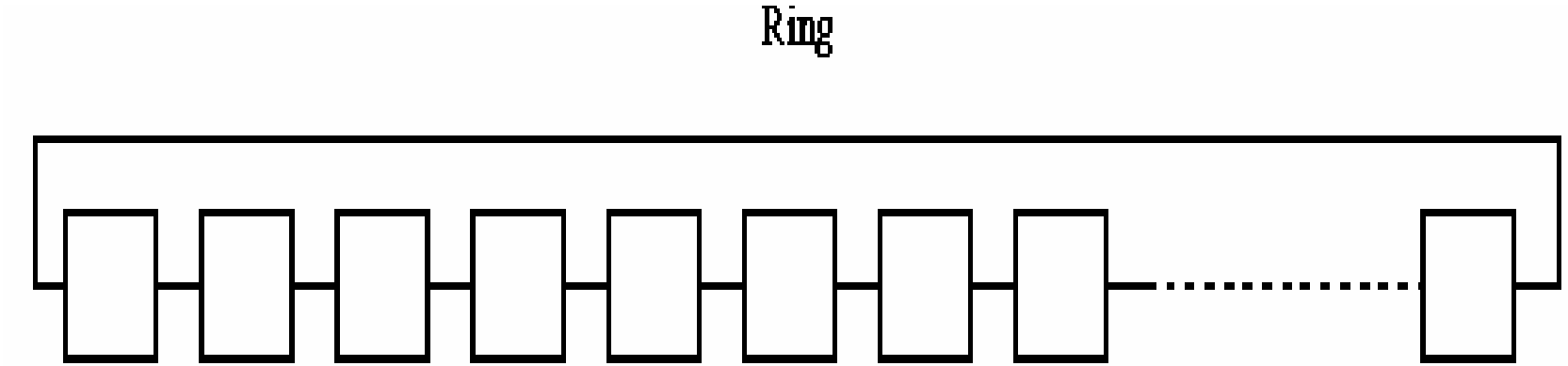


Figure 1.6 Linear array

Topologías de interconexión

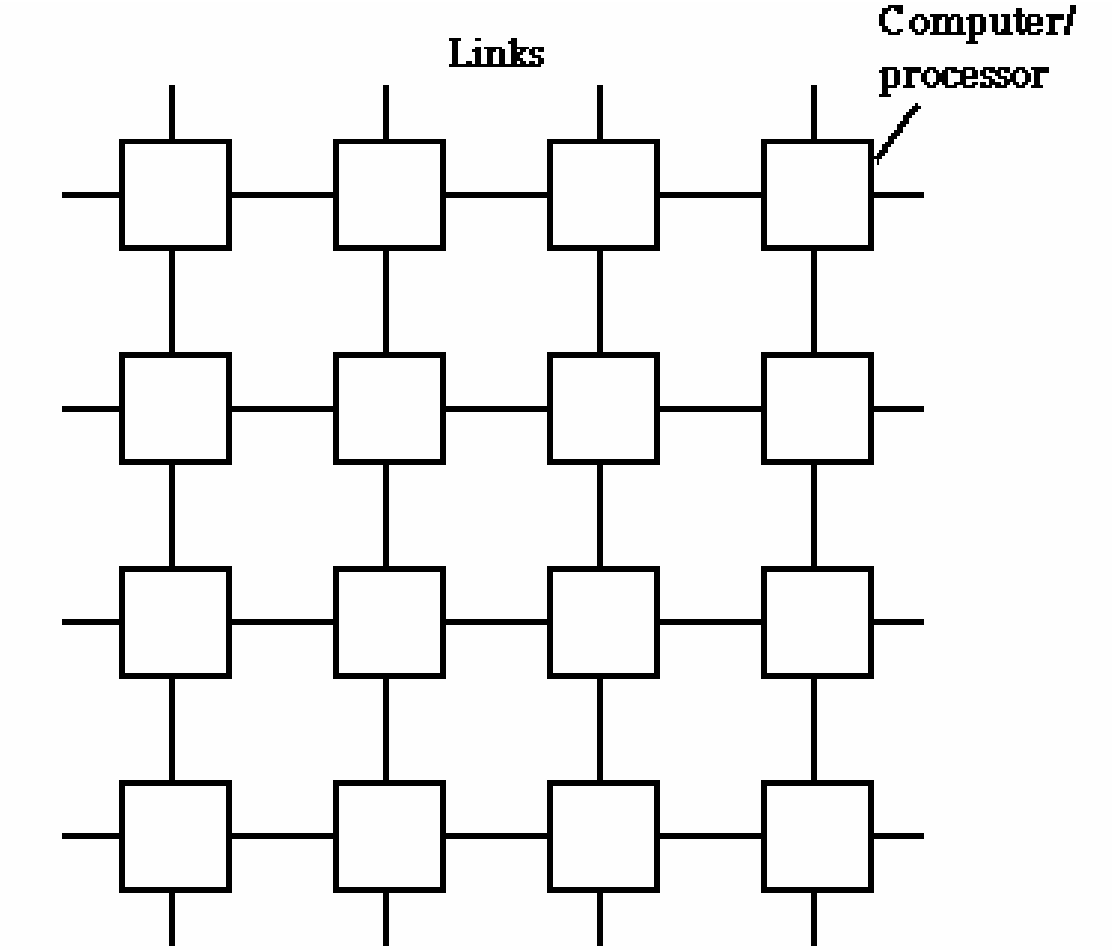


Figure 1.7 Two-dimensional array (mesh)

Topologías de interconexión

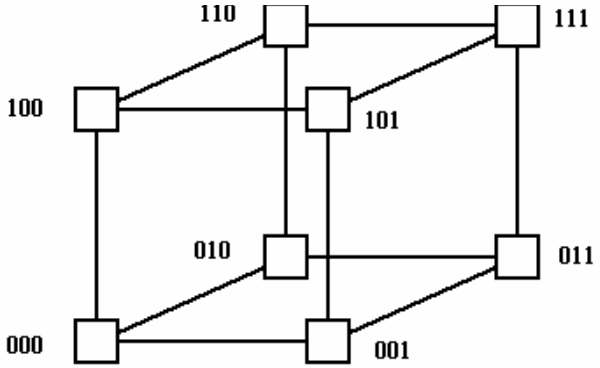


Figure 1.10 Three-dimensional hypercube

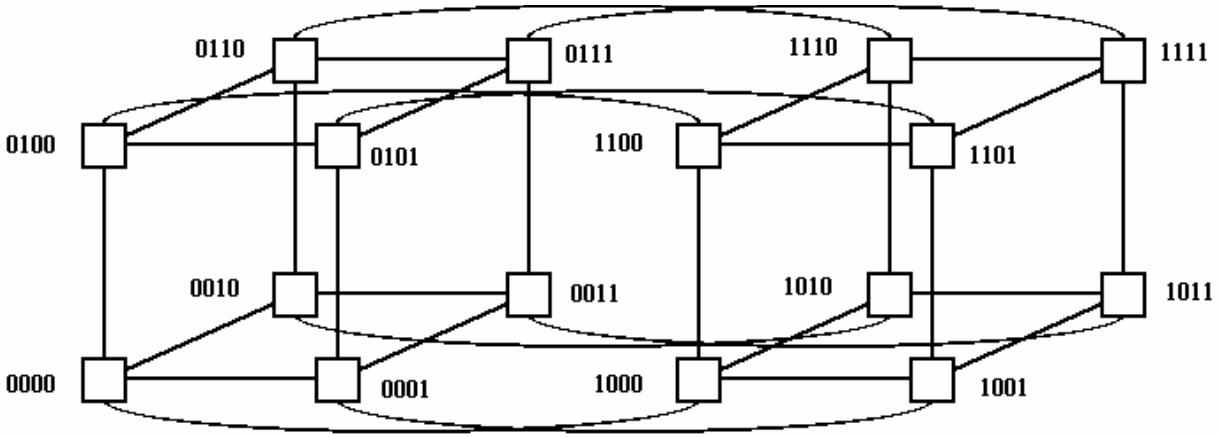


Figure 1.11 Four-dimensional hypercube

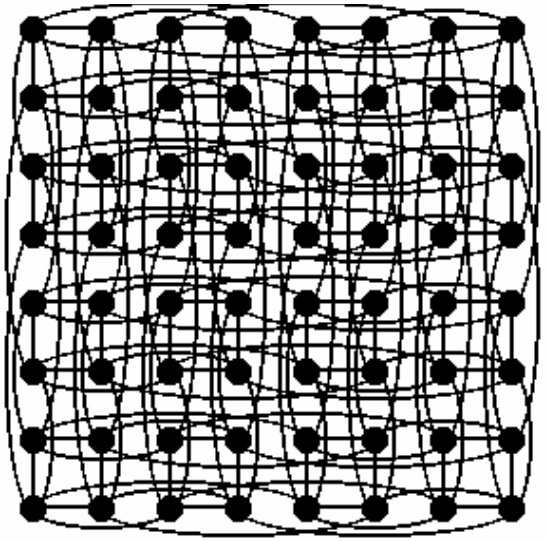


Figure 1.12 Six-dimensional hypercube laid out in one plane

Topologías de interconexión

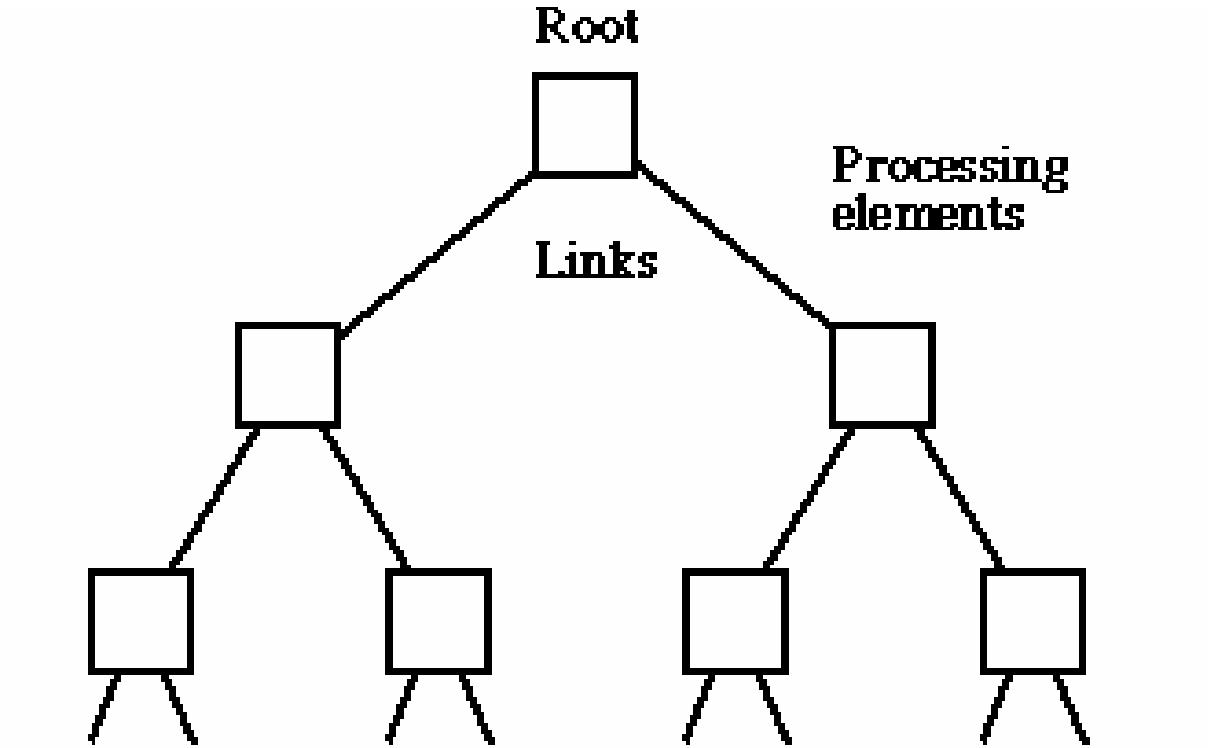


Figure 1.9 Tree structure

Topologías de interconexión estáticas

- Resumen de características

Table 2.1 A summary of the characteristics of various static network topologies connecting p nodes.

Network	Diameter	Bisection Width	Arc Connectivity	Cost (No. of links)
Completely-connected	1	$p^2/4$	$p-1$	$p(p-1)/2$
Star	2	1	1	$p-1$
Complete binary tree	$2\log((p+1)/2)$	1	1	$p-1$
Linear array	$p-1$	1	1	$p-1$
2-D mesh, no wraparound	$2(\sqrt{p}-1)$	$\sqrt{p}$	2	$2(p-\sqrt{p})$
2-D wraparound mesh	$2\lfloor\sqrt{p}/2\rfloor$	$2\sqrt{p}$	4	$2p$
Hypercube	$\log p$	$p/2$	$\log p$	$(p\log p)/2$
Wraparound k-ary d-cube	$d\lfloor k/2\rfloor$	$2k^{d-1}$	$2d$	dp

Topologías de interconexión

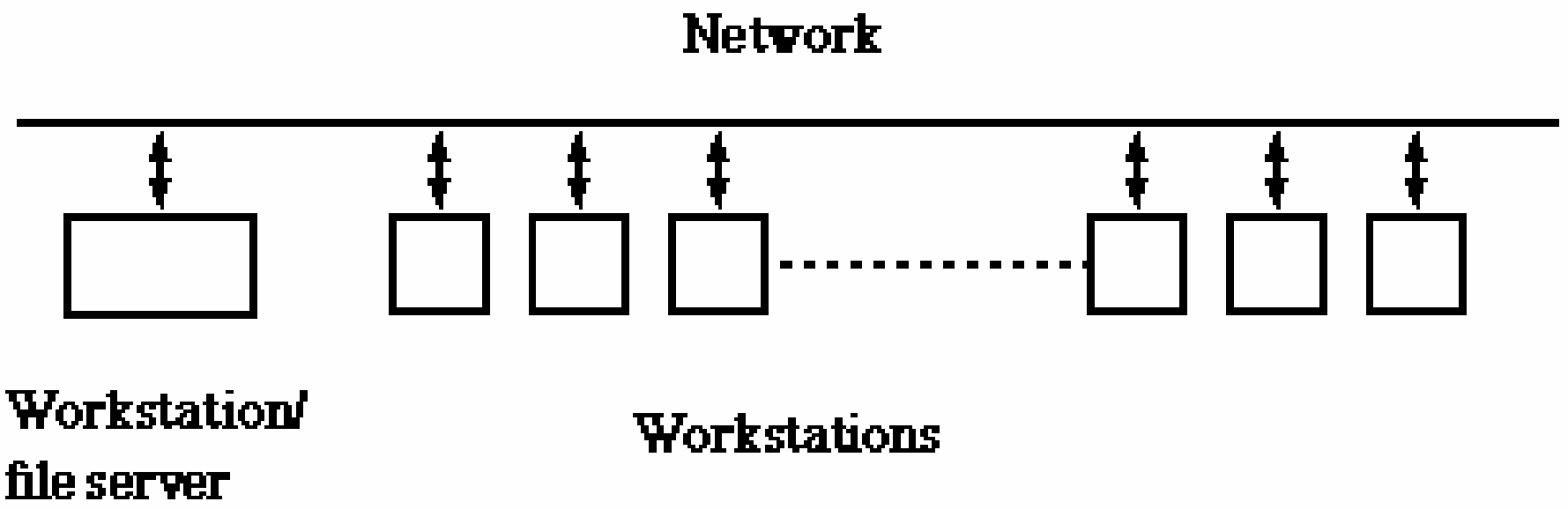


Figure 1.15 Ethernet type single wire network