

# IL MULTIPROCESSING

*Il problema: necessità di aumento della potenza di calcolo.*

La velocità di propagazione del segnale (20 cm/ns) impone limiti strutturali all'incremento della velocità dei processori (es: 10GHz → max 2 cm)

Tendenza attuale: **distribuire il calcolo tra più processori.**

***processori strettamente accoppiati:***

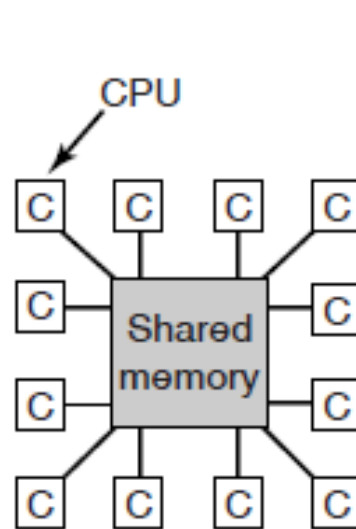
**sistemi multiprocessore:** tutte le CPU condividono clock e una memoria fisica comune; comunicazione attraverso memoria condivisa: Uniform Memory Access UMA (SMP, crossbar, . . . ), Non Uniform Memory Access NUMA (Cache Coherent-NUMA, No Cache-NUMA), Cache Only Memory Access COMA;

***processori debolmente accoppiati:***

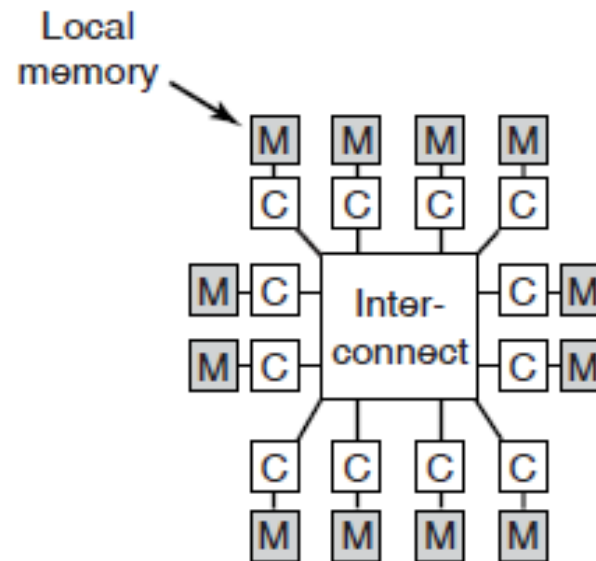
**sistemi multicalcolatore:** le CPU non condividono clock e/o memoria; ogni CPU ha la sua memoria privata e comunicano attraverso canali asincroni molto più lenti

**sistemi distribuiti:** reti.

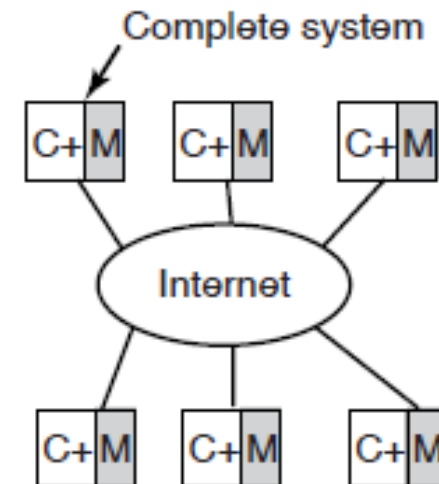
# MULTIPROCESSORE, MULTICOMPUTER, SISTEMA DISTRIBUITO



(a)



(b)



(c)

## Differenze:

- ☞ *costo*
- ☞ *scalabilità*
- ☞ *complessità di programmazione/utilizzo*

## MULTIPROCESSORS CLASSIFICATION

- ↳ A multiprocessors is made by two or more processors connected through a communication system.
  - ☞ Processors may be identical in terms of their functionality (**homogeneous multiprocessor**): in such a case any available processor can be used to run any process. Processors may be different (**heterogeneous multiprocessor**): only programs compiled for a given processor's instruction set could be run on that processor.
  - ☞ The way a processor access the memory may be the same for all processors (**uniform memory access or UMA**) or may be different for each processor (**non uniform memory access or NUMA**).
  - ☞ Even within a homogeneous multiprocessor, a process needing for a device attached to the private bus of a processor must run on that processor.
- ↳ If several identical processors are available, then **load sharing** can occur.
- ↳ From the general point of view, a processing structure can be classified according to different characteristics.
- ↳ The most accepted classification (**Flynn taxonomy**) considers the **control-flow** and the **data-flow** concepts: in fact, an elementary processing is based on a sequence of instructions (control flow), each of one operating on a couple of data (data flow).

## MULTIPROCESSORS CLASSIFICATION ACCORDING WITH FLYNN

A processing structure may be:

- ↳ Single Instruction flow and Single Data flow (**SISD**): that is the classical single CPU architecture of a Von Neumann machine;
- ↳ Single Instruction flow and Multiple Data flow (**SIMD**): that is the architecture of an array processor;
- ↳ Multiple Instruction flow and Single Data flow (**MISD**): that is the architecture of a pipeline processor;
- ↳ Multiple Instruction flow and Multiple Data flow (**MIMD**): the more complex architecture, which may, in turn, range from:
  - ↳ an architecture with a single common storage accessed by all the processors → a **tightly coupled architecture** usually named **true multiprocessor system**;
  - to
  - ↳ an architecture with a private storage for each processor → a **loosely coupled architecture** usually named **multicomputer** or **true distributed system**.

**Grande varietà di architetture** proposte e realizzate

Non sempre si riesce a sfruttarne le potenzialità adeguatamente

La vera difficoltà non è realizzare architetture parallele, ma sviluppare applicazioni parallele.

# TRUE MULTIPROCESSOR ARCHITECTURE

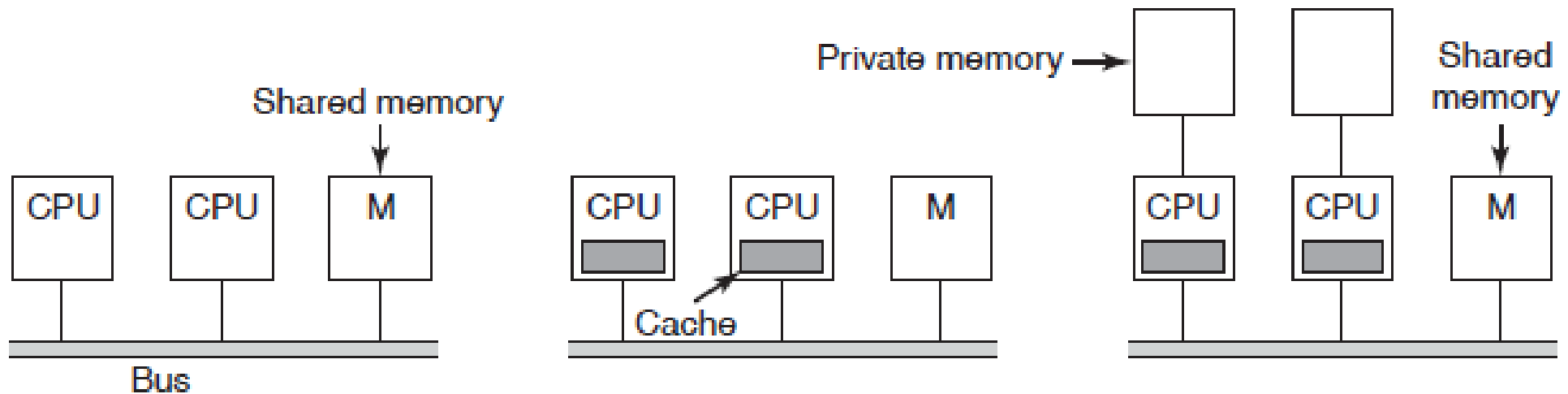
## Symmetric MultiProcessing (SMP-UMA)

Un multiprocessore vero è caratterizzato da due o più processori che condividono un unico clock ed un'unica memoria comune tramite un sistema di comunicazione che consenta ridottissimi tempi di accesso alla memoria.

Possibili organizzazioni del multiprocessore:

- **a bus comune**  
il sistema di comunicazione è costituito da un bus ed ogni processore esegue uno dei processi che risiede nella memoria comune;
- **a griglia di interconnessione (crossbar grid)**  
il sistema di comunicazione è costituito dalla griglia, a sua volta formata da  $n^2$  commutatori ad alta velocità, che consentono ad ognuno degli  $n$  processori di poter accedere ad uno degli  $n$  moduli in cui la memoria è suddivisa.

## THE COMMON BUS MULTIPROCESSORS ARCHITECTURE



La memoria ed il bus rappresentano dei “colli di bottiglia” dell’architettura, in quanto condivisi da tutti i processori.

Ciò rende inefficiente l’uso di un numero di processori superiore a qualche unità, così come avviene nelle architetture di mercato. **Limitata scalabilità** (max 16 CPU, solitamente meno di 8). Per andare oltre, si devono usare reti crossbar o omega, o si passa a NUMA.

Per ridurre l’accesso al bus ed alla memoria, ogni processore deve essere dotato di una cache memory in grado di aggiornare immediatamente il contenuto della memoria (**write through cache**) o di aggiornare il contenuto della cache quando quello della memoria cambia (Problemi di coerenza  $\Rightarrow$  uso di **bus snooping cache**).

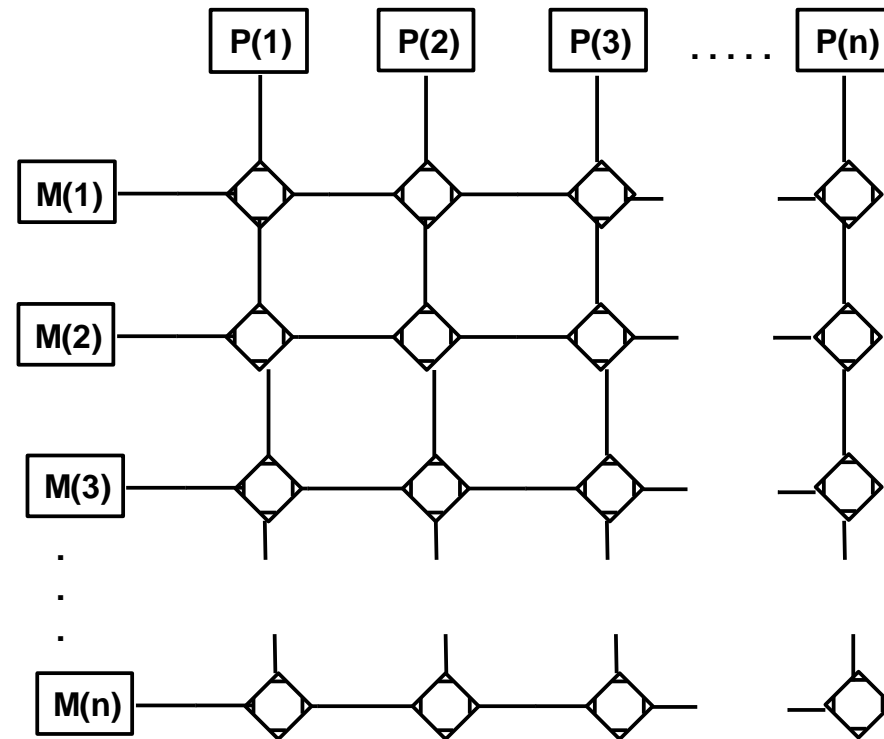
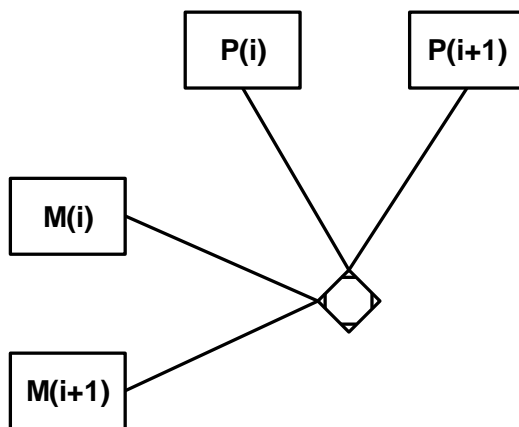
## THE CROSSBAR GRID MULTIPROCESSORS ARCHITECTURE

Il numero dei processori e dei moduli di memoria può arrivare al migliaio.

Il tempo di accesso alla memoria è funzione quadratica del numero di commutatori.

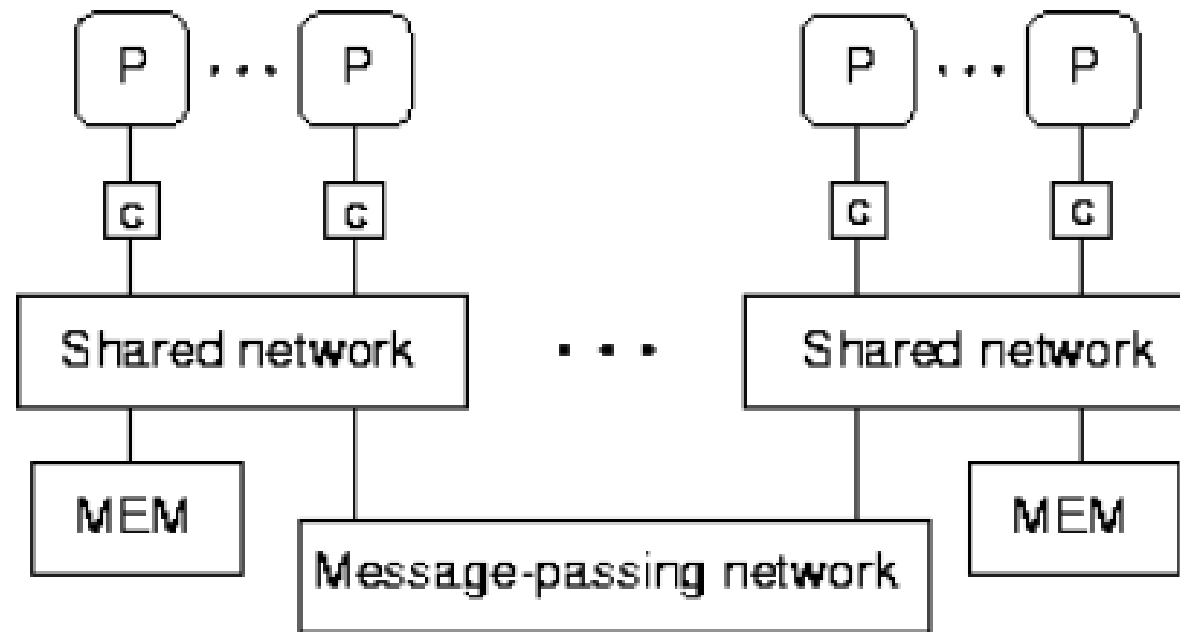
Il costo della griglia è alto.

Per ridurli si usano commutatori a 2 ingressi e 2 uscite.



## HARDWARE MULTIPROCESSORE: NUMA

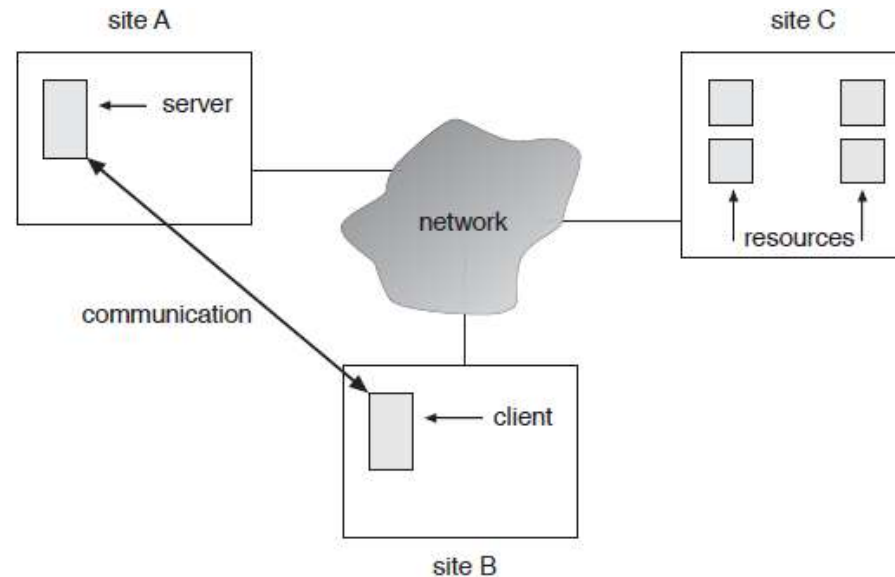
- ➡ Applicabili a 100+ processori. Due o più tipi di memorie:
  - *locale*: privata ad ogni CPU/gruppo SMP di CPU
  - *remota*: condivisa tra le CPU; tempo di accesso 2 ÷ 15 volte quello locale.
- ➡ *Reindirizzamento via rete/bus*, risolto in hardware (MMU)
- ➡ Eventualmente, *una cache locale per la memoria remota* (CC-NUMA)





## SISTEMI DISTRIBUITI (MULTI-COMPUTER)

Sono sistemi lascamente accoppiati, normalmente più orientati verso la comunicazione (= accesso a risorse remote), che al calcolo intensivo.



### Motivazioni per i sistemi distribuiti

- ✚ **Condivisione delle risorse**
  - condividere e stampare file su siti remoti
  - elaborazione di informazioni in un database distribuito
  - utilizzo di dispositivi hardware specializzati remoti
- ✚ **Accelerazione dei calcoli: bilanciamento del carico**
- ✚ **Affidabilità:** individuare e recuperare i fallimenti di singoli nodi, sostituire nodi difettosi
- ✚ **Comunicazione:** passaggio di messaggi tra processi su macchine diverse, similamente a quanto succede localmente.

## TRUE DISTRIBUTED SYSTEM ARCHITECTURE

Un sistema distribuito vero è caratterizzato da due o più processori, ciascuno con la propria memoria privata ed un proprio bus, che sono connessi tramite un sistema di comunicazione.

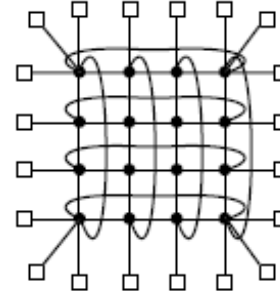
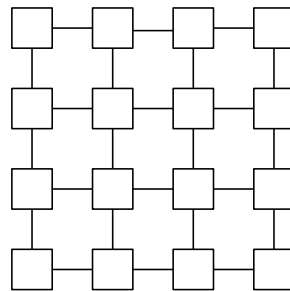
Possibili topologie del sistema distribuito:

- **a rete di interconnessione**

l'architettura in questione è la più comunemente usata, per connettere più calcolatori sia localmente (tramite una LAN) sia geograficamente (tramite una WAN): cfr. appunti “Introduzione alle reti di calcolatori”;

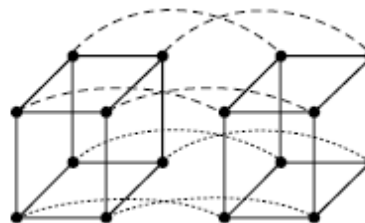
- **a matrice di interconnessione**

i calcolatori sono planarmente disposti secondo una matrice;



- **ad ipercubo** (molto usati perché diametro =  $\log_2$  nodi)

i calcolatori sono disposti spazialmente ai vertici di uno o più cubi; ogni vertice è connesso all'omologo vertice di un altro cubo. Il numero dei cubi stabilisce l'ordine dell'ipercubo.



## THE O.S. OF TRUE MULTIPROCESSORS

Le caratteristiche del SO di un multiprocessore vero sono sostanzialmente simili a quelle del SO di un singolo processore. Infatti la presenza della memoria comune consente l'utilizzo degli stessi meccanismi o algoritmi di comunicazione, sincronizzazione e gestione del deadlock già esaminati.

L'unica **sostanziale differenza** deriva dalla possibilità che più processi possano trovarsi nello stato di “run”.

Ciò richiede l'adozione di politiche di **scheduling** diverse da quelle considerate, sia per i processi applicativi che per quelli del SO stesso:

- ☞ associando ad ogni processore una specifica coda di “ready”: questa soluzione potrebbe portare alla non desiderabile situazione di un processore “idle” e di un altro sovraccarico di processi;
- ☞ prevedendo un'unica coda di “ready” per tutti i processi e facendo eseguire un processo su qualunque processore disponibile al momento. In tal caso è possibile organizzare due distinti meccanismi di scheduling:
  - ogni processore è autoschedulante (**symmetric multiprocessing**): ciò richiede l'adozione di meccanismi di accurata sincronizzazione dei processori nell'accesso a strutture dei dati condivise (la coda di “ready”);
  - un processore ha la funzione di scheduler degli altri (**asymmetric multiprocessing**), secondo una tipica struttura master-slave: questa soluzione, molto più semplice da realizzare, è però anche la più inefficiente.

## THE O.S. OF TRUE MULTIPROCESSORS

Indipendentemente dalla strategia di scheduling adottata, il *context switching* comporta, in un multiprocessore, un **significativo rallentamento** a causa del più intenso uso del sistema di connessione e della memoria comune, che sarà inizialmente richiesto dal processore interessato, fino a quando la cache memory non sarà stata aggiornata.

Per evitare tale rallentamento, nel caso di context switching relativo a processi utenti si preferisce in alcuni casi posporlo, **attendendo per una frazione di tempo** che il processo andato in wait torni rapidamente nello stato di ready, al fine di poterne riprendere l'esecuzione.

## THE O.S. OF TRUE DISTRIBUTED SYSTEMS

I SO per sistemi distribuiti possono suddividersi in due categorie principali:

### sistemi operativi per reti di calcolatori

- ✓ sono **più semplici da realizzare**, ma **più difficili da usare**: gli utenti possono accedere a risorse remote, facendo “login” remoto sul computer appropriato e trasferendo dati da esso;
- ✓ tipiche funzioni di tali sistemi sono:
  - **login remoto** tramite comando **telnet**: viene quindi stabilita una connessione tra il calcolatore locale e quello remoto e, dopo aver fornito lo user-name e la password di identificazione, è possibile effettuare tutte le operazioni consentite sul calcolatore remoto;
  - **file transfer remoto** tramite un opportuno protocollo (**ftp**), che consente solo operazioni di copia di file (tramite i comandi **get**, **put** ed **ls**) ad utenti forniti di user-name e password riconosciuti e che conoscano la dislocazione dei file. Ad un utente non dotato di diritti di accesso è invece consentita, qualificandosi come utente **anonymous**, la sola copia dei file di pubblico dominio;

### sistemi operativi distribuiti

- ✓ sono **più complessi da realizzare**, ma **molto più semplici da usare** in quanto offrono una pluralità di servizi.