Curs 1

Programare Paralela si Distribuita

Continutul cursului

(realizat si pe baza pe http://grid.cs.gsu.edu/~tcpp/curriculum/)

Teoretic

- Notiuni introductive: arhitecturi, concurenta, paralelism
- Etape in dezvoltarea programelor paralele
- Evaluarea performantei programelor paralele
- Modele de programare paralela
  - Diferenta intre cele bazate pe memorie partajata si memorie distribuita
- *Patterns*
  - Pt programare paralela
  - Pt programare distribuita

Practic

- Java threads (low level API)
- C++ (>=C++11) threads
- High-level API: pachete Java-> java.util.concurrent packages.
- Java streams

- OpenMP (C++)
- CUDA (C++)
- MPI –Message Passing Interface
  - exemplificari C, C++

Bibliografie

- Ian Foster. Designing and Building Parallel Programs, Addison-Wesley 1995.
- Berna L. Massingill, Timothy G. Mattson, and Beverly A. Sanders,Addison A Pattern Language for Parallel Programming. Wesley Software Patterns Series, 2004.
- Michael McCool, Arch Robinson, James Reinders, Structured Parallel Programming: Patterns for Efficient Computation," Morgan Kaufmann,, 2012 .
- D. Culler, J. Pal Singh, A. Gupta. Parallel Computer Architecture: A Hardware/Software Approach. Morgan Kaufmann. 1998.
- Grama, A. Gupta, G. Karypis, V. Kumar. Introduction to Parallel Computing, Addison Wesley, 2003.
- D. Grigoras. Calculul Paralel. De la sisteme la programarea aplicatiilor. Computer Libris Agora, 2000.
- V. Niculescu. Calcul Paralel. Proiectare si dezvoltare formala a programelor paralele. Presa Univ. Clujana, 2006.
- B. Wilkinson, M. Allen, Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers, Prentice Hall, 2002
- A. Williams. C++ Concurrency in Action PRACTICAL MULTITHREADING. Manning Publisher.2012.
- Tutoriale Java: http://docs.oracle.com/javase/tutorial/essential/concurrency/further.html
- C++11 http://en.cppreference.com/w/
- OpenMP: http://openmp.org/
- MPI: http://www.mpi-forum.org/

Evaluare

- Laborator
  - (30%) NL->Programe/proiecte
  - (5%) NS-> Exercitii in timpul laboratorului (“in class”)

In sesiune

- (25%) NT -> Test practic
- (40%) NE -> Examen teoretic NE>=4.5

Procesare Paralela

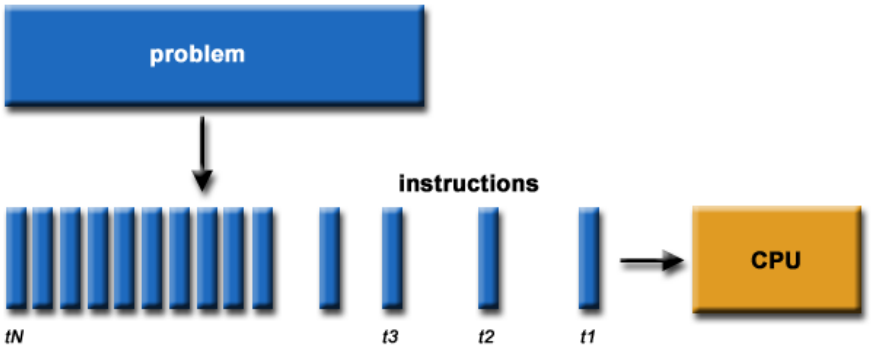
- **Un *calculator paralel* este un calculator (sistem) care foloseste multiple elemente de procesare simultana intr-o maniera cooperativa pentru a rezolva o problema computationala.**

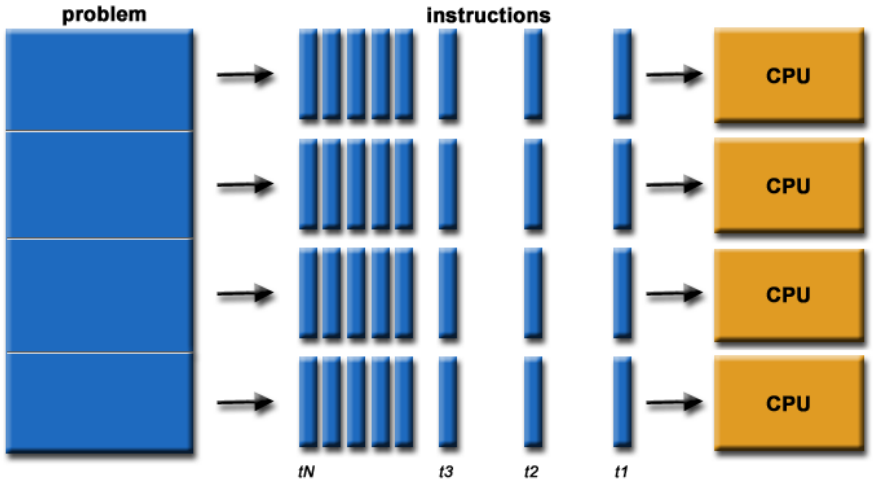
- Procesarea Paralela include tehnici si tehnologii care fac posibil calculul in paralel
  - Hardware, retele, SO, biblioteci, limbaje, compilatoare, algoritmi …

- PERFORMANTA
  - *Parallelism is very much about performance!*

Calcul Serial vs. Paralel

(images from **Introduction to Parallel Computing** *Blaise Barney*)





“It would appear that we have reached the limits
of what it is possible to achieve with computer technology,
although one should be careful with such statements,
as they tend to sound pretty silly in 5 years. “

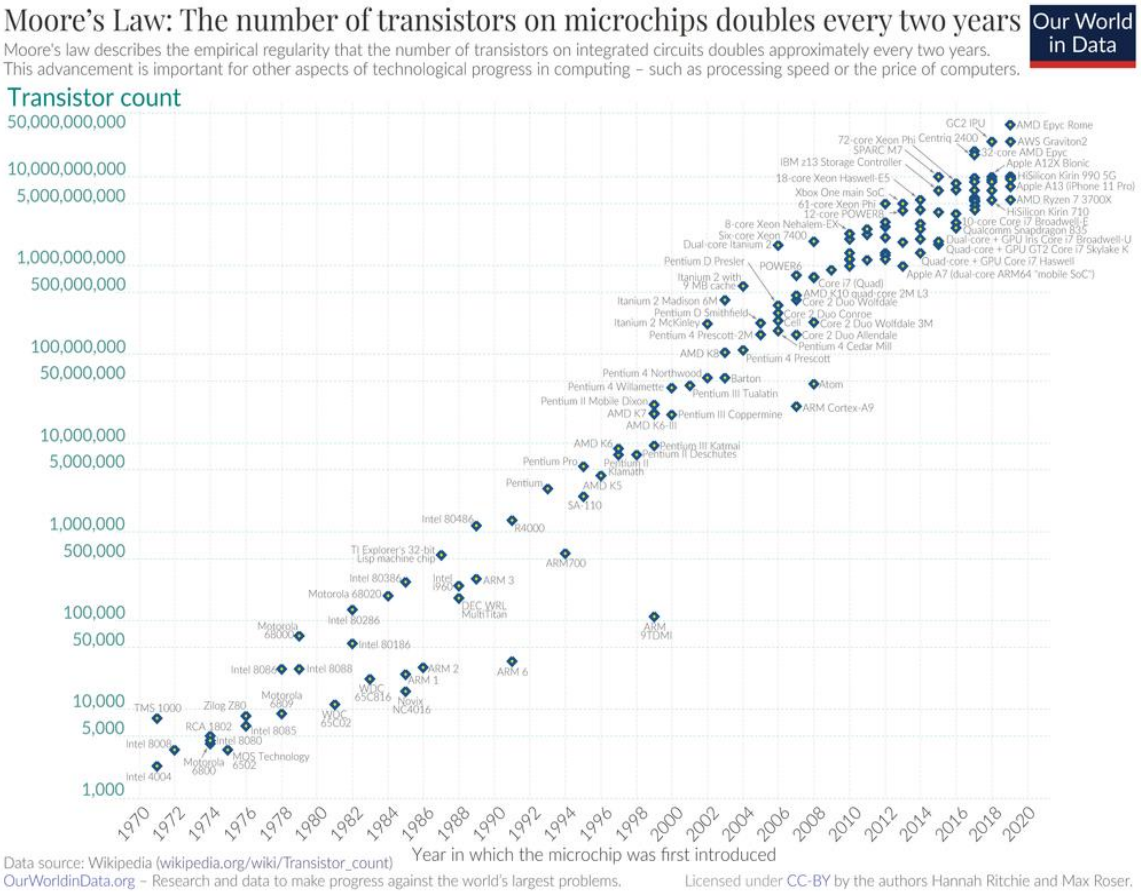
(John von Neumann, 1949)

Limite ale programarii seriale

- Viteza de transmisie –
  - Viteza luminii (30 cm/nanosecond),
  - limita de transmisie pe fir de cupru (9 cm/nanosecond).

- Limitarea miniaturizarii – numar de trazistori pe chip.
  - **Legea lui Moore**: numărul de tranzistori care pot fi plasati un circuit integrat (per square inch chip) se dubleaza la fiecare 2 ani.
  - Impune costuri mari.

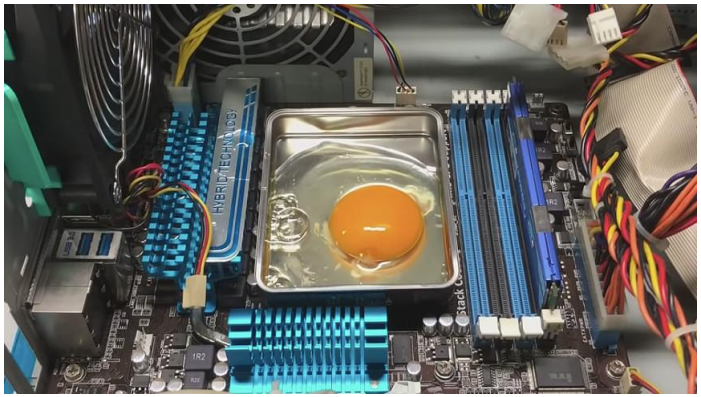
- Limitari economice



Istoric

- Cresterea performantei procesor prin cresterea frecventei ceasului CPU (CPU clock frequency)
  - Riding Moore's law
- Probleme : incalzirea puternica a chipurilor!
  - Frecventa ceas mai mare $\Rightarrow$ consum electric mai mare

  ❍ *Frying an egg on a computer*



- Solutie – adugare mai multor core-uri pt a ajunge la performanta dorita
  - Se pastreaza frecventa de ceas la fel sau chiar micsorare
  - nu creste consumul.

Niveluri de paralelism

1. **paralelism la nivel de job:**

- intre joburi;
- intre faze ale joburilor;

2. **paralelism la nivel de program:**

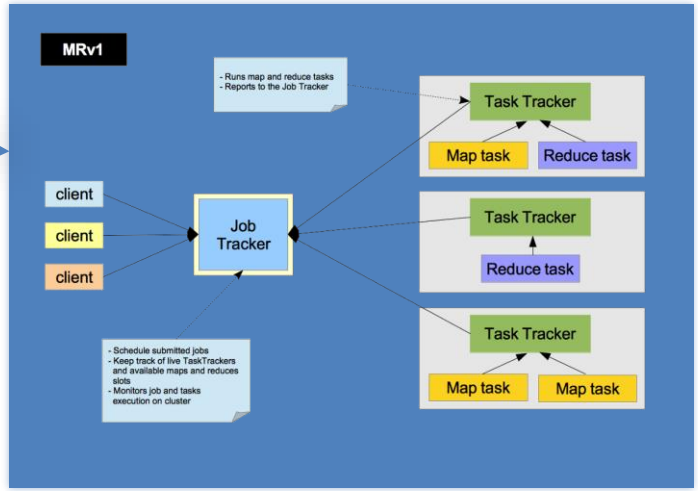
- intre părţi ale programului;
- in anumite cicluri;

3. **paralelism la nivel de instrucţiune:**

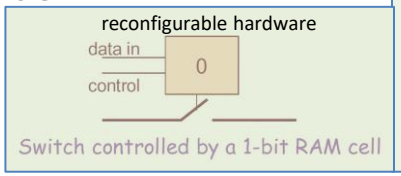
- *Instruction-level parallelism (ILP)*

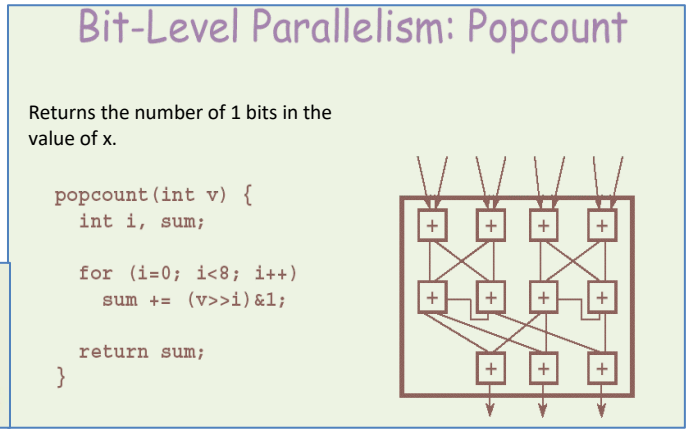
4. **paralelism la nivel aritmetic şi la nivel de bit:**

- intre elemente ale unei operaţii vectoriale;
- intre circuitele logicii aritmetice.



ILP allows the compiler and the processor to overlap the execution of multiple instructions or even to change the order in which instructions are executed.





Hardware parallelism

- Arhitecturile curente se bazeaza tot mai mult pe paralelism la nivel hardware
  pentru a imbunatati performanta

Metrici performanta procesor=>

- ✓ *number of instruction issues per machine cycle (IPC)*
- ✓ *number of instructions per second = IPC x clock rate (cycles per sec given in Hertz)*
- ✓ *floating point operations per second*

(DAR performanta depinde si de software si de configuratia intregii masini – !!! memorie (memory hierarchy))

Modalitati de imbunatatire:

- Multiple execution units
- Pipelined instructions
- Multi-core

Paralelism <-> Concurenta

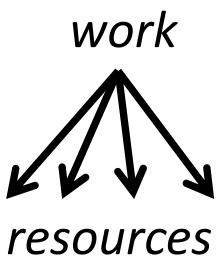
- Consideram mai multe taskuri care trebuie executate pe un calculator
- Taskurile se considera a fi ***pur paralele*** daca:
  - Se pot executa in acelasi timp (*parallel execution*)
- Dependente -> executie concurenta:
  - Un task are nevoie de rezultatele altora;
  - Un task trebuie sa se execute dupa ce o anumita conditie e indeplinita
  - Mai multe taskuri incearca sa foloseasca aceeasi resursa

    => Forme de sincronizare trebuie folosite pentru a satisface conditiile/dependentele
- Concurenta este fundamentala in *computer science*
  - Sisteme de operare, baze de date, networking, …

Paralelism vs. Concurenta

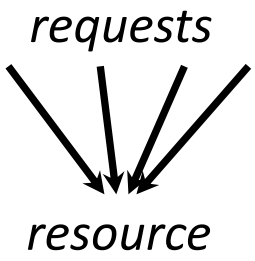
Paralelism:

Se folosesc mai multe resurse pentru a rezolva o problema mai rapid



Concurenta:

Gestiunea corecta si eficienta a accesului la resurse comune



Obs:

- Se pot folosi *threaduri* sau procese in ambele cazuri
- Daca un calcul paralel necesita acces la resurse comune atunci este nevoie sa se gestioneze corect concurenta

$\Rightarrow$ Paralelismul poate implica concurenta

$\Rightarrow$ Concurenta nu exista fara paralelizare

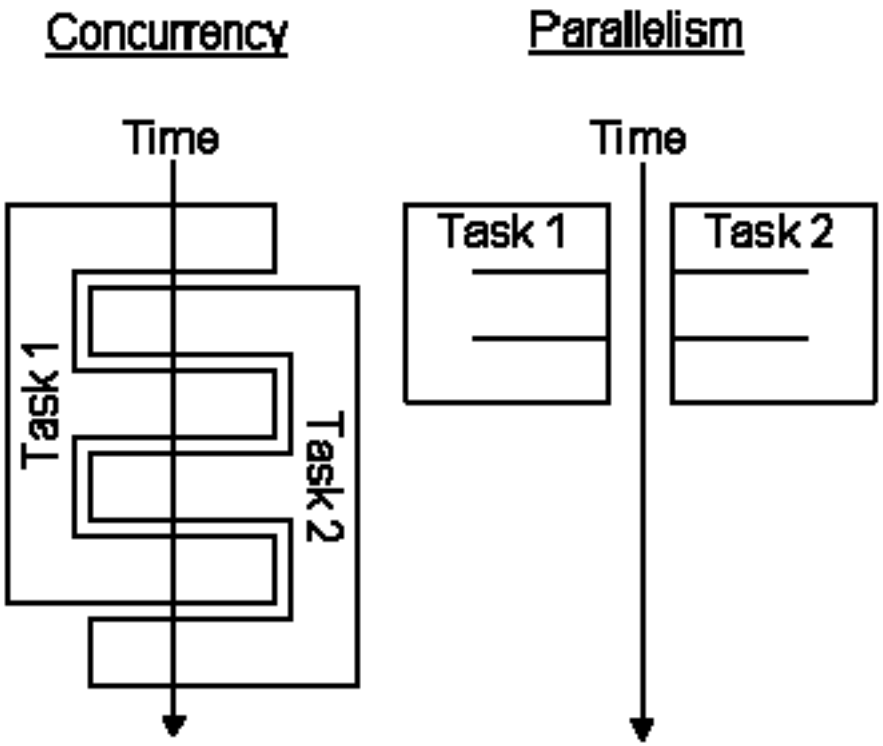
Concurenta si Paralelism

- Concurent vs. paralel
- Executie Paralela:
  - Taskurile se executa efectiv in acelasi timp;
  - Este necesara existenta de multiple resurse de calcul
- **Paralelism = concurenta + hardware “paralel”**

Paralelism

- Exista mai multe niveluri de *software parallelism* :
  - Procese, threads, routine, instructiuni, …
- Trebuie sa fie suportate de resursele hardware
  - Procesoare, nuclee(cores), … (executia instructiunilor)
  - Memorii, DMA, retele , … (operatii asociate)

o abordare simplista - (*debatable)*



http://www.java-programming.info/tutorial/pdf/java/11-Java-Multithreaded-Programming.pdf

De ce sa folosim programare paralela?

- Motive primare:
  - Timp de calcul mai rapid (*response time*)
  - Rezolvarea problemelor ‘mari’ de calcul (in timp rezonabil de calcul)
- Motive secundare:
  - Folosirea efectiva a resurselor de calcul
  - Costuri reduse
  - Reducerea constrangerilor asociate memoriei
  - Limitarile masinilor seriale

- **Paralelism = concurenta + hardware ‘paralel’+ performanta**

- **Rezolvarea problemelor dificile, mari:**
  - "Grand Challenge" (en.wikipedia.org/wiki/Grand_Challenge) problems requiring PetaFLOPS and PetaBytes of computing resources.
  - Web search engines/databases processing millions of transactions per second

- **Folosirea resurselor non-locale:**
  - SETI@home (setiathome.berkeley.edu) uses over 330,000 computers for a compute power over 528 TeraFLOPS (as of August 04, 2008)
  - Folding@home (folding.stanford.edu) uses over 340,000 computers for a compute power of 4.2 PetaFLOPS (as of November 4, 2008)

Grand Challenge Problems

- The solution or simulation of fundamental problems in science and engineering, with a strong scientific and economic impact, known as Grand Challenge Problems (GCPs), have been the driving force for Parallel Computing.
- Typically, GCPs simulate phenomena that cannot be measured by experimentation:

✓ I Global Climate modeling
✓ I Biology: genomics; protein folding, drug design
✓ I Astrophysical modeling
✓ I Computational Chemistry
✓ I Earthquake and structural modeling
✓ I Computational fluid dynamics (airplane design)
✓ I Crash simulation
✓ I Financial and economic modeling

New Data-Intensive Applications

- Currently, large volumes of data data are produced and their processing and analysis also require high performance computing.
- Some examples:

✓ I Data mining
✓ I Web search
✓ I networked video
✓ I Video games and virtual reality
✓ I Computer aided medical diagnosis
✓ I ...

- Similarly, data is collected and stored at enormous speeds(GByte/hour).
- Some examples:

✓ I sensor data streams
✓ I telescope scanning the skies
✓ I micro-arrays generating gene expression data

Directii in procesarea paralela

- Arhitecturi paralele
  - Necesitati Hardware
  - Computer system design
- Sisteme de operare (Paralelism/concurenta)
- Gestionarea aspectelor sistem pentru un calculator paralel
- Programare paralela
  - Biblioteci (low-level, high-level)
  - Limbaje
  - Medii de dezvoltare
  - Software
- Algoritmi Paraleli
- Evaluarea performantei programelor paralele
- Testarea vs. asigurarea corectitudinii
- *Parallel tools:*
  - Performanta, analize, vizualizare, …

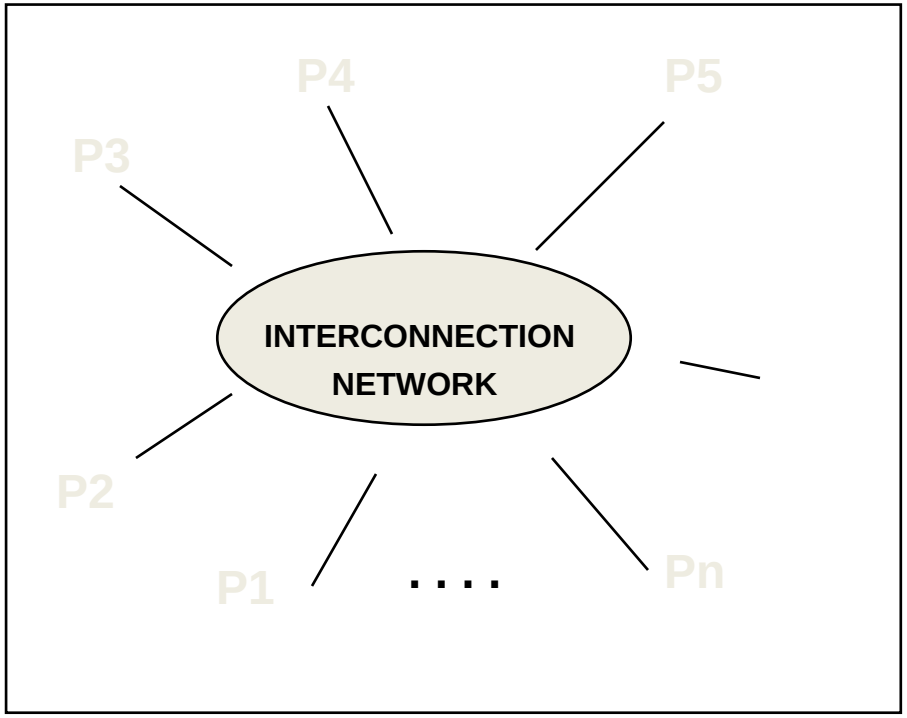
De ce sa studiem programare paralela?

- Arhitecturi de calcul
  - Inovatiile conduc la noi modele de programare
- Convergenta tehnologica
  - "killer micro" este peste tot
  - Laptop-urile si supercomputere sunt fundamental similare
  - Trend-urile actuale conduc la convergenta abordarilor diverse
- Trendurile tehnologice fac calculul paralel inevitabil
  - Multi-core processors!
  - Acum orice sistem de calcul este paralel
- Intelegerea principiilor fundamentale !!!
  - Programare, comunicatii, memorie, …
  - Performanta
- **"Parallelism is the future of computing" -** Blaise Barney
  - M. Andrews, J. S. Walicki. "Concurrency and parallelism—future of computing" in Proceeding of ACM '85 Proceedings of the 1985 ACM annual conference on The range of computing : mid-80's perspective. pp.224-231.

Inevitabilitatea Procesarii Paralele

- Cerintele pt aplicatii
  - Necesitatea uriasa de cicluri de calcul
- Trenduri tehnologice
  - Procesare si memorie
- Trenduri Architecturale
- Factori economici
- Treduri actuale:
  - *Today's microprocessors have multiprocessor support*
  - *Servers and workstations available as multiprocessors*
  - *Tomorrow's microprocessors are multiprocessors*
  - *Multi-core is here to stay and #cores/processor is growing*
  - *Accelerators (GPUs, gaming systems)*

Programare paralela vs. Programare distribuita



TiPURI DE MULTIPROCESARE
PARALLEL DISTRIBUTED

ASPECTE TEHNICE

- **PARALLEL COMPUTERS (- IN MOD UZUAL) LUCREAZA BAZAT PE**
  - ***CUPLARE STRANSA*,**
  - **in general bazate pe SINCRONICITATE,**
  - **CU UN SISTEM DE COMUNICATIE FOARTE RAPID SI FIABIL**
  - **Spatiu unic de adresare (intr-o masura mare)**

- **DISTRIBUTED COMPUTERS**
  - **MAI INDEPENDENTE,**
  - **COMUNICATIE MAI PUTIN FRECVENTA SI mai putin RAPIDA (ASINCRONA)**
  - **COOPERARE LIMITATA**
  - **NU EXISTA CEAS GLOBAL**
  - **"Independent failures"**

SCOPURI

- **PARALLEL COMPUTERS COOPEREAZA PENTRU A REZOLVA MAI EFICIENT PROBLEME DIFICILE**

- **DISTRIBUTED COMPUTERS AU SCOPURI INDIVIDUALE SI ACTIVITATI PRIVATE.**

DOAR UNEORI INTERCOMUNICAREA ESTE NECESARA

PARALLEL COMPUTERS: COOPERARE IN SENS "*POZITIV*"

DISTRIBUTED COMPUTERS: COOPERARE IN SENS "*NEGATIV*" --
DOAR ATUNCI CAND ESTE NECESARA

In general …

Aplicatii paralele

Suntem interesati sa rezolvam problemele *mai rapid* in paralel

Aplicatii distribuite

Suntem interesati sa rezolvam anumite probleme specifice :

- **COMMUNICATION SERVICES**
  ROUTING
  BROADCASTING

- **MAINTENANCE OF CONTROL STUCTURE**
  TOPOLOGY UPDATE
  LEADER ELECTION

- **RESOURCE CONTROL ACTIVITIES**
  LOAD BALANCING
  MANAGING GLOBAL DIRECTORIES

Sistemele Distribuite

-pot fi folosite pentru -

- Aplicatii distribuite implicit
  - BD Distribuite, rezervari bilete avion/etc. sistem bancar
- Informatii partajate intre useri
- Partajare resurse
- Raport cost / performanta mai bun pt aplicatii paralele
  - Pot fi folosite eficient pt. aplicatii cu granularitate mare(*coarse-grained*) si/sau pt aplicatii paralele de tip *embarrassingly parallel applications*
- Fiabilitate (*Reliability)*
- Scalabilitate
  - Cuplare slaba (Loosely coupled connection) ; hot plug-in
- Flexibilitate
  - Reconfigurare sistem pentru a intruni cerintele

Performanta/Scalabilitate

Spre deosebire de sistemele paralele cele distribuite implica:
- mediu mai putin rapid de transfer al datelor (retea mai putin rapida)
- Heterogenitate

Solutii:

- Procesare *batch* a mesajelor:
  - Se evita interventia SO pt fiecare transfer de mesaj.
- Cache data
  - Se evita repetarea transferului aceleiasi date
- Evitatea entitatilor si a algoritmilor centralizati
  - Evitarea saturarii retelei
- Realizare operatii “post” la nivelul clientului
  - Evitarea traficului intens intre clienti si servere
- ….

Securitate

- Nu exista doar un singur punct de control
- Probleme:
  - Mesaje, furate, modificate, copiate, …
    - Solutie : folosire Criptografie
  - Failures
    - Fault Tolerance solutions

Un punct de vedere…

Parallel v.s. Distributed Systems

(from M. FUKUDA CSS434 System Models)

	Parallel Systems	**Distributed Systems**
Memory	**Tightly coupled shared** memory UMA, NUMA	**Distributed** memory Message passing, RPC, and/or used of distributed shared memory
Control	**Global clock** control SIMD, MIMD	**No global clock** control Synchronization algorithms needed
Processor interconnection	Order of **Tbps** Bus, mesh, tree, mesh of tree, and hypercube (-related) network	Order of **Gbps** Ethernet(bus), token ring and SCI (ring), myrinet(switching network)
Main focus	**Performance** Ex. - Scientific computing	Performance(**cost and scalability**) **Reliability/availability** **Information/resource sharing**