Curs 1

Programare Paralela si Distribuita

Continutul cursului

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

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
  - (40%) NL->Programe/proiecte NL >=4.5
  - (10%) NS-> Exercitii in timpul laboratorului ("in class")
- (10%) NT -> Test practic
- (40%) NE -> Examen teoretic NE>=4.5

- Informatii curs
  - http://www.cs.ubbcluj.ro/~vniculescu/didactic/PPD/CursPPD.html

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.**

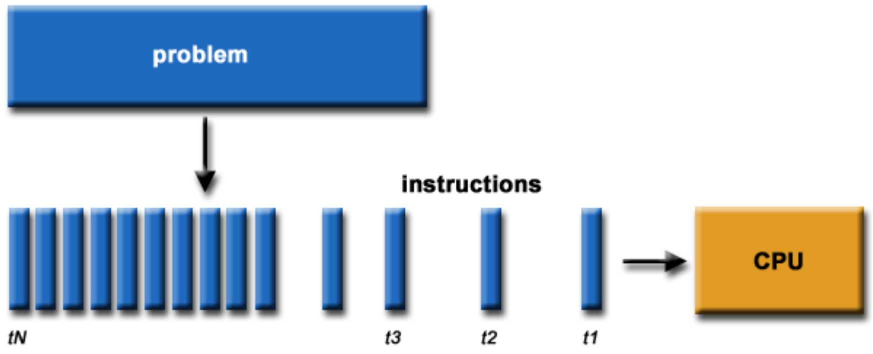
- Procesarea Paralela include tehnici si tehnologii care fac posibil calculul in paralel
  - Hardware, retele, SO, biblioteci, limbaje, compilatoare, algoritmi …

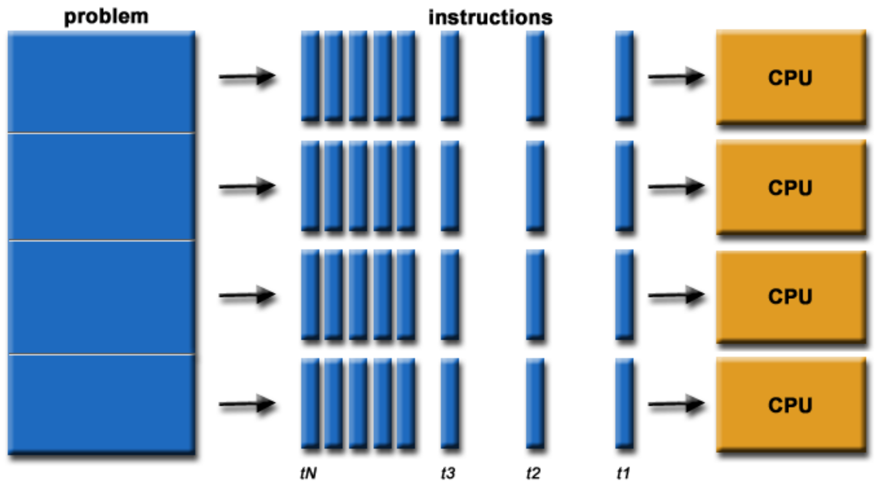
- Paralelismul este natural.

- 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. “

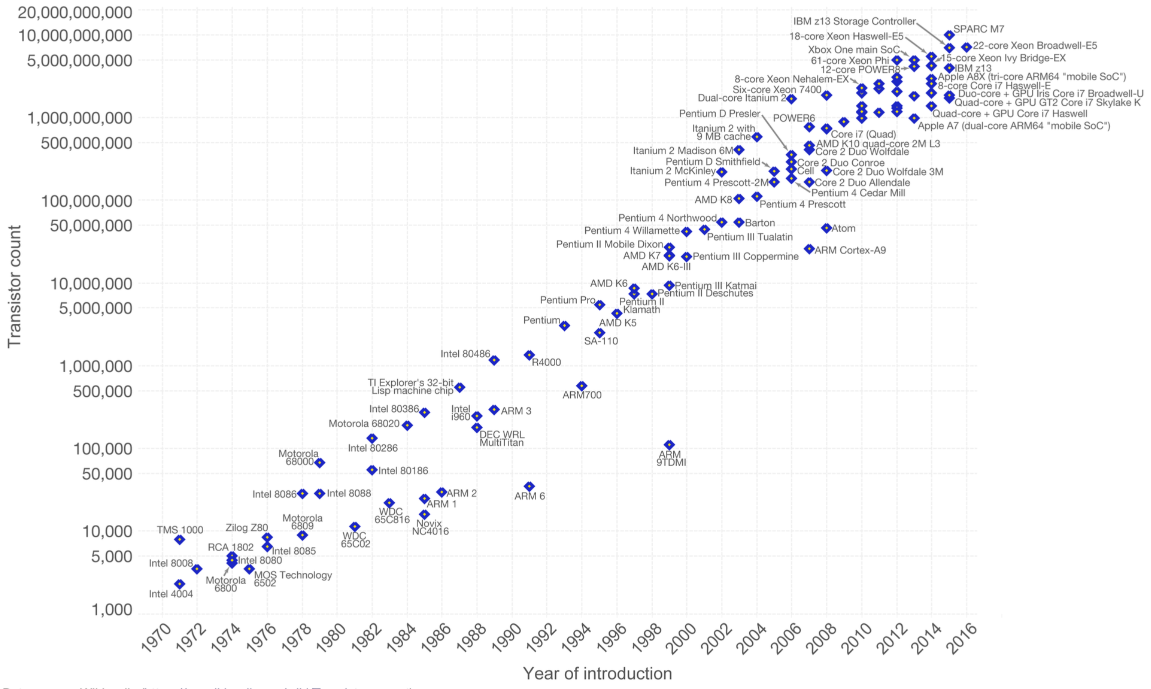
(John von Neumann, 1949)

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 pe un singur circuit integrat (per square inch chip) se dubleaza la fiecare 2 ani.
  - Impune costuri mari.
- Limitari economice

Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

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 ⇒ consum electric mai mare

  (Pentium 4 heat sink ❍ Frying an egg on a Pentium 4)

- 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:**

- intre diferite faze de execuţie ale unei instrucţiuni;

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

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

- Arhitecturile curente se bazeaza tot mai mult pe

paralelism la nivel hardware pentru a imbunatati performanta :

  - Multiple execution units
  - Pipelined instructions
  - Multi-core

Paralelism <-> Concurenta

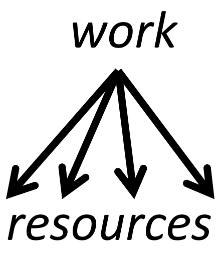
- Consideram mai multe taskuri care trebuie executate pe un calculator
- Taskurile se considera a fi ***pur paralele*** daca:
  - Se pot executa in acelasi timp (*parallel execution*)
- Dependente -> executie concurenta:
  - Un task are nevoie de rezultatele altora;
  - Un task trebuie sa se execute 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

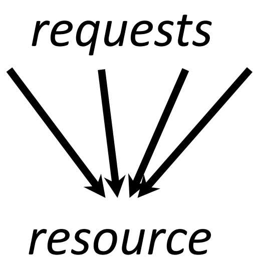
Paralelism:

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

=> Paralelismul poate implica concurenta

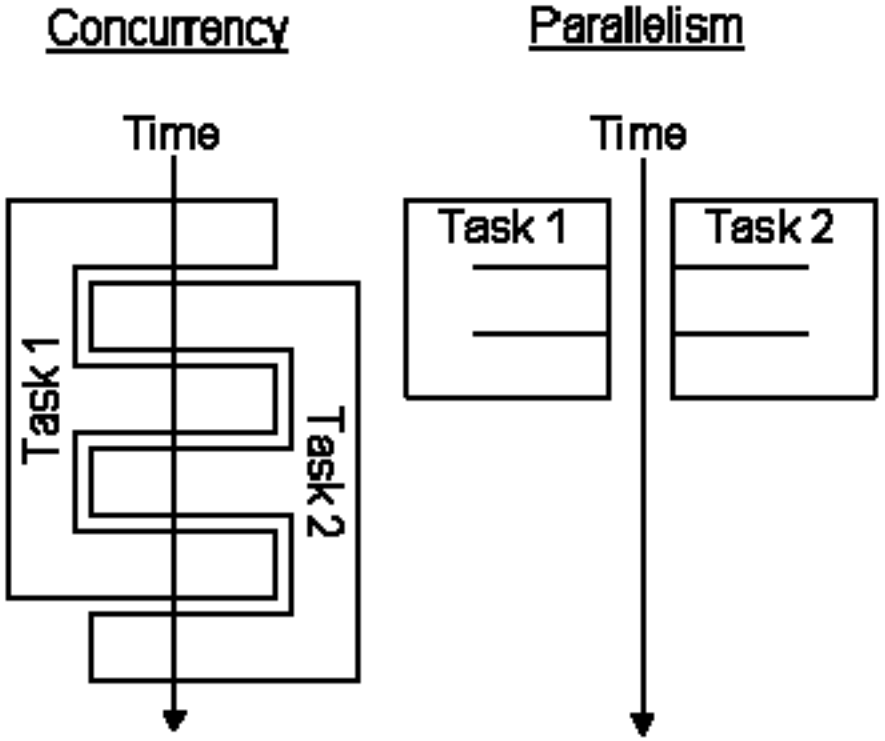
Concurenta si Paralelism

- 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 paralelism:
  - 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)

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)*

exemple…

- Procesor AMD Ryzen Threadripper 1950X ~ 1300USD (2017)
  - Memorie Cache 40 MB
  - Frecventa procesor (MHz) 3500
  - Turbo Boost pana la 4000 MHz
  - Numar nuclee 16 Nuclee =>32 Threads

- Intel® Xeon® Processor E7 v4 Family ~ 8000USD (2017)
  - # of Cores=24
  - # of Threads=48
  - Processor Base Frequency=2.40 GHz
  - Max Turbo Frequency=3.40 GHz
  - Cache= 60 MB

UBB CLUSTER – IBM Intelligent Cluster

- Hybrid architecture
  - HPC system +
  - private cloud

HPC – IBM NextScale

- Rpeak 62 Tflops, Rmax 40 Tflops
- 68 noduri NX360 M5, din care
  - 12 nodes with 2 GPU Nvidia K40X,
  - 6 nodes with Intel Phi
- 2 processors E5-2660 v3 with 10Cores per node
- 128 GB RAM per node, 2 HDD SATA de 500 Gb / node
- Subscription rate 1:1 between nodes based on Switch: IB Melanox SX6512 with 216 ports
- Storage NetApp E5660, 120 HDD SAS cu 600 Gb/Hdd => total 72Tb
  - IBM GPFS 4.x -parallel file system
- IBM TS3100 Tape library for data archivation
- Operating systems on each node : RedHat Linux 6 with subscription
- Management Software: IBM Platform HPC 4.2

Private Cloud – IBM Flex System

- 10 virtualization servers Flex System x240
  - 128 Gb RAM / server
  - Procesoare 2 x Intel Xeon E5-2640 v2 / server
  - 2 x SSD SATA 240 Gb / server
- 1 management server
- Software for private cloud: IBM cloud manager with OpenStack 4.2
- Software for monitorizing and management: IBM Flex System Manager software stack
- Virtualization software: Vmware vSphere Enterprise 5.1