CURS 0x09

TIPURI DE PARALELISM

Instruction-level parallelism (ILP)

- **Instruction Pipelining**
- **Register Renaming**
- **Speculative Execution**
- **Branch Prediction**
- **Value Prediction**
- **Memory Dependence Prediction**
- **Cache Latency Prediction**
- **Out-of-order Execution**
- **Dataflow Analysis/Execution**

Task parallelism (ILP)

- **multi-thread**
- **multi-process**

PERFORMANȚA MULTI-CORE

- **ce se întâmplă dacă avem un sistem multi-core?**
  - putem rula mai multe programe simultan
    - același program, instanțe diferite
    - diferite programe
  - putem rula un program mai eficient (paralelizând părți ale sale)
    - presupunem că avem *s* procesoare
    - presupunem că *p*% din program poate beneficia teoretic de paralelizare/îmbunătățire (unele secțiuni de cod sunt doar secvențiale, acolo nu se poate face nimic)
    - **legea lui Amdahl** (speed-up *S*):

$$S = \frac{1}{(1-p) + \frac{p}{s}}$$

    - **legea lui Gustafson** (speed-up *S*):

$$S = 1 - p + \delta p$$

  - de multe ori, implementările paralele au nevoie să comunice date (asta poate câteodată domina calculul)

Ierarhia memoriei

t(RAM) = 50 ns
t(L1) = 1 ns si miss rate m(L1) = 10%
t(L2) = 5 ns si miss rate m(L2) = 1%
t(L3) = 10 ns si miss rate m(L3) = 0.2%

verificăm în RAM: 50 ns

verificăm în L1:
1 ns + (0.1 x 50 ns) = 6 ns

verificăm în L2:
1 ns + (0.1 x (5 ns + (0.01 x 50 ns))) = 1.55 ns

verificăm în L3:
1 ns + (0.1 x (5 ns + (0.01 x (10 ns + (0.002 x 50 ns))))) = 1.5101 ns