ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Introduction to Parallel Computing

Parallel Programming in Scala

Aleksandar Prokopec

What is Parallel Computing?

Parallel computing is a type of computation in which many calculations are performed at the same time.

What is Parallel Computing?

Parallel computing is a type of computation in which many calculations are performed at the same time.

Basic principle: computation can be divided into smaller subproblems, each of which can be solved simultaneously.

What is Parallel Computing?

Parallel computing is a type of computation in which many calculations are performed at the same time.

Basic principle: computation can be divided into smaller subproblems, each of which can be solved simultaneously.

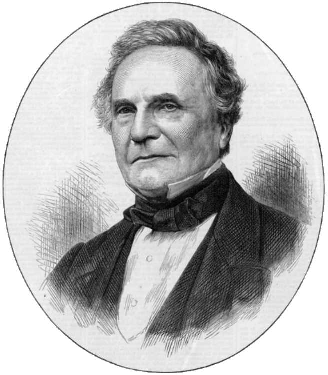
Assumption: we have parallel hardware at our disposal, which is capable of executing these computations in parallel.

History

Parallel computing was present since the early days of computing.

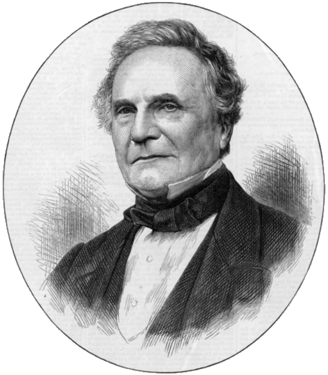
History

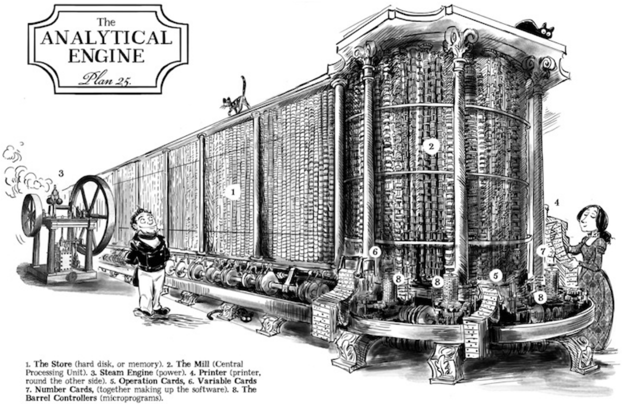
Parallel computing was present since the early days of computing.



History

Parallel computing was present since the early days of computing.





1. **The Store** (hard disk, or memory). 2. **The Mill** (Central Processing Unit). 3. **Steam Engine** (power). 4. **Printer** (printer, round the other side). 5. **Operation Cards,** 6. **Variable Cards** 7. **Number Cards,** (together making up the software). 8. **The Barrel Controllers** (microprograms).

History

Parallel computing was present since the early days of computing.

History

Parallel computing was present since the early days of computing.

In the 20th century, researchers from IBM built some of the first commercial parallel computers.

History

Parallel computing was present since the early days of computing.

In the 20th century, researchers from IBM built some of the first commercial parallel computers.

> *For over a decade prophets have voiced the contention that the organization of a single computer has reached its limits and that truly significant advances can be made only by interconnection of a multiplicity of computers.*
> *Gene Amdahl, 1967.*

History

Parallel computing was present since the early days of computing.

In the 20th century, researchers from IBM built some of the first commercial parallel computers.

> *For over a decade prophets have voiced the contention that the organization of a single computer has reached its limits and that truly significant advances can be made only by interconnection of a multiplicity of computers.*
> *Gene Amdahl, 1967.*

At the time, parallel computing was confined to niche communities and used in high performance computing.

Recent History

At the beginning of the 21st century processor frequency scaling hit the *power wall*.

Recent History

At the beginning of the 21st century processor frequency scaling hit the *power wall*.

Processor vendors decided to provide multiple CPU cores on the same processor chip, each capable of executing separate instruction streams.

Recent History

At the beginning of the 21st century processor frequency scaling hit the *power wall*.

Processor vendors decided to provide multiple CPU cores on the same processor chip, each capable of executing separate instruction streams.

Common theme: parallel computing provides computational power when sequential computing cannot do so.

Why Parallel Computing?

Parallel programming is much harder than sequential programming.

Why Parallel Computing?

Parallel programming is much harder than sequential programming.

- Separating sequential computations into parallel subcomputations can be challenging, or even impossible.

Why Parallel Computing?

Parallel programming is much harder than sequential programming.

- Separating sequential computations into parallel subcomputations can be challenging, or even impossible.
- Ensuring program correctness is more difficult, due to new types of errors.

Why Parallel Computing?

Parallel programming is much harder than sequential programming.

- Separating sequential computations into parallel subcomputations can be challenging, or even impossible.
- Ensuring program correctness is more difficult, due to new types of errors.

Speedup is the only reason why we bother paying for this complexity.

Parallel Programming vs. Concurrent Programming

Parallelism and concurrency are closely related concepts.

Parallel Programming vs. Concurrent Programming

Parallelism and concurrency are closely related concepts.

Parallel program – uses parallel hardware to execute computation more quickly. Efficiency is its main concern.

Parallel Programming vs. Concurrent Programming

Parallelism and concurrency are closely related concepts.

Parallel program – uses parallel hardware to execute computation more quickly. Efficiency is its main concern.

Concurrent program – *may* or *may not* execute multiple executions at the same time. Improves modularity, responsiveness or maintainability.

Parallelism Granularity

Parallelism manifests itself at different granularity levels.

Parallelism Granularity

Parallelism manifests itself at different granularity levels.

- bit-level parallelism – processing multiple bits of data in parallel

Parallelism Granularity

Parallelism manifests itself at different granularity levels.

- bit-level parallelism – processing multiple bits of data in parallel
- instruction-level parallelism – executing different instructions from the same instruction stream in parallel

Parallelism Granularity

Parallelism manifests itself at different granularity levels.

- bit-level parallelism – processing multiple bits of data in parallel
- instruction-level parallelism – executing different instructions from the same instruction stream in parallel
- task-level parallelism – executing separate instruction streams in parallel

Parallelism Granularity

Parallelism manifests itself at different granularity levels.

- bit-level parallelism – processing multiple bits of data in parallel
- instruction-level parallelism – executing different instructions from the same instruction stream in parallel
- task-level parallelism – executing separate instruction streams in parallel

In this course, we focus on task-level parallelism.

Classes of Parallel Computers

Many different forms of parallel hardware.

Classes of Parallel Computers

Many different forms of parallel hardware.

- multi-core processors

Classes of Parallel Computers

Many different forms of parallel hardware.

- multi-core processors
- symmetric multiprocessors

Classes of Parallel Computers

Many different forms of parallel hardware.

- multi-core processors
- symmetric multiprocessors
- general purpose graphics processing unit

Classes of Parallel Computers

Many different forms of parallel hardware.

- multi-core processors
- symmetric multiprocessors
- general purpose graphics processing unit
- field-programmable gate arrays

Classes of Parallel Computers

Many different forms of parallel hardware.

- multi-core processors
- symmetric multiprocessors
- general purpose graphics processing unit
- field-programmable gate arrays
- computer clusters

Classes of Parallel Computers

Many different forms of parallel hardware.

- multi-core processors
- symmetric multiprocessors
- general purpose graphics processing unit
- field-programmable gate arrays
- computer clusters

Our focus will be programming for multi-cores and SMPs.

Summary

Course structure:

- week 1 – basics of parallel computing and parallel program analysis
- week 2 – task-parallelism, basic parallel algorithms
- week 3 – data-parallelism, Scala parallel collections
- week 4 – data structures for parallel computing