Advanced Topics

Chapter 17

Parallelism

Topics

- Introduction
- Characterizations of Parallelism
- Microscopic Vs Macroscopic
- Symmetric Vs. Asymmetric
- Fine Grain Vs Coarse-grain
- Explicit Vs. Implicit
- Parallel Architectures
- SISD
- SIMD

Topics

- MIMD
- Symmetric Multiprocessors
- Communication, Coordination, Contention
- Performance of Multiprocessors
- Consequence for Programmers
- Locks and Mutual Exclusion
- Programming Explicit and Implicit Parallel Computers
- Redundant Parallel Architectures
- Distributed and Cluster Computers

Introduction

- Previous parts covered
  - processors
  - memory
  - I/O
- Now
  - advanced topics
  - use of parallelism to increase speed
  - parallel hardware, taxonomy, limitations etc.
- Parallel and pipelined architectures
  - two fundamental techniques or basis of increasing speed

Characterizations of Parallelism

- Parallel and non-parallel are the extremes
- The grey region in-between describes amount and type of parallelism
  - microscopic vs macroscopic
  - symmetric vs. asymmetric
  - fine grain vs coarse-grain
  - explicit vs. implicit

Microscopic Vs Macroscopic

- Microscopic
  - parallelism in a computer remains hidden inside subcomponents
  - parallel hardware within specific component
- Macroscopic
  - parallelism is a basic premise around which the system is designed

Examples

- Microscopic
  - ALU, registers, physical memory, parallel bus architecture.
- Macroscopic
  - use parallelism across multiple large-scale components of a computer systems, multiple identical processors, multiple dissimilar processors

Symmetric Vs. Asymmetric

- Symmetric
  - replication of identical elements
  - e.g. dual processors
- Asymmetric
  - multiple elements that function at the same time but differ from each other
  - PC with graphics coprocessor and math coprocessor

Fine Grain Vs Coarse-grain

- Fine Grain
  - parallelism at the level of individual instructions or individual data elements
  - e.g. graphics processor that uses 16 hardware units to update 16 bytes of an image at the same time
- Coarse-grain
  - parallelism on the level of programs or large blocks of data
  - e.g. dual processor, using a processor each to print and compose an email simultaneously

Explicit Vs. Implicit

- Implicit
  - hardware handles parallelism automatically without requiring a programmer to initiate or control parallel execution
- Explicit
  - programmer must control each parallel unit

Parallel Architectures

- Parallel architectures: parallelism of the central features around which the entire system is designed
- Flynn's classification

Name	Meaning
SISD	**Single Instruction Single Data stream**
SIMD	**Single Instruction Multiple Data streams**
MIMD	**Multiple Instructions Multiple Data streams**

Terminology used to characterize computers according to the amount and type of parallelism. Note, hybrids that span multiple types also exist.

SISD

- Single Instruction Single Data stream
- Also called sequential or uniprocessor architecture
- Follows Von Neumann’s architecture
- Does not support macroscopic parallelism, can use parallelism internally

SIMD

- Single Instruction streams Multiple Data streams
- Each instruction specifies a single operation applied to many data items simultaneously
- Vector, array processors

```
for i from 1 to N {
    V[i] ← V[i] × Q;
}
```

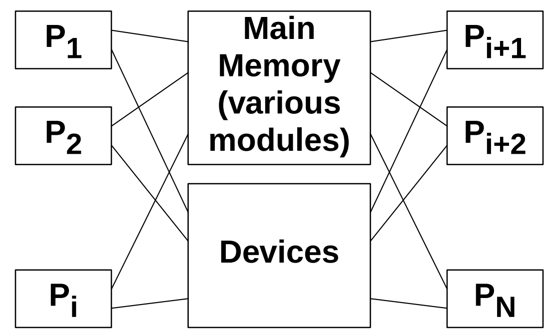
Algorithm for vector normalization in a sequential computer.

- Same in a SIMD computer V <- V X Q
- Graphic processors

MIMD

- Multiple Instruction streams Multiple Data streams
- Each processor performs independent computations at the same time
- Example of MIMD
  - symmetric multiprocessors (SMP)
  - asymmetric multiprocessors (AMP)
  - math graphics coprocessors
  - I/O processors
  - CDC peripheral processors

Symmetric Multiprocessors



Symmetric multiprocessor with N identical processors. Each processor has access to memory I/O devices.

Performance of Multiprocessors

- Speedup = execution time on a single processor/execution time required on a multiprocessor

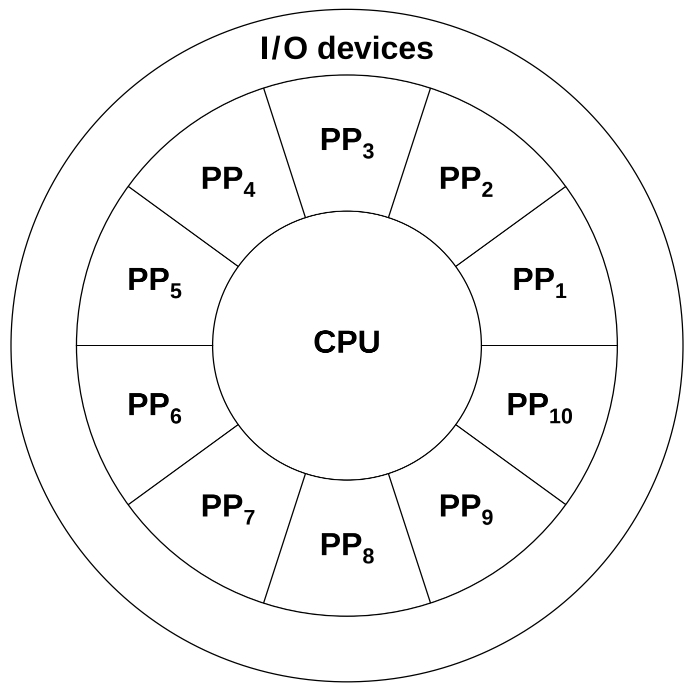


Illustration of the ideal and typical performance of a multiprocessor as the number of processors is increased. Values on the y-axis list the relative speedup compared to a single processor.

Performance of Multiprocessors

- Reasons why multiprocessor architectures have not fulfilled the promise of scalable, high performance computing.
  - OS bottlenecks
  - memory contention
  - I/O bottlenecks

  When used for general-purpose computing, a multiprocessor does not perform well. In some cases, added overhead means performance decreases as more processors are added.

Consequence for Programmers

- Locks and mutual exclusion
- Programming explicit and implicit parallel computers
- Programming symmetric and asymmetric multiprocessors

Locks and Mutual Exclusion

- Shared variables in multiprocessors face the following problem.
  - What happens if two processors attempt to increment at the same time?
  - Variable is incremented only once instead of the correct two times.
- One solution: locks

Locks and Mutual Exclusion

```
loadx, R5
incrR5
storeR5, x
```

An example sequence of machine instructions used to increment a variable in memory. In most architectures, increment entails a load and store.

Locks

```
lock17
loadx, R5
incrR5
storeR5, x
release 17
```

Illustration of the instructions used to guarantee exclusive access to a variable. A separate lock is assigned to each shared item.

- Only one copy of lock exists, only one processor can gain access to lock at a time.

Locks

- Problems
  - programmers forget to apply locks to shared variables
  - errors due to simultaneous access depend on timing and are rare to detect
  - locking can reduce performance
  - locking adds overhead

Programming Explicit and Implicit Parallel Computers

From a programmer's point of view, a system that uses explicit parallelism is significantly more complex to program that a system that uses implicit parallelism.

Redundant Parallel Architectures

- Use of parallel hardware to improve reliability and prevent failure
- Multiple copies of a hardware unit that operate in parallel to perform an operation (same data and same operation on all units)
- What happens if redundant copies of hardware disagree?
  - votes?
  - error message.

Distributed and Cluster Computers

- Tightly coupled
  - parallel hardware units are located inside the same computer system
- Loosely coupled
  - multiple computer systems that are interconnected by a communication mechanism
  - e.g. distributed architectures, cluster computer

Distributed and Cluster Computers

- Distributed architecture
  - set of computers connected by a computer network or Internet
- Cluster computer
  - set of independent computers connected by a high-speed computer network that are all dedicated to solving one problem at a time
  - also called network cluster

Summary

- Parallelism is one of the fundamental optimization techniques used to increase hardware performance
- Explicit parallelism gives a programmer control over the use of parallel facilities; implicit parallelism handles parallelism automatically
- SIMD architecture allows an instruction to operate on an array of values
  - vector processors
  - graphic processors

Summary

- MIMD employs multiple, independent processors that operate simultaneously and can each execute a separate program.
  - symmetric multiprocessors
  - asymmetric multiprocessors
- Speedup
  - theoretically machine with N processors should perform N times as fast as a single processor
  - in practice, speedup does not increase linearly with number of processors due to memory contention and communication overhead.
- Alternatives to SIMD and MIMD are redundant, distributed, and cluster architectures