

Embedded Parallel Computing

Lecture 1 - Introduction

Tomas Nordström

Course webpage: <<http://www.hh.se/DO8003>>

Course responsible and examiner: Tomas Nordström,
Tomas.Nordstrom@hh.se; Room E313; Tel: +46 35 16 7334

Other teachers: Zain-ul-Abdin <Zain-UI-Abdin@hh.se>

Welcome to the world of ***embedded parallel*** computing!

Why Parallel?

- Keep Moores “law” running but..
- Power and heat limit clock speed
- Chip size limits synchronized signaling distance
- Software and tools limits “free” parallelism
- ...

Near Future

- Explicit parallelism -- dual/quad/eight core is here
- Sixteen-cores coming soon
- Many-core is appearing
- Multithreading
- Virtualization
- Streaming and Vector-operations -- CBE/GPGPU

What is Special About Embedded?

The typical characteristics of an embedded system include:

- ▶ Timing guaranties and real-time operation
- ▶ Tough reliability and security requirements: working in safety-critical and/or mission-critical situations
- ▶ Tight integration of software and hardware component often developed together
- ▶ Often single chip solutions (still with many cores) leading to constraints for memory, I/O bottlenecks and on chip communication solutions like network-on-chip.
- ▶ Energy efficiency and battery life-time issues are paramount for operation of the system
- ▶ Often specific hardware augmentation is added and heterogeneous cores are used

Course Objectives

Upon completion of the course, the student shall be able to:

Knowledge and understanding

- ▶ describe and explain the most important parallel architecture models, as well as parallel programming models, and discuss their respective pros, cons, and application opportunities

Course Objectives

Skills and abilities

Upon completion of the course, the student shall be able to:

- ▶ program parallel computer systems intended for embedded applications
- ▶ describe, evaluate, and discuss how the choice of programming model and method influences, e.g., execution time and required resources
- ▶ read and understand scientific articles in the area, to review and discuss them and to make summaries and presentations; as well as identify relevant research publications and research groups in the area

Course Objectives

Upon completion of the course, the student shall be

Judgement and approach

able to: ▶ **relate the state of the art in the area to the current research and development, in particular such research and development that is important for the design of embedded systems**

Teaching Formats

- **Lecture** part (L1, L2, etc. in the course planning).
 - ▶ Overview presentation will be given as a preparation for the students' own reading of the material and problem solving;
 - ▶ Discussion sessions will be provided in the following lecture after the students' own reading and problem solving.
- **Seminar** part (S1, S2, etc. in the course planning).
 - ▶ Seminars prepared by the students: Presentations of current "hot topics" followed by discussions.
- **Project** part (P1, P2, etc. in the course planning).
 - ▶ Hands-on parallel programming of multiprocessor on a chip, in laboratory sessions followed by a small project.

Lectures

- Lecture 1: Introduction to Advanced Computer Architecture and Parallel Processing
- Lecture 2: Parallelism in microprocessors
- Lecture 3: Multiprocessors Interconnection Networks
- Lecture 4: Multiple-instruction multiple-data streams (MIMD) parallel architectures
- Lecture 5: The anatomy of a modern multiprocessor, the multi-core processors
- Lecture 6: Fundamentals of embedded many-core architectures
- Lecture 7: Programming models and methodologies for parallel embedded processors
- Lecture 8: Energy efficiency in modern embedded parallel processors

Laboratory and Project

- Two lab sessions are planned in the course. Each session consist of an introductory lecture followed by a laboratory exercise a few days later. The students are supposed to work in groups of two.
 - ▶ In the first sessions, students will use MPI library to program some simple tasks and calculate speedup by executing their application on a parallel machine.
 - ▶ In the second session, students get the opportunity to implement some basic signal processing functions in Ambric structured programming model and simulate their results in the eclipse integrated development environment provided by Ambric.
- In the project part, students are expected to program a specified task in either MPI or Ambric programming model according to their choice.

Seminars

- You will be provided a list of different topic studied during the course, along with an article associated with the topic, and you choose one of these topics to present.
- Each student is given a quiz based on the article. Presentations are performed in groups of two or three student. *All the students are expected to have read the article and prepare questions or points to be raised for discussion after the presentation.*

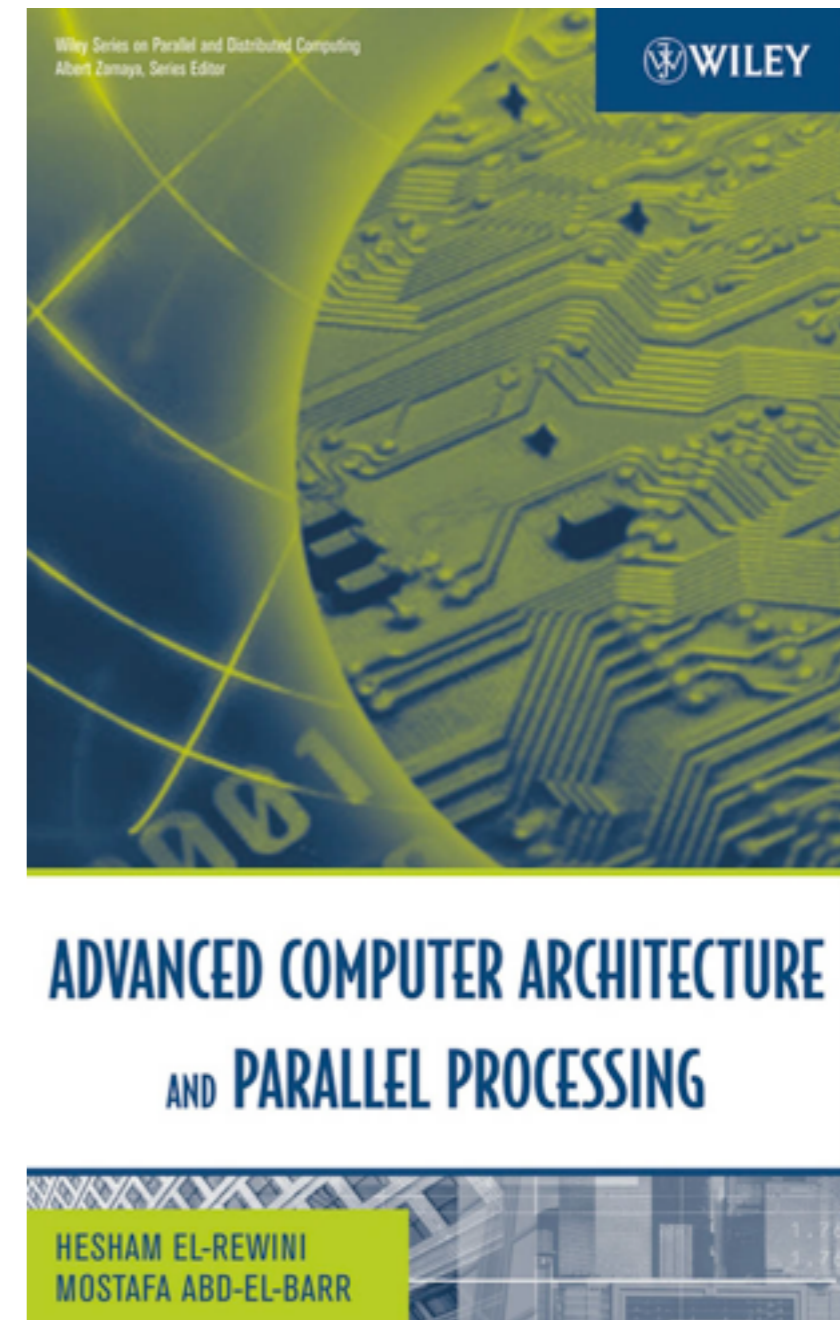
Examination

- The laboratory exercises and programming project are obligatory parts of the course. The programming project is examined based on the reports submitted by the students.
- The student needs to make at least one seminar presentation to pass the course. *Bonus points for the written exam may be earned through participation in the seminars and providing correct answers on the written quizzes.*
- For the lectures part, the students will be assessed in a written examination at the end of the course.

The overall grade given is Fail, 3, 4, or 5 based on the performance of the students in course examination, seminar quizzes, and their performance in presenting their own seminar

Course Literature

- [ACAPP] H. El-Rewini & M. Abd-el-Barr, "Advanced Computer Architecture and Parallel Processing", John Wiley & Sons, 2005, ISBN 0-471-46740-5.
- The book is electronically available in full-text form through the database ebrary, available through the University's library.



Course Literature

- **Additional Books, recommended reading**
 - ▶ Kornaros, G., Multi-core Embedded Systems (Embedded Multi-core Systems), 2010.
 - ▶ Wolf, W., High-performance Embedded Computing: Architectures, Applications, and Methodologies, Morgan Kaufmann, 2007.
 - ▶ Hennessy, J.L. and D.A. Patterson, Computer Architecture: A Quantitative Approach, Fourth Edition, Morgan Kaufmann, 2006.
 - ▶ Almasi, G.S. and A. Gottlieb, Highly Parallel Computing, 2nd Edition, Benjamin/Cummings, 1994.
 - ▶ Zomaya, A.Y.H. (ed), Parallel and Distributed Computing Handbook, McGraw-Hill, 1996.
- **Wikipedia**
- **Scientific Publications in Journals and Conferences**
 - ▶ Find them via google scholar, university library, ieeexplore, acm, ...

Introduction to Advanced Computer Architecture and Parallel Processing

- [ACAPP] Chapter One

1. Introduction to Advanced Computer Architecture and Parallel Processing	1
1.1 Four Decades of Computing	2
1.2 Flynn's Taxonomy of Computer Architecture	4
1.3 SIMD Architecture	5
1.4 MIMD Architecture	6
1.5 Interconnection Networks	11
1.6 Chapter Summary	15
Problems	16
References	17

- [ACAPP] Chapter Three

3. Performance Analysis of Multiprocessor Architecture	51
3.1 Computational Models	51
3.2 An Argument for Parallel Architectures	55
3.3 Interconnection Networks Performance Issues	58
3.4 Scalability of Parallel Architectures	63
3.5 Benchmark Performance	67
3.6 Chapter Summary	72
Problems	73
References	74

Five Decades of Computing

- "Most computer scientists agree that there have been four distinct paradigms or eras of computing"

TABLE 1.1 Four Decades of Computing

Feature	Batch	Time-Sharing	Desktop	Network
Decade	1960s	1970s	1980s	1990s
Location	Computer room	Terminal room	Desktop	Mobile
Users	Experts	Specialists	Individuals	Groups
Data	Alphanumeric	Text, numbers	Fonts, graphs	Multimedia
Objective	Calculate	Access	Present	Communicate
Interface	Punched card	Keyboard and CRT	See and point	Ask and tell
Operation	Process	Edit	Layout	Orchestrate
Connectivity	None	Peripheral cable	LAN	Internet
Owners	Corporate computer centers	Divisional IS shops	Departmental end-users	Everyone

Q: What is the characteristics of the fifth decade?

Flynn's Taxonomy of Computer Architectures

- According to Flynn's classification (1966), either of the instruction or data streams can be single or multiple. Computer architecture can be classified into the following four distinct categories
 - ▶ single-instruction single-data streams (SISD);
 - ▶ single-instruction multiple-data streams (SIMD);
 - ▶ multiple-instruction single-data streams (MISD); and
 - ▶ multiple-instruction multiple-data streams (MIMD).

Single-instruction multiple-data streams (SIMD)

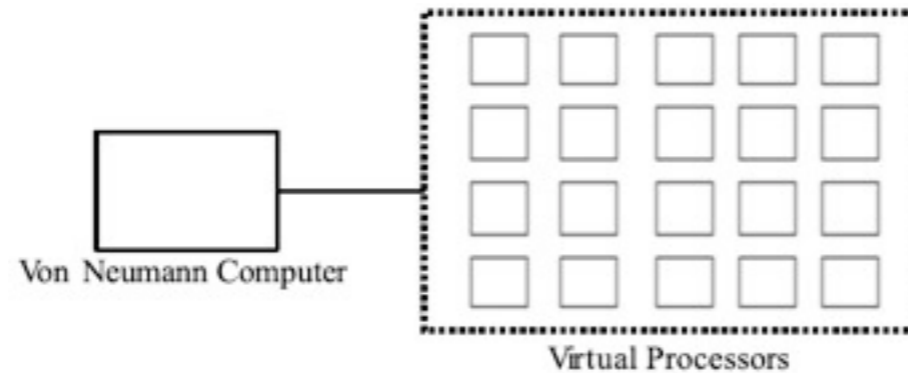
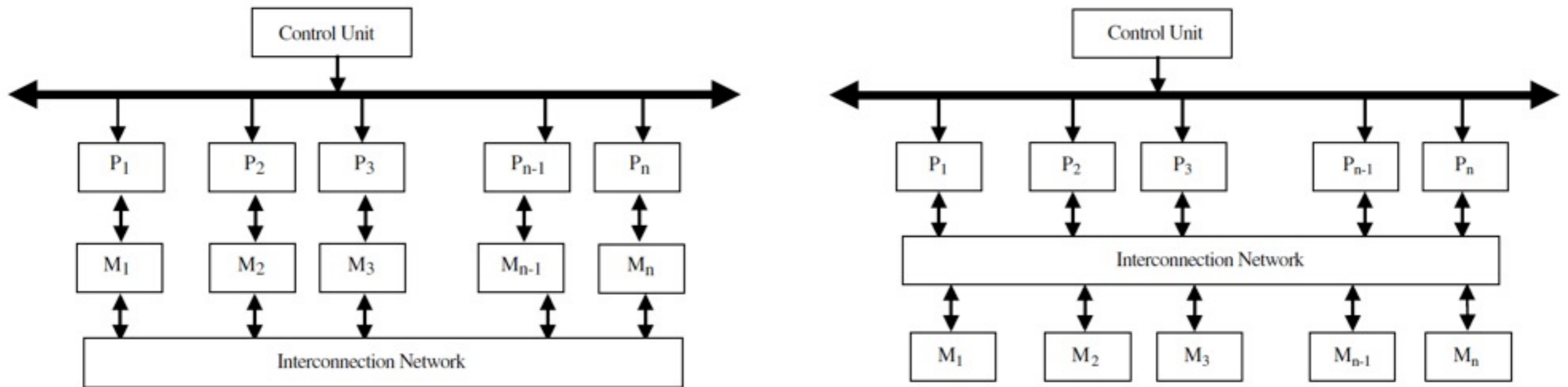
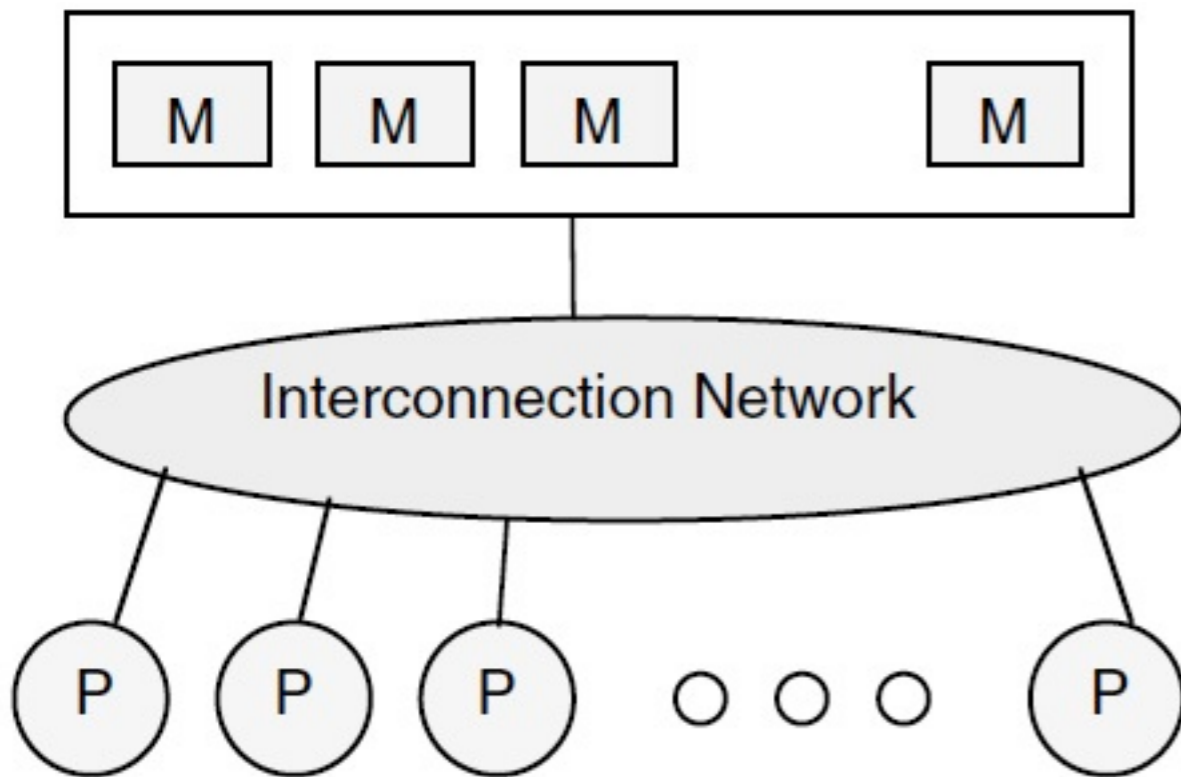


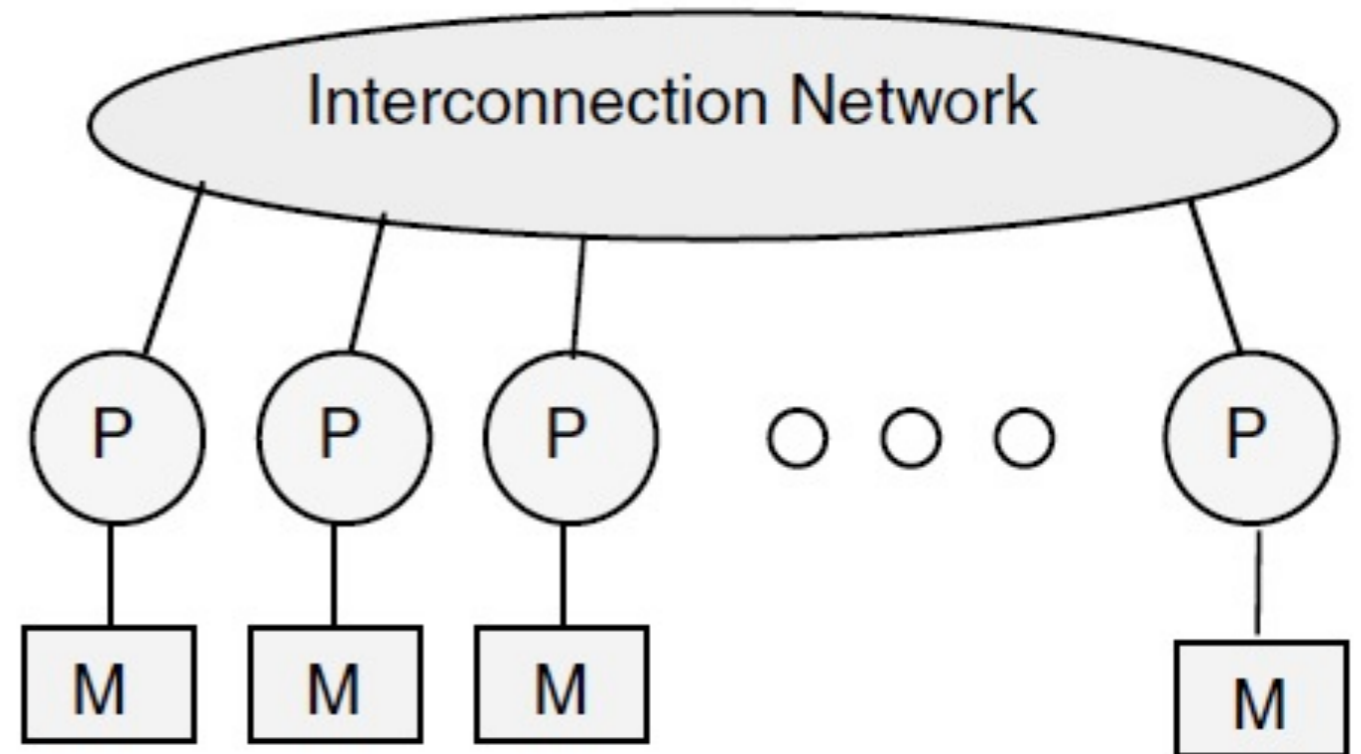
Figure 1.4 SIMD architecture model.



Multiple-instruction multiple-data streams (MIMD)



Shared Memory MIMD Architecture



Message Passing MIMD Architecture

Interconnection Networks

- Mode of operation: synchronous vs. asynchronous
- Control Strategy: centralized vs decentralized
- Switching Techniques: circuit vs packet switching
- Topology

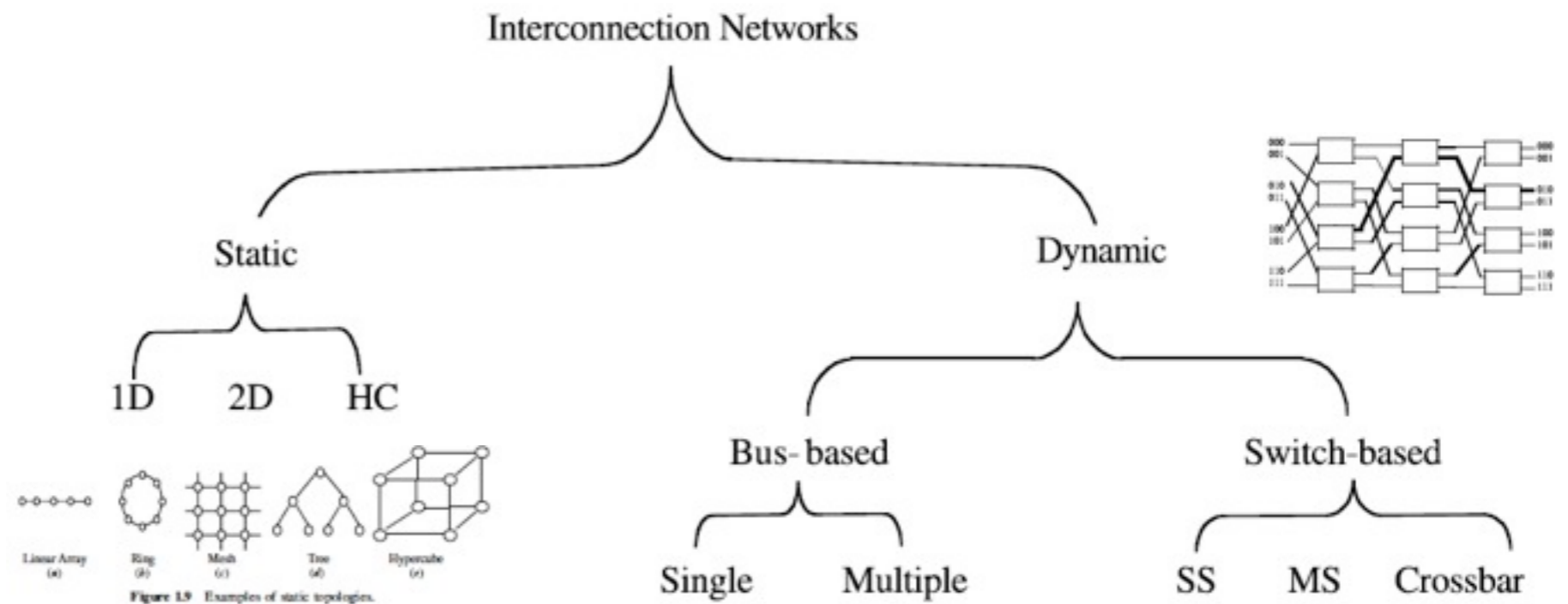


Figure 2.1 A topology-based taxonomy for interconnection networks.

Q: what performance characteristics can be used?

Performance Measures of Parallel Architectures

- [ACAPP] Chapter Three

3. Performance Analysis of Multiprocessor Architecture	51
3.1 Computational Models	51
3.2 An Argument for Parallel Architectures	55
3.3 Interconnection Networks Performance Issues	58
3.4 Scalability of Parallel Architectures	63
3.5 Benchmark Performance	67
3.6 Chapter Summary	72
Problems	73
References	74

Computational Model

- Two computational models:
 - ▶ equal duration and
 - ▶ parallel computations with serial sections

(the speedup and efficiency is computed with and without the effect of the communication overhead)

Argumenting for Parallel Architectures

- Ahdahls' Law
- Gustafson-Barsis's Law
- Karp–Flatt metric [Wikipedia]
- Performance per watt [Wikipedia]

Not everything can be done in parallel!

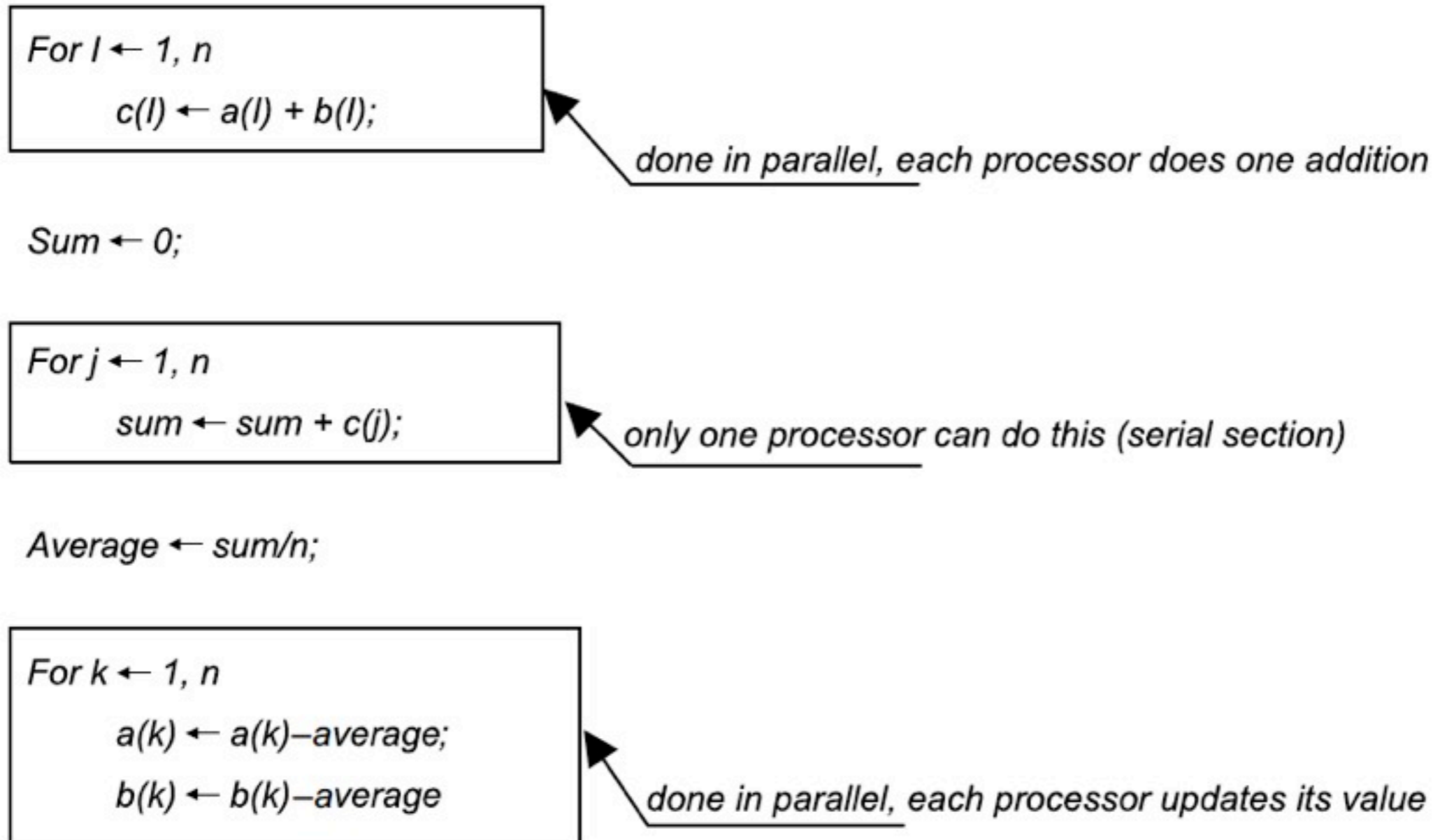
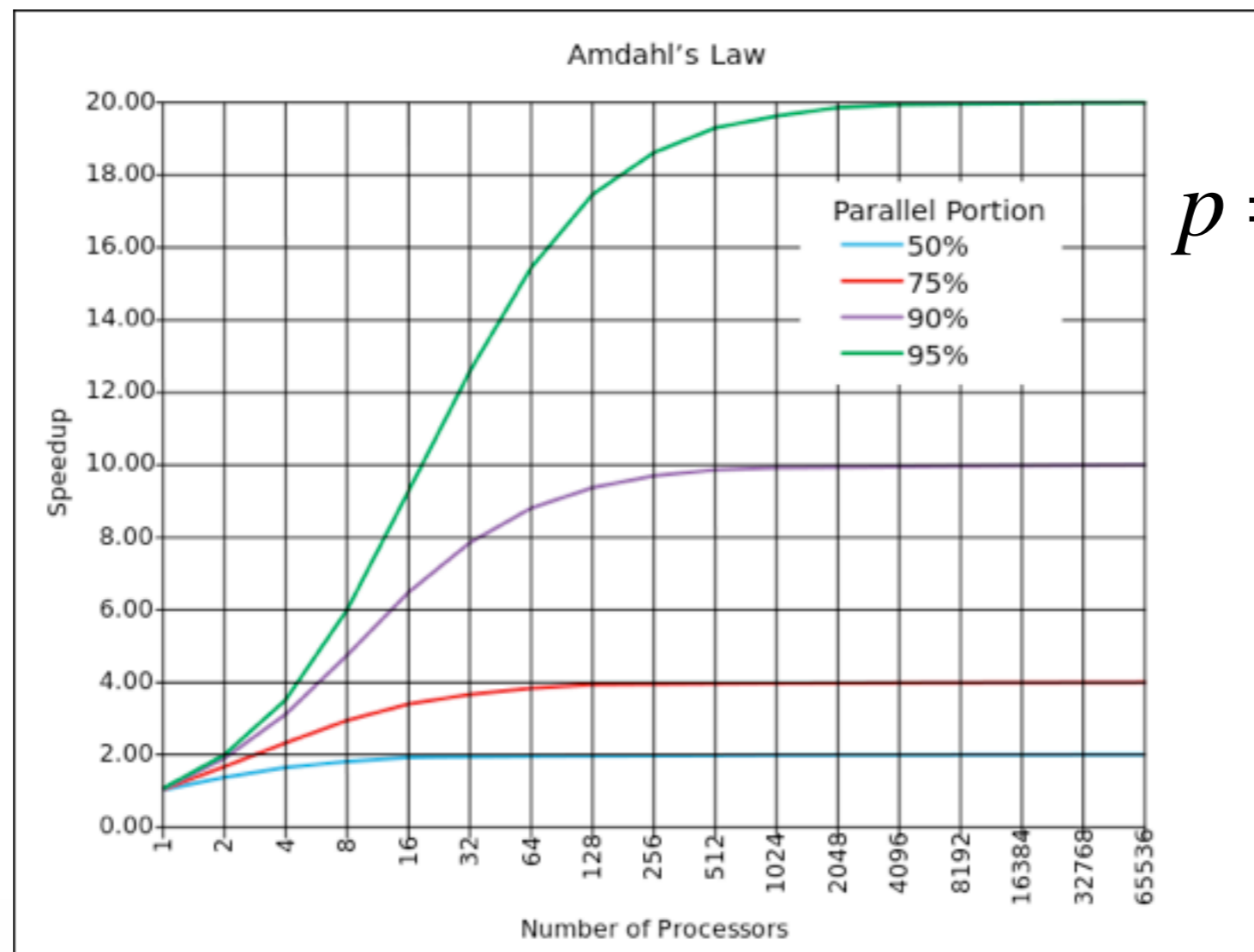


Figure 3.1 Example program segments.

Amdahl's Law

$$S(n) = \frac{t_s}{t_m} = \frac{t_s}{ft_s + (1-f)\frac{t_s}{n}} = \frac{n}{1 + (n-1)f}$$



$$p = (1 - f)$$

Amdahl's Law

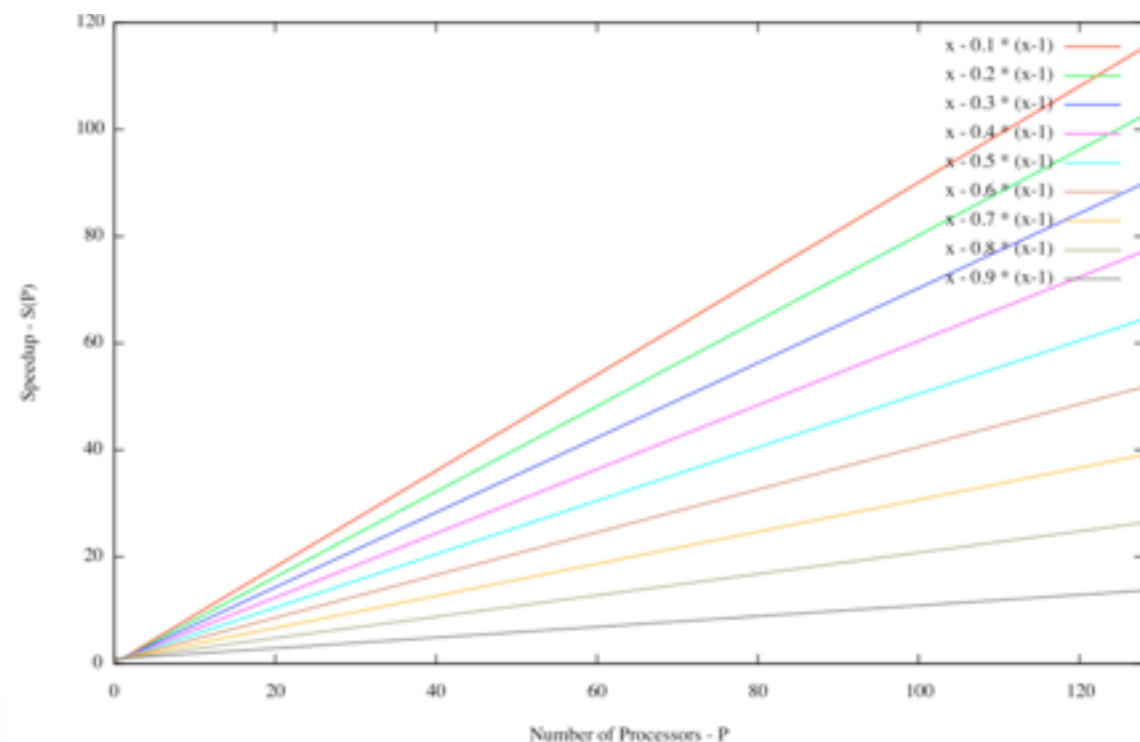
- Observe that f is considered fixed
- This is however not always the case in many real-world examples
- Instead if f goes to zero as we scale up the problem with the number of processors we can get a linear speedup!

Gustafson-Barsis's Law

- They postulated that if s and p represent respectively the serial and the parallel time spent on a parallel system, then $s + p \times n$ represents the time needed by a serial processor to perform the computation.
- They therefore, introduced a new factor, called the scaled speedup factor,

$$SS(n) = \frac{s + p \times n}{s + p} = s + p \times n = s + (1 - s) \times n = n + (1 - n) \times s$$

- This equation shows that the resulting function is a straight line with a slope = $(1 - n)$.



Gustafson: Speedup should be measured by scaling the problem to the number of processors, not by fixing the problem size

Performance per Watt

- In computing, **performance per watt** is a *measure of the energy efficiency* of a particular computer architecture or computer hardware. Literally, it measures the rate of computation that can be delivered by a computer for every watt of power consumed.
 - ▶ FLOPS (Floating Point Operations Per Second) per watt
 - ▶ 3DMark2006 score per watt
- While performance per watt is useful, absolute power requirements are also important. Claims of improved performance per watt may be used to mask increasing power demands.

Scalability of Parallel Architectures

- A parallel architecture is said to be **scalable** if it can be expanded (reduced) to a larger(smaller) system with a linear increase (decrease) in its performance (cost)
- In terms of speed, a scalable system is capable of increasing its speed in proportion to the increase in the number of processors.

Summary

- [ACAPP]
- Chapter 1: We have gone over a number of concepts and system configurations related to obtaining high-performance computing via **parallelism**. In particular, we have provided the general concepts and terminology used in the context of multiprocessors. The popular **Flynn's taxonomy** of computer systems has been provided. An introduction to **SIMD** and **MIMD** systems was given. Both **shared-memory** and the **message passing systems** and their **interconnection networks** were introduced.
- Chapter 3: We have covered a number of important issues related to the performance of multiprocessor systems. Two computational models: equal duration and **parallel computations with serial sections** have been first introduced. In each case the speedup and efficiency have been computed with and without the effect of the communication overhead. A rebuttal to a number of critical views about the effectiveness of parallel architectures has been made. This includes Grosch's and **Amdahl's laws**. In addition, the **Gustafson–Barsis law**, which supports the use of multiprocessor architecture, has been introduced. The **scalability** of parallel architectures in terms of **speed and efficiency** has been discussed.

Links

Parallel Computing <http://en.wikipedia.org/wiki/Parallel_computing>

Embedded systems <http://en.wikipedia.org/wiki/Embedded_system>

Flynn's taxonomy <http://en.wikipedia.org/wiki/Flynn%27s_taxonomy>;

SIMD <<http://en.wikipedia.org/wiki/SIMD>>;

MIMD <<http://en.wikipedia.org/wiki/MIMD>>;

Shared-memory systems <http://en.wikipedia.org/wiki/Shared_memory>;

Message passing systems <http://en.wikipedia.org/wiki/Message_passing>;

Multi-core processor <http://en.wikipedia.org/wiki/Multi-core_%28computing%29>;

Interconnection networks <http://en.wikipedia.org/wiki/Network_topology>

Amdahl's laws <http://en.wikipedia.org/wiki/Amdahl%27s_law>;

Gustafson–Barsis law <http://en.wikipedia.org/wiki/Gustafson%27s_law>;

<<http://www.cis.temple.edu/~shi/docs/amdahl/amdahl.html>>;

Karp–Flatt metric <http://en.wikipedia.org/wiki/Karp%E2%80%93Flatt_metric>;

Performance per watt <http://en.wikipedia.org/wiki/Performance_per_watt>;

Scalable parallelism <http://en.wikipedia.org/wiki/Scalable_parallelism>;

Benchmarking: SPEC <<http://www.spec.org/>> <http://en.wikipedia.org/wiki/Standard_Performance_Evaluation_Corporation>