Modeller for parallelle beregninger/maskiner --- BSP og PRAM ---

- **Pensum**
  - Models of Parallel Computation, Notat, Lasse Natvig
  - High-level Simulation of BSP-architectures, notat, Lasse Natvig
- **Innhold**
  - Models
  - Motivation for using models
  - The P-RAM model
  - Properties of good models
  - Valiant´s bridging BSP model
    - as a bridge between SW and HW
    - central concepts
    - architecture parameters
    - BSP programming
    - portability through parametrization

Motivation

- **Increasingly complex systems**
  - cheap HW-components and modules
  - better tools
  - free software
- **=> ABSTRACTION**
- **Models**
  - of various kinds
    - programming
    - description, understanding
    - cost models
    - performance models
  - at different levels of detail
- **Teoretisk databehandling (Eng. Theoretical Computer Science (TCS))**
  - introdusere
  - "gi pekere"

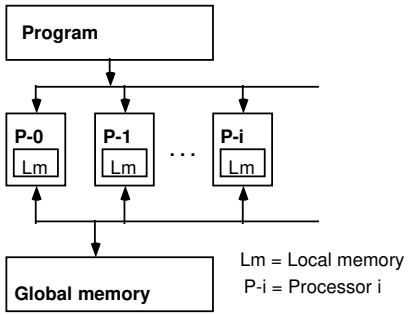
Models

- **Sekvensielle modeller**
  - (RAM)
- **Parallelle modeller**
  - a large number, and
    - different names at the same
    - different interpretations of the same
  - our main tool for abstraction
  - may provide a bridge to TCS
  - model / standard
  - remember the essentials, forget
    - most details
    - *i.e.* selective forgetting

Theoretical models

- **a large number of models within the TCS community**
  - to a large extent "unknown" by the practitioner and the engineer
  - see the references, and Handbook of TCS
- **the most popular model of parallel computation is the P-RAM model!**

The PRAM Model



- **Fortune and Wyllie 1978**
- **Synchronous processors**
- **Shared memory with ?R?W-access**
- **one program, BUT MIMD**

Motivation for using models

- **Abstractions simplify**
  - programming model
  - when building hardware
  - when integrating systems
- **Common platform**
  - project work
  - portability of results
    - HW
    - SW
    - research results
    - teaching
- **Performance**
  - analysis
  - prediction
  - f.eks. detaljerte modeller som gir ytelsen til et system som funksjon av sentrale arkitektur-parametre
    - ekstrapolering forutsetter skalerbarhet

Properties of good models

1) **easy to understand**
2) **easy to use**
3) **well defined**
4) **general**
5) **performance representative**
6) **realistic**
7) **durable**
8) **mathematically convenient**

Model tradeoffs ?

- **These are conflicting requirements:**
  - Easy programming vs. efficient execution
  - General purpose model vs. special purpose model
  - Easy to use vs. mathematical convenience
  - Easy to analyse vs. easy to build

Bridging models in CS

- **Bridging models in general**
  - simple and general
    - ➪ common understanding
  - sufficiently "advanced"
    - ➪ challenging for a long period
  - technology-independent
    - technology becomes old-fashioned very fast
  - sufficiently undetailed
    - room for architectural and technological improvements
- **Examples:**
  - von Neumann model (RAM)
    - *A bridge between HW and SW*
  - relational model for databases
    - *A bridge between users and* DBMS-developers
  - **Bulk Synchronous Parallel (BSP)** model for parallel computers
    - *A bridge between parallel-HW* and (parallel)-SW

Bakgrunn: Marked for parallel programmering?

- Å utvikle programmer er vanskelig (flaskehals, "software crisis")
- Å utvikle parallelle programmer er **enda vanskeligere**
- Å utvikle effektive parallelle programmer er **enda vanskeligere**
- Å utvikle effektive og skalerbare parallelle programmer er **enda vanskeligere**
- Å utvikle effektive, skalerbare og portable parallelle programmer er **enda vanskeligere**

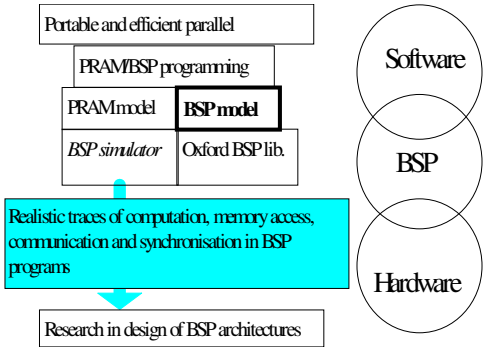
- **ergo;**
  - sekvensielle programmer bør antakelig brukes der det er mulig
  - tungregning vil kreve høytelses applikasjoner
    - » gir marked for parallel programmering
- **parallell-programmering bør gjøres lettest mulig**
- **parallell-programmering er ekstra utsatt for å bli "smittet" av ikke-flyttbar kode og triks**
- **skrikende behov for standardisering**
- **for umodent fagområde til å standardisere**

Valiant´s bridging BSP model overview

- **Provide convenient programming**
  - (to be accepted for a long time)
- **Challenging for the HW-developers for a long time**
- **Main parts of the BSP-model**
  - 1) Processing **components** with processing and/or memory functions
  - 2) **router**
  - 3) **synchronization facility**
    - may synchronise all or a subset of the components at regular intervals
    - periodicity-parameter l
- **Superstep**
  - set of computations and communication performed by one processor between two synchronisation points
    - » these are executed at full speed in parallell asynchronously
    - » = bulk of instructions

» "synchronous at bulk-level"

The Role of a BSP-simulator



McColl's superstep

- **William McColl [McC95] defines a superstep as**
  - "A sequence of steps, followed by a barrier synchronization at which point any memory accesses take effect. During a superstep, each processor has a set of programs or threads which it has to carry out, and it can do the following:
    - » perform a number of computation steps, from its set of threads, on values held locally at the start of the superstep
    - » send and receive a number of messages corresponding to non-local read and write requests"
- **One synchronization call ends a superstep**
- **Interference-free supersteps**

BSP-computer, 4 main parameters

	Name	Explanation
p	number of processors	
s	processor speed	number of local operations performed by a processor per second
l	synchronization periodicity	minimal number of time steps between successive synchronization operations
g	global computation/com-munication balance	(total number of local operations performed by all processors in one second) / (total number of words delivered by the communications network in one second)

Parallel slackness and communication slackness

- **Parallel slackness** ("overskuddsparallellitet")
  - "parallellism in program (#of virtual processors) compared to #of physical processors"
- **Communication slackness**
  - "overskudd av behov for lokale data framfor globale data
  - litt ulogisk navn;
    - bedre: (?) computation slackness, locality slackness

BSP world wide - pågående standardisering-

- **Oxford BSP library**
  - bsp_start, bsp_finish
  - bsp_fetch, bsp_store
  - bsp_sstep, bsp_sstep_end
- **Oxford BSP lib programming model**
  - Data parallel program structure
  - SPMD program text
  - Direct mode of global memory management
  - Remote assignment semantics for data access between processors
  - Static processor allocation
  - Single threaded process management
- **WWW**
  - http://www.scs.carleton.ca/~palepu/BSP.html

Portability through parametrizing

- **BSP algorithms typically have the four main BSP architectural parameters as input-parameters**
- **This makes it possible for the algorithm to choose strategies that depends on the overall architectural performance of the underlying BSP-machine used to run the algorithm**

Copyright © 1995 Lasse Natvig, IDT/NTH/UNIT

BSP-simulator; intended use

- **Exploration of the BSP design space**
- **Exploration of BSP variants and alternatives**
- **Research in BSP systems and hardware**
  - can a high-level model written in C++ use or interface to submodels written in Verilog or VHDL
- **Modelling of communication in message passing computers**
- **BSP software**

Copyright © 1995 Lasse Natvig, IDT/NTH/UNIT