Parallel Computing Survey

- Basic Definitions
- Illustrative Examples
- Computer models
- Speedup
- Engineering principles
- Multiple processors
- Parallel programming models
- Evolution of Performances

Parallel Computing

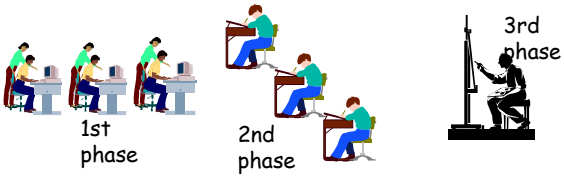
- Parallel computer + parallel program = parallel computing
- A parallel computer is a set of processors that are able to work cooperatively to solve a common computational problem.
  - parallel supercomputers with hundreds or thousands of processors,
  - networks of workstations,
  - multiple-processor workstations, and
  - embedded systems.
- A parallel program is a code being able to be executed by parallel computers.

Parallel Terminology

- There are many different expressions used in the literature for similar principles.
- Processor, Computer, Node, etc.
- Parallel execution, Concurrent execution, Distributed execution
- Multicomputer, Multiprocessor, Parallel processor, Distributed processor
- Communication channel, Link,
- We will use parallel computer when communication by messages is used and parallel processor if shared memory is used for communication.

Illustrative Examp.1-Writing Projects

- Writing projects (some communication in the first phase, no communication during work - all writers are independent, and no communication on grading).



- Finally, several projects have been done in the time needed to finish a single project plus some overhead because of initial personalised communication.

Illustrative Examp.2 - Rowing a Boat

- Synchronisation -> helmsman & oarsmen, Communication -> oarsman- oarsman, On-line corrections -> oarsmen and helmsman

- The boat is moving faster if the crew is larger but not proportionally and only for some extent. To much people could even sink the boat.

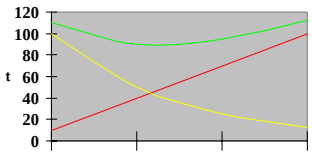
Fundamental Problems to be Solved

- Problem decomposition,
- Functional and data dependencies,
- Synchronisation, Communication,
- Partitioning,
- Visualisation of results,
- Performance measurements.

Parallel Applications

- Numerical simulations of complex systems such as weather, climate, mechanical devices, electronic circuits,
- Control of manufacturing processes, and chemical reactions,
- Commercial applications for large amounts of data,
- Video servers (real-time video).
- The need for faster computers is driven by the demands of both data-intensive applications in commerce and computation-intensive applications in science and engineering.

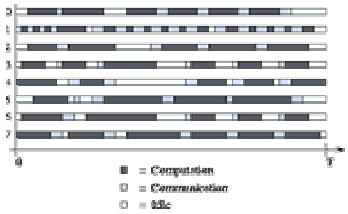
Computation + Communication



Total execution time =
Communication time
+ Idle time
+ Calculation time

- execution time =
  computation + communication + idle time
- some optimal number of processors p exist for a given problem size N,
- increase processor performances (technology),
- shorten communication and idle time (algorithm).

Computation + Communication (cont.)

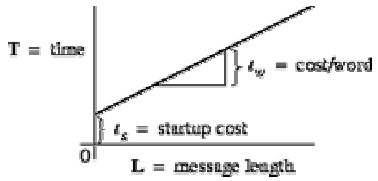


Activity plot during execution of a parallel program on eight processors. Each processor spends its time computing, communicating, or idling.

Overlapping Computation with Communication

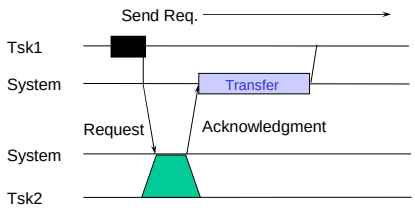
- A worker that has sent a request to the input task waits for the parameters to arrive.
- In some cases, efficiency can be improved by pre-fetching, that is, requesting the next parameter before it is needed.
- The worker can then perform computation while its request is being processed by the input task.
- A part of Computation and Communication can be executed in parallel - overlapped in time.
- In such a way idle time can be reduced.

Communication time - Model



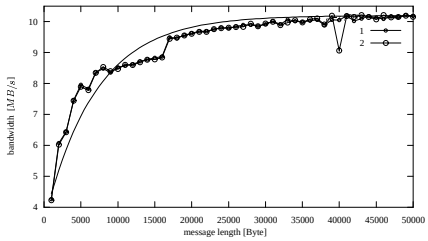
-Time T versus message length L is given, the slope of the line corresponds to the cost per word transferred tw (1/tw is bandwidth). T(0) is the message startup cost ts.
-This idealised model of communication performance is adequate for many purposes but does break down in some situations (congestion of communication channels).

Communication time (exm.)



- After a send request a fixed period is spend to service (ts). Then data start to transfer (tw).

Communication time (cont.)



Measured bandwidth by 100Mb/s FE (two runs & theory).
- **Short messages** have lower bandwidth because of startup time ts.
- By **longer messages** the bandwidth is approaching to the theoretical value (11 MB/s).

Speedup and Efficiency

- Execution time of a program on a single processor isT1.
- Execution time of the same parallelized program on p (possibly equally loaded) processors is Tp.
- Tp is composed of probably shorter execution time, some communication time, and some idle time.
- Speedup S = T1/Tp > 1,
- Efficiency E = S/p < 1.
- Ideally, S=p and E=1, typically, speedup increases slower than p and efficiency decreases.
- Isoefficiency function tells us how the amount of computation must scale with P to keep E constant. Algorithm with isoef. fun. of O(P) is highly scalable.

Performance Analysis

- We estimate the total computation time for a calculation domain of dimension f(N) by analysing the code and the number of floating point operations.

$$T_{comp} = t_c * f(N)$$

- The communication time depends on the number of messages M(N,P) per calculation step and on the amount of data D(N,P) to be transferred. Clearly both parameters depend on the number of processors.

$$T_{comm} = P * M(N,P) * D(N,P)$$

- If idle time is 0, because of well balanced computation the execution time of a parallelized program is:

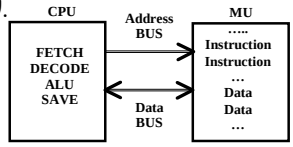
$$Tp = (T_{comp} + Tcomm) / P$$

How to increase computer performances ?

- By increasing the data parallelism in a processor (parallel busses, etc.)
- By using multiple functional units and pipelining,
- By using vectorized processors able to operate on several vectorized items at the same time,
- By using multiple computing and memory units,
- By increasing the system clock.
- ???

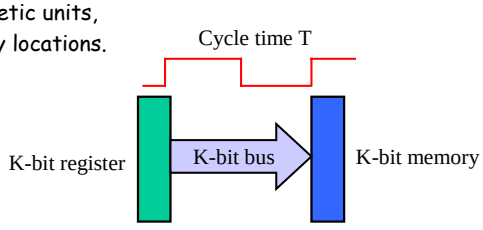
Single Computer Model (Von Neuman)

- Control unit (CU) - fetch and decode instructions,
- Processing Unit (PU) - execute instructions,
- CU + PU = CPU,
- Random Access Memory (RAM) Unit (MU) - save data and instructions,
- CPU read instructions and data from MU and write results back to the MU.



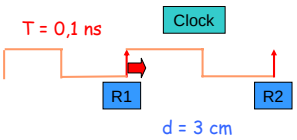
Data Parallelism

- Parallel bits ⇒ bytes ⇒ words ⇒ etc.
- Registers, data buses, address busses,
- Arithmetic units,
- Memory locations.



Physical Limits

- Speed of light and **electromagnetic field** $v = 3*10^8$ m/s,
- $f = 10^9$ Hz = 1 GHz $\Rightarrow$ $T = 10^{-9}$ s = 1ns $\Rightarrow$ Distance $d = v*T = 3*10^8$ [m/s] $*10^{-9}$ [s] = **30 cm**,
- For **10 GHz** we get the distance of **3cm**,



Pipelining

- Suppose that for finishing a complex repetitive task (e.g., car production) time *T* is needed for one person.
- If the task can be divided into *k* equally time-demanding subtasks, *k* workers can work in parallel, each its own subtask.
- The resources needed for the implementation of our task can be exploited totally with k workers.
- The first car will come from the pipe in time *T*, but the second car will be out already after *T+T/k*, and third after an additional delay of *T/k*.

Pipelining (cont.)

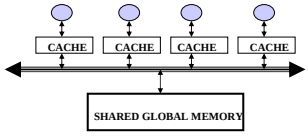
Time T needed for a single person to complete the task

1/5 of task	1/5 of task	1/5 of task	1/5 of task	1/5 of task
1/5 of task	1/5 of task	1/5 of task	1/5 of task	1/5 of task
1/5 of task	1/5 of task	1/5 of task	1/5 of task	1/5 of task
1/5 of task	1/5 of task	1/5 of task	1/5 of task	1/5 of task
1/5 of task	1/5 of task	1/5 of task	1/5 of task	1/5 of task
1/5 of task	1/5 of task	1/5 of task	1/5 of task	1/5 of task

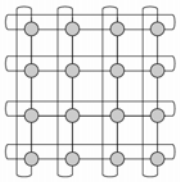
- What will happen if subtasks have diff. time complexity?

Increasing number of processors

- **Shared memory processors** (SMP)
  (number of processors is limited (< 6) because of shared memory access).



- **Distributed memory computers** or
  Message Passing Processors (MPP) (Network Of Workstations (NOW), Clusters (COW), GRIDs, Constellations (clusters of SMP or MPP)
  (communication channels are needed).



Properties of SMP

- Advantages
  - no need for explicit communication mechanisms
  - the data distribution is completely transparent to the user
  - simpler and more compact program code
- Disadvantages
  - the connection of the CPUs to each other and to the memory (caches can alleviate the bandwidth problem)
  - speed of the memory
  - in practice the number of processors limited to 10-20

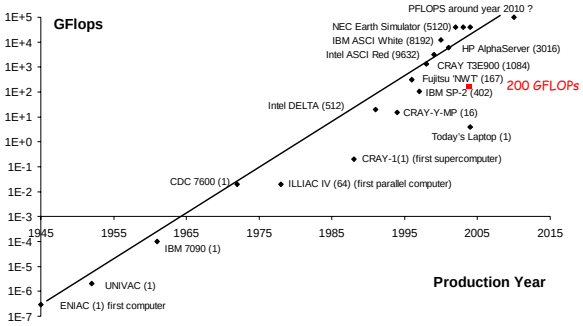
Properties of MPP

- Advantages
  - communication **bandwidth scales** up automatically with the number of processors
  - **no need** for very fast memory
  - **unlimited** number of processors
- Disadvantages
  - **global communication** is much slower (synchronization slower too)
  - **user has to distribute** the data over the processors
  - data **exchange** between processors has to be performed explicitly by calling a communication routine.
  - more **complex program code** (actual speed-up << theoretical speed-up)

Performance Measures

- Floating Point Operation is an arithmetic operation on 64 bit wide data.
- Number of Floating Point Operations per second (R): Mflop/s = 10^6, Gflop/s = 10^9, Tflop/s = 10^{12}
- Clock of the Central Processing Unit (MHz = 10^6/s, GHz = 10^9/s),
- Memory (Cache, Bulk, Back-up) (Mbit=Mb, Mbyte=MB, Mword=Mw).
- A problem has order of magnitude f(N)=O(g(N)) if there is a positive constant c such that $|f(N)| \leq c^*|g(N)|$ (e.g., for $2/3^*N^2+N$ operations the order of magnitude is $O(N^2)$, because terms of lower order than N^2 become insignificant if N becomes large.

R_{max} in the period 1945 - 2004

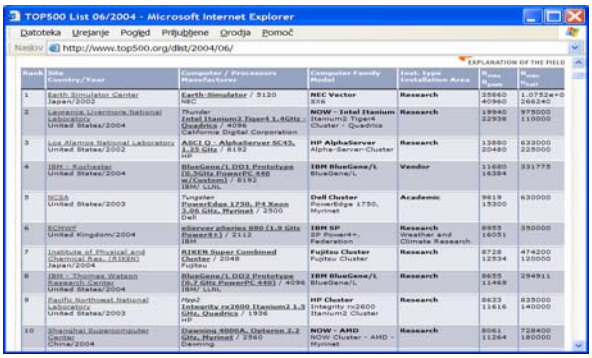


Moore's Law

- We can learn from the previous graphs that the computer performance increased **exponentially**.
- From 3^*10^{-7} GFlops in 1945 to 4.1^*10^4 GFlops in 2004 (**factor of 1.4^*10^{11} in 59 years**).
- $(R_2/R_1)=2^{(t2-t1)/Td}$

  (Td=time for double performance; log2=0.3)
- $Td=59^*\log(2)/\log(2^*10^{11}) =$ **1.57**
- **Moore's law** states that the computer performance doubles each **1.5 year** (long-term agreement OK).

Top10 from Top500 list June 2004



Rmax and **Rpeak** values are in GFlops. **Rmax** of the cluster in Itzling is about 200 GFlops (200 times smaller than No.1).