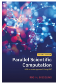
The Bulk Synchronous Parallel Model

Sections 1.1–1.2 of Parallel Scientific Computation, 2nd edition

Rob H. Bisseling

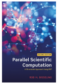
Utrecht University



What is a parallel computer?

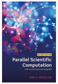
Frontier supercomputer at Oak Ridge National Lab, USA
Source: `https://www.ornl.gov`

A parallel computer consists of a set of processors that work together on solving a computational problem.



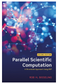
Single-processor speeds do not improve anymore

- The clock frequency of a single processor has reached its maximum at about 4 GHz in 2007. Since then no further increase, due to leakage of currents.
- Moore's Law ('the number of transistors on a chip doubles every 18 months') says we can still have more processors (cores) on a chip.
- Current supercomputers are all parallel computers.
- The Frontier supercomputer has 8 730 112 cores, running at 2 GHz, and it is the fastest supercomputer on earth (Top 500, June 2022). It can reach a speed of 1.1 Exaflop/s; 1 Exaflop $= 10^{18}$ floating-point operations.
- Frontier's power consumption is 21.1 MW, so it delivers 52 Gflop/watt.



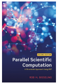
Why parallel computing?

- Today, almost every computer is a parallel computer:
  - octacore smartphones,
  - dualcore or quadcore laptop computers,
  - multicore desktop PCs,
  - compute servers,
  - massively parallel supercomputers.
- Higher speeds can only be obtained by exploiting parallelism, not by making single processors faster.

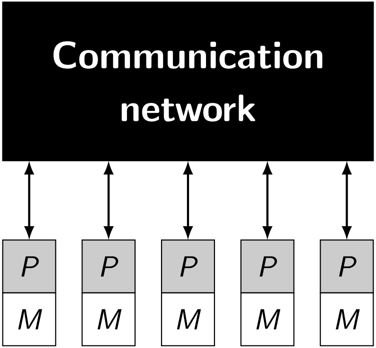


Why not?

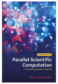
- It is more difficult to write parallel programs than to write sequential ones (i.e., for one processor). The work has to be distributed evenly over the processors and the amount of communication between the processors has to be minimized.
- But not much more difficult. That's why we have this course.
- Nonportable parallel programs may run fast on certain architectures, but surprisingly slow on others.
- That's why we teach portable parallel algorithm design and implementation.



Parallel computer: abstract model

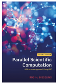


Bulk synchronous parallel (BSP) computer.
Proposed by Leslie Valiant, 1989.

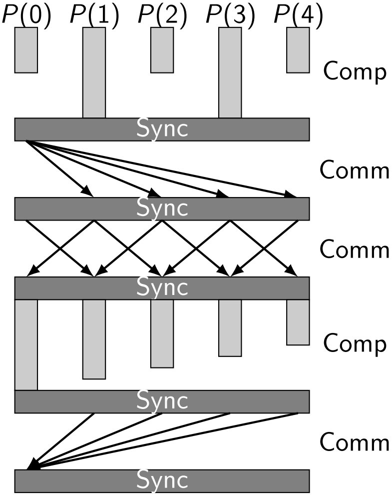


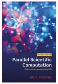
BSP computer

- A BSP computer consists of a collection of processors, each with its own memory. It is a distributed-memory computer.
- Access to local memory is fast and to remote memory slower, but uniform in time.
- No need to open the black box of the communication network. Algorithm designers should not worry about network details, only about global performance.
- Algorithms designed for a BSP computer are portable: they can be run efficiently on many different parallel computers, either with distributed memory, shared memory, or both.



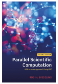
Parallel algorithm: supersteps





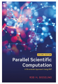
BSP algorithm

- A BSP algorithm consists of a sequence of supersteps.
- A computation superstep consists of many small steps, such as the floating-point operations (flops) addition, subtraction, multiplication, division.
- In scientific (numeric) computation, flops are the common unit for expressing computation cost. In nonnumeric computation, we can just call the operations ops.
- A communication superstep consists of many basic communication operations, each transferring a data word such as a real or integer from one processor to another.



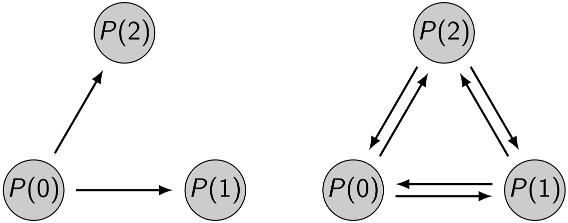
Mixed supersteps

- A mixed superstep has both computation and communication.
- In our theoretical algorithms, we prefer pure (nonmixed) supersteps. This helps in the design and analysis of parallel algorithms.
- For irregular computations such as in graph algorithms, it is more convenient to allow mixing.
- In our practical programs, we can freely mix computation and communication in each superstep.
- The BSP system then separates the two, by delaying all communications until after the computations have finished.

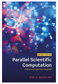


Communication superstep: *h*-relation

2-relations:



- An *h*-relation is a communication superstep in which every processor sends and receives at most h data words: $h = \max\{h_{\mathrm{s}}, h_{\mathrm{r}}\}$.
- h_{s} is the maximum number of data words sent by a processor.
- h_{r} is the maximum number of data words received by a processor.

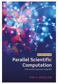


Cost of a communication superstep

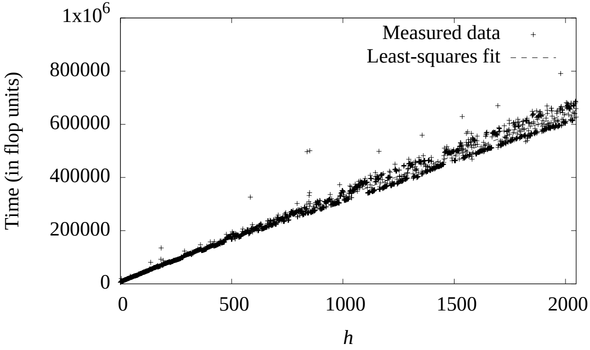
- The cost of an h-relation is

$$T(h) = hg + l,$$

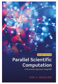
  where g is the time per data word and l the global synchronization time.
- g stands for the gap between sending successive data words, and l stands for latency.
- Motivation for hg: h determines the communication time, since entry/exit of processor is the bottleneck.
- Motivation for l: l contains fixed overhead such as start-up costs of sending data, costs of checking whether all data have arrived, and costs of the synchronization mechanism itself.



Time of an h-relation on a quadcore Apple iMac desktop



- $r = 8.44$ Gflop/s, $p = 4$, $g = 311$ flop (37 ns), $l = 16\,807$ flop (2.0 μs).
- 3.1 GHz Intel Core i5, running MulticoreBSP for C (v2.0.4β).

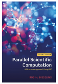


Cost of a computation superstep

- The cost of a computation superstep is

$$T = w + l,$$

  where w is the maximum number of flops of a processor.
- Processors with less than w flops have to wait. This waiting time is called idle time.
- To measure T, a wall clock is needed that gives elapsed time. Straightforwardly using a CPU timer will not work, since it does not measure idle time.
- Synchronizing the processors before every time measurement helps, but it takes time to synchronize!

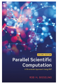


Cost of a mixed superstep

- The cost of a mixed superstep is

$$T = w + gh + l.$$

- For simplicity, we take the same l for all types of superstep.

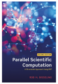


Cost of a BSP algorithm

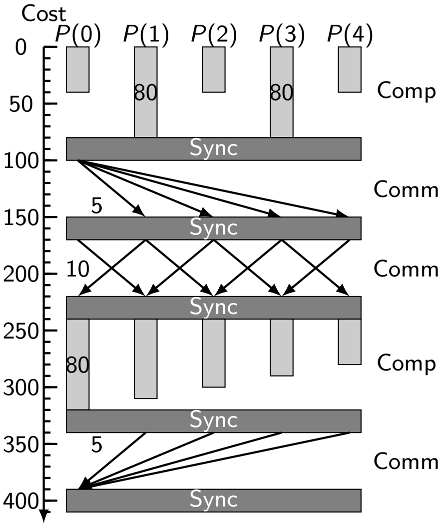
- The cost of a BSP algorithm is an expression of the form

$$a + bg + cl.$$

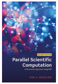
  This cost is obtained by adding the costs of all the supersteps.
- Note that $g = g(p)$ and $l = l(p)$ are a function of the number of processors p.
- The parameters a, b, c generally depend on p and a problem size n.



Parallel algorithm: supersteps



Cost for $p = 5$, $g = 2.5$, $l = 20$ is 410 flops.



Summary

- An abstract BSP machine is just a BSP(p, r, g, l) computer. This is all we need to know about the machine for developing algorithms. The parameters are:

p	number of processors
r	computing rate (in flop/s)
g	communication cost per data word (in flop time units)
l	global synchronization cost (in flop time units)

- The BSP model consists of
  - a distributed-memory architecture with a black box communication network providing uniform-time access to remote memories;
  - an algorithmic framework formed by a sequence of supersteps;
  - a cost model giving cost expressions of the form $a + bg + cl$.