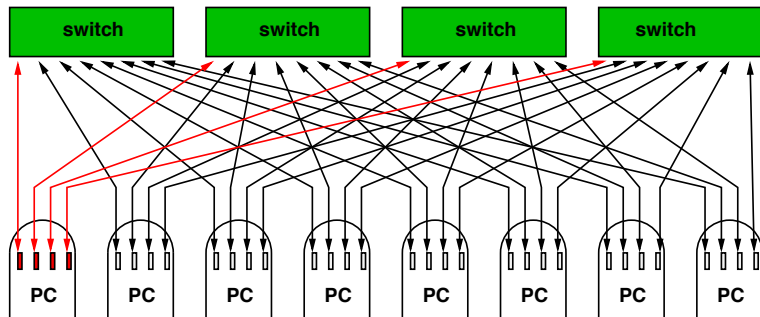# The Bulk Synchronous Parallel Model
(PSC §1.2)

What is a parallel computer?

A parallel computer consists of a set of processors (such as a cluster of PCs) that work together on solving a computational problem.

Moore's law has broken down

- Single-processor speed cannot continue to improve. Moore's Law ('Speed doubles every 18 months') has broken down in 2007, because of increasing power consumption $\mathcal{O}(f^3)$, where f is the clock frequency. Therefore, improvements can only come from using multiple processors or processor cores.

- Current supercomputers are all parallel computers. Chinese Tianhe-2 (Milkyway-2) has 3.12 million processor cores and is the fastest supercomputer on earth (Top 500, June 2014). It can reach a speed of 33.8 Petaflop/s; 1 Petaflop $= 10^{15}$ floating-point operations. Its power consumption is 17.8 MW.

Why parallel computing?

- It is almost as cheap to put two processor cores onto a chip (giving a dual-core chip) as putting just one. The doubled speed of a dual-core PC makes for fine advertising.

- Two cores running at clock frequency $f/2$ use only $1/4$ of the power of one core running at frequency f.

- But it is hard to achieve the same total computation speed in practice.

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 kept within limits.

- But not much more difficult. That's why we have this course.

- Parallel programs may run fast on certain architectures, but surprisingly slow on others.

- But not if you program in a portable fashion. That's what we try to teach.

# Parallel computer: abstract model
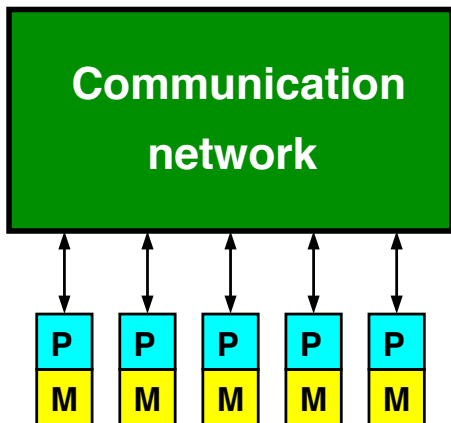
Bulk synchronous parallel (BSP) computer.
Proposed by Leslie Valiant, 1989.

BSP computer

- A BSP computer consists of a collection of processors, each with its own memory. It is a distributed-memory computer.
- Access to own memory is fast, to remote memory slower.
- Uniform-time access to all remote memories.
- 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.

Parallel algorithm: supersteps
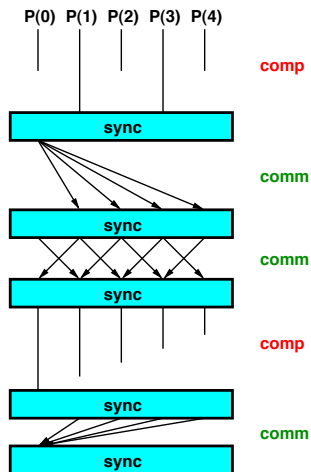
BSP algorithm

- 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 computing, flops are the common unit for expressing computation cost.
- 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.
- In our theoretical algorithms, we distinguish between the two types of supersteps. This helps in the design and analysis of parallel algorithms.
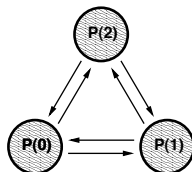- In our practical programs, we drop the distinction and mix computation and communication freely in each superstep.

# Communication superstep: *h*-relation

2-relations:
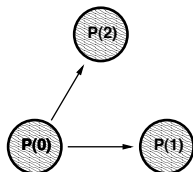
(a)   (b)

- ▶ An *h-relation* is a communication superstep in which every processor sends and receives at most *h* data words: $h = \max\{h_s, h_r\}$.
- ▶ 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.
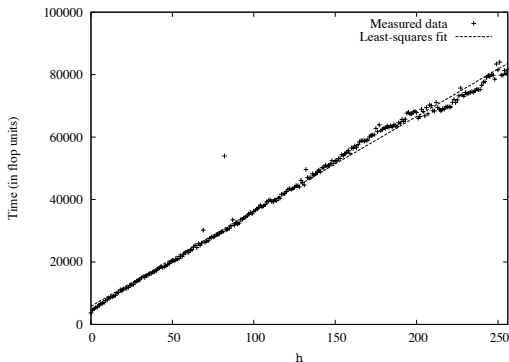
Cost of communication superstep

- $T(h) = hg + l$, where g is the time per data word and l the global synchronisation time.
- Motivation hg: h determines communication time, since entry/exit of processor is the bottleneck .
- Motivation l: contains fixed overhead such as start-up costs of sending data, costs of checking whether all data have arrived at their destination, and costs of the synchronisation mechanism itself.

Time of an *h*-relation on an 8-processor IBM SP2
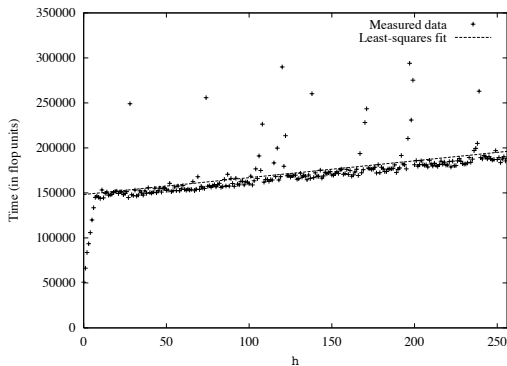
$r = 212$ Mflop/s, $p = 8$, $g = 187$ flop (0.88μs),
$l = 148212$ flop ($698\ \mu$s). Year 2000.

Time of an *h*-relation on a 4-core Apple iMac desktop

$r = 6.8$ Gflop/s, $p = 4$, $g = 304$ flop (44 ns),
$l = 5796$ flop (0.85 μs). Year 2014.
3.4 GHz Intel Core i5, 4 cores, running MulticoreBSP for C.

Cost of computation superstep

- $T = w + l$, where w is the maximum number of flops of a processor in the superstep.
- Processors with less than w flops have to wait. This waiting time is called idle time.
- To measure T, a wall clock is needed, giving the elapsed time. Straightforwardly using a CPU timer will not work, since it does not measure idle time.
- Synchronising the processors before every time measurement helps, but it takes time to synchronise!
- Same l as in communication superstep, for simplicity.

Cost of algorithm

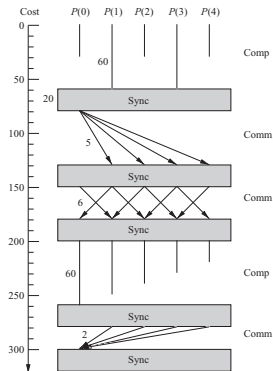- The cost of a BSP algorithm is an expression of the form

$$a + bg + cl.$$

  This cost is obtained by adding the costs of all the supersteps.

- Note that $g = g(p)$ and $l = l(p)$ are in general a function of the number of processors p.

- The parameters a, b, c depend in general on p and on a problem size n.

Parallel algorithm: supersteps

Cost of BSP algorithm for $p = 5$, $g = 2.5$, $l = 20$ is 320 flops.
First computation superstep costs $60 + 20 = 80$ flops.
First communication superstep costs $4 \cdot 5 \cdot 2.5 + 20 = 70$ 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 synchronisation 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 $a + bg + cl$.