

# Distributed and Parallel Computing

## Lecture 02

Alan P. Sexton

University of Birmingham

Spring 2016

# Measuring Parallel Speedup

- **Latency:** the time from initiating to completing a task
  - Units of time
- **Work:** a measure of what the CPU/GPU has to do for a particular task
  - number of floating point operations
  - number of images processed
  - number of pixels processed
  - number of simulation steps
- **Throughput:** work done per unit time

# Speedup and Efficiency

- **Speedup<sub>P</sub>,  $S_P$** : The ratio of the latency for solving a problem with 1 hardware unit to the latency of solving it with  $P$  units
  - $S_P = \frac{T_1}{T_P}$
  - Perfect linear speedup:  $S_P = P$
- **Efficiency,  $E_P$** : The ratio of the latency for solving a problem with 1 hardware unit to  $P$  times the latency of solving it on  $P$  units
  - This measures how well the individual hardware units are contributing to the overall solution
  - $E_P = \frac{T_1}{P \times T_P} = \frac{S_P}{P}$
  - Perfect linear efficiency:  $E_P = 1$

# Interpreting speedup and efficiency

- Sub-linear speedup and efficiency is normal
  - Overhead in parallelizing a problem
- Super-linear speedup is possible, but usually due to special conditions
  - e.g. Serial version does not fit in CPU cache but each of the parallel sub-problems do.
- Important to compare the best serial version of the program with the parallel version
  - Serial algorithm A is fast but hard to parallelize
  - Serial algorithm B is slow but gives Parallel algorithm B
  - To measure speedup/efficiency, compare Serial A to Parallel B
  - Both must solve the same problem, but allow for minor differences
    - e.g. small round-off differences (but be aware of differing precision on host and on GPU!)
    - differences due to different execution order

# Strong Scalability

Gene Amdahl, in 1967, argued that the time spent executing a program is composed of the time spent doing non-parallelizable work plus the time spent doing parallelizable work:

$$T_1 = T_{\text{ser}} + T_{\text{par}}$$

Therefore, if the speedup on  $P$  units of the parallel part only is  $s$

$$T_P = T_{\text{ser}} + \frac{T_{\text{par}}}{s}$$

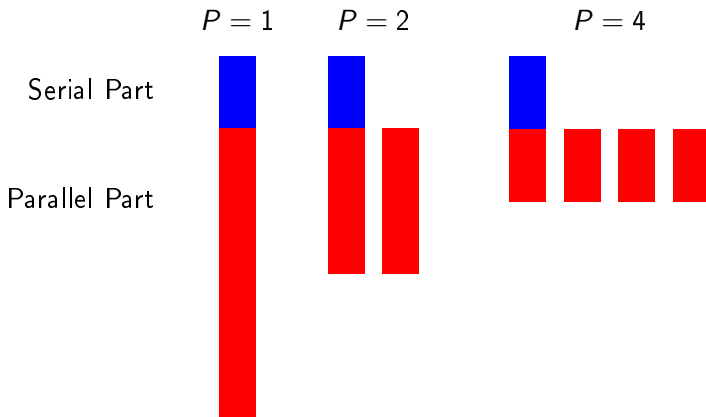
Hence the overall speed up given the speedup of the parallel part is  $s$  is:

$$S_P = \frac{T_{\text{ser}} + T_{\text{par}}}{T_{\text{ser}} + \frac{T_{\text{par}}}{s}}$$

If we let  $f$  be the fraction of a program that is parallelizable, then  $T_{\text{ser}} = (1 - f)T_1$  and  $T_{\text{par}} = fT_1$ . Hence

$$S_P = \frac{(1 - f)T_1 + fT_1}{(1 - f)T_1 + \frac{fT_1}{s}} = \frac{1}{1 - f + \frac{f}{s}} \quad (\text{Amdahl's Law})$$

# Amdahl's Law Graphically



# Interpreting Amdahl's Law

Amdahl's law seems to say that there is a limit to parallel speedup

$$\lim_{s \rightarrow \infty} S_P = \lim_{s \rightarrow \infty} \frac{1}{1 - f + \frac{f}{s}} = \frac{1}{1 - f}$$

or, in other words

$$\begin{aligned} \lim_{s \rightarrow \infty} \frac{T_1}{T_P} &= \frac{1}{1 - f} \\ \Rightarrow \lim_{P \rightarrow \infty} T_P &= T_{\text{ser}} \quad \text{assuming } P \rightarrow \infty \Rightarrow s \rightarrow \infty \end{aligned}$$

John Gustafson and Edwin Barsis, in 1988, argued that Amdahl's law did not give the full picture

- Amdahl kept the task fixed and considered how much you could shorten the processing time by running in parallel
- Gustafson and Barsis kept the processing time fixed and considered how much larger a task you could handle in that time by running in parallel
- This was motivated by observing that, as computers increase in power, the problems that they are applied to often increase in size



Assume that  $W$  is the workload that can be executed without parallelism in time  $T$ . If  $f$  is the fraction of the workload that is parallelizable, then

$$W = (1 - f)W + fW$$

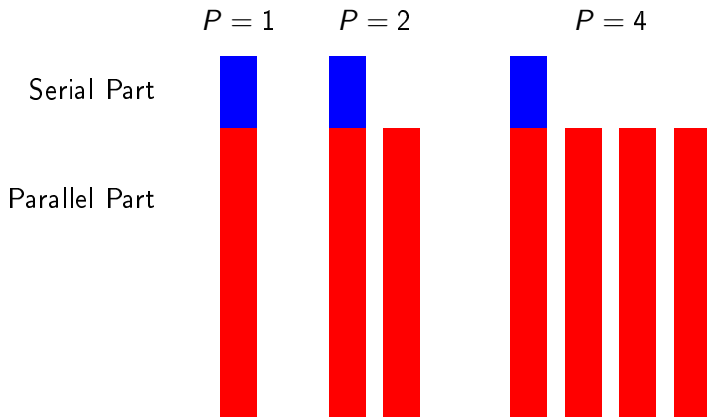
With speedup  $s$ , we can run  $s$  times the parallelizable part in the same time, although we don't change the amount of work done in the non-parallelizable part:

$$W_s = (1 - f)W + sfW$$

If we do  $W_s$  in time  $T$ , we are, on average, doing  $W$  amount of work in time  $\frac{TW}{W_s}$ . Hence the total speedup is:

$$S_s = \frac{T}{TW/W_s} = \frac{W_s}{W} = 1 - f + fs \quad (\text{Gustafson-Barsis})$$

# Gustafson-Barsis Law Graphically



- Both Amdahl (A) and Gustafson-Barsis (GB) are correct
- The two together give guidance on what kinds of problems can benefit from parallelization and how
- You can only go so much faster on a fixed problem by parallelization
- You can avoid Amdahl's limit on speedup if you can increase the size of the parallelizable part of the problem faster than you increase the size of the non-parallelizable part