The limits of parallelism: Amdahl's law

Speedup is defined as

$$\frac{\text{time for serial execution}}{\text{time for parallel execution}}$$

or, more precisely, as

$$\frac{\text{time for serial execution of best serial algorithm}}{\text{time for parallel execution of our algorithm}}$$

Give two reasons why it is better to define it the second way than the first.

[§4.3.1] If some portions of the problem don't have much concurrency, the speedup on those portions will be low, lowering the average speedup of the whole program.

Exercise: Submit your answers to the questions below.

Suppose that a program is composed of a serial phase and a parallel phase.

- The whole program runs for 1 time unit.
- The serial phase runs for time *s*, and the parallel phase for time 1–*s*.

Then regardless of how many processors *N* are used, the execution time of the program will be at least ___

and the speedup will be no more than ___. This is known as *Amdahl's law.*

For example, if 25% of the program's execution time is serial, then regardless of how many processors are used, we can achieve a speedup of no more than __.

Efficiency is defined as

$$\frac{\text{speedup}}{\text{number of processors}}$$

Let us normalize computation time so that

- the serial phase takes time 1, and
- the parallel phase takes time p if run on a single processor.

Then if run on a machine with N processors, the parallel phase takes p/N.

Let α be the ratio of serial time to total execution time. Thus

$$\alpha = \frac{1}{1 + p/N} = \frac{N}{N + p}.$$

For large N, α approaches __, so efficiency approaches ___.

Does it help to add processors?

Gustafson's law: But this is a pessimistic way of looking at the situation.

In 1988, Gustafson et al. noted that as computers become more powerful, people run larger and larger programs.

Therefore, as N increases, p tends to increase too. Thus, the fraction of time $1-\alpha$ does not necessarily shrink with increasing N, and efficiency remains reasonable.

There may be a maximum to the amount of speedup for a given problem size, but since the problem is "scaled" to match the processing power of the computer, there is no clear maximum to "scaled speedup."

Gustafson's law states that any sufficiently large problem can be efficiently parallelized.