

APPENDIX E

AMDAHL'S LAW

William Stallings

Copyright 2014

Supplement to
Operating Systems, Eighth Edition
Pearson 2014
<http://williamstallings.com/OperatingSystems/>

When considering system performance, computer system designers look for ways to improve performance by improvement in technology or change in design. Examples include the use of parallel processors, the use of a memory cache hierarchy, and speedup in memory access time and I/O transfer rate due to technology improvements. In all of these cases, it is important to note that a speedup in one aspect of the technology or design does not result in a corresponding improvement in performance. This limitation is succinctly expressed by Amdahl's law.

Amdahl's law was first proposed by Gene Amdahl¹ in 1967 and deals with the potential speedup of a program using multiple processors compared to a single processor. Consider a program running on a single processor such that a fraction $(1 - f)$ of the execution time involves code that is inherently serial and a fraction f that involves code that is infinitely parallelizable with no scheduling overhead. Let T be the total execution time of the program using a single processor. Then the speedup using a parallel processor with N processors that fully exploits the parallel portion of the program is as follows:

$$\text{Speedup} = \frac{\text{time to execute program on a single processor}}{\text{time to execute program on } N \text{ parallel processors}} = \frac{T(1-f) + Tf}{T(1-f) + \frac{Tf}{N}} = \frac{1}{(1-f) + \frac{f}{N}}$$

¹ Amdahl, G. "Validity of the Single-Processor Approach to Achieving Large-Scale Computing Capability." *Proceedings, of the AFIPS Conference*, 1967.

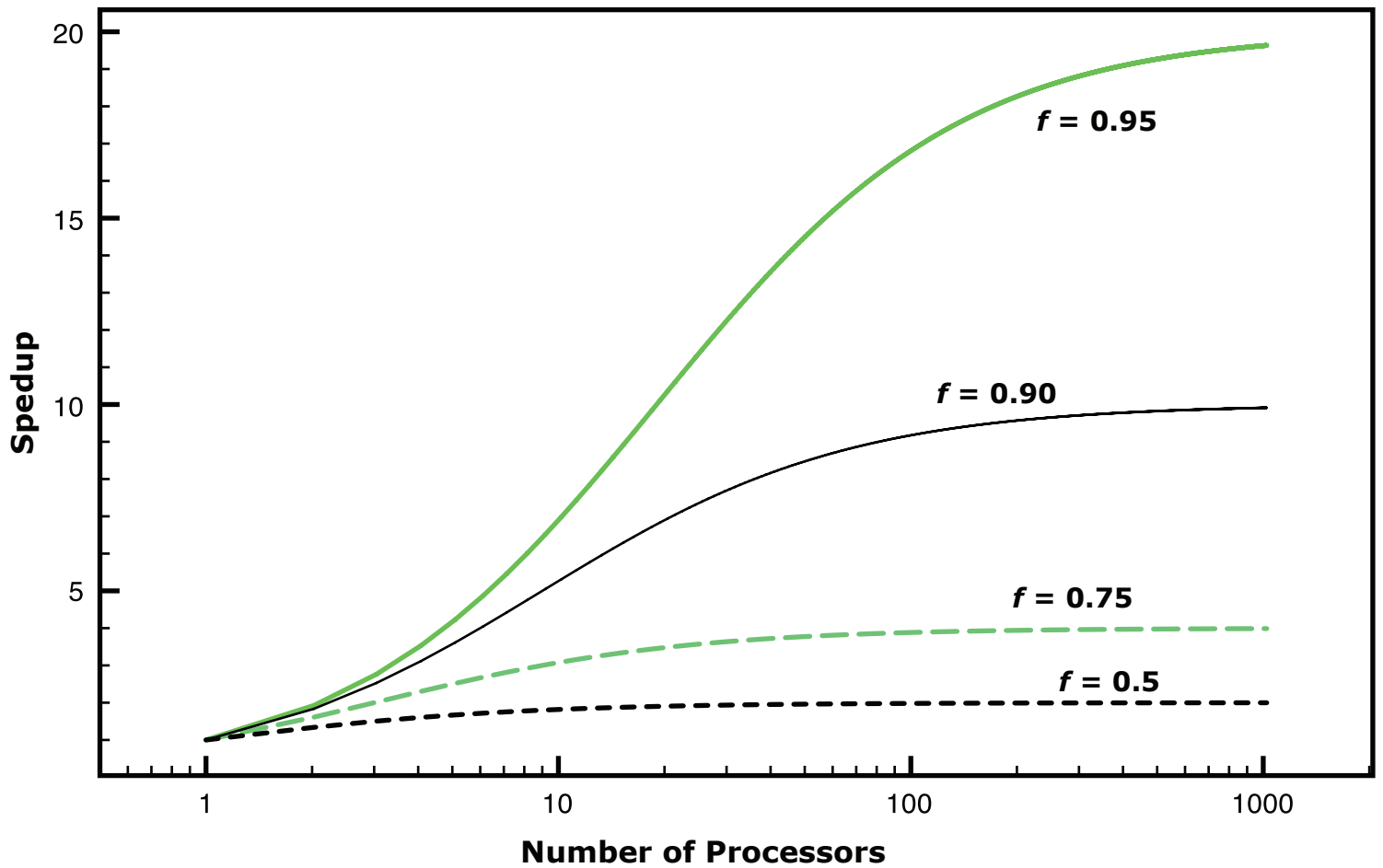


Figure E.1 Amdahl's Law for Multiprocessors

This equation is illustrated in Figure E.1. Two important conclusions can be drawn:

1. When f is small, the use of parallel processors has little effect.
2. As N approaches infinity, speedup is bound by $1/(1 - f)$, so that there are diminishing returns for using more processors.

These conclusions are too pessimistic, an assertion first put forward in [GUST88]. For example, a server can maintain multiple threads or multiple tasks to handle multiple clients and execute the threads or tasks in parallel up to the limit of the number of processors. Many database applications involve computations on massive amounts of data that can be split up into multiple parallel tasks. Nevertheless, Amdahl's law illustrates the problems facing industry in the development of multicore machines with an ever-growing number of cores: The software that runs on such machines must be adapted to a highly parallel execution environment to exploit the power of parallel processing.

Amdahl's law can be generalized to evaluate any design or technical improvement in a computer system. Consider any enhancement to a feature of a system that results in a speedup. The speedup can be expressed as

$$\text{Speedup} = \frac{\text{Performance after enhancement}}{\text{Performance before enhancement}} = \frac{\text{Execution time before enhancement}}{\text{Execution time after enhancement}}$$

Suppose that a feature of the system is used during execution a fraction of the time f , before enhancement, and that the speedup of that feature after enhancement is SU_f . Then the overall speedup of the system is

$$Speedup = \frac{1}{(1-f) + \frac{f}{SU_f}}$$

For example, suppose that a task makes extensive use of floating-point operations, with 40% of the time is consumed by floating-point operations. With a new hardware design, the floating-point module is speeded up by a factor of K . Then the overall speedup is:

$$Speedup = \frac{1}{0.6 + \frac{0.4}{K}}$$

Thus, independent of K , the maximum speedup is 1.67.