Amdahl's law

Gene Myron Amdahl (November 16, 1922 – November 10, 2015) was an American computer architect and high-tech entrepreneur, chiefly known for his work on mainframe computers at IBM and later his own companies, especially Amdahl Corporation. He formulated Amdahl's Law…
(https://en.wikipedia.org/wiki/Gene_Amdahl)

"The overall performance improvement gained by optimizing a single part of a system is limited by the fraction of time that the improved part is actually used".

Speed up

Old system

20 Sec — INPUT/OUTPUT

50 Sec — MEMORY OPERATION

30 sec — ALU OPERATION

100 Sec

New system

(Memory upgraded only)

20 Sec — INPUT/OUTPUT

25 Sec — Memory operation

30 sec — ALU OPERATION

75 Sec

$$\text{Speed up, S} = \frac{\text{Run time in Old system}}{\text{Rum time in partially enhanced system}} = 100/75 = 1.33$$

Old system

20 Sec — INPUT/OUTPUT

50 Sec — MEMORY OPERATION

Fraction of time

F = 50/100 = 0.5

30 sec — ALU OPERATION

100 Sec

New system (Memory upgraded only)

20 Sec — INPUT/OUTPUT

25 Sec — Memory operation

Acceleration factor

s = 50 Sec/25 Sec = 2

30 sec — ALU OPERATION

75 Sec

Normalized Rum time = 1

Amdahl's law

$$\text{Speed up, } S = \frac{1}{(1 - F) + F/s} = \frac{1}{(1 - 0.5) + 0.5/2} = 1.33$$

$$\text{Speed up, } S = \frac{\text{Run time in Old system}}{\text{Rum time in partially enhanced system}}$$

= 100/75 = 1.33

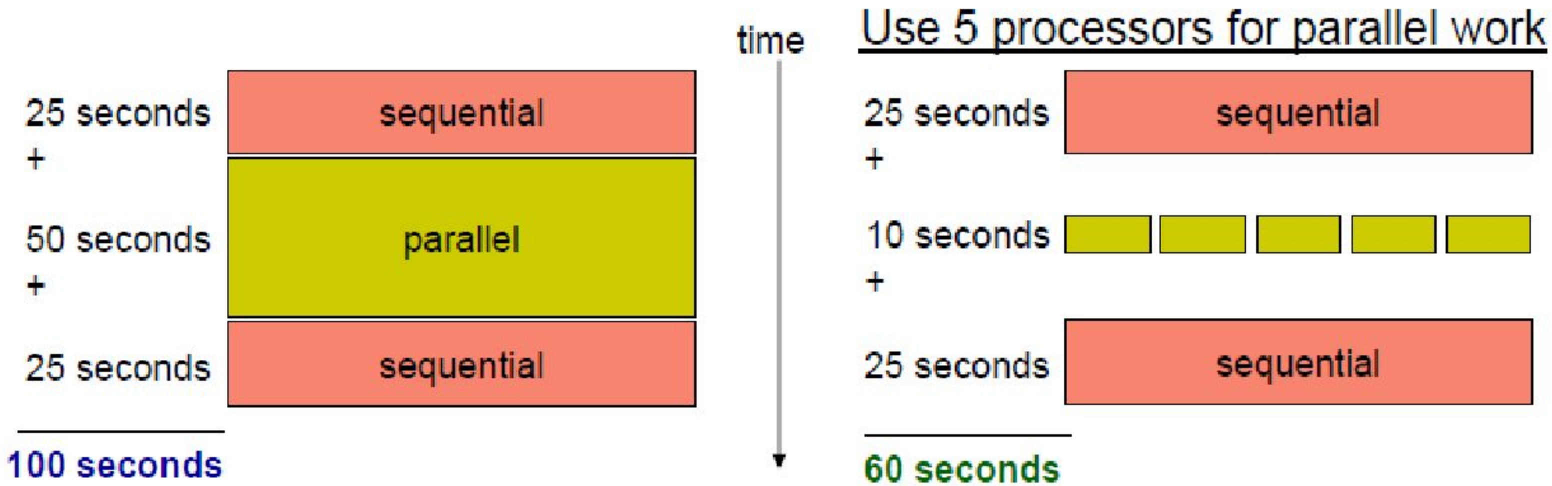
Amdahl's law

- Amdahl's law is often used in parallel computing to predict the theoretical speedup when using multiple processors.
- For example, if a program needs 20 hours to complete using a single thread, but a one-hour portion of the program cannot be parallelized, therefore only the remaining 19 hours execution time can be parallelized, then regardless of how many threads are devoted to a parallelized execution of this program, the minimum execution time cannot be less than one hour.
- Hence, the theoretical speedup is limited to at most 20 times the single thread performance

Parallel Architecture

$$\text{Parallel Speedup} = \frac{\text{Time to execute the program with 1 processor}}{\text{Time to execute the program with } N \text{ processors}}$$

Amdahl's Law



- Speedup = old running time / new running time
  = 100 seconds / 60 seconds
  = 1.67
  (parallel version is 1.67 times faster)

Amdahl's Law

- p = fraction of work that can be parallelized
- n = the number of processor

$$speedup = \frac{\text{old running time}}{\text{new running time}}$$

$$= \frac{1}{(1-p) + \frac{p}{n}}$$

fraction of time to complete sequential work

fraction of time to complete parallel work

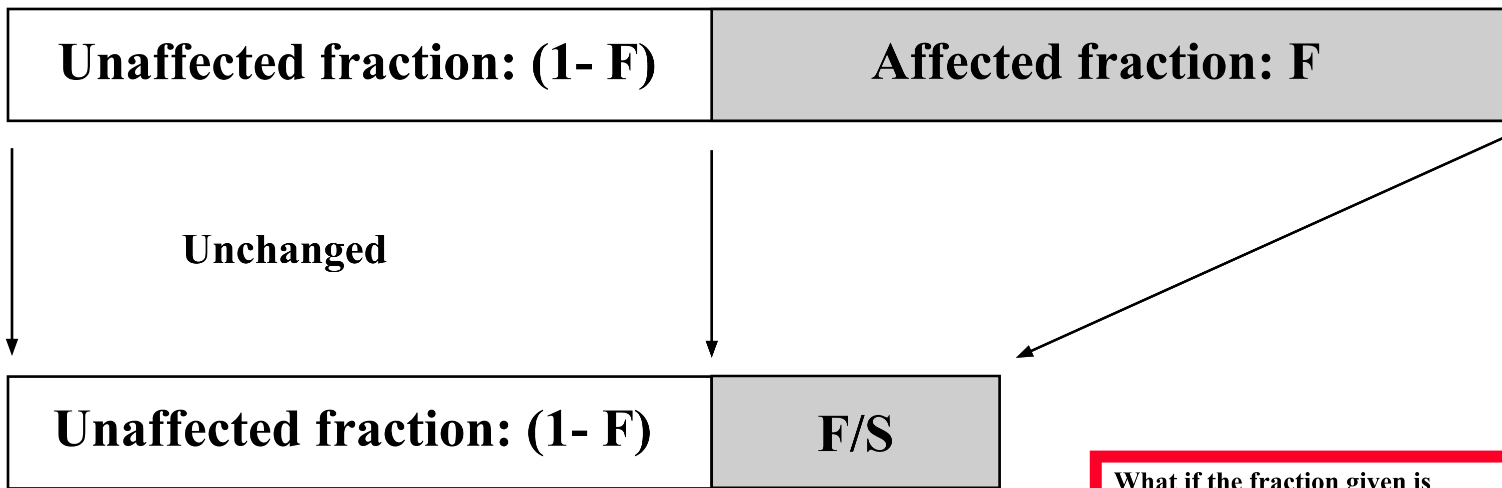
Amdahl's Law

Enhancement E accelerates fraction F of original execution time by a factor of S

Before:

Execution Time without enhancement E: (Before enhancement is applied)

- **shown normalized to 1 = (1-F) + F =1**



What if the fraction given is after the enhancement has been applied? How would you solve the problem? (i.e find expression for speedup)

After:

Execution Time with enhancement E:

$$\text{Speedup(E)} = \frac{\text{Execution Time without enhancement E}}{\text{Execution Time with enhancement E}} = \frac{1}{(1 - F) + F/S}$$

Amdahl's Law

- The performance enhancement possible due to a given design improvement is limited by the amount that the improved feature is used

Performance improvement or speedup due to enhancement E:

$$\text{Speedup(E)} = \frac{\text{Execution Time without E}}{\text{Execution Time with E}} = \frac{\text{Performance with E}}{\text{Performance without E}}$$

- Suppose that enhancement E accelerates a fraction F of the original execution time by a factor S and the remainder of the time is unaffected then:

$$\text{Execution Time with E} = ((1\text{-}F) + F/S) \times \text{Execution Time without E}$$

Hence speedup is given by:

$$\text{Speedup(E)} = \frac{\cancel{\text{Execution Time without E}}}{((1 - F) + F/S) \times \cancel{\text{Execution Time without E}}} = \frac{1}{(1 - F) + F/S}$$

- Amadahl's law concerns the speedup achievable from an improvement in computation that affects a fraction *F* of the computation, where the improvement has a speedup of *S*.

Before improvement	1 – F	F
After improvement	1 – F	F / S

- Execution time before improvement: $(1 - F) + F = 1$
- Execution time after improvement: $(1 - F) + F / S$
- Speedup obtained:

$$\boxed{\text{Speedup} = \frac{1}{(1 - F) + F / S}}$$

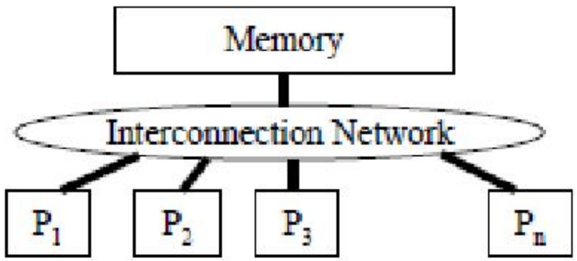
- As $S \rightarrow \infty$, $Speedup \rightarrow 1 / (1 - F)$
  - The fraction *F* limits the maximum speedup that can be obtained.
- Illustration of law of diminishing returns:
  - Let $F = 0.25$.
  - The table shows the speedup ($= 1 / (1 - F + F / S)$) for various values of *S*.

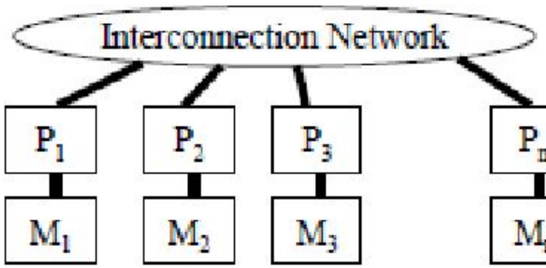
$1 / (1 - 0.25) = 1.33$

S	Speedup
1	1.00
2	1.14
5	1.25
10	1.29

S	Speedup
50	1.32
100	1.33
1000	1.33
100,000	1.33

- Two primary patterns of multicore architecture design
  - Shared memory
    - Ex: Intel Core 2 Duo/Quad
    - One copy of data shared among many cores
    - Atomicity, locking and synchronization essential for correctness
    - Many scalability issues
  - Distributed memory
    - Ex: Cell
    - Cores primarily access local memory
    - Explicit data exchange between cores
    - Data distribution and communication orchestration is essential for performance





- Some examples:
  - We make 10% of a program 90X faster, speedup = 1 / (0.9 + 0.1 / 90) = 1.11
  - We make 90% of a program 10X faster, speedup = 1 / (0.1 + 0.9 / 10) = 5.26
  - We make 25% of a program 25X faster, speedup = 1 / (0.75 + 0.25 / 25) = 1.32
  - We make 50% of a program 20X faster, speedup = 1 / (0.5 + 0.5 / 20) = 1.90
  - We make 90% of a program 50X faster, speedup = 1 / (0.1 + 0.9 / 50) = 8.47

- Suppose we are running a set of programs on a RISC processor, for which the following instruction mix is observed:

Operation	Frequency	CPI_i
Load	20 %	5
Store	8 %	3
ALU	60 %	1
Branch	12 %	2

$W_i * CPI_i$	% Time
1.00	0.48
0.24	0.12
0.60	0.29
0.24	0.11

CPI = 2.08

1 / 2.08

We carry out a design enhancement by which the CPI of Load instructions reduces from 5 to 2. What will be the overall performance improvement?

Fraction enhanced F = 0.48

Fraction unaffected 1 – F = 1 – 0.48 = 0.52

Enhancement factor S = 5 / 2 = 2.5

Therefore, speedup is

$$\frac{1}{(1-F) + F/S} = \frac{1}{0.52 + 0.48/2.5} = 1.40$$

- Alternate way of calculation:
  - Old CPI = 2.08
  - New CPI = 0.20 * **2** + 0.08 * 3 + 0.60 * 1 + 0.12 * 2 = 1.48

$$\boxed{\begin{aligned} Speedup &= \frac{XT_{orig}}{XT_{new}} = \frac{IC * CPI_{old} * C}{IC * CPI_{new} * C} \\ &= \frac{CPI_{old}}{CPI_{new}} = \frac{2.08}{1.48} = 1.40 \end{aligned}}$$

- The execution time of a program on a machine is found to be 50 seconds, out of which 42 seconds is consumed by multiply operations. It is required to make the program run 5 times faster. By how much must the speed of the multiplier be improved?
  - Here, F = 42 / 50 = 0.84
  - According to Amadahl's law,

$$5 = 1 / (0.16 + 0.84 / S)$$

or, $0.80 + 4.2 / S = 1$

or, $S = 21$

- The execution time of a program on a machine is found to be 50 seconds, out of which 42 seconds is consumed by multiply operations. It is required to make the program run 8 times faster. By how much must the speed of the multiplier be improved?
  - Here, F = 42 / 50 = 0.84
  - According to Amadahl's law,

$$8 = 1 / (0.16 + 0.84 / S)$$

or, $1.28 + 6.72 / S = 1$

or, $S = -24$

No amount to speed improvement in the multiplier can achieve this.

Maximum speedup achievable:
$1 / (1 - F) = 6.25$

- Suppose we plan to upgrade the processor of a web server. The CPU is 30 times faster on search queries than the old processor. The old processor is busy with search queries 80% of the time. Estimate the speedup obtained by the upgrade.
  - Here, $F = 0.80$ and $S = 30$
  - Thus, speedup $= 1 / (0.20 + 0.80 / 30) = 4.41$

- The total execution time of a typical program is made up of 60% of CPU time and 40% of I/O time. Which of the following alternatives is better?
  a) Increase the CPU speed by 50%
  b) Reduce the I/O time by half

  Assume that there is no overlap between CPU and I/O operations.

CPU	I/O	CPU	I/O	CPU	I/O

- Increase CPU speed by 50%
  - Here, $F = 0.60$ and $S = 1.5$
  - Speedup $= 1 / (0.40 + 0.60 / 1.5) = 1.25$
- Reduce the I/O time by half
  - Here, $F = 0.40$ and $S = 2$
  - Speedup $= 1 / (0.60 + 0.40 / 2) = 1.25$

Thus, both the alternatives result in the same speedup.

- Suppose that a compute-intensive bioinformatics program is running on a given machine X, which takes 10 days to run. The program spends 25% of its time doing integer instructions, and 40% of time doing I/O. Which of the following two alternatives provides a better tradeoff?
  a) Use an optimizing compiler that reduces the number of integer instructions by 30% (assume all integer instructions take the same time).
  b) Optimizing the I/O subsystem that reduces the latency of I/O operations from 10 μsec to 5 μsec (that is, speedup of 2).

- Alternative (a):
  - Here, $F = 0.25$ and $S = 100 / 70$
  - Speedup $= 1 / (0.75 + 0.25 * 70 / 100) = 1.08$

- Alternative (b):
  - Here, $F = 0.40$ and $S = 2$
  - Speedup $= 1 / (0.60 + 0.40 / 2) = 1.25$

Extension to Multiple Enhancements

- Suppose we carry out multiple optimizations to a program:
  - Optimization 1 speeds up a fraction F_1 of the program by a factor S_1
  - Optimization 2 speeds up a fraction F_2 of the program by a factor S_2

$1 - F_1 - F_2$	F_1	F_2

$1 - F_1 - F_2$	F_1 / S_1	F_2 / S_2

Speedup

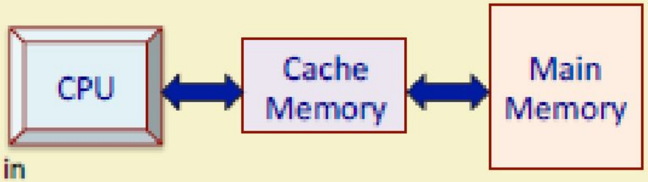
$$\frac{1}{(1 - F_1 - F_2) + F_1 / S_1 + F_2 / S_2}$$

- General expression:
  - Assume m enhancements of fractions $F_1, F_2, \ldots, F_m$ by factors of $S_1, S_2, \ldots, S_m$ respectively.

$$\boxed{\text{Speedup} = \frac{1}{(1 - \sum_{i=1}^{m} F_i) + \sum_{i=1}^{m} \frac{F_i}{S_i}}}$$

Consider an example of memory system.

- Main memory and a fast memory called cache memory.
- Frequently used parts of program/data are kept in cache memory.
- Use of the cache memory speeds up memory accesses by a factor of 8.
- Without the cache, memory operations consume a fraction 0.40 of the total execution time.
- Estimate the speedup.



Solution

$$\textbf{Speedup} = \frac{1}{(1-F) + F/S} = \frac{1}{(1-0.4) + 0.4/8} = 0.91$$

- Now we consider two levels of cache memory, L1-cache and L2-cache.

  Assumptions:
  - Without the cache, memory operations take 30% of execution time.
  - The L1-cache speeds up 80% of memory operations by a factor of 4.
  - The L2-cache speeds up 50% of the remaining 20% memory operations by a factor of 2.

  We want to find out the overall speedup.

- Solution:
  - Memory operations = 0.3
  - $F_{L1} = 0.3 * 0.8 = 0.24$
  - $S_{L1} = 4$
  - $F_{L2} = 0.3 * (1-0.8) * 0.5 = 0.03$
  - $S_{L2} = 2$

Speedup

$$\frac{1}{(1-F_{L1}-F_{L2}) + F_{L1}/S_{L1} + F_{L2}/S_{L2}}$$

$$\frac{1}{(1-0.24-0.03) + 0.24/4 + 0.03/2}$$

$$= \mathbf{1.24}$$

Performance Enhancement Example

- **For the RISC machine with the following instruction mix given earlier:**

Op	Freq	Cycles	CPI(i)	% Time
ALU	50%	1	.5	23%
Load	20%	5	1.0	45%
Store	10%	3	.3	14%
Branch	20%	2	.4	18%

CPI = 2.2

- **If a CPU design enhancement improves the CPI of load instructions from 5 to 2, what is the resulting performance improvement from this enhancement:**

Fraction enhanced = F = 45% or .45

Unaffected fraction = 1- F = 100% - 45% = 55% or .55

Factor of enhancement = S = 5/2 = 2.5

Using Amdahl's Law:

$$\text{Speedup(E)} = \frac{1}{(1 - F) + F/S} = \frac{1}{.55 + .45/2.5} = 1.37$$

An Alternative Solution Using CPU Equation

Op	Freq	Cycles	CPI(i)	% Time
ALU	50%	1	.5	23%
Load	20%	5	1.0	45%
Store	10%	3	.3	14%
Branch	20%	2	.4	18%

CPI = 2.2

- **If a CPU design enhancement improves the CPI of load instructions from 5 to 2, what is the resulting performance improvement from this enhancement:**

Old CPI = 2.2

New CPI of load is now 2 instead of 5

New CPI = .5 x 1 + .2 x 2 + .1 x 3 + .2 x 2 = 1.6

$$\text{Speedup(E)} = \frac{\text{Original Execution Time}}{\text{New Execution Time}} = \frac{\cancel{\text{Instruction count}} \times \text{old CPI} \times \cancel{\text{clock cycle}}}{\cancel{\text{Instruction count}} \times \text{new CPI} \times \cancel{\text{clock cycle}}}$$

$$= \frac{\text{old CPI}}{\text{new CPI}} = \frac{2.2}{1.6} = 1.37$$

Which is the same speedup obtained from Amdahl's Law in the first solution.

$$T = I \times CPI \times C$$

Performance Enhancement Example

- **A program runs in 100 seconds on a machine with multiply operations responsible for 80 seconds of this time. By how much must the speed of multiplication be improved to make the program four times faster?**

$$\text{Desired speedup} = 4 = \frac{100}{\text{Execution Time with enhancement}}$$

→ Execution time with enhancement = 100/4 = 25 seconds

25 seconds = (100 - 80 seconds) + 80 seconds / S

25 seconds = 20 seconds + 80 seconds / S

→ 5 = 80 seconds / S

→ S = 80/5 = 16

Alternatively, it can also be solved by finding enhanced fraction of execution time:

F = 80/100 = .8

and then solving Amdahl's speedup equation for desired enhancement factor S

$$\text{Speedup(E)} = \frac{1}{(1 - F) + F/S} = 4 = \frac{1}{(1 - .8) + .8/S} = \frac{1}{.2 + .8/s}$$

Hence multiplication should be 16 times faster to get an overall speedup of 4.

Solving for S gives S= 16

Machine = CPU

Performance Enhancement Example

- For the previous example with a program running in 100 seconds on a machine with multiply operations responsible for 80 seconds of this time. By how much must the speed of multiplication be improved to make the program five times faster?

$$\text{Desired speedup} = 5 = \frac{100}{\text{Execution Time with enhancement}}$$

→ Execution time with enhancement = 100/5 = 20 seconds

20 seconds = (100 - 80 seconds) + 80 seconds / s

20 seconds = 20 seconds + 80 seconds / s

→ 0 = 80 seconds / s

No amount of multiplication speed improvement can achieve this.

Extending Amdahl's Law To Multiple Enhancements

n enhancements each affecting a different portion of execution time

- **Suppose that enhancement E_i accelerates a fraction F_i of the original execution time by a factor S_i and the remainder of the time is unaffected then:**

i = 1, 2, …. n

$$Speedup = \frac{\cancel{\text{Original Execution}}\ \text{Time}}{\left(\left(1-\sum_i F_i\right)+\sum_i \frac{F_i}{S_i}\right) X \text{Original}\ \cancel{\text{Execution Time}}}$$

Unaffected fraction

$$Speedup = \frac{1}{\left(\left(1-\sum_i F_i\right)+\sum_i \frac{F_i}{S_i}\right)}$$

What if the fractions given are after the enhancements were applied? How would you solve the problem? (i.e find expression for speedup)

Note: All fractions F_i refer to original execution time before the enhancements are applied.

Amdahl's Law With Multiple Enhancements: Example

- Three CPU performance enhancements are proposed with the following speedups and percentage of the code execution time affected:

  $\text{Speedup}_1 = S_1 = 10 \qquad \text{Percentage}_1 = F_1 = 20\%$
  $\text{Speedup}_2 = S_2 = 15 \qquad \text{Percentage}_1 = F_2 = 15\%$
  $\text{Speedup}_3 = S_3 = 30 \qquad \text{Percentage}_1 = F_3 = 10\%$

- While all three enhancements are in place in the new design, each enhancement affects a different portion of the code and only one enhancement can be used at a time.
- What is the resulting overall speedup?

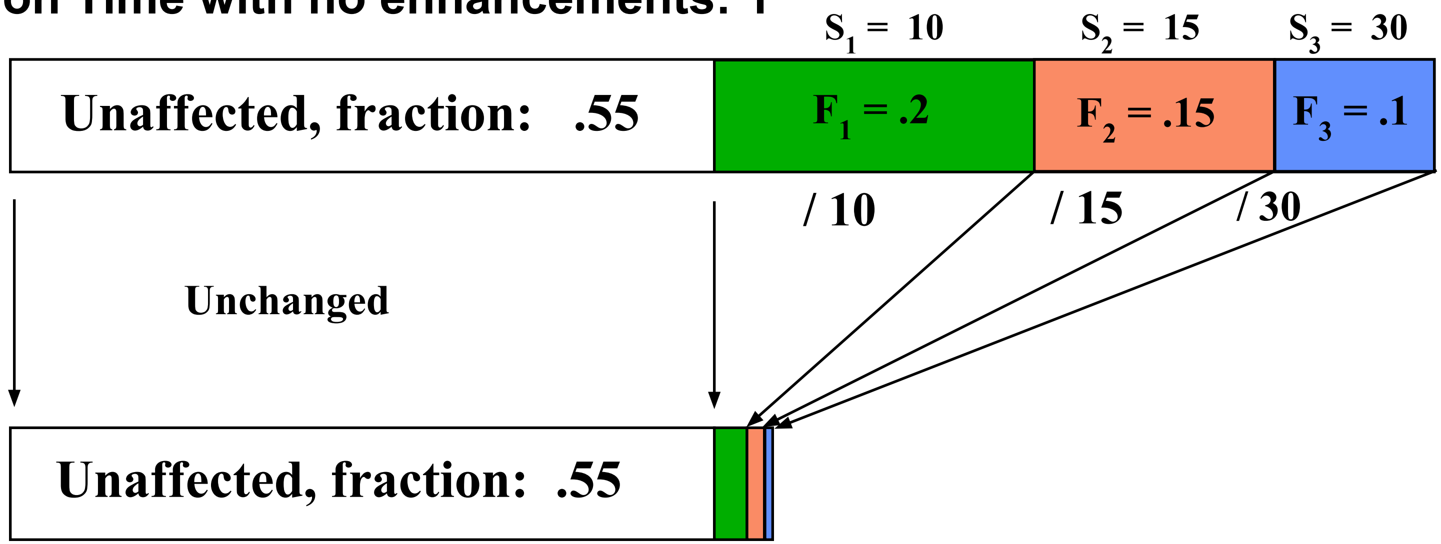
$$Speedup = \frac{1}{\left((1 - \sum_i F_i) + \sum_i \frac{F_i}{S_i}\right)}$$

- Speedup = 1 / [(1 - .2 - .15 - .1) + .2/10 + .15/15 + .1/30)]
  = 1 / [.55 + .0333]
  = 1 / .5833 = 1.71

Pictorial Depiction of Example

i.e normalized to 1

Before:
Execution Time with no enhancements: 1



After:
Execution Time with enhancements: .55 + .02 + .01 + .00333 = .5833

Speedup = 1 / .5833 = 1.71

What if the fractions given are after the enhancements were applied? How would you solve the problem?

Note: All fractions F_i refer to original execution time.

- http://www.eecs.harvard.edu/cs146-246/