Amdahl's Law Calculator

Table of contents:

Our Amdahl's law calculator helps you calculate a task's **theoretical speedup** when a fraction of it is sped up. It will give you the insight you need to decide whether **expanding the resources to speedup** a particular fraction of the task is worth the cost. You can also assess the factor by which a part of the task has to be faster to improve the speed of the task.

Are you curious how it can do such wonders? Stick around, and we'll explore the simple yet elegant formula for Amdahl's law and some examples, understand its implications, and how far we can rely on its speedup formula.

What is Amdahl's law and its formula?

Amdahl's law is an observation on the improvement in the execution time of a task by speeding up a particular portion of a task. We formulate it as a set of two equations:

$$T_{new} = (1 - p)T_{old} + \frac{p}{s}T_{old}$$

$$S = \frac{T_{old}}{T_{new}} = \frac{1}{1 - p + \frac{p}{s}}$$

where:

- T_{new} – Execution time of the overall task *after* improvements have been made to some part of the task;
- p – Proportion of the task that has been improved, sped up, or parallelized (sometimes denoted by α);
- T_{old} – Execution time of the overall task *prior* to any improvements;
- s – Factor by which p proportion of the task has been improved or sped up (sometimes denoted by k); and
- S – Theoretical speedup of the overall task (sometimes denoted by $S_{latency}$).

Often in computing, this *speeding up* involves parallelizing a portion of the task by using multiple processors or threads. With that in mind, we call the part of the task that will benefit from parallel execution/processing as the **parallelizable** part, while the part that cannot or doesn't benefit from parallel execution/processing is the **non-parallelizable** or **sequential** part.

🔎 In this Amdahl's law calculator, the formula is expressed in terms of p, the proportion of **parallelizable task**, which gives $(1 - p)$ as the proportion of **sequential task**, according to the general usage. Seldom, authors may prefer to

express Amdahl's law formula the other way around, that is, in terms of b, the proportion of **sequential task**, thereby $(1 - b)$ representing the proportion of **parallelizable task**.

Number of threads/processors: Sometimes, the speedup factor s is associated with the number of threads or processors used for parallel processing the parallelizable part. While this may be preferable, this is not realistic if you're using processors with unequal processing power.

For example, you can achieve a speedup by a factor of 2 with one additional processor of the same capacity as the original processor or with two additional processors of half the power of the original processor each. Hence we leave it up to you to determine the number of processors required to achieve the speedup factor s.

Examples of Amdahl's law and its implications

After learning what Amdahl's law is, let's take a look at some examples. If a task takes **2 min** to complete, and you can **speed up 40%** of the task up by **4 times**, then $T_{old} = 2 \text{ mins} = 120 \text{ s}$, $p = 0.4$, and $s = 4$. If we put it into the speedup formula, we get:

$$S = \frac{1}{1 - 0.4 + \frac{0.4}{4}}$$

$$S \approx 1.43$$

and...

$$T_{new} = \frac{T_{old}}{S} = \frac{120}{1.43}$$

$$T_{new} = 83.9 \text{ s}$$

Although 40% of the task is four times faster, the task **hasn't received a significant boost** in execution time. The new overall time is 83.9 s which is 30% faster than the original. Our percentage decrease calculator can help you calculate the time reduction in percentage.

Also, note that even if unlimited resources are available to boost the parallelizable part of the task, $s \to \infty$, then the Amdahl's law speedup would be:

$$\lim_{s \to \infty} S = \frac{1}{1 - p}$$

This equation shows a **limit to how much speedup** we can hope to achieve **by improving the parallelizable fraction** of the task. Ironically, that is **dependent on the proportion of the task we have *not* improved**, $1 - p$. You can estimate this quantity S_{max} in this Amdahl's law calculator's `advanced mode`.

In our previous example, the maximum speedup we could achieve with infinite resources to boost 0.4 fraction of the task would be $= \frac{1}{1-0.4} \approx 1.67$, which is not that different from what we achieved by boosting this part of the task by a factor of four!

Consider another task with an overall execution time of **10 hours**. If we boost **60%** of the task with expanded resources by a **factor of 5**, then we would get:

$$S = \frac{1}{1 - 0.6 + \frac{0.6}{5}}$$

$$S \approx 1.92$$

and...

$$T_{new} = \frac{T_{old}}{S} = \frac{9}{1.92}$$

$$T_{new} \approx 4.7 \text{ hours}$$

A speedup of 1.92 times is not significant compared to the substantial speedup we've provided to a significant percentage of the task. This is an observation of Amdahl's law you should keep in mind:

> **Only when a substantial portion of the task is improved do we get a significant reduction in overall execution time**.

Multiple parallelizable subtasks

Sometimes, the task at hand may contain multiple parallelizable subtasks, which can be sped up at different rates. In that case, Amdahl's law speedup is given by:

$$S = \frac{1}{1 - \sum p + \sum \frac{p}{s}}$$

For example, if **25%** of the task is sped up **5 times**, **10%** of the task is sped up **8 times**, and **40%** of the task is sped up **3 times**, then we would get:

$$S = \frac{1}{1 - 0.75 + \frac{0.25}{5} + \frac{0.1}{8} + \frac{0.4}{3}}$$
$$S \approx 2.24$$

In this calculator, you can enter up to 5 parallelizable subtasks and their speedup factors to compute the task's speedup.

Optimization of the sequential part

The crucial role, or rather, the hindering role of the **sequential part** in reducing the overall execution time, is evident from the abovementioned examples and implications. One way to reduce the execution time of the sequential part is by **optimizing** its code to do the job faster. Amdahl's law would then become:

$$T_{new} = (1 - p)\frac{T_{old}}{O} + \frac{p}{s}T_{old}$$

$$S = \frac{T_{old}}{T_{new}} = \frac{1}{\frac{1-p}{O} + \frac{p}{s}}$$

where:

- O **– Optimization factor –** Factor by which the execution of the sequential part has improved.

You can calculate this Amdahl's law speedup in our calculator's `advanced mode`.

> 💡 Are you interested in tools that help optimize time? Try our data transfer calculator and download time calculator!
> They will help you estimate the time needed to upload or download any file.

Applications and limitations of Amdahl's law

Given the simplicity in its formulation, it is remarkable how applicable Amdahl's law truly is. In essence, it can answer three crucial questions regarding the improvement of system performance:

- **Am I focusing on the right portion of the task?** Amdahl's law speedup formula helps you figure out which part of the task should perform faster to improve the overall task's performance effectively.
- **Am I devoting enough resources?** It helps you predict whether you have an opportunity to economically improve the performance of the overall system by speeding up the correct portion of the task.
- **Am I wasting resources?** It also shows whether speeding up the chosen subtask has any additional improvement in the system performance. Sometimes the performance increase can stem from the optimization of the sequential part!

Although it is most useful in the computation field to design faster programs, the insights offered by Amdahl's law are also valuable for someone trying to reduce lead time or improve performance in a particular field.

That said, there are caveats to Amdahl's law:

- **Does not reflect actual gains.** The theoretical speedup calculated from Amdahl's law may not reflect the actual speedup measured in practice. Other factors such as memory speed, cache memory, network cards, etc., have a role in the smooth execution of the task. So it is good to use this law to explore any opportunities for improvement and optimization, but we must measure actual speedup separately.
- **Amdahl's law assumes that the workload is fixed**. In practice, expanded resources may need to handle more work. This increase in workload is usually in the parallelizable part of the task, as befitting the expanded resources. You can use **Gustafson's law** in such cases for a more realistic appraisal of the increase in parallel processing workload. It also finds the speedup in the task's execution time using a different method.

Krishna Nelaturu

People also viewed...

16:10 aspect ratio

This convenient 16:10 aspect ratio calculator allows you to resize your files to fit the 16:10 aspect ratio.

16:10 Aspect Ratio Calculator →

KD (kill-to-death ratio)

The KD calculator computes your kill/death/assist ratio for you.

KD Calculator (Kill/Death Ratio) →

Lost socks

Socks Loss Index estimates the chance of losing a sock in the laundry.

Schwarzschild radius

Calculate the gravitational acceleration at the event horizon of a black hole of a given mass using the Schwarzschild