Throughput vs Latency

This lessons discusses throughput and latency in the context of concurrent systems.

Throughput

Throughput is defined as the *rate of doing work* or how much work gets done per unit of time. If you are an Instagram user, you could define throughput as the number of images your phone or browser downloads per unit of time.

Latency

Latency is defined as the *time required to complete a task or produce a result*. Latency is also referred to as *response time*. The time it takes for a web browser to download Instagram images from the internet is the latency for downloading the images.

Throughput vs Latency

The two terms are more frequently used when describing networking links and have more precise meanings in that domain. In the context of concurrency, throughput can be thought of as time taken to execute a program or computation. For instance, imagine a program that is given hundreds of files containing integers and asked to sum up all the numbers. Since addition is commutative each file can be worked on in parallel. In a single-threaded environment, each file will be sequentially processed but in a concurrent system, several threads can work in parallel on distinct files. Of course, there will be some overhead to manage the state including already processed files. However, such a program will complete the task much faster than a single thread. The performance difference will become more and more apparent as the number of input files increases. The throughput in this example can be defined as the number of files processed by the program in a minute. And latency can be defined as the total time taken to completely process all the files. As you observe in a multithreaded implementation throughput will go up and latency will go down. More work gets done in less amount of time. In general, the two have an inverse relationship.