Performance Optimization - Cheat Sheet

Amdahl's Law

https://en.wikipedia.org/wiki/Amdahl%27s_law

$$S_{\text{latency}} = \frac{1}{1 - p + \frac{p}{s}}$$

Where S_{latency} is the theoretical speedup, s is the speedup of the part with improved system resources and p is the percentage of execution time that benefits from the improved resources.

Describes the theoretical speedup in latency of a task with a fixed workload when the system resources are improved. Usually used to describe speedup if a task is run on multiple processors.

One can use Amdahl's law to see if an upgrade is worth it or not.

Example

By exchanging the disks in a server, one hopes to improve the overall performance. 30% of the system's performance relies on the disks. The new disks are twice as fast as the previous disks.

$$p = 0.3, s = 2$$
$$S_{latency} = \frac{1}{1 - 0.3 + \frac{0.3}{2}} \approx 1.18$$

Amdahl's law states that the latency of the system will be improved by 18%.

Types of workloads

Online Transaction Processing (OLTP)

This tasks revolve around interactive user activities. Generally OLTP includes measurements of a user's *think time*. The goal is usually to have a small response time.

The performance metric for OLTP is defined as the supported users N:

$$N_{\text{concurrent}} = N_{\text{active}} \left(\frac{\text{Response time}}{\text{Response time} + \text{think time}} \right)$$

To calculate the supported concurrent users one can use *real* workload (collected online) or *synthetic* workload. It is composed based on hypothetical use cases - usually described by stochastic processes.

Batch jobs

Specifies the amount of work to be processed without the necessity of intervention (non-interactive).

The performance metric for batch jobs is *throughput* - the number of tasks completed within a given time period.

Calculating the throughput of batch jobs is relatively easy - assume that the system is always at maximum load.

Queuing theory (QT)

Queuing theory is generic for computing and optimizing the efficiency of any system that achieves its objectives by consuming multiple resources. In customer call centers, computer systems etc. there's a limited amount of resources that should be optimally used to achieve maximum possible efficiency.

Provides a way to unify all metrics with quantative performance laws.

A metaphor - visiting a banking center.

- **Server**: banking center fulfilling the customer's service requests
- **Customer**: the initiator of service requests
- **Wait time**: the time duration a customer has to spend in line
- **Service time:** the time duration from when the teller starts the service a customer to when the customer is leaving the teller and the next customer is called in for a service
- **Arrival rate**: the rate at which customers arrive for service
- **Service rate**: the rate at which customers are serviced
- **Utilization**: the portion of a teller's time actually serving customers rather than idling
- **Queue length**: the total number of customer waiting or being serviced
- **Response time**: how long customers are in the system overall, on average. Usually called *latency*. Equal to wait time + service time
- **Residence time**: same as response time but when the teller handles multiple transactions per client
- **Throughput**: the number of users served per second
- **Demand**: number of visits $*$ service time

RAID

RAID is a technology to set up an array of disks for different purposes. For example, RAID 1 uses one or more mirrored disks to achieve reduncancy. RAID 0 uses two or more disks and write across them, a method sometimes called *striping*. RAID 5 uses at least three disks, two for RAID 0 and one for parity. RAID 5 is able to recover errors that may occur in one of the other disks.

Memory Hierarchy

A computer has the following memory types:

1. Processor registers - the fastest type of memory around, very small
2. Cache Level 0-4. Ranges from a few KB with hundres of GB/s access to over 128 MB with tens

of GB/s
3. Primary memory. Several GB, at least 10 GB/s access
4. Secondary memory. Multiple TB, hundres of MB/s to some GB/s

Server Oriented Architecture (SOA)

A reusable atomic service that does something indepentently or in a federated fashion - programming is about using services.

The user doesn't care about the *black box* - it sees the service.

Web Service Description Language (WSDL) is an XML file that describes the web service.

Writing a new application is not coding from scratch - it takes a whole bunch of ready-made "black-box" services.

Optimization and tuning

Optimization is identifying and eliminating internal inefficient designs and implementations (code).

Tuning is establishing the optimal setting for every possbile external configuration parameter (configuration).

Little's law

Throughput, response time and queuing length completely quantifies the performance of a queuing system.

$$N_i = X_i R_i$$

N is the number of customers waiting or receiving service, X_i is the throughput and R_i the response time.

Hyper-threading

Dual processor systems work with seperated sockets (physical processors).

Hyper-threading processors have shared ALUs and cache. The only seperated parts are the logical processors and registers.

There is a need to synchronize logical CPUs. Single-threaded applications can actually slow down single-threaded applications. It benefits multi-threaded applications.

Not the same as two physical processors.

The effectiveness of hyper-threading depends on how busy the SUT without hyper-threading is under the indended load.

The OS sees a physical core and a logical core (from hyper-threading) appears as the same unit.

Multicore

Multicore is a lot more powerful since a dual-core processor is closer to two processors than a single-core hyper-threading processor.

These cores share nothing above the L2 cache. Both single and multi-core applications can benefit from multicore.

Increases front-side-bus speed (CPU and memory controller) and size of L2/L3 cache.

More cores will help for CPU-bound workloads and when computation can be done in parallel.