Dimensioning

CGS - Hugo Miranda

2021

Ciências ULisboa
Faculdade de Ciências da Universidade de Lisboa

Motivation

- Understand how to answer question such as:
  - Why is (will be) the system slow?
    - What is (are) the bottleneck(s)
  - Will the system handle the extra load?
    - How many replicas should I buy?
  - How does load flow from one component to the other?
  - What load to expect?

Steps for Understanding Performance

1. Make a model
   - Divide into components
2. Populate with performance metrics
   - Estimates
   - Observed values
   - Hardware/software vendors
3. Analyze the model
   - Simulation
   - Mathematical analysis
4. Tune the model, restart

Workload

Workload The amount of work per unit of time encountered by the component

- Unit: work/time (packets/s)
- Not necessarily the same for different components
  - 1 HTTP request
    - Multiple IP packets at the firewall
    - Even more frames (e.g. ARP)

Capacity

Capacity The maximum possible throughput of the component

- Unit: work/time

Response Time

Response Time The amount of time it takes for the component to respond to a request

- Unit: time
- Response time is not constant
  - Depends on the request
  - Cached web page vs DB query

Requests Outcome

Loss Rate The fraction of requests that do not receive a response or receive an erroneous response

- Unit: %

Throughput (goodput) The amount of work per unit of time that receives a normal response through the component.

- Unit: work/time

Utilization

Utilization The fraction of time the component is busy

- Unit: %
- Rule of thumb: aim to 30%

Lets Forget Computers

https://www.youtube.com/watch?v=MVm1KcrHM6s
Excerpt of "The Soup Nazi" (Seinfeld, S07E06, 1995)

Our Restaurant Model

(Minutes as the unit of time)

Operations table

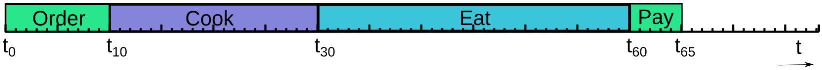
Order	Operation	Time(m)	Component
1	Take order	10	Waiter
2	Cook meal	20	Chef
3	Eat	30	Table
4	Pay	5	Waiter
		65	

Our Restaurant Model

Resources

Component	Instances
Waiter	3
Chef	5
Table	20

Graphical Representation



Metrics Applied

Assume we have 10 clients/h

- Q: What is the throughput of the system?
  - A: If everything goes well, $10 clients/h = \frac{1}{6}$ clients/m
- Q: Is the workload equal for all components?
  - A: Throguhput is the same for all components ($\frac{1}{6} clients/m$) but components have a different number of instances
- Q: What is the response time?
  - A: See the table above:
    - For a meal: 65m
    - For a Chef: 20m

Metrics Applied

Is the system properly sized? (or can it handle the workload? or Is the utilization < 1?)

Utilization Law

The average utilization of any component in the computer system is the throughput of the system multiplied by the amount of service time each request requires at that component

Utilization Law In Practice

Component	Equation	Utilization
Waiter	$\frac{1}{6} \times \frac{10+5}{3}$	.83
Chef	$\frac{1}{6} \times \frac{20}{5}$	.66
Table	$\frac{1}{6} \times \frac{30}{20}$	.25

- Notes:
  - $\frac{1}{6} = \text{throughput}$
  - 10+5->order+pay
  - second factor always divided by the number of replicas

Utilization Law In Practice

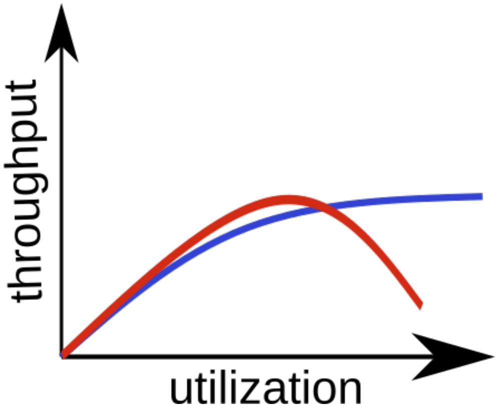
How to alleviate Waiters?

$$U = throughput \times \frac{time}{replicas}$$

- Lower the throughput
  - I.e. admit less clients
- Lower the time
  - Handle clients faster
- Increase the replicas
  - Hire more waiters

Utilization vs Capacity

What happens when Utilization $\approx$ Capacity?



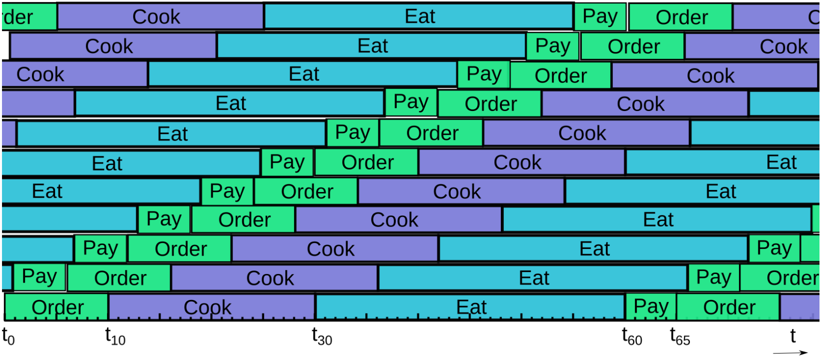
Metrics Applied

How many clients are, on each moment, on the restaurant?

Little's Law

The average number of pending requests in any computer system equals the average rate of arrival of requests multiplied by the average time spent by the request in the system.

Graphical Representation



Little's Law In Practice

- On average, how many clients are inside the restaurant?

$$c = \mathit{throughput} \times \mathit{response} = \frac{1}{6} \times 65 = 10.8$$

- What would have to be the throughput to get 15 clients inside the restaurant?

$$15 = t \times 65 \Rightarrow t = .23$$

- Homework: what about utilization?

Metrics Applied

How many clients is each component handling?

Forced Flow Law

The throughput through different components of a system is proportional to the number of times that component needs to handle each request.

Or

If the throughput of a component x is T_x and the throughput of the system S is T_s, each request visits $\frac{T_x}{T_y}$ the component.

Forced Flow Law in Practice

Component	Clients/m
Waiter	$\frac{1}{6} \times \frac{2}{3} = \frac{1}{9}$
Chef	$\frac{1}{6} \times \frac{1}{5} = \frac{1}{30}$
Table	$\frac{1}{6} \times \frac{1}{20} = \frac{1}{120}$

- Notes:
  - Throughput: $\frac{1}{6}$
  - $\frac{2}{3} \Rightarrow$ (Order + Pay) divided by 3 waiters
  - Each waiter sees a new client on average every 9m

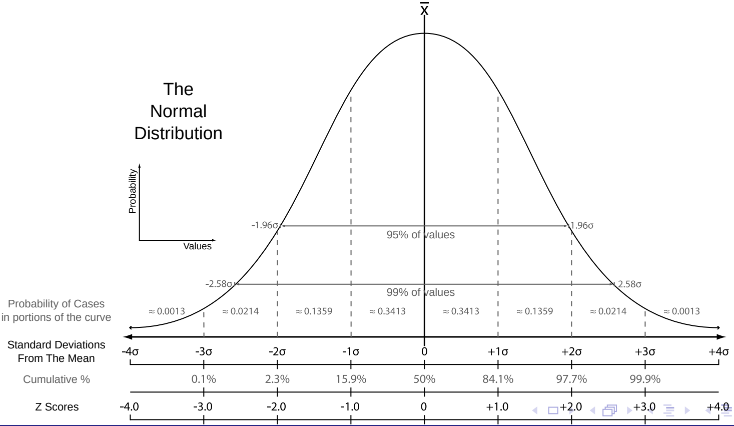
Oversizing

Is the restaurant well sized?

The $3 - \sigma$ rule

Most of the values in any distribution with a mean of m and standard deviation of σ lie within the range $[m - 3\sigma, m + 3\sigma]$

The Normal Distribution



The $3 - \sigma$ rule in practice

- How to avoid lines in front of my restaurant?
  1. Map the arrivals into a distribution
  2. Size the resources appropriately
- What if standard deviation is 4?
  - Expect 22 clients/h. Recalculate utilization

Waiting Queues Theory

- So far clients were "well behaved"
    - Arrived at well-known and properly defined intervals
- What if they don't?

Waiting Queues

- Requests don't arrive equally spaced
  - Instead they arrive with some probability p within time interval t
- Popular distributions
  - Deterministic
  - General
  - Poisson

Poisson Distribution

$$f(n, \lambda) = \frac{\lambda^n e^{-\lambda}}{n!}$$

- Probability of receiving n requests when the expected are λ
- The probability of not receiving a request on time interval t is given by $e^{-\lambda * t}$
- Probability of getting a request is independent of the remaining requests

Waiting Queue Notation

- A/B/C
- Where

  - **A** entry distribution
  - **B** service distribution
  - **C** number of replicas

- Typical distributions

  - **M** Poisson
  - **D** Deterministic
  - **G** General

Applications

With λ the parameter of the first distribution and μ the parameter of the second

M/M/1

Arrival Rate λ

Service rate μ

Utilization $\rho = \frac{\lambda}{\mu}$

- If $\rho < 1$

  Average Delay $\frac{1}{\mu(1-\rho)}$

  Pending Requests $\frac{\rho}{1-\rho}$

Applications

With λ the parameter of the first distribution and μ the parameter of the second

M/D/1

Arrival rate λ

Service rate μ mas com μ constante e fixo

Utilization $\rho = \frac{\lambda}{\mu}$

If $\rho < 1$

Average Delay $\frac{1}{2(\mu - \lambda)} + \frac{1}{2\mu}$

Pending Requests $1 + \frac{\rho^2}{2(1-\rho)}$

A few things on resilience

- Buzzwords
- Understanding/estimating resilience

Buzzwords related with resilience

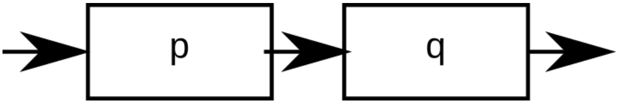
MTBF Mean Time Between Failures. The predicted time between failures of some hardware component

N 9's The proportion of time the system is up (when needed)

- 99.999% (5 nines): 5.26m unavailable per year
- 99.99% (4 nines): 52m unavailable per year

Serial (alt A)

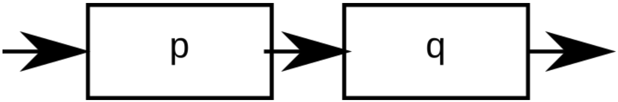
Let P_p and P_q the probability of two components (respect.) p and q be correct



The system will be correct if p AND q are both correct.
Numerically:
$P_p \times P_q$

Serial (alt B)

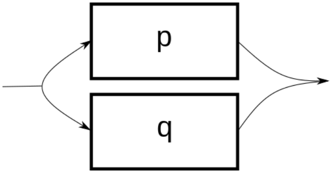
Let P_p and P_q the probability of two components (respect.) p and q be incorrect



The system will be correct if p AND q are both correct.
Numerically:
$(1 - P_p) \times (1 - P_q)$

Parallel (alt A)

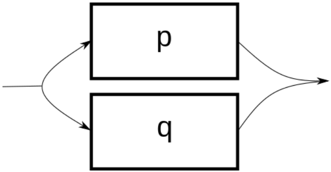
Let P_p and P_q the probability of two components (respect.) p and q be correct



The system will be correct if (p AND q are correct) OR (p is correct AND q not) OR (p is not correct AND q is correct) $\rightarrow$ $(P_p \times P_q) + [P_p \times (1 - P_q)] + [(1 - P_p) \times P_q]$

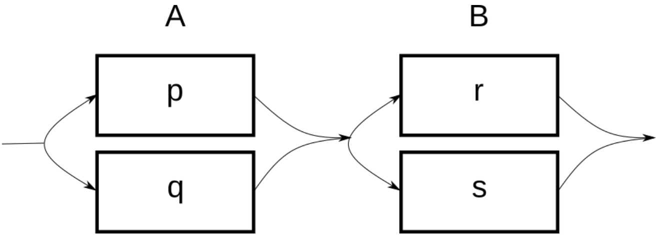
Parallel (alt B)

Let P_p and P_q the probability of two components (respect.) p and q be correct



The system will be incorrect when p AND q are incorrect $\rightarrow$ $(1 - P_p) \times (1 - P_q)$. It will be correct otherwise $\rightarrow$ $1 - [(1 - P_p) \times (1 - P_q)]$

Blocks



Handle each block in separate. Compose blocks in serial.

Wrap Up

- Learn the metrics
  - Throughput
  - Utilization
  - Response time
- Understand the dependencies between them
- Be able to justify that parallel is more resilient than serial