Open Street Map vs Google maps

Pros of Open source:

The main difference between these two services is that every edit you make to OSM is owned by you and the community

More granuality and quality of maps in especially in less developed areas

Free

Cons of Open Source:

Not enough tutorials and documentation not user friendly – define user friendly!

Pros of GMaps:

Backed by a huge company

Good documentation and easy to follow tutorial

Cons of GMap:

everychange you make to Google Maps will be owned by Google.

http://geoawesomeness.com/google-bans-community-map-edit-after-urinating-robot-prank/

Talk about potential designs of the sensor so one sensor maybe be able to detect 5 bays.

<http://www.dailymail.co.uk/news/article-2071527/Parking-space-London-costs-96-000-13-000-average-house-Middlesbrough.html>

<https://en.wikipedia.org/wiki/Traffic_engineering_(transportation)>

<https://en.wikipedia.org/wiki/Queueing_theory>

<http://waset.org/publications/9999252/the-application-of-the-queuing-theory-in-the-traffic-flow-of-intersection>

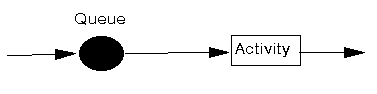
<https://www.researchgate.net/post/Can_I_use_queueing_theory_to_optimize_parking_area>

# Queueing theory

Queuing theory deals with problems which involve queuing (or waiting). Typical examples might be:

* banks/supermarkets - waiting for service
* computers - waiting for a response
* failure situations - waiting for a failure to occur e.g. in a piece of machinery
* public transport - waiting for a train or a bus

In essence all queuing systems can be broken down into individual sub-systems consisting of *entities* queuing for some *activity* (as shown below).



Typically we can talk of this individual sub-system as dealing with ***customers*** queuing for ***service***. To analyse this sub-system we need information relating to:

* **arrival process:**
  + how customers arrive e.g. singly or in groups (batch or bulk arrivals)
  + how the arrivals are distributed in time (e.g. what is the probability distribution of time between successive arrivals (the ***interarrival time distribution***))
  + whether there is a finite population of customers or (effectively) an infinite number

The simplest arrival process is one where we have completely regular arrivals (i.e. the same constant time interval between successive arrivals). A Poisson stream of arrivals corresponds to arrivals at random. In a Poisson stream successive customers arrive after intervals which independently are exponentially distributed. The Poisson stream is important as it is a convenient mathematical model of many real life queuing systems and is described by a single parameter - the average arrival rate. Other important arrival processes are scheduled arrivals; batch arrivals; and time dependent arrival rates (i.e. the arrival rate varies according to the time of day).

* **service mechanism:**
  + a description of the resources needed for service to begin
  + how long the service will take (the ***service time distribution***)
  + the number of servers available
  + whether the servers are in series (each server has a separate queue) or in parallel (one queue for all servers)
  + whether preemption is allowed (a server can stop processing a customer to deal with another "emergency" customer)

Assuming that the service times for customers are independent and do not depend upon the arrival process is common. Another common assumption about service times is that they are exponentially distributed.

* **queue characteristics:**
  + how, from the set of customers waiting for service, do we choose the one to be served next (e.g. FIFO (first-in first-out) - also known as FCFS (first-come first served); LIFO (last-in first-out); randomly) (this is often called the *queue discipline*)
  + do we have:
    - balking (customers deciding not to join the queue if it is too long)
    - reneging (customers leave the queue if they have waited too long for service)
    - jockeying (customers switch between queues if they think they will get served faster by so doing)
    - a queue of finite capacity or (effectively) of infinite capacity

The Poisson process is one of the most widely-used counting processes. It is usually used in scenarios where we are counting the occurrences of certain events that appear to happen at a certain rate, but completely at random (without a certain structure). For example, suppose that from historical data, we know that earthquakes occur in a certain area with a rate of 22 per month. Other than this information, the timings of earthquakes seem to be completely random.

https://ace-ebert.shinyapps.io/queue\_simulator\_mmk/

<https://pdfs.semanticscholar.org/848f/a1f48ad9d3edb24b05667f15cfc633eb8f69.pdf> -> page 12: It is quite rare, except for elementary or Markovian systems, that the distributions can be computed. Usually their mean or transforms can be calculated – Use this in car parks?

<https://pdfs.semanticscholar.org/848f/a1f48ad9d3edb24b05667f15cfc633eb8f69.pdf> -> page 13/14 – equations and notation systems

<https://pdfs.semanticscholar.org/848f/a1f48ad9d3edb24b05667f15cfc633eb8f69.pdf->> page 17 - An M/M/1 queueing system is the simplest non-trivial queue where the requests arrive according to a Poisson process with rate λ, that is the interarrival times are independent, exponentially distributed random variables with parameter λ. The service times are also assumed to be independent and exponentially distributed with parameter µ. Furthermore, all the involved random variables are supposed to be independent of each other

<http://www.csus.edu/indiv/b/blakeh/mgmt/documents/opm101supplc.pdf> -> page C4, reference baulking(?)

Data of machine learning can enhance the availability of car parks by applying the data and using it in queueing theory (<https://pdfs.semanticscholar.org/848f/a1f48ad9d3edb24b05667f15cfc633eb8f69.pdf> -> page 41)

reference email Kathryn

download appyparking and look at their layout

TO ADD:

--- Using the variables for the logistic regression: Peak times in the downloaded report: <https://www.racfoundation.org/wp-content/uploads/2017/11/spaced_out-bates_leibling-jul12.pdf>

--- stochastic gradient decsent

--- Add equations and define the M/M/1 convention used in queueing theory.

--- add traffic engineering to literature review?

--- Breadboard circuit layout diagram

--- Look for more examples in terms of layout ui, finding requirements etc.

--- XLMiner, compare results of machine learning coefficients with other libraries out there etc

--- Add SSL to connection <- feature ot add

--- Calculate average on previous data, alternative to logistic regression

--- Make sure to add reference to code copied

--- Talk about Arduino

---Include small snippets in implementation section

---KNN: The KNN classifier is also a non parametric and instance-based learning algorithm.

Non-parametric means it makes no explicit assumptions about the functional form of h, avoiding the dangers of mismodeling the underlying distribution of the data. For example, suppose our data is highly non-Gaussian but the learning model we choose assumes a Gaussian form. In that case, our algorithm would make extremely poor predictions.

Instance-based learning means that our algorithm doesn’t explicitly learn a model. Instead, it chooses to memorize the training instances which are subsequently used as “knowledge” for the prediction phase. Concretely, this means that only when a query to our database is made (i.e. when we ask it to predict a label given an input), will the algorithm use the training instances to spit out an answer. https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/

-Robo3t

ML:

<http://library-collections-search.westminster.ac.uk/primo_library/libweb/action/search.do;jsessionid=99100513B6005005CF219B5126BA88F3?fn=search&ct=search&initialSearch=true&mode=Basic&tab=local&indx=1&dum=true&srt=rank&vid=WST_VU1&frbg=&tb=t&vl%28freeText0%29=Machine+learning+in+java&scp.scps=scope%3A%28WST_CALM_DS%29%2Cscope%3A%28WST_ML_DS%29%2Cscope%3A%28WST_ALMA_DS%29>

Among the different machine learning approaches, there are three main ways of learning, as shown in the following list:

* Supervised learning
* Unsupervised learning
* Reinforcement learning

Given a set of example inputs, *X*, and their outcomes, *Y*, supervised learning aims to learn a general mapping function, *f*, that transforms inputs to outputs, as *f: X Y -🡪 Chapter 1*

*Supervised Learning definition – Chapter1.6*