Replication ~~and Redundancy~~

~~Definition~~ Reason 1 : Redundancy

Redundancy is a key technique to increase availability. If one server crashes, then there is a replica that can still be used. Thus, failures are tolerated by the use of redundant components.

Physical
E.g. use 3 engines even though you only need 1

Time
Perform an action again, and again, ...

Information
E.g. send extra bits over the network, allow recovery

Probability of availability

$$\begin{aligned} P(\text{service is available}) &= 1 - P(\text{all replicas have failed}) \\ &= 1 - (P(\text{replica 1 has failed}) \\ &\quad \cdot P(\text{replica 2 has failed}) \\ &\quad \cdot P(\text{replica 3 has failed}) \\ &\quad \cdot \text{ and so on }) \end{aligned}$$

Reason 2: $P(\text{replica } x \text{ has failed}) = \frac{\text{mean time to repair failure}}{\text{mean time between failures} + \text{mean time to repair failure}}$

Performance

- By placing a copy of the data closeby the process using them, the time to access the data decreases. This is also useful to achieve scalability.
- One approach to this is caching: web browsers store locally a copy of a previously fetched web page to avoid the latency of fetching the resource again.

Consistency problem

- If a copy (i.e. a replication) is modified, this copy becomes different from the rest. Consequently, modifications have to be carried out on all copies, to ensure consistency.
- When and how these modifications need to be carried out determines the price of replication.