Self-organising P2P

Antonio Bucchiarone

Fondazione Bruno Kessler, Trento – Italy
bucchiarone@fbk.eu

06 November 2019

P2P Paradigm

- P2P for building distributed systems.
- P2P allows the construction of systems with unprecedented *size* and *robustness.*
  - *Decentralisation* and *Redundant* structure.
- For databases the P2P approach offers new possibilities:
  - utilisation of a large number of resources:
  - ***storage space*** or ***processing power*** of peers in the network.
- *Massive scale* and *very high dynamism* makes it impossible to capture and maintain a complete picture of the entire P2P network.
- A peer is only able to maintain a *partial or estimated view* of the system.

Database Scenario

- **Data distribution:** how to partition the data among the peers**.**
- A peer introducing *new data*, or creating a *new replica*, has to decide which of the other peers in the network is *the most suitable to host the data*.
- Distributed Hash Table (DHT) approach assumes that *all peers are similar and have equal capabilities* for maintaining data.
- The distribution of resources among the peers is uniform.
- This is not the case in real-life systems:
  - number of connections,
  - uptime,
  - available bandwidth,
  - storage space,
- usually exhibit the so called *scale-free* or *heavy-tails* properties.

Large-Scale Distributed Storage System

- System's topology and replica placement dynamically adapts to reflect the heterogeneities in the network and peer properties.

- Assumptions:
  - The data is persistent and highly replicated
  - The system keeps track of all replicas so that their owners are able to update or delete them.
  - The replica are placed in the most reliable, high performance peers only.
  - The data is required much more frequently than updated.

- **Self-organising neighbourhood selection algorithm** that
  - Clusters peers with *similar reliability and performance characteristics*
  - Generates a network topology that helps to solve the problem of *dynamic replica placement*.

Peer Reliability Metrics

- To address the persistent data requirements for a distributed system deciding **where to store the data.**

- Two Extremes:
  - To store all data in a *centralised server* (not scalable).
  - To *partition the data among a set of peers* using some indexing scheme (DHT).

- Many existing P2P systems assume that all peers have *identical capabilities and responsibilities*, and the data and *load distribution is uniform* among all nodes.

- **Problem:** the use of peers with lower bandwidth/stability/trust to store data would degrade the performance of the entire network.

Peer Reliability Metrics

- To allow data to be stored on the fastest, highest bandwidth, and most reliable trusted peers, called **superpeers**.

- **Problem:** How *to identify and select the superpeers* from the set of peers in the system - without a global knowledge of the system.

- **Possible Solutions**:
  - *Flooding*: it requires communication with all N nodes in the system.
  - *Hard-wiring* them in the system or *configuring them manually*.

- These solutions **are in conflict** with the assumption of *self-management*, *decentralisation*, and *the lack of a central authority* that controls the structure of the system.

- **Adaptive self-organising system**
  - The peers automatically and dynamically elect superpeers, accordingly to the demand, available resources and other runtime constraints.

Peer Selection

- The selection of peers for replica placement are based on criteria such as:
  - Stability.
  - Available bandwidth and latency.
  - Storage Space.
  - Processing Performance.
  - Open IP address and willingness to share resources.

- **Peer reputation model:** only the most trusted peers might be allowed to host a replica.

- **Peer's reliability**: weighted sum of the above parameters.

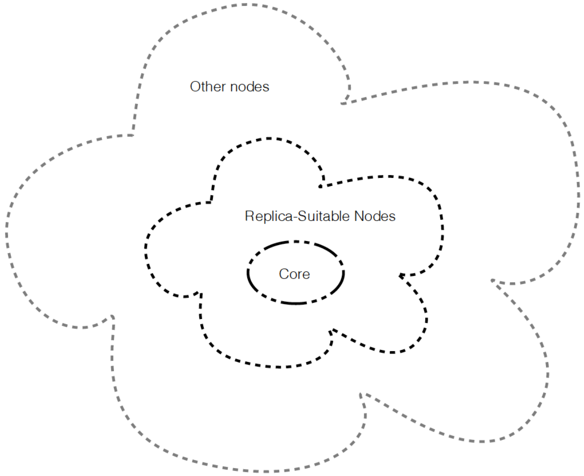
Closed vs Open System

- **Closed System:** where all peers trust each other, it is sufficient that every peer evaluates its own as reliability level.
  - Neighbouring peers can exchange the reliability information without any verification procedure, since trust is assumed.
- **Open, untrusted environment**: the system should be protected against malicious peers providing fake reliability information.
  - The system should be also robust against *cliques* or *greedy* nodes.
- **Persistent data is stored by the most reliable peers**.
- **The system tries to maximize data availability, security and the quality of service by placing data replicas on the most reliable hosts.**

Neighbour Selection Algorithm

FONDAZIONE BRUNO KESSLER

- Unstructured P2P architecture where reliable peers, maintaining persistent data, are highly connected with each other and form a *logical core of the network.*
- The network around the core is composed by *less reliable peers*.
- *Grouping reliable peers* have the following advantages:
  - Searching for reliable peers maintaining replicas, is less expensive.
  - The overhead for replica synchronization is reduced since the replicas are located close to each other.
  - Routes between peers storing data are more stable and up-to-date.
  - Trust evaluation between peers storing data is less expensive.



Replication Strategy

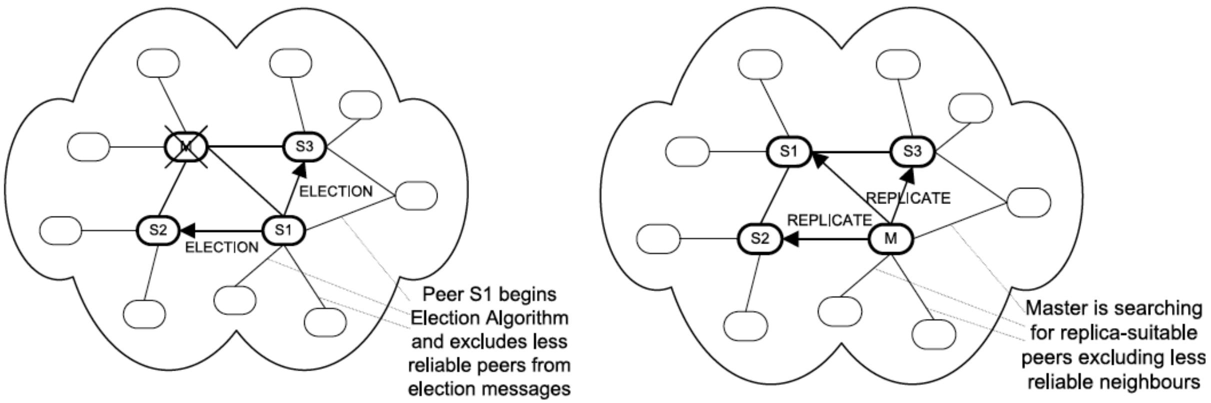
FONDAZIONE BRUNO KESSLER

- Each peer can potentially create an independent database, and replicate it over the P2P network - to improve is availability and persistence guarantees.
- A peer that creates the first copy of a database (*master replica*), becomes the *database owner*.
- Subsequent replicas of the database hosted by other peers are called *slave replicas.*
- The users issue queries to the database that can be resolved by any replica.
- The *owner*, and potentially other authorised users, can also *update* or *delete* a database.
- There is *only one master replica* - responsible for handling and synchronising updates.
- The set of peers that are allowed to create slave replicas are restricted to those with reliability above *replica-suitable threshold*.

Replication Strategy

1. A peer *accepting a slave replica* may require from the peer initiating the placement a certain level of reliability, above a some threshold, which we call the **replica creation threshold.**

2. The master replica may require that the slave replicas are created only by peers located in the replica-suitable core of the network, i.e., ***replica acceptance thresholds*** – *No consensus between peers on the threshold values is required,* since the thresholds can be determined by each peer individually.



(a) Peers S1, S2 and S3 compare their reliability to elect a new master.

(b) Master compares the reliability of peers S1, S2 and S3 to select the best peer for slave replica placement.

Replica Synchronization

- Database replicas must *be synchronised between the master and the slaves* after update operations.
- **Constraint:** the updates are only performed on the master, while queries can be handled by any slave.
- If an **update** is delivered to an ordinary replica, the *replica forwards it* to the master, and the master propagates the update to all replicas.
- **Concurrent updates** from different peers *are serialised* and sent in the same order to all copies of the database (no write-write conflicts).
- The updates can be propagated either *instantaneously*, or in a *lazy fashion*, by *periodic gossiping*.
- The design can be also improved by allowing the replicas to construct a hierarchy, a **spanning tree for spreading the updates**.

Master Election

- Peers have *relative positions* in the topology, defined by their **reliability metric**.

- **Election Algorithm**
  - Peers can use a **heuristic** that excludes peers with *lower reliability*.
  - The heuristic does not guarantee that *the most reliable peer will become master* unless all peers in the core are fully connected.
  - **Gossiping election model**.
  - The election initiating peer sends the election messages to a certain number of neighbouring peers with lower reliability (inside the core).
  - Given high enough connectivity between nodes in the core, within a certain probability **the node with the highest reliability should win the election**.

Replica Discovery

FONDAZIONE BRUNO KESSLER

- A searching mechanism is needed for peers to discover nearby replicas of a DB they request access to.

- **Search in unstructured P2P**: random walk, iterative deepening, routing indices

- **Probabilistic adaptive algorithm** where routing is based on two main factors:
  - Heuristic values learned by the system;
  - Neighbour reliability heuristic to effectively route queries towards the core of the network.

Main Loop of the Simulation in Repast

Algorithm 1: Main loop of the simulation

```
for N steps of the simulation do
    increase the number of peers by 1%;
    probabilistically remove peers according to their reliability;
    forall peers p in the network do
        p.step();
    end
end
```

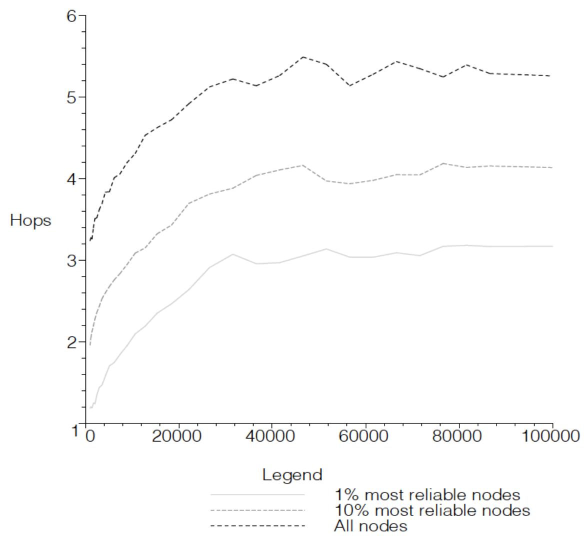
Agent Step in Repast

Algorithm 2: Agent step

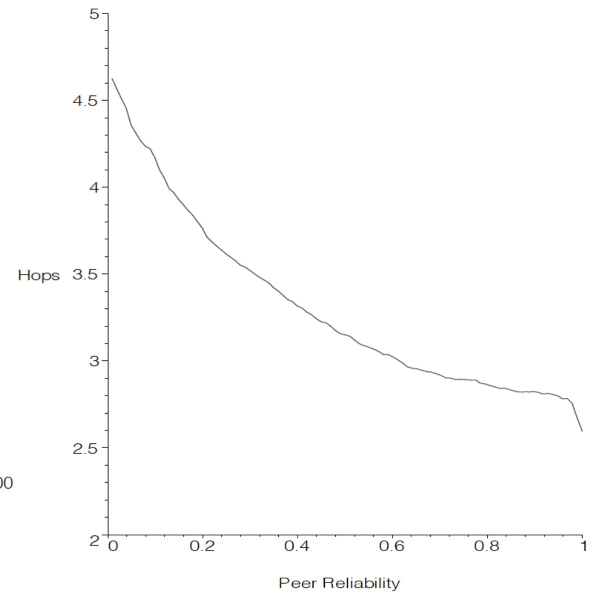
```
if number of neighbours = MAX_NEIGHBOURS then
    disconnect random neighbour;
end
if number of similar neighbours < MAX_SIMILAR then
    choose randomly neighbour p from all known neighbours;
    get all neighbours n1..nk from p;
    choose peer n with the most similar reliability from n1..nk;
    connect to n;
end
if number of random neighbours < MAX_RANDOM then
    choose randomly neighbour p from all known neighbours;
    get all random neighbours n1..nk of p;
    choose randomly peer n from n1..nk;
    connect to n;
end
```

Experimental Results

- The average path length between peers varies with peer reliability.
- The average distance between the most reliable peers is lower than between less reliable peers.
- The most reliable peers are highly connected with each other and form a reliable core of the network.



(a) Average distance between peers as a function of the network size.



(b) Average distance between peers as a function of node reliability, network size 100,000 peers.

Self-organising P2P

Antonio Bucchiarone

Fondazione Bruno Kessler, Trento – Italy
bucchiarone@fbk.eu

06 November 2019