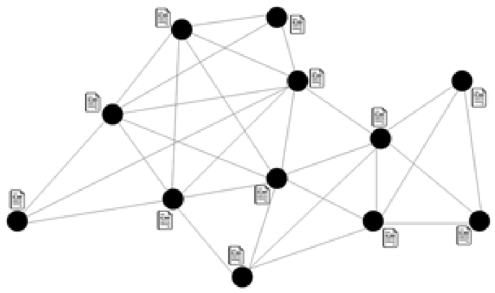
Decentralized state machine using Nakamoto (probabilistic) consensus.

What is a decentralized state machine ?

Goal : have a network of nodes hold a record of numbered logs.



All nodes should hold the same record (**consensus hypothesis**).

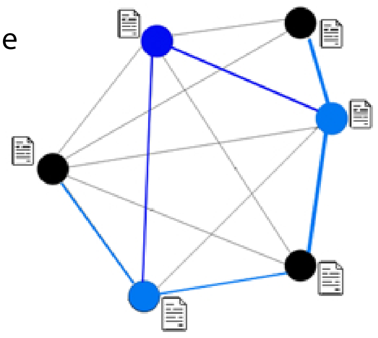
A client can query any node to get the logs, or ask to add a new log.

Fault tolerance : if some nodes fail, the system should still be functionning.

Snowball Consensus Protocol (Nakamoto)

Step 1 : A node receives a query from a client asking to add a log to the record. If it is possible for this node, it accepts the new log, selects a bunch of nodes in the network and sends the same query.

Step 2 : Each node checks is new log is acceptable. If it is, reply ok to the sender and propagate.

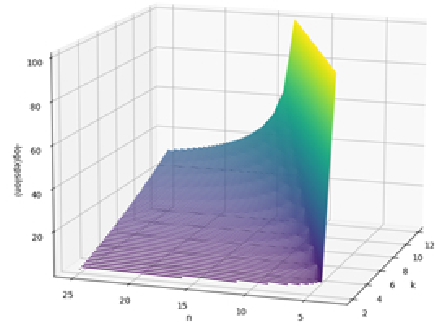


Parameters :
n : number of nodes in the network
k : number of nodes that a node queries each round
β : number of rounds before a nodes stops querying (for one log)

Comparison with classical consensus

Classical consensus was represented with the Raft consensus algorithm

Probability (ε) of not reaching consensus with Snowball (β = 2) **:**



By choosing the right parameters, ε can be made negligible.

In Raft, each leader failure triggers a leader election which can be costly in requests. Normal functionning also involves regular heartbeat requests which are avoided with Snowball.

Snowball allows better byzantine fault tolerance because it is leaderless. In raft, if leader is byzantine (malicious), the whole system might fall.

Author : Nicolas Couraud
Contact : nicolas.couraud@etu.emse.fr