Large Scale Computing

Lab 5 - CAP Theory

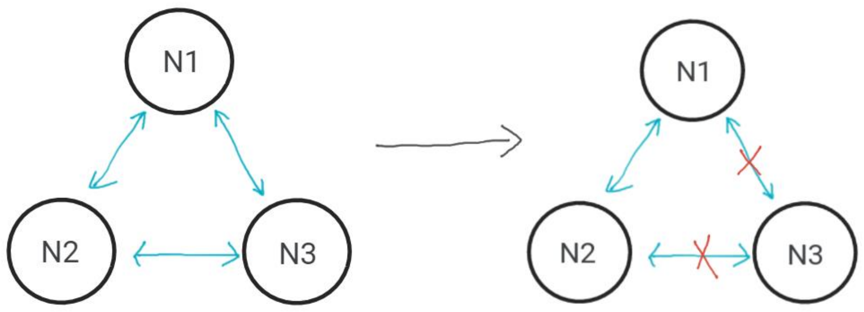
Author: Adrian Żerebiec

Assignment 1

Task: (3p) Answer the questions found above in the description of Exercises 1,2,3.

Question from Exercise 1

Explain if and why you were/were not able to get/increase the value of AtomicLong in each step on particular nodes.



Answer

1. **Before network partition**

All nodes can see AtomicLong value, and they can update value because they are connected with each other. Any increment of AtomicLong value is immediately visible on all nodes.

2. **During Network Partition**

Node 1 and Node 2 can read the value of AtomicLong, and they are changing value because between them the quorum exists. Node 3 is isolated from the other two, so it cannot change read and change value of AtomicLong.

3. **After Network Partition Healing**

After partitioning, nodes must synchronize. Latest consistent value left assigned on all nodes, for example,

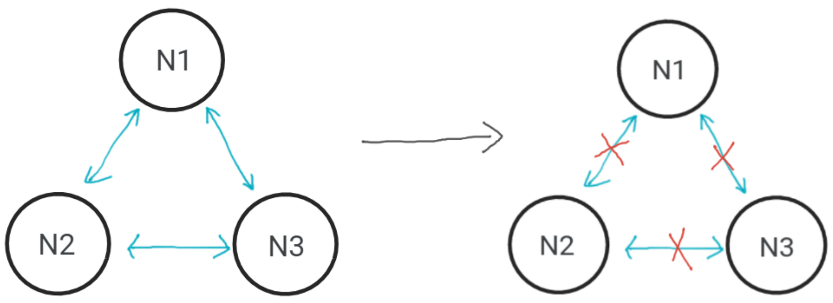
Before partitioning: Node 1 -> 1, Node 2 -> 1, Node 3 -> 1

During Partition: Node 1 -> 3, Node 2 -> 3

After healing: Node 1 -> 3, Node 2 -> 3, Node 3 -> 3.

Question from Exercise 2

Explain if and why you were/were not able to get/increase the value of AtomicLong in each step on particular nodes.



Answer

1. **Before network partition**

All nodes can see AtomicLong value, and they can update value because they are connected with each other. Any increment of AtomicLong value is immediately visible on all nodes.

2. **During Network Partition**

None of the nodes can update and read the AtomicLong Value. This is because the nodes are isolated from each other and there is no way to form a quorum with other nodes.

3. **After Network Partition Healing**

As in the first task, after the repair, all nodes synchronize. They then assume the AtomicLong value from before the partition. For example,

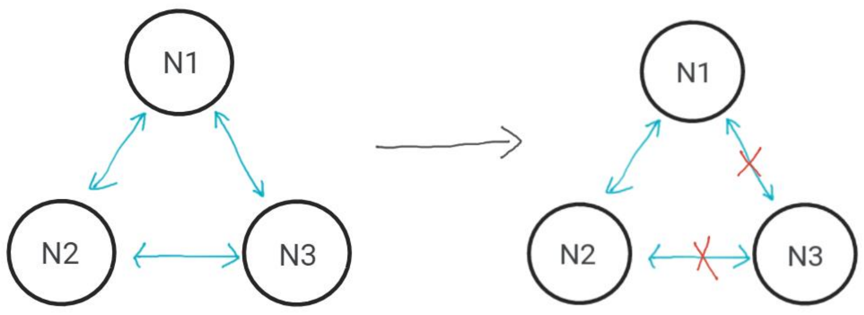
Before partition: Node 1 -> 1, Node 2 -> 1, Node 3 -> 1

During partition: Cannot read any value

After healing: Node 1 -> 1, Node 2 -> 1, Node 3 -> 1

Question from Exercise 3

Explain if and why you were/were not able to get/increase the value of PNCounter in each step on particular nodes.



Answer

In the case of AP, nodes can always update the PNCounter value and read it. When the partition is performed, the values are no longer consistent. Connected nodes update their values (if one increases the value, the other sees it). In the case of an unconnected node, the value remains unchanged. Example:

Before partition: Node 1 -> 1, Node 2 -> 1, Node 3 -> 1

During partition: Node 1 -> 4, Node 2 -> 4, Node 3 -> 2

After healing: Node 1 -> 4, Node 2 -> 4, Node 3 -> 2

Assignment 2

<u>Task: (2p) Which data structures -- AP or CP -- require the Raft algorithm? Why is this algorithm needed?</u>

Answer

CP data structures require the RAFT algorithm. This is a consensus algorithm. It is crucial for maintaining the "C" in CP (consistency). RAFT helps maintain a consistent state of data even if the event on node failures or communication interruptions.

CP systems care most about keeping the same data on all nodes, even if the network has problems. Raft helps with this by choosing a leader and making sure all changes happen in the same order everywhere.

Assignment 3

Task: (2p) Read article *Session Guarantees for Weakly Consistent Replicated Data* and explain what it means that the PN Counter in Hazelcast provides *Read-Your-Writes* (RYW) and *Monotonic reads* guarantees and why they are *session guarantees*.

Answer

PN Counter is a structure with Conflict-free Replicated Data Type. This allows each Hazelcast structure to increment and decrement a value and the changes are reflected on all replicas.

Read-Your-Writes guarantees that updates are visible immediately within the same session. A client cannot read an outdated value, for example from before its update, even if the nodes are not fully synchronized.

Monotonic Reads guarantees that if a client reads a value, then in subsequent reads the client cannot read an outdated or old value.

The session guarantee means that the RYW and MR guarantees provided by PNCounter apply only within the context of a single continuous session with the distributed system. This means that if the system does not detect a replica with the previously presented state, it will throw an appropriate exception informing the application. If the session is terminated, access to previous replicas will be lost.