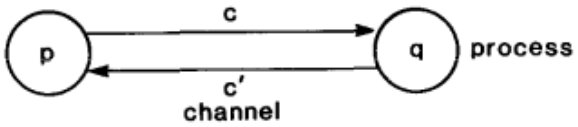
Paper Review

"Distributed Snapshots: Determining Global States of Distributed Systems"

1. Summary

In this paper, the author proposed a model to detect global states of distributed systems, which helped detect service stability by identifying error states such as deadlock and computation termination. The highlight of this paper is the modelization of states, events, and global states.

The algorithm aimed to compute the global state of the distributed system at a discrete-time point. The obvious problem is that there are ambiguous states. For example, process **p** sent a token to process **q** as 1 shown. **c** and **c′** are channels for communication between **p** and **q**, which may have delays and messages need times to reach q.



There are four states:

- Token **in p**
- Token **in c**
- Token **in c′**
- Token **in q**

When p sends a token to q, p knows that token is in channel c, so p thinks the global state is **in c**. However, q may not know that token is in channel c, so q thinks the global state is **in p**. Here's the ambiguity, and it can be impossible for us to detect global state and errors because we do not know where the token is or which part went wrong if the token is missing since we cannot merge two different states.

To solve this problem, the author proposed an idea.

1. P will send a marker when p changed its state (p will change its state to **in-c** in this case). Q will receive a token mixed with the marker. If the marker is received, q will know that the token is q.
2. When q receives marker, it will know the token is **in-q** and will send a marker through channel c′ to p.
3. When q received marker from p and p received the second marker from q, the sending of the token is finished.

Feiran Qin qinfr@shanghaitech.edu.cn 2019533161

In this process, the token successfully traverses from p to q, and the states are clear even in a distributed system.

The real algorithm is as follows:

- **Sender P**
  1. (Node or process) P record its state(computational result or system properties) and send a marker to each channel.
- **Receiver Q**
  1. If q's state haven't be recorded, then q record its state and set channel c's state to ∅
  2. If q's state has been recorded, then q set channel c's state messages from the first message q received to the marker.

I want to summarize with **Chandy-Lamport**. We can use the algorithm to create a snapshot of a distributed system. The token cannot be lost when running this algorithm since the token is either in q or channel, so as the snapshot. In Apache Spark, one node will send a message with a marker and listen to all input channels for return markers and snapshot data from other nodes.

2. Advantages

+ A global clock and shared memory are not necessary.
+ The process of the snapshot will not affect computing.
+ It is scalable because when a new vertice is added and we do a snapshot, the marker will only transfer to the node has a direct path to it, which is a part of the DAG of the whole graph and will not add so much cost to take the snapshot.

3. Disadvantages

– It assumes that the channel buffer is infinite and all channels are error-free, which is not common in real distributed systems.

4. Brainstorming

We are going to build a system to collect data from all nodes. We cannot combine all metrics without a global clock because they are different system clock metrics. This algorithm may help us because it is self-diffusing and can get metrics of relatively moments at the adjacent nodes.