OhMyDB!

CS 739 P2 Project Report

Raft Vs Prolonged Network Partitions

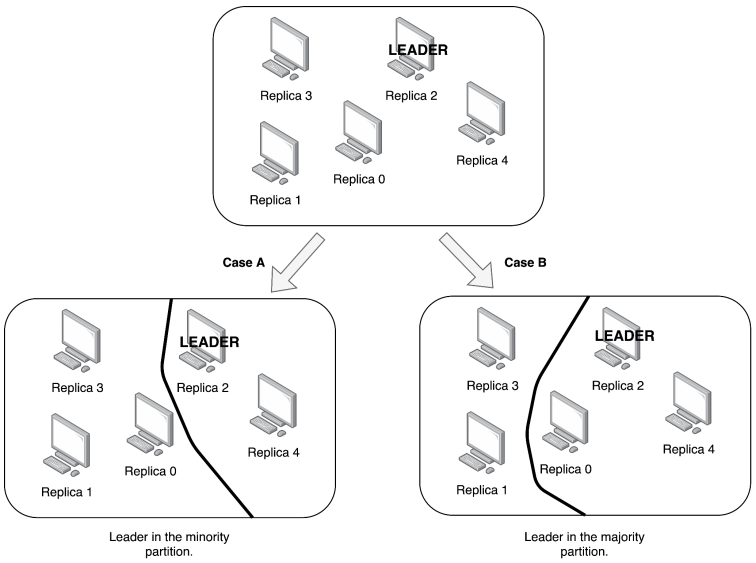
Aditya Jain, Hayden Coffey, and Tzu-Tao Chang

While there weren't any outstanding questions during the presentation, there was some discussion on the case of Raft coping up with a prolonged network partition which we had simulated using OhMyDB's fault injection features. In this report, we explore the problem in more detail and attempt to come up with a mathematical model.

[1] Overview

Let us consider the case when the network gets partitioned into two partitions. We can denote such a network partition as $\Delta(n, m)$ where n is the number of nodes in the larger (majority) partition and m be the number of nodes in the smaller (minority) partition.

There can be two cases as shown in the figure below.



In Case A, the old leader ends up in the minority partition which makes the majority elect a new leader while the old leader continues to believe it is the leader in the minority partition. Upon reconnection, the majority leader causes the minority leader to step down. No new elections take place in this case.

Interesting things happen in Case B, though. Here, the leader stays in the majority partition and continues to work normally. However, the minority is left without a leader and it keeps trying to elect a new leader thereby driving its term up dramatically.

Let us assume that the partition lasts long enough such that the minority's term ends up being significantly higher than the majority's term. In such a case, upon reconnection, the minority makes the majority leader step down. Secondly, the minority nodes can enter election but they can't win it. Furthermore, the majority nodes will either not win election or they will need to immediately step down upon winning election. This will continue to happen until the system converges to a term. This is a useful case to study as it can be observed not only in actual network partitions but also when the system attempts to add a new replica.

The question here is: **What is the expected number of elections the system will go through before stabilizing upon reconnection after a prolonged network partition in which the minority partition was left without a leader?**

[2] State upon reconnection

We are now ready to describe the state of the system upon reconnection. Shortly after reconnection when the majority leader has stepped down, the state of the system can be described as below:

1. Minority nodes have a term T'. These nodes have a high term but can't win elections.

2. Dethroned majority leader also has a term T' as it must have had its term updated before stepping down.
3. Rest of majority nodes with a term $T \ll T'$.

[3] Formulation

Let us now define three types of nodes.

1. Type-1 nodes are minority nodes. These have a high term but can't win elections.
2. Type-2 nodes are those majority nodes that have caught up with the minority's term. If these trigger election, they will win and the system will be stabilized. Note that in the state we described above, the dethroned majority leader is a Type-2 node. It can enter election and win it to stabilize the system.
3. Type-3 nodes are majority nodes that lag behind in term. In the state described above, all majority nodes except their dethroned leader are Type-3 nodes. They can enter elections and may even win but they will be made to step down soon after. Overall, the system doesn't get stabilized.

The state of the system can be written as $S(i, n, m)$ where i denotes the number of type-2 nodes, n denotes the total number of nodes in majority partition, and m denotes the total number of nodes in the minority partition. Overall, if N_j denotes the number of nodes of type j, then we can say:

$$
\begin{aligned}
i &= N_2 \\
n &= N_2 + N_3 \\
m &= N_1
\end{aligned}
$$

Note: At this point we note that the state of the system soon after reconnection as described in the section [2] can be denoted as $S(1,n, m)$ using the notation that we have just defined.

[4] System Evolution

Let us consider how the system evolves when an election happens.

1. If a type-1 node enters election, it loses the election. But it converts all majority nodes to type-2.
$$S(i, n, m) \xrightarrow{\text{type-1 election}} S(n, n, m)$$
2. If a type-2 majority node with a term T' enters election, it will win it. The system will stabilize. Let us denote this stable state as ϕ.
$$S(i, n, m) \xrightarrow{\text{type-2 election}} \phi$$
3. If a type-3 majority node enters election, it will get converted to a type-2 node. (Note that it may win election, but it will be made to step down quickly and the system can't be considered stable yet.)
$$S(i, n, m) \xrightarrow{\text{type-3 election}} S(i + 1,n, m)$$

We have now defined the evolution of system when an election happens. Now we need to define its evolution in time. For this we need to consider when an election happens. Clearly, every node uses a random timeout to trigger an election. Furthermore, we can assume that the mean of these timeouts is approximately the same although the values themselves are picked at random. This key property allows us to model occurrence of elections in each node as a Poisson Process with a rate λ.

With some work, it can be shown that independent election poisson processes running on different nodes can be aggregated to form a compound process with a rate $N\lambda$ where $N = n + m$ is the total number of nodes. While doing so, we assume that elections are instantaneous and therefore don't overlap.

Let E be an event in this compound process, the probability that it belongs to a specific process is $\frac{1}{N}$. Therefore, if there are N_j nodes of the type j and let S_j denote the set of events due to processes corresponding to these nodes, we can write:

$$P(E \in S_j) = \frac{N_j}{N}$$

[5] Solution: Expected number of elections to reach stable state

The problem we are trying to answer can now be defined more formally.

Let $E(i, n, m)$ denote the expected number of elections required for system to translate from state $S(i, n, m)$ to stable state ϕ. Then, given a partition $\Delta(n, m)$, we are interested in calculating $E(1,n, m)$.

We can now write the following recurrence relation based on the state transitions described in section [4].

$$\begin{aligned} E(i, n, m) &= \frac{n-i}{n+m}(1 + E(i+1,n, m)) + \frac{m}{n+m}(1 + E(n, n, m)) + \frac{i}{n+m}(1) \\ &= 1 + \frac{n-i}{n+m}E(i+1,n, m) + \frac{m}{n+m}E(n, n, m) \end{aligned}$$

To complete this recurrence relation, we need to calculate the termination case of $E(n, n, m)$ when the entire majority has turned into type-2. This can be solved as follows, let $X = E(n, n, m)$ then:

$$X = \frac{n}{n+m}(1) + \frac{m}{n+m}(1 + X)$$

Solving we get:

$$E(n, n, m) = X = \frac{n+m}{n}$$

Substituting in expression for $E(i, n, m)$, we get the solution:

$$\boxed{E(i, n, m) = \begin{cases} 1 + \frac{m}{n} + \frac{n-i}{n+m}E(i+1,n, m), & i \neq n \\ \frac{n+m}{n}, & i = n \end{cases}}$$

[6] Result: the case of nearly equal partitions: $\Delta(n, n-1)$

When the partitions are of nearly the same size, we observe that the maximum expected number of elections that can be observed is bounded and is given by:

$$\lim_{n \to \infty} E(1,n, n-1) = 4$$

Furthermore, for the particular case that we simulated and discussed during the presentation with $N = 5$ nodes was $\Delta(3,2)$ and:

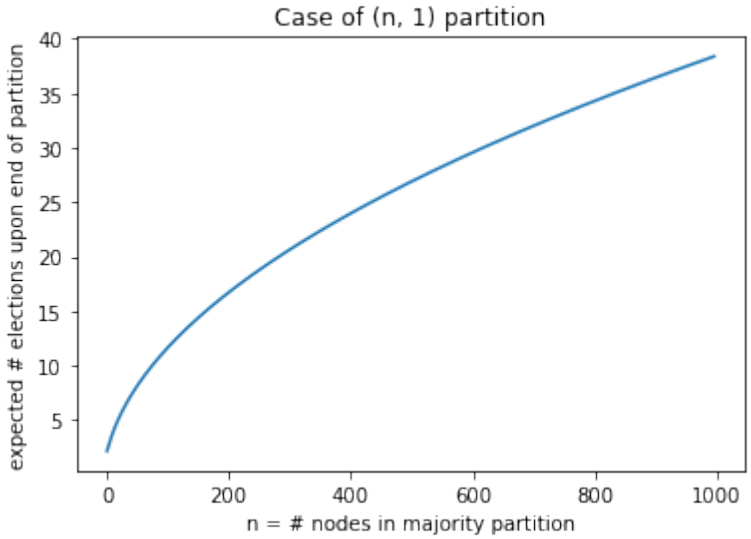
$$E(1,3,2) = 2.46$$

[7] Result: the case of small partitions: $\Delta(n,1)$

This case is much more problematic than the previous one. First of all we observe that the expected number of elections grow unbounded with increase in n.

$$\lim_{n \to \infty} E(1,n,1) \to \infty$$

The following plot shows the dramatic increase in election



This result shows that a single follower upon reconnection after a prolonged partition can cause significant churn in the system. For instance, a system with 50 nodes can expect to undergo 8 elections before stabilizing!

We believe that such a case is likely to be much more common and can even happen when a new node is being added i.e. when there was no real partition.

[8] The Fix: Pre-Voting Phase

A fix for these problems which seems to be widely used across implementations while not being very actively discussed is to have nodes check (pre-voting phase) if they can connect to a majority number of nodes before going for election. Having such a check would prevent minority partition nodes from entering into elections repeatedly during partition. This implies a smooth and graceful reconnection without additional elections at the end of partition.

[9] Conclusion

Our model and analysis suggests that network partitions are much worse than what one might expect. Pre-voting phase is a critical component of Raft and should not be skipped while implementing.