On Using Raft Over Networks: Improving Leader Election

Kostas Choumas and Thanasis Korakis, *Member, IEEE*

***Abstract*—Raft is a state-of-the-art consensus algorithm for state replication over a distributed system of nodes. According to Raft, all state updates occurring anywhere in the system are forwarded to the leader, which is elected among the system nodes to collect and replicate these updates to all other nodes. Thus, the time required for the state replication, named as system response time, depends on the delays between the leader and all other nodes. After multiple node failures and leadership transitions, each node can be leader with a probability that affects the expected response time. The leadership probabilities, in turn, are affected by the random intervals that nodes are waiting, after detecting a leader failure and before competing for the successive leadership. The Raft designers suggest the ranges of these intervals to be equal for all nodes. However, this may result in increased expected response time. In this paper, mathematical models are presented for estimating the ranges resulting in the desired leadership probabilities. The presented theoretical results are also confirmed by testbed experimentation with an open-source and widely used Raft implementation.**

***Index Terms*—Raft, distributed-system, clustering, testbed-experimentation.**

I. Introduction

DISTRIBUTED systems receive extensive attention nowadays, that networking technologies are flourishing and time-critical system functions are spread over multiple interconnected nodes. The emerging Software Defined Networking (SDN) excels in assisting distributed systems, while in parallel is assisted by distributed systems, since distributed SDN controller clusters are more efficient than single-instance controllers [2]. The most popular open-source SDN controllers, such as *ODL* [3] and *ONOS* [4], are fundamentally designed to support clustering. Similarly, *Necklace* [5] provides an architecture for distributed Service Function Chaining that performs surprisingly well. Finally, *Kubernetes* [6], *OpenStack* [7] and *Hyperledger-Fabric* [8] are a few examples of widely used systems with increased scalability and efficiency, due to their distributed operation, which is assisted by *etcd* [9] with distributed key-value store. However, these systems require a protocol for reaching consensus between their nodes.

Raft [10], [11] is the common point of all the aforementioned distributed systems [3]–[9] for reaching consensus. It inherits high performance and correctness from its predecessor and less understandable Paxos, which is assumed as the "gold standard" for distributed consensus [12]. Raft is a leader-based protocol, meaning that commands received for execution by a system node have to be replicated to the whole system through a leader, which is elected among the system nodes. Commands received by non-leaders are redirected to the leader at the time cost of their forwarding. After multiple node failures and leadership transitions, the expected response time of a Raft-operated system, which is the total time required for a command execution, is obviously affected by the leader election process. Leaders receiving more commands or being closer to other nodes are more efficient regarding the system response time, thus they should be boosted by this process.

According to the Raft election process, each node campaigns for the leadership, when it realizes that there is no existing leader and after waiting for a random time interval. This interval is called election timeout and the node with the lowest one, when multiple nodes campaign together, most probably wins. Raft designers propose the selection of this random timeout from the same time range for all nodes, which results in identical leadership winning probabilities for all nodes, when the delays between them are identical. Although this is the case for many scenarios that the distributed nodes are collocated as virtual machines in the same data-center, there are also many other cases where the nodes are connected through networks which are extended over large geographic areas and the delays between them are unequal.

In this paper, *the leadership probabilities of the nodes are modeled as functions of the time ranges of their random election timeouts*. Appropriate configurations of the timeout ranges result in increased leadership probabilities of the nodes being closer to or receiving more commands than the other system nodes, which in turn decreases the expected response time. For example, if all nodes receive equal number of commands per time unit, the response time benefits from a central node as a leader (Figure 1.a), whereas if one node receives almost all commands, then this one should be the leader, despite its potentially high delays to the other nodes (Figure 1.b). In the presented experiments, being agnostic to the receiving command rates at the nodes of each distributed system (Figure 1.c), we use potential networks interconnecting

Manuscript received February 17, 2021; revised November 1, 2021 and January 26, 2022; accepted January 27, 2022. Date of publication January 31, 2022; date of current version June 10, 2022. The research leading to these results has received funding from the European Horizon 2020 Programme for research, technological development and demonstration under Grant Agreement Number No 824994 (H2020 EMPOWER). A part of this work is published in IEEE NetSoft 2020 [doi: 10.1109/NetSoft48620.2020.9165534]. The associate editor coordinating the review of this article and approving it for publication was C. Avin. *(Corresponding author: Kostas Choumas.)*

The authors are with the Department of Electrical and Computer Engineering, University of Thessaly, 382 21 Volos, Greece (e-mail: kohoumas@uth.gr; korakis@uth.gr).

Digital Object Identifier 10.1109/TNSM.2022.3147958

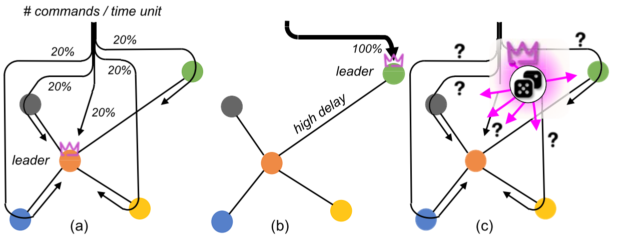


Fig. 1. (a) Nodes receive equal number of commands per time unit, (b) one node receives all commands (c) unknown rates of receiving commands.

these nodes to exemplify how the proposed models can be used for adjusting the leadership probabilities as we wish. As a proof of concept, we *theoretically estimate and experimentally validate the timeout ranges that share equally the leadership over two network examples*. Although equal leadership probabilities do not necessarily result in minimum response time, the presented examples show the capability of the proposed models to succeed every set of leadership probabilities, including the one that minimizes the response time under whichever set of receiving command rates. For specific systems, given their networks and receiving command rates, the same models can be used for achieving the probabilities that minimize the response time.

The paper is divided into the following sections. Section II presents related work suggesting Raft enhancements. Section III highlights insights into the Raft protocol. Sections IV and V model the leadership probabilities and the expected response time of a Raft-operated system running over a network, assuming two different types of node failures. Section VI provides results of testbed experimentation confirming the theoretical analysis of the previous sections. Finally, Section VII concludes the paper.

II. Related Work

In [13] and [14], authors study the placement of a distributed cluster of SDN controllers that use Raft for their consensus. Controller placement is optimized for minimum overall control traffic, based on the bandwidth requirements of the Raft control messages. The leader election is not considered, assuming that all controllers, leaders or not, have almost the same long-term bandwidth requirements for their control messaging. In [1], authors focus on the leader election process and study the effect of the underlying network on the election process and hence on the Raft performance, presenting preliminary results of this work.

The transformation of the network from problem to solution is presented in [15], proposing the usage of dedicated P4-based network devices for the partial offloading of the Raft operation to the underlying network. The modeling and the numerical evaluation of the Raft-operated distributed SDN clusters is given in [16], using Stochastic Activity Networks and estimating the effect of various hardware and software controller failures on the system response time. However, the differentiation of the delays on the connections between the nodes is not considered in this work, as well as their impact on the election process and the response time. The distances and the corresponding propagation delays between the ONOS controllers are considered by the authors of [17] for the master controller election, however, they always suggest the most central node as the best master controller, which is the Raft leader, regardless of the receiving command rates at the controllers.

The election timeouts are adjusted in [18] and [19], as we do in this paper, for the purpose of avoiding useless elections, since the system is unavailable during the elections. In [18], the dynamic increase of the election timeouts eliminates the elections that could happen because the still-alive leader is overloaded. The rationale is that overloaded leaders cannot send in-time their control messages and the time intervals between them vary a lot. The other nodes expect the intervals between the receiving messages to be less than the election timeout, otherwise they trigger elections assuming that the last leader has failed, although it has not. Similarly, in [19], two algorithms are proposed for minimizing the number of elections without a winner (due to split vote, that will be presented later in detail). Finally, in [20], the election timeouts are optimized for minimizing the probability of the cluster's majority being unreachable for the leader due to packet losses on the network links. All these approaches use the election timeouts for optimizing Raft, however, only the last one aims at electing the most efficient leader, but it does not consider the network delays effect.

In this work, we present models for adjusting the election timeouts and, in turn, the leadership probabilities according to the underlying network delays. Given the rates of receiving commands at the nodes of a cluster, the presented models are able to estimate the election timeouts that minimize the response time of this cluster. Although preliminary results are given in [1], this paper presents in more detail extra models covering the response time and two failure types instead of one, as well as extra experiments related to the response time.

III. Insights Into Raft

In a Raft-operated cluster (or distributed system) of nodes, all nodes are *state machines* computing identical copies of the same state and continue operating even if some of them are down. Replicated state machines are used to solve a variety of fault tolerance problems in distributed systems. They are typically implemented using a replicated *series of commands*, which has to be executed by each state machine in order. Keeping the replicated commands consistent is the responsibility of the Raft consensus algorithm.

Raft relies on the election of a leader, which is responsible for managing the replicated commands. The leader has to be elected when a cluster starts or when the previous leader has failed. At any given time, each node is either *leader*, *follower* or *candidate*. Between the elections, there is at most one leader and the other nodes are followers, while during the elections there is no leader and some nodes become candidates, which exist only during the elections. The followers are passive, meaning that they do not issue requests on their own, but only respond to requests from leaders and candidates. The leader handles all incoming commands, even the ones received by the followers, by replicating them across the cluster and forcing each follower to have the same series of commands with the leader. The leader election and the command replication processes will be presented in more detail in the following subsections.

A. Leader Election

When nodes start up, they begin as followers. A node remains follower as long as it receives requests from a leader or a candidate. The leader sends periodic *heartbeat requests* to all followers in order to maintain its authority. The *heartbeat timeout* is the period between the heartbeats, which should be slightly higher than the average time it takes a node to send requests in parallel to all cluster nodes and receive their responses. If a follower receives no heartbeat over a random time period called *election timeout*, which should be an order of magnitude higher than the heartbeat timeout, then the follower assumes that there is no viable leader and starts an election. This happens to all nodes when they start up, since none of them is leader.

The follower starts an election by transitioning to candidate, voting for itself and issuing *vote requests* in parallel to all other nodes. Then, either (a) the candidate becomes the new leader being voted by more than half of the cluster, or (b) another candidate establishes itself as leader and this candidate transitions back to follower, after receiving a heartbeat from the new leader, or (c) there is no winner because of a *split vote*, since none of the candidates collects the required number of votes to be the new leader. When split vote happens, each candidate waits again for a random election timeout to send another round of vote requests and initiate new election.

If the election timeout was not random but fixed and equal for all nodes, a split vote could be repeated indefinitely (e.g., all cluster nodes become candidates simultaneously and vote for themselves repeatedly). Thus, Raft uses random election timeouts to ensure that split votes are rare and resolved quickly. The random election timeout must be higher than the heartbeat timeout, otherwise heartbeat messages do not have the time to reach all followers and useless elections are triggered. Simultaneously, it should be low enough for quick detection of leader failures and decreased periods of system unavailability, over which no leader exists to serve the commands.

B. Command Replication and Response Time

Once a leader is elected, it serves all incoming commands to the cluster. Commands received by the leader, as well as commands received by the followers and redirected to the leader, are appended to its series of commands. Leader with newly appended commands issues a *first round* of *append requests* in parallel to all other nodes to replicate these commands. When the leader gets the replies from more than half of the cluster, the commands are safely replicated and the leader applies the commands to its state machine. Now, the commands are considered as *committed*. Then, the leader sends a *second round* of append requests to the followers, in order for them to apply the already replicated commands to their state machines. If followers crash or run slowly, or if network packets are lost, the leader retries sending append requests infinitely until all followers eventually store all commands. The *response time* of the cluster is defined as the delay introduced between the moment that a command is received by a cluster node, either leader or follower, and the moment that the command is applied to its state machine.

IV. Raft Over Network (Assuming Instant Failures)

Although Raft designers focused on a cluster deployed over a fully connected mesh network with links featuring equal and almost negligible delays, many real Raft-operated clusters rely on networks with unequal and remarkable delays on their links. In the latter case, the nodes with lower sum of delays to the other nodes, named as central nodes, most probably collect first the required votes to become leaders than the nodes on the edge. Although the election of a central node facilitates faster communication between the leader and the majority of the followers, it is not certain that it is minimizing the expected response time, which also depends on the rates of the incoming commands to the cluster nodes.

More specifically, assuming a cluster $\mathscr{N}$ with N nodes, the expected response time for the commands received by each cluster node $n_i \in \mathscr{N}$ is

$$t_i^{\text{rep}} = \sum_{n_l \in \mathscr{N}} \pi_l t_{il}^{\text{rep}}, \tag{1}$$

where t_{il}^{rep} is the response time for a command sent to n_i given that $n_l \in \mathscr{N}$ is the current leader after multiple leadership transitions, which happens with probability π_l, named as *leadership probability* of n_l. Time t_{il}^{rep} is spent in order (a) n_i to redirect its incoming command to leader n_l, (b) n_l to replicate this command to the cluster through the first round of append requests, (c) n_l collect the append replies from the majority of the cluster nodes and finally (d) n_l to inform n_i, and the whole cluster, that the command is committed through the second round of append requests. Then, the expected response time over all nodes, using also Equation (1), is modeled as

$$t^{\text{rep}} = \sum_{n_i \in \mathscr{N}} \lambda_i t_i^{\text{rep}} = \sum_{n_i \in \mathscr{N}} \sum_{n_l \in \mathscr{N}} \lambda_i \pi_l t_{il}^{\text{rep}}, \tag{2}$$

where λ_i is the normalized rate of incoming commands received by n_i (the sum of normalized rates for all nodes is $\sum_{i=1}^{N} \lambda_i = 1$). Table I summarizes the notation that is used in the following sections.

Given a network with stable response times t_{il}^{rep} and incoming command rates $\boldsymbol{\lambda} = [\lambda_1, \lambda_1, \ldots, \lambda_N]$, t^{rep} can be minimized by configuring the leadership probabilities $\boldsymbol{\pi} = [\pi_1, \pi_1, \ldots, \pi_N]$. For the sake of simplicity, it is assumed that the network topology, the delays on its links and the incoming command rates are stable over time, as well as that the packet loss ratios are negligible. The presented results can be easily extended to scenarios that some of these parameters do not hold. For example, under these assumptions, if all rates λ_i are equal to $1/N$ and n_c is the most central node, then t^{rep} is minimized for $\pi_c = 1$ (Figure 1.a), while if all commands are sent to another node n_r and $\lambda_r = 1$, then t^{rep} is minimized

TABLE I
NOTATION SUMMARY

$\mathscr{N}$	cluster of N nodes $\{n_1, n_2, \ldots, n_N\}$
$\mathscr{N}'$	subset of $\mathscr{N}$
$\mathscr{N}^{-n_i}$	set of subsets of $\mathscr{N} - \{n_i\}$ with cardinality $\lceil (N-1)/2 \rceil$
t_{il}^{rep}	response time for a command sent to n_i, when n_l is leader
t_i^{rep}	expected response time for a command sent to n_i
t^{rep}	expected response time for all commands
t_o^{out}	fixed minimum election timeout for all nodes
t_i^{out}	random part of election timeout of node n_i
α_i	maximum t_i^{out} (default 1, always ≤ 1)
$f(\tau)$	PDF of standard uniform distribution
$\bar{F}(\tau)$	CCDF of standard uniform distribution
d_{ij}	delay on the link between n_i and n_j
d_{l*}	$\lceil (N-1)/2 \rceil^{\text{th}}$ highest delay from n_l to all other nodes
$d_{liz}^{\mathscr{N}'}$	the lowest $t_z^{\text{out}} - t_i^{\text{out}}$, in order n_i to be voted at least by $\{n_i\} \cup \mathscr{N}'$ after the failure of leader n_l
p_{li}	probability that n_i is the next leader after n_l
π_i	probability that n_i is leader after multiple transitions
λ_i	normalized rate of commands sent to n_i ($\sum_{i=1}^{N} \lambda_i = 1$)

for $\pi_r = 1$ (Figure 1.b), whichever is the network topology and the centrality of n_r. The latter is true since the existence of a leader different from n_r would result in extra delays for redirecting all commands from n_r to the other leader.

In the three following subsections, it is also assumed that the leadership transitions happen only due to *instant failures* of the leaders. Failures are called instant when the leader remains off for a period of time almost equal to the average election timeout, thus, it is up again to participate in the following election. In each subsection is presented how the leadership probabilities $\boldsymbol{\pi}$ are modeled as functions of the ranges of the random election timeouts. Starting with the simplest network with unequal delays between the nodes, which is a 3-node bus network, simple models are presented for estimating theoretically the leadership probabilities and the expected response time.

A. 3-Node Bus Network

Let's assume a cluster $\mathscr{N}$ with $N = 3$ interconnected nodes n_1, n_2 and n_3 connected through a bus network, where n_2 is located in the middle with equal distances to n_1 and n_3, as it is depicted in Figure 2(a). The delay of the link connecting two nodes is mainly due to the signal propagation, thus it is proportional to its length, which means that the delays from n_1 to n_2 and n_3 are d and $2d$ respectively. Each node $n_i \in \mathscr{N}$ chooses its election timeout $t_o^{\text{out}} + t_i^{\text{out}}$, where t_o^{out} is fixed and t_i^{out} is sampled from the standard uniform distribution with Probability Density Function (PDF) $f(\tau) = 1$ for $\tau \in [0, 1]$ and 0 otherwise.

Leadership Probabilities: Assuming that n_1 is the leader that has failed, n_3 becomes the successive leader, if and only if its vote request is received by n_2 before n_2 attempts to send its own vote requests. Otherwise, n_2 votes for itself and also gets the vote of n_1 that is closer to n_2, thus n_2 wins the leadership with 2 votes out of 3. The lower part of Figure 2(a) depicts this fact that n_3 wins the vote of n_2, by illustrating the timeout of n_3 ($t_o^{\text{out}} + t_3^{\text{out}}$) ending at least time d before the timeout of n_2 ($t_o^{\text{out}} + t_2^{\text{out}}$). Moreover, the heartbeat requests of n_1 are received by n_2 and n_3 with delays d and $2d$ respectively. Both followers start their timeouts once they receive the last heartbeat of n_1, thus n_2 has a head start of time d, as it is depicted in the upper part of Figure 2(a). Obviously, n_3 becomes the next leader, if and only if $d + t_o^{\text{out}} + t_2^{\text{out}} > 2d + t_o^{\text{out}} + t_3^{\text{out}} + d \Rightarrow t_2^{\text{out}} > t_3^{\text{out}} + 2d$. As follows, assuming that $2d \leq 1$, the probability that n_3 is the successive leader after n_1 is

$$p = \Pr\left[t_2^{\text{out}} > t_3^{\text{out}} + 2d\right] = \int_{-\infty}^{\infty} \int_{\tau_3 + 2d}^{\infty} f(\tau_3) f(\tau_2)\, d\tau_2\, d\tau_3$$

$$= \int_0^{1-2d} \int_{\tau_3 + 2d}^{1} d\tau_2\, d\tau_3 = \frac{(1-2d)^2}{2}. \tag{3}$$

Using Markov chain to model the leadership transitions from one node to the other, occurred due to multiple instant failures, we get the transition probability matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 1-p & p \\ 1/2 & 0 & 1/2 \\ p & 1-p & 0 \end{bmatrix},$$

where the probability in row x and column y is the transition probability from node n_x to node n_y. The matrix is symmetric because of the symmetric network topology. The transition probability from n_3 to n_1 is the same as the one from n_1 to n_3, that is p. Similarly, the transition probabilities from n_2 to either n_1 or n_3 are the same and equal to $1/2$. As follows, the steady-state probabilities are given by the vector

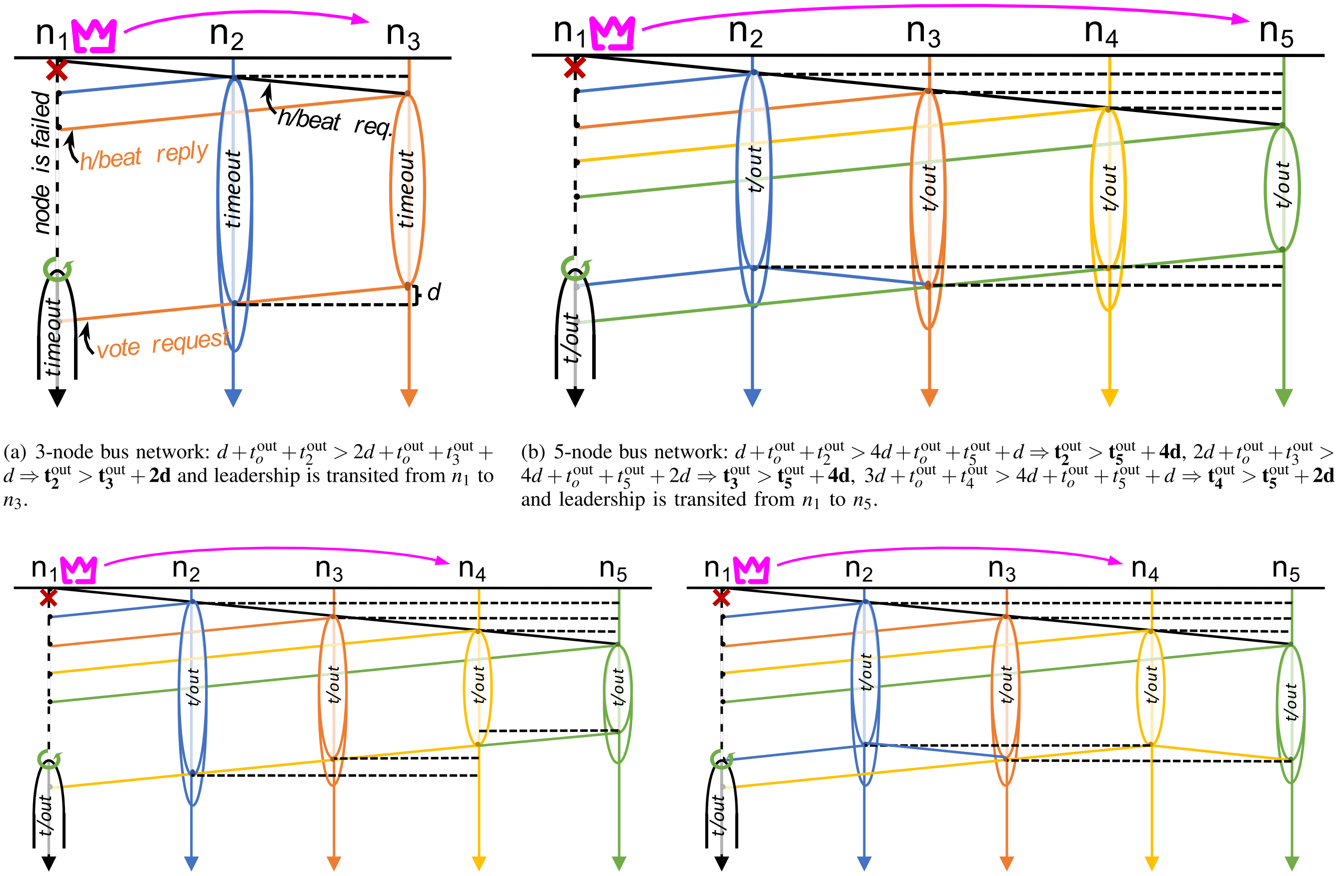
$$\boldsymbol{\pi} = \left[\frac{1}{4-2p},\quad \frac{1-p}{2-p},\quad \frac{1}{4-2p}\right]$$

$$= \left[\frac{1}{3+4d-4d^2},\quad \frac{1+4d-4d^2}{3+4d-4d^2},\quad \frac{1}{3+4d-4d^2}\right], \tag{4}$$

where the steady-state probability π_i models the leadership probability of n_i. Apart from the case of zero delay $d = 0$, where $p = 1/2$ and all leadership probabilities are equal to $1/3$, for all other delays, the leadership probability of the central node is higher than that of the nodes on the edges.

Response Time: The response time t_{11}^{rep} is the time needed by n_1, when it is both leader and command receiver, to send the first round of append requests to all other nodes and collect at least 2 replies out of 3, which are its own reply (zero delay) and the reply of the nearby n_2 (two times delay d for the exchanged request and reply between n_1 and n_2). Thus, $t_{11}^{rep} = d + d = 2d$. For n_2 and n_3, the corresponding response times are the same and equal to $t_{22}^{\text{rep}} = t_{33}^{\text{rep}} = 2d$. On the other hand, commands received by follower n_1, when n_2 is the leader, increase the response time to $t_{12}^{\text{rep}} = d + t_{22}^{\text{rep}} + d = 4d$, since extra delay d is added twice to the response time t_{22}^{rep} due to the command redirection from follower n_1 to leader n_2 and the append request of the second round sent from leader n_2 to n_1. Similarly, $t_{13}^{\text{rep}} = 2d + t_{33}^{\text{rep}} + 2d = 6d$, since the delay on the connection between n_1 and n_3 is $2d$. Finally, $t_{32}^{\text{rep}} = t_{12}^{\text{rep}} = 4d$, $t_{31}^{\text{rep}} = t_{13}^{\text{rep}} = 6d$ and $t_{21}^{\text{rep}} = t_{23}^{\text{rep}} = t_{33}^{\text{rep}} + 2d = 4d$. The expected response time is modeled by Equation (2), using the previously estimated probabilities $\boldsymbol{\pi}$ and the given incoming command rates $\boldsymbol{\lambda}$.

Equalized Leadership Probabilities: Given the uncontrollable $\boldsymbol{\lambda}$, the expected response time can be minimized by only



(a) 3-node bus network: $d+t_o^{\text{out}}+t_2^{\text{out}} > 2d+t_o^{\text{out}}+t_3^{\text{out}}+d \Rightarrow \mathbf{t_2^{out}} > \mathbf{t_3^{out}}+\mathbf{2d}$ and leadership is transited from n_1 to n_3.

(b) 5-node bus network: $d+t_o^{\text{out}}+t_2^{\text{out}} > 4d+t_o^{\text{out}}+t_5^{\text{out}}+d \Rightarrow \mathbf{t_2^{out}} > \mathbf{t_5^{out}}+\mathbf{4d}$, $2d+t_o^{\text{out}}+t_3^{\text{out}} > 4d+t_o^{\text{out}}+t_5^{\text{out}}+2d \Rightarrow \mathbf{t_3^{out}} > \mathbf{t_5^{out}}+\mathbf{4d}$, $3d+t_o^{\text{out}}+t_4^{\text{out}} > 4d+t_o^{\text{out}}+t_5^{\text{out}}+d \Rightarrow \mathbf{t_4^{out}} > \mathbf{t_5^{out}}+\mathbf{2d}$ and leadership is transited from n_1 to n_5.

(c) 5-node bus network: $\mathbf{t_2^{out}} > \mathbf{t_4^{out}}+\mathbf{4d}$, $\mathbf{t_3^{out}} > \mathbf{t_4^{out}}+\mathbf{2d}$, $\mathbf{t_5^{out}} > \mathbf{t_4^{out}}-\mathbf{2d}$ and leadership is transited from n_1 to n_4, since the latter is voted at least by itself, n_2 and n_3.

(d) 5-node bus network: $\mathbf{t_2^{out}} > \mathbf{t_4^{out}}+\mathbf{2d}$, $\mathbf{t_3^{out}} > \mathbf{t_4^{out}}+\mathbf{2d}$, $\mathbf{t_5^{out}} > \mathbf{t_4^{out}}$ and leadership is transited from n_1 to n_4, since the latter is voted at least by itself, n_3 and n_5.

Fig. 2. 3-node and 5-node bus networks assuming instant failures. The vertical axis depict the time sequence of the messaging and the duration of the election timeouts.

adjusting the leadership probabilities $\boldsymbol{\pi}$, which in turn happens by editing the ranges of the random election timeouts t_i^{out}. Being agnostic to $\boldsymbol{\lambda}$, as it is already illustrated in Figure 1, we cannot provide the $\boldsymbol{\pi}$ minimizing the expected response time, however, we can show, e.g., how the leadership can be shared equally between the three nodes, just for the purpose of showcasing the capabilities of manipulating the leadership probabilities.

By editing t_1^{out} and t_3^{out} to be sampled from a uniform distribution in the shorter range $[0, \alpha]$ (with PDF $(1/\alpha)f(\tau/\alpha)$ and $\alpha < 1$), the average election timeouts of the nodes at the edge, n_1 and n_3, become lower than that of the central node n_2, thus n_1 and n_3 win more elections and the probabilities π_1 and π_3 are increased, while π_2 is decreased. The value of α, for which the increased π_1 and π_3 are equalized with the decreased π_2, is estimated below. If $\alpha \leq 1-2d$, then the transition probability of Equation (3) changes to

$$\begin{aligned} p &= \int_{-\infty}^{\infty}\int_{\tau_3+2d}^{\infty} \frac{1}{\alpha} f\left(\frac{\tau_3}{\alpha}\right) f(\tau_2)\, d\tau_2\, d\tau_3 \\ &= \frac{1}{\alpha}\int_0^{\alpha}\int_{\tau_3+2d}^{1} d\tau_2\, d\tau_3 = 1-2d-\frac{\alpha}{2} \end{aligned}$$

and the leadership probabilities are modeled as

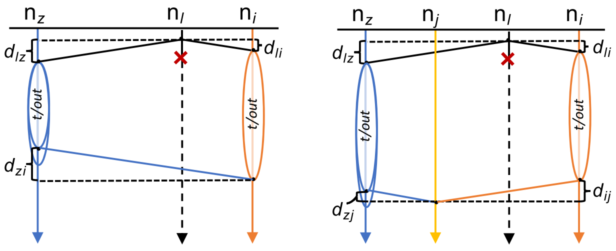
$$\boldsymbol{\pi} = \left[\frac{1}{2+4d+\alpha}, \quad \frac{4d+\alpha}{2+4d+\alpha}, \quad \frac{1}{2+4d+\alpha}\right].$$

If $4d < 1$ and $\alpha = 1-4d$, then all nodes have equal leadership probabilities $\pi_1 = \pi_2 = \pi_3$.

Now, let's generalize by estimating the leadership probabilities and the expected response time of a cluster operating over any network.

B. General Case Network

Let's assume a cluster $\mathcal{N}$ with N nodes, where the random part t_i^{out} of the election timeout of each node $n_i \in \mathcal{N}$ is sampled from the range $[0, \alpha_i]$ with PDF $(1/\alpha_i)f(\tau/\alpha_i)$. The Complementary Cumulative Distribution Function (CCDF) of the standard uniform distribution is denoted as $\bar{F}(\tau) = \int_\tau^\infty f(\tau')d\tau' = 1-\tau$ for $\tau \in [0, 1]$, 1 for $\tau < 0$ and 0 otherwise. Each node n_i has its own CCDF $\bar{F}(\tau/\alpha_i)$, which is the probability that $t_i^{\text{out}} \geq \tau$. The delay between two nodes $n_i, n_j \in \mathcal{N}$ is $d_{ij} = d_{ji}$, with $d_{ii} = 0$. The heartbeat timeout is twice the maximum delay between every couple of



(a) n_i becomes candidate.

(b) n_i is voted by n_j.

Fig. 3. General case network: Relationship of the election timeouts of n_i and every other n_z, apart from the leader n_l that has failed.

nodes and it is an order of magnitude lower than the minimum election timeout, which is equal to the fixed t_o^{out} for zeroed random part, thus $\max_{n_i,n_j \in \mathscr{N}} d_{ij} \ll t_o^{\text{out}}$.

Leadership Probabilities: Given a leader $n_l \in \mathscr{N}$ that has failed, two followers $n_i \neq n_l$ and $n_z \neq n_i, n_l$[1] initiate their election timeouts when they receive the last heartbeat of n_l. This happens with delay d_{li} and d_{lz} respectively after n_l sending this heartbeat. Thus, as it is depicted in Figure 3(a), n_i becomes candidate and votes for itself, if its timeout plus d_{li} is less than the timeout of every other follower n_z plus d_{lz} plus the delay on the forwarding of the vote request of n_z to n_i, which is equivalent to

$$d_{li} + t_i^{\text{out}} < d_{lz} + t_z^{\text{out}} + d_{zi},\ \forall n_z \neq n_i, n_l. \tag{5}$$

Otherwise, n_i remains follower voting for another follower n_z, which changed to candidate and sent the vote request earlier. In addition, candidate n_i is voted by follower $n_j \neq n_i$, if the vote request of n_i goes earlier to n_j than the vote request of every other candidate n_z, as it is illustrated in Figure 3(b), which is equivalent to

$$\begin{aligned} &d_{li} + t_i^{\text{out}} + d_{ij} < d_{lz} + t_z^{\text{out}} + d_{zj},\ \forall n_z \neq n_i, n_l,\ \text{or} \\ &t_z^{\text{out}} > t_i^{\text{out}} + d_{li} - d_{lz} + d_{ij} - d_{zj},\ \forall n_z \neq n_i, n_l. \end{aligned} \tag{6}$$

Keep in mind that Equation (5) is given by Equation (6) by setting $n_j = n_i$, since n_i is voted by itself if and only if it becomes candidate.

Based on the above, n_i is voted by itself and at least a set of other nodes $\mathscr{N}' \subseteq \mathscr{N} - \{n_i\}$, if Equation (6) holds for n_i and every node $n_j \in \mathscr{N}'$, or equivalently

$$\begin{aligned} &t_z^{\text{out}} > t_i^{\text{out}} + d_{li} - d_{lz} + \max_{n_j \in \{n_i\} \cup \mathscr{N}'} \{d_{ij} - d_{zj}\}, \\ &\qquad \forall n_z \neq n_i, n_l,\ \text{or} \\ &t_z^{\text{out}} > t_i^{\text{out}} + d_{liz}^{\mathscr{N}'},\ \forall n_z \neq n_i, n_l, \end{aligned} \tag{7}$$

where

$$d_{liz}^{\mathscr{N}'} = d_{li} - d_{lz} + \max_{n_j \in \{n_i\} \cup \mathscr{N}'} \{d_{ij} - d_{zj}\}. \tag{8}$$

From Equation (7), $d_{liz}^{\mathscr{N}'}$ is the lower bound of the timeout difference $t_z^{\text{out}} - t_i^{\text{out}}$. If Equation (7) is satisfied, then n_i *is voted at least by*[2] all nodes in $\{n_i\} \cup \mathscr{N}'$ after the failure of leader n_l. This happens with probability

$$\begin{aligned} p_{li}^{\mathscr{N}'} &= \Pr[\{n_i\ v.l.\ \{n_i\} \cup \mathscr{N}'\}] \\ &= \Pr\left[\bigcap_{n_z \neq n_i, n_l} \left\{t_z^{\text{out}} > t_i^{\text{out}} + d_{liz}^{\mathscr{N}'}\right\}\right] \\ &= \int_{-\infty}^{\infty} \frac{1}{\alpha_i} f\left(\frac{\tau_i}{\alpha_i}\right) \left(\prod_{n_z \neq n_i, n_l} \int_{\tau_i + d_{liz}^{\mathscr{N}'}}^{\infty} \frac{1}{\alpha_z} f\left(\frac{\tau_z}{\alpha_z}\right) d\tau_z\right) d\tau_i \\ &= \frac{1}{\alpha_i} \int_0^{\alpha_i} \prod_{n_z \neq n_i, n_l} \bar{F}\left(\frac{\tau_i + d_{liz}^{\mathscr{N}'}}{\alpha_z}\right) d\tau_i, \end{aligned} \tag{9}$$

Assuming that $\mathscr{N}^{-n_i}$ is the family of sets over $\mathscr{N} - \{n_i\}$ that have cardinality equal to $\lceil (N-1)/2 \rceil$, if n_i *v.l.* itself and a set of nodes $\mathscr{N}' \in \mathscr{N}^{-n_i}$, then n_i is the next leader, since it is voted by the cluster majority. As follows, n_i is the successive leader after n_l with probability

$$p_{li} = \Pr\left[\bigcup_{\mathscr{N}' \in \mathscr{N}^{-n_i}} \{n_i\ v.l.\ \{n_i\} \cup \mathscr{N}'\}\right]. \tag{10}$$

Besides that, an election may end with split vote, in which case it is repeated and most probably the previous leader is re-elected. This happens because the previous leader initiated its election timeout during the first election that ended with split vote, while the other nodes restart their timeouts for the repeated election. In this way, the timeout of the previous leader most probably ends first and much earlier than the others. As follows, the probability of no leadership transition is approximately the probability of split vote, which is the complement of having leadership transition, thus

$$p_{ll} = 1 - \sum_{n_i \neq n_l} p_{li}. \tag{11}$$

The transition probability matrix is

$$\mathbf{P} = [p_{li} : \forall n_l, n_i \in \mathscr{N}] \tag{12}$$

and the leadership probabilities $\boldsymbol{\pi}$ are modeled as the eigenvector of $\mathbf{P}$ with eigenvalue equal to one ($\boldsymbol{\pi}\mathbf{P} = \boldsymbol{\pi}$), which always exists due to the PerronâĂŞFrobenius theorem.

Response Time: The response time is

$$t_{il}^{\text{rep}} = 2(d_{il} + d_{l*}), \tag{13}$$

where d_{l*} is the delay d_{lz}, between the leader n_l and the node $n_z \neq n_l$ that is ranked $\lceil (N-1)/2 \rceil^{\text{th}}$ (ties broken arbitrarily), when all cluster nodes apart from n_l are increasingly ordered according to their delay to n_l. The response time is actually the sum of (a) delay d_{il} on the command redirection from n_i to the leader, plus (b) delay $2d_{l*}$ in order for the leader to send the first round of append requests and receive the corresponding replies from the cluster majority formed by the $\lceil (N-1)/2 \rceil$ most nearby nodes, plus (c) delay $d_{li} = d_{il}$ for sending back to n_i the append request of the second round. The expected response time is modeled again by Equation (2).

[1] $n_i \neq n_l$ and $n_z \neq n_i, n_l$ are equivalent to $n_i \in \mathscr{N} - \{n_l\}$ and $n_z \in \mathscr{N} - \{n_i, n_l\}$ respectively.

[2] From now on, *"is voted at least by"* is written as *"v.l."*

Back to 3-Node Bus Network: Let's now see how the previous models for the leadership probabilities and the response time, applicable to any network, conclude to the same results with Section IV-A, when they are specified for the 3-node bus network. Having in mind that $\mathscr{N}^{-n_3} = \{\{n_1\},\{n_2\}\}$ is the set of subsets of $\mathscr{N} - \{n_3\}$ with cardinality equal to $\lceil (N-1)/2 \rceil = 1$,

$$\begin{aligned} p_{13} &= \Pr\left[\bigcup_{\mathscr{N}' \in \mathscr{N}^{-n_3}} \{n_3 \ v.l. \ \{n_3\} \cup \mathscr{N}'\}\right] \\ &= \Pr[\{n_3 \ v.l. \ n_3, n_1\} \cup \{n_3 \ v.l. \ n_3, n_2\}]. \end{aligned}$$

The event $\{n_3 \ v.l. \ n_3, n_2\}$ is equivalent to $\{n_3 \text{ is voted by } n_3, n_2, n_1\}$ and superset of $\{n_3 \ v.l. \ n_3, n_1\}$, since there is no way n_3 to be voted by n_2 and not be voted by n_1. If n_2 votes for n_3, then the vote request of n_3 has reached n_2 before its election timeout expires, which means that n_2 does not request votes, thus n_1 receives vote request only from n_3 and certainly votes for it. Thus, using also Equation (9),

$$\begin{aligned} p_{13} &= \Pr[\{n_3 \ v.l. \ n_3, n_1\} \cup \{n_3 \ v.l. \ n_3, n_2\}] \\ &= \Pr[\{n_3 \ v.l. \ n_3, n_2\}] = p_{13}^{\{n_2\}} = \\ &= \frac{1}{\alpha_3} \int_0^{\alpha_3} \prod_{n_z \neq n_3, n_1} \bar{F}\left(\frac{\tau_3 + d_{13z}^{\{n_2\}}}{\alpha_z}\right) d\tau_3, \\ &= \frac{1}{\alpha_3} \int_0^{\alpha_3} \bar{F}\left(\frac{\tau_3 + 2d}{\alpha_2}\right) d\tau_3, \end{aligned}$$

since $d_{132}^{\{n_2\}} = d_{13} - d_{12} + \max_{n_j \in \{n_3, n_2\}}\{d_{3j} - d_{2j}\} = 2d - d + \max\{-d, d\} = 2d$. Assuming that $\alpha_2 = \alpha_3 = 1$,

$$\begin{aligned} p_{13} &= \int_0^1 \bar{F}(\tau_3 + 2d) d\tau_3 \\ &= \int_0^1 (1 - \tau_3 - 2d) d\tau_3 = \frac{(1-2d)^2}{2}, \end{aligned}$$

while if $\alpha_2 = 1$ and $\alpha_3 \leq 1 - 2d$,

$$\begin{aligned} p_{13} &= \frac{1}{\alpha_3} \int_0^{\alpha_3} \bar{F}(\tau_3 + 2d) d\tau_3 \\ &= \frac{1}{\alpha_3} \int_0^{\alpha_3} (1 - \tau_3 - 2d) d\tau_3 = 1 - 2d - \frac{\alpha_3}{2}. \end{aligned}$$

As follows, the probabilities are the same with the ones of Section IV-A, if α and p are mapped to α_3 and p_{13} respectively. It is trivial to show that the response times of both sections are identical, thus this part is dismissed.

In the following section, the models for the leadership probabilities and the response time are specified for the less trivial 5-node bus network, and then they are used for equalizing again the leadership probabilities. Other indicative network topologies, like symmetric star or fully connected mesh networks with equal delays between the nodes, are trivial cases for the equalization exercise since they already feature equal leadership probabilities under the default election timeout ranges. Arbitrary network topologies, coming from realistic scenarios of the Raft usage, would be investigated in our future work.

C. 5-Node Bus Network

Let's assume a cluster $\mathscr{N} = \{n_1, n_2, \ldots, n_5\}$ with $N = 5$ nodes, where $n_i \in \{n_1, n_2, n_3, n_4\}$ is before n_{i+1} and the delay between them is d, as it is depicted in Figure 2(b). With respect to the heartbeat and election timeout constraints described in the previous section, it is assumed that $\max_{n_i, n_j \in \mathscr{N}} d_{ij} \ll t_o^{\text{out}} \Rightarrow 4d \ll t_o^{\text{out}}$.

Leadership Probabilities: Starting with the estimation of the transition probabilities, and especially of p_{15}, $\mathscr{N}^{-n_5}$ is the set of subsets of $\mathscr{N} - \{n_5\}$ with cardinality equal to $\lceil (N-1)/2 \rceil = 2$, which means that $\mathscr{N}^{-n_5} = \{\{n_1, n_2\}, \{n_1, n_3\}, \{n_1, n_4\}, \{n_2, n_3\}, \{n_2, n_4\}, \{n_3, n_4\}\}$. From Equation (10),

$$\begin{aligned} p_{15} = \Pr[&\{n_5 \ v.l. \ n_5, n_1, n_2\} \cup \{n_5 \ v.l. \ n_5, n_1, n_3\} \\ &\cup \{n_5 \ v.l. \ n_5, n_1, n_4\} \cup \{n_5 \ v.l. \ n_5, n_2, n_3\} \\ &\cup \{n_5 \ v.l. \ n_5, n_2, n_4\} \cup \{n_5 \ v.l. \ n_5, n_3, n_4\}] \\ = \Pr&[\{n_5 \ v.l. \ n_5, n_3, n_4\}] = p_{15}^{\{n_3, n_4\}}, \end{aligned}$$

since event $\{n_5 \ v.l. \ n_5, n_3, n_4\}$ is superset of all previous events. The rationale is that there is no way n_5 to be voted by n_1 or n_2 and not be voted by itself and the most nearby n_3 and n_4 (explanations have been given for the analogous case of leadership transition from n_1 to n_3 in the 3-node bus network). From Equations (8) and (9),

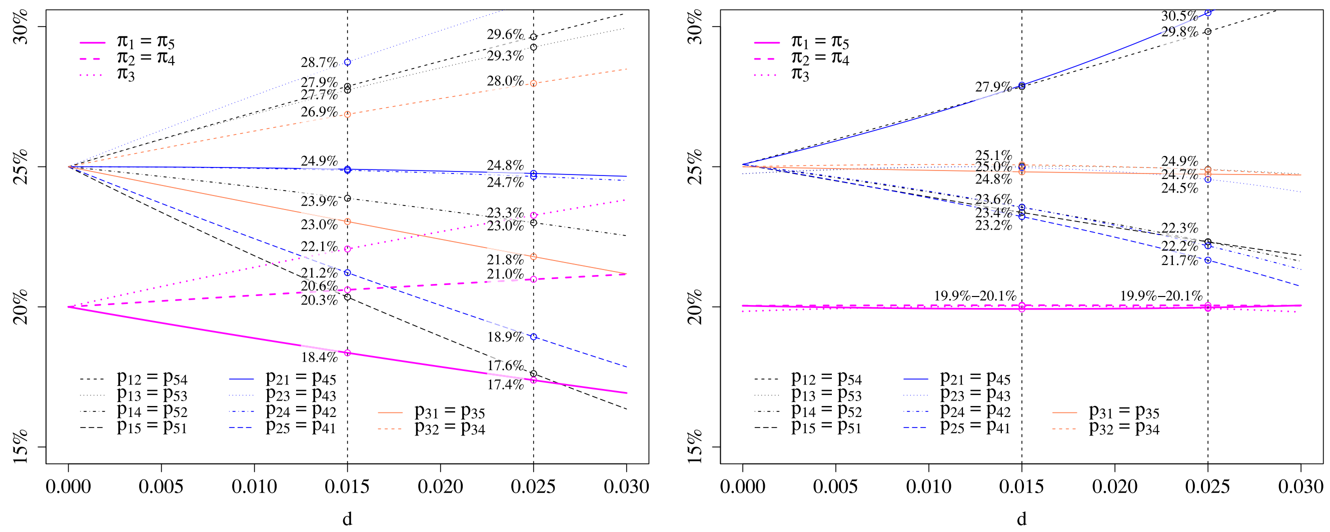
$$\begin{aligned} p_{15}^{\{n_3, n_4\}} = \frac{1}{\alpha_5} \int_0^{\alpha_5} &\bar{F}\left(\frac{\tau_5 + 4d}{\alpha_2}\right) \bar{F}\left(\frac{\tau_5 + 4d}{\alpha_3}\right) \\ &\times \bar{F}\left(\frac{\tau_5 + 2d}{\alpha_4}\right) d\tau_5, \end{aligned}$$

given that $d_{152}^{\{n_3, n_4\}} = d_{15} - d_{12} + \max_{n_j \in \{n_3, n_4, n_5\}}\{d_{5j} - d_{2j}\} = d_{15} - d_{12} + d_{35} - d_{32} = 4d - d + 2d - d = 4d$, $d_{153}^{\{n_3, n_4\}} = 4d$ and $d_{154}^{\{n_3, n_4\}} = 2d$. Figure 2(b) illustrates how much higher must be t_2^{out}, t_3^{out} and t_4^{out} comparing to t_5^{out}, or in other words, which are the lower bounds $d_{152}^{\{n_3, n_4\}}$, $d_{153}^{\{n_3, n_4\}}$ and $d_{154}^{\{n_3, n_4\}}$ of the corresponding timeout differences, in order n_5 to be the successive leader after n_1.

Similarly, p_{14} is estimated as the probability that n_4 is voted by itself and at least n_2 and n_3 or n_3 and n_5. Thus, from Equations (8), (9) and (10),

$$\begin{aligned} p_{14} &= \Pr[\{n_4 \ v.l. \ n_4, n_2, n_3\} \cup \{n_4 \ v.l. \ n_4, n_3, n_5\}] \\ &= \Pr[\{n_4 \ v.l. \ n_4, n_2, n_3\}] + \Pr[\{n_4 \ v.l. \ n_4, n_3, n_5\}] \\ &\quad - \Pr[\{n_4 \ v.l. \ n_4, n_2, n_3, n_5\}] \\ &= p_{14}^{\{n_2, n_3\}} + p_{14}^{\{n_3, n_5\}} - p_{14}^{\{n_2, n_3, n_5\}} \\ &= \frac{1}{\alpha_4} \int_0^{\alpha_4} \bar{F}\left(\frac{\tau_4 + 4d}{\alpha_2}\right) \bar{F}\left(\frac{\tau_4 + 2d}{\alpha_3}\right) \bar{F}\left(\frac{\tau_4 - 2d}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_4 + 2d}{\alpha_2}\right) \bar{F}\left(\frac{\tau_4 + 2d}{\alpha_3}\right) \bar{F}\left(\frac{\tau_4}{\alpha_5}\right) \\ &\qquad - \bar{F}\left(\frac{\tau_4 + 4d}{\alpha_2}\right) \bar{F}\left(\frac{\tau_4 + 2d}{\alpha_3}\right) \bar{F}\left(\frac{\tau_4}{\alpha_5}\right) d\tau_4. \end{aligned}$$

For the first term of the integrated sum above, the lower bounds of the timeout differences, namely $d_{142}^{\{n_2, n_3\}} = 4d$, $d_{143}^{\{n_2, n_3\}} = 2d$ and $d_{145}^{\{n_2, n_3\}} = -2d$, are illustrated in Figure 2(c). The same



(a) When the ranges of the election timeouts are equal.

(b) When the ranges of the election timeouts equalize the leadership probabilities.

Fig. 4. Transition percent probabilities $\mathbf{P}$ and leadership percent probabilities $\boldsymbol{\pi}$ over the 5-node bus network assuming instant failures.

holds for the lower bounds $d_{142}^{\{n_3,n_5\}} = 2d$, $d_{143}^{\{n_3,n_5\}} = 2d$ and $d_{145}^{\{n_3,n_5\}} = 0$ of the second term and Figure 2(d). Following the same process (more details in Appendix), all transition probabilities $\mathbf{P}$ are estimated and leadership probabilities $\boldsymbol{\pi}$ are modeled by the corresponding eigenvector of $\mathbf{P}$.

Response Time: Starting with delays d_{1*} and d_{5*}, n_3 is ranked as $\lceil (N-1)/2 \rceil = 2^{\text{nd}}$ regarding the node delays to n_1 and n_5 respectively, thus $d_{1*} = d_{13} = 2d$ and $d_{5*} = d_{53} = 2d$. Similarly, $d_{2*} = d_{3*} = d_{4*} = d$, since the delays from the other three nodes n_2, n_3 and n_4 to the corresponding 2^{nd} most nearby node is d. From Equation (13), $t_{i1}^{\text{rep}} = 2(d_{i1} + d_{1*}) = 2d(|i-1|+2)$, $t_{i2}^{\text{rep}} = 2d(|i-2|+1)$, $t_{i3}^{\text{rep}} = 2d(|i-3|+1)$, $t_{i4}^{\text{rep}} = 2d(|i-4|+1)$ and $t_{i5}^{\text{rep}} = 2d(|i-5|+2)$. The expected response time is modeled by Equation (2).

Equalized Leadership Probabilities: Let's now estimate the ranges of the random election timeouts that equalize the leadership probabilities. Figure 4(a) depicts the transition and leadership probabilities for various delays $d \in [0, 0.03]$, when all random parts t_i^{out} of the election timeouts are sampled from the same range $[0,1]$ ($\alpha_i = 1$), as it is suggested by the Raft designers. Delays $d \in [0, 0.03]$ conform to the constraint $4d \ll t_o^{\text{out}}$, assuming that $t_o^{\text{out}} = 1$. Especially for the delays $d = 0.015$ and 0.025, the corresponding probabilities are marked in all relative figures, since these two cases will be further investigated in our experimentation.

As a proof of concept that leadership probabilities can be adjusted as we wish, even for minimizing the expected response time for any given $\boldsymbol{\lambda}$, we compute the ranges $[0, \alpha_i]$ of the random part t_i^{out} of the election timeouts that equalize the leadership probabilities. In particular, for each delay $d \in [0, 0.03]$ that is integral multiple of 0.001, exhaustive search for the minimum sum of absolute differences between the leadership probabilities ($|\pi_1 - \pi_2| + |\pi_1 - \pi_3|$) is done. The area of search is the 2-dimensional discrete space produced by α_1 and α_2 by increasing them from 0 to 1 with step 0.001. It is assumed that $\alpha_3 = 1$, as well as that $\alpha_4 = \alpha_2$ and $\alpha_5 = \alpha_1$ because of the symmetric positions of the corresponding nodes. Using polynomial modeling, the relationships of the equalizing α_i with d are expressed as $\alpha_5 = \alpha_1 = 0.99 - 11.47d$ ($R^2 = 99.7\%$) and $\alpha_4 = \alpha_2 = 0.99 - 4.29d$ ($R^2 = 99.4\%$), assuming that $\alpha_3 = 1$. Figure 4(b) shows the transition probabilities and the almost equal leadership probabilities, when the ranges of the random part of the election timeouts are given by these linear functions of delay d.

V. Raft Over Network (Assuming Long-Term Failures)

In this section, the main difference compared with the previous Section IV is that leadership transitions happen due to *long-term failures* of the leaders. Failures are called long-term when the leader goes down and it is up again after the new leader has been elected and before a new failure occurs. Thus, there is no possibility of the failed leader to be also the successive leader.

Leadership Probabilities: Assuming again a cluster $\mathscr{N}$ with N nodes, after the long-term failure of leader $n_l \in \mathscr{N}$, $n_i \neq n_l$ is voted by itself and at least a cluster subset $\mathscr{N}' \subset \mathscr{N} - \{n_l, n_i\}$, if Equation (7) holds for $\mathscr{N}'$ with probability given by Equation (9). However, there are two main differences in this case. One difference is that cluster subset $\mathscr{N}'$, although it has the same cardinality with before and equal to $\lceil (N-1)/2 \rceil$, belongs to a new family of sets over $\mathscr{N} - \{n_l, n_i\}$, noted as $\mathscr{N}^{-n_l,n_i}$, which is a subset of the previous larger family of sets $\mathscr{N}^{-n_i}$. This happens because the failed leader n_l does not vote for the successive leader. The second difference is that the election winner after a split vote is any other node apart from the failed leader.

In case of no split vote, the transition probability from n_l to n_i is given by the modified Equation (10), that is

$$p_{li}^{\text{no-split}} = \Pr\left[\bigcup_{\mathscr{N}' \in \mathscr{N}^{-n_l,n_i}} \{ n_i \ v.l.\ \{n_i\} \cup \mathscr{N}' \} \right].$$

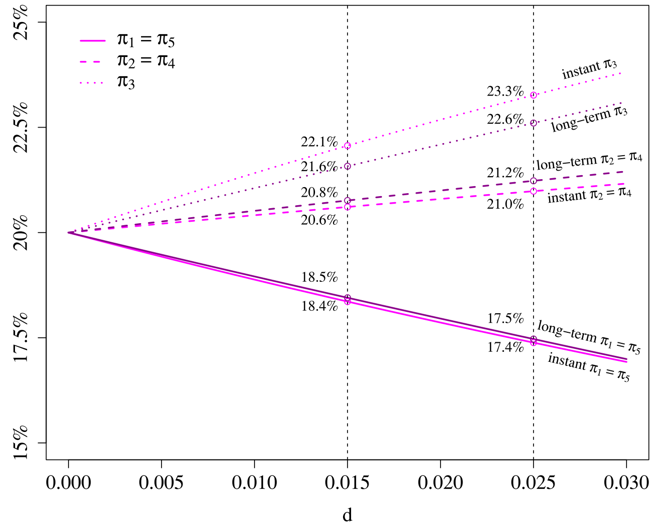


Fig. 5. Leadership percent probabilities $\boldsymbol{\pi}$ over the 5-node bus network using equal ranges of the election timeouts and assuming either instant or long-term failures.

Then, the split vote probability is

$$p_l^{\text{split}} = 1 - \sum_{n_i \neq n_l} p_{li}^{\text{no-split}}$$

and the transition probability from n_l to n_i is

$$\begin{aligned} p_{li} &= p_{li}^{\text{no-split}} + p_l^{\text{split}}\left(p_{li}^{\text{no-split}} + p_l^{\text{split}}\left(p_{li}^{\text{no-split}} + \cdots\right)\right) \\ &= p_{li}^{\text{no-split}}\left(1 + p_l^{\text{split}} + \left(p_l^{\text{split}}\right)^2 + \cdots\right) = \frac{p_{li}^{\text{no-split}}}{1 - p_l^{\text{split}}}, \end{aligned}$$

since the election is repeated after each split vote with probability p_l^{split} and node n_i wins each of the repeated elections with probability $p_{li}^{\text{no-split}}$. The transition probability matrix $\mathbf{P}$ is given by Equation (12) with $p_{ll} = 0$, while the leadership probabilities $\boldsymbol{\pi}$ are modeled again by the eigenvector of $\mathbf{P}$ with eigenvalue equal to one.

Response Time: No changes between the two types of failure, regarding the response time.

Equalized Leadership Probabilities Over the 5-Node Bus Network: The same exercise is repeated here for showcasing the usage of the new model, searching again the election timeout ranges that equalize the leadership probabilities over the 5-node bus network. Figure 5 depicts the leadership probabilities for the same delays $d \in [0, 0.03]$, when the random parts t_i^{out} of the election timeouts are sampled from the same range [0, 1] ($\alpha_i = 1$) and the failures are either instant (pink lines) or long-term (deep-purple lines). Obviously, the probabilities of each failure type are almost equal between them, thus both models for leadership probabilities can be used without considering if all failures are instant or long-term. After repeating the same search with the one in Section IV-C, the relationship between the equalizing α_i and delay d are $\alpha_5 = \alpha_1 = 1 - 10.13d$ ($R^2 = 99.98\%$) and $\alpha_4 = \alpha_2 = 1 - 2.70d$ ($R^2 = 99.97\%$), assuming that $\alpha_3 = 1$.

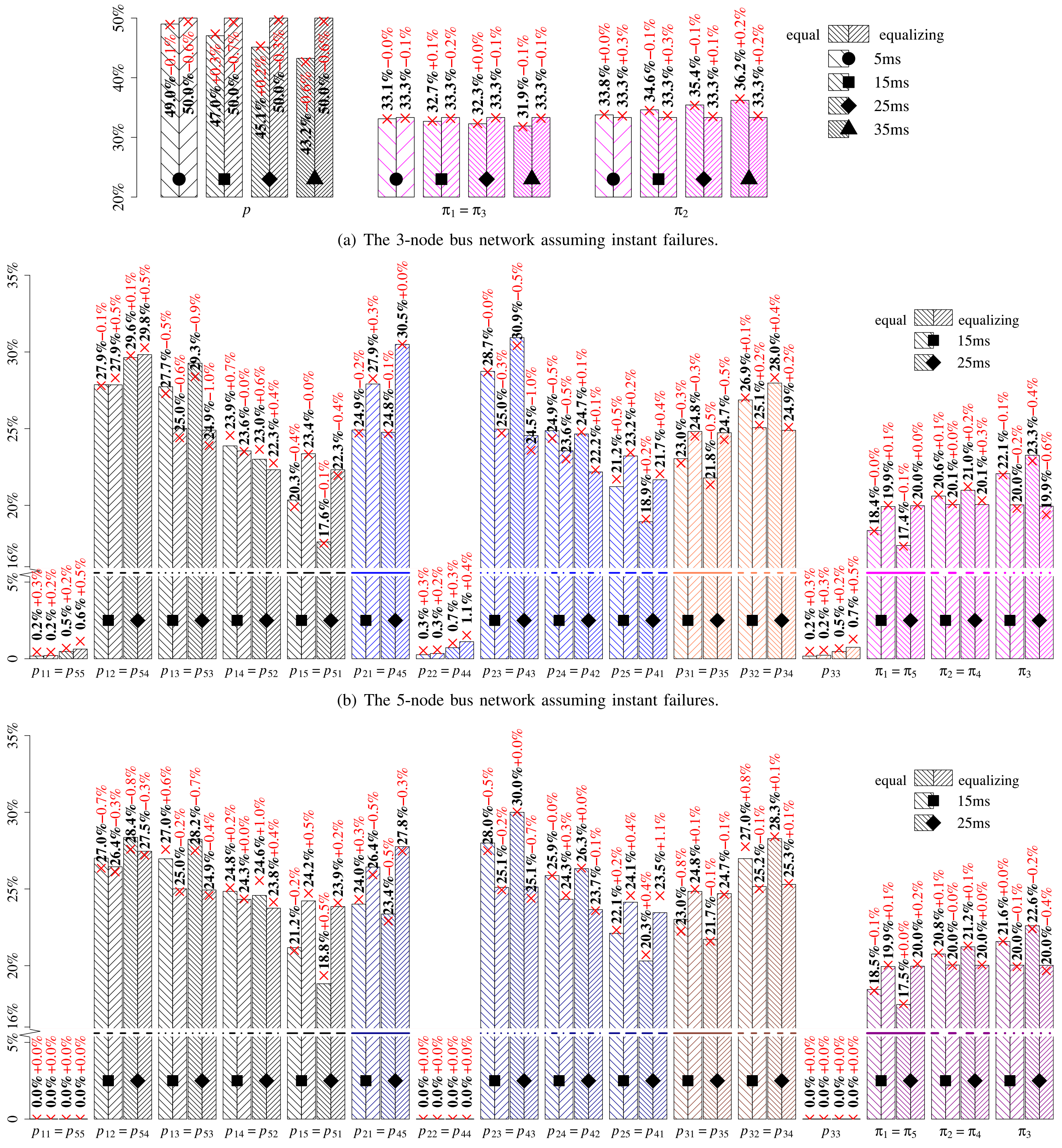
VI. Testbed Experimentation

Extended experimentation has been performed over the NITOS testbed [21]. A distributed key-value data store is built using the open-source Raft implementation of etcd [9]. More specifically, *raftexample* is used, which is an example usage of the Raft library of etcd. According to etcd version 3.3.0−rc.0, time is slotted and the heartbeat/vote requests are sent when slots start. The election timeouts are integral multiples of slot and the slot duration is 100 milliseconds. The fixed minimum election timeout is 10 slots, thus $t_o^{\text{out}} = 1$ second. The random part of the election timeout, by default, is uniformly sampled between 0 and 9 slots, which is the discretized form of the continuous range [0, 1] second. In the following experiments, Raft is executed over the 3-node and 5-node bus networks and all probabilities and response times of our interest are measured and compared to the results of the already presented models.

Leadership Probabilities Over the 3-Node Bus Network, Assuming Instant Failures: The instant failures are emulated by modifying the code of raftexample, in order each leader to send only 8 heartbeats. Followers assume that leader has failed after sending its 8^{th} heartbeat, but it is always up and votes for the successive leader, as it would happen after an instant failure. The failures are repeated more than 40,000 times. The transition and leadership probabilities are measured by tracking the etcd output and counting how many times each node has been leader and successive leader of another one that has failed. Moreover, random delays sampled between 0 and 99 milliseconds are introduced before nodes send their requests, emulating the fact that the node timers are not necessarily synchronized in a real geo-distributed cluster.

Figure 6(a) presents the transition and leadership probabilities of a cluster operating over the 3-node bus network, when delay *d* corresponds to either 5, 15, 25 or 35 milliseconds. The red ‘x’-marks depict the *measured (through experimentation)* probabilities, the bars illustrate their *theoretically expected* values and the red numbers show their differences. The right bar of each couple corresponds to *equal* election timeout ranges, while the left bar corresponds to ranges that *equalize* the leadership probabilities. The line density and symbol of each couple corresponds to a delay given by the legend. The measured value of *p* is the average of the measured p_{13} and p_{31}, which are very close to each other and theoretically expected to be equal due to the network symmetry. The same holds for the presented as experimentally measured $\pi_1 = \pi_3$ that is the average of the measured π_1 and π_3. The measured values of all probabilities are very close to their theoretically expected values.

When nodes use equal ranges ($\alpha = \alpha_2 = 1$), π_2 increases and $\pi_1 = \pi_3$ decreases with *d*. On the other hand, when nodes use the equalizing ranges given at the end of Section IV-A ($\alpha = 1 - 4d$ and $\alpha_2 = 1$), π_2 and $\pi_1 = \pi_3$ are always equal. The absolute differences between the measured and the theoretically expected values of all leadership probabilities are less than 0.3%. It is worth mentioning that delay *d* should be expressed in seconds, as the rest time variables do, thus 5 milliseconds correspond to delay d = 0.005. For a delay



(a) The 3-node bus network assuming instant failures.

(b) The 5-node bus network assuming instant failures.

(c) The 5-node bus network assuming long-term failures.

Fig. 6. Transition and leadership percent probabilities over the 3-node and 5-node bus networks assuming either instant or long-term failures. The left and right bar of each couple correspond respectively to equal and equalizing (regarding leadership probabilities) election timeout ranges. Couples with different density and symbol correspond to different delays. The bar heights and the black-bold numbers show the theoretically expected values, the red 'x'-marks depict the measured (through experimentation) values and the red numbers give the differences between experimentation and theory.

equal to 25 milliseconds ($d = 0.025$) and $\alpha = 1 - 4 \cdot 0.025 = 0.9$ second or 9 slots, the random part of the election timeout is sampled between 0 and 8 slots. For α that is not integral multiple of slot, various integral multiples are used from election to election as upper bounds of the random part, so that their average is equal to α. For example, if delay is 5 milliseconds and $\alpha = 1 - 4 \cdot 0.005 = 0.98$ second or 9.8 slots, 1 second is used as upper bound four times more than 0.9 second.

Leadership Probabilities Over the 5-Node Bus Network Assuming Instant Failures: Figure 6(b) shows the respective results over the 5-node bus network, for delay d corresponding to either 15 or 25 milliseconds. The theoretically expected and black-written values, on top of the bars, are the same with the ones already presented in Figure 4. In addition, the types and colors of the lines in the middle of the bars indicate the corresponding lines of Figure 4. The experimentation results are grouped again, presenting the averages of some couples (e.g., the averages of p_{15} and p_{51} or π_1 and π_5), since the probabilities of each of these couples are almost equal, as it is theoretically expected. The inequality among the leadership probabilities increases with the delay, when all nodes use equal ranges (all $\alpha_i = 1$). On the contrary, the leadership probabilities are almost equal for both delays, when nodes use the equalizing ranges given at the end of Section IV-C ($\alpha_5 = \alpha_1 = 0.99 - 11.47d$, $\alpha_4 = \alpha_2 = 0.99 - 4.29d$ and $\alpha_3 = 1$). The absolute differences between experimentation and theory regarding the leadership probabilities are less than 0.6%. Once again, when the upper bounds of the random part of the election timeouts are not integral multiples of slot, the same approach is used with the 3-node bus network.

Leadership Probabilities Over the 5-Node Bus Network Assuming Long-Term Failures: Figure 6(c) shows the respective results over the 5-node bus network, assuming long-term failures. Deeper colours are used for indicating the different failure type, as it happens in Figure 5 that probabilities of the long-term failures are colored deeper than the ones of the instant failures. In this round of experiments, the code is modified to disable leaders replying to vote requests. In this way, the long-term failures are emulated and repeated as before measuring again the probabilities of our interest. The results regarding the distribution of the leadership probabilities are the same with before, when assuming instant failures, either with usage of equal ranges (all $\alpha_i = 1$) or the equalizing ranges given at the end of Section V ($\alpha_5 = \alpha_1 = 1 - 10.13d$, $\alpha_4 = \alpha_2 = 1 - 2.70d$ and $\alpha_3 = 1$). The absolute differences between experimentation and theory regarding the leadership probabilities are less than 0.4%.

Response Time of the 3-Node Bus Network: The last round of the presented experiments refers to the response time. The code is modified once again, changing the *HTTP PUT* requests sent to the Raft nodes to be blocking and return only after the inserted value is updated at the receiving node. In this way, the time spent in each request is proportional to the response time, which is measured with Postman [22]. Requests are repeated for more than 20,000 times and the measured response time is the average of these repeats. The elections are repeatedly triggered due to instant failures and the requests are sent to either a central (n_2) or an edge (n_1) node of the 3-node bus network (results for n_3 are the same with the ones for n_1). The *average response times* are presented in Table II, as well as the theoretically *expected response times*, which are inside the parentheses.

TABLE II
RESPONSE TIME (IN MILLISECONDS) OVER THE 3-NODE BUS NETWORK

delay (milliseconds)	5	15	25	35
$\lambda_1 = 1$, equal ranges	72 (20)	112 (60)	154 (100)	193 (140)
$\lambda_1 = 1$, eq/ing ranges	70 (20)	111 (60)	151 (100)	192 (140)
$\lambda_2 = 1$, equal ranges	70 (17)	104 (50)	133 (82)	170 (115)
$\lambda_2 = 1$, eq/ing ranges	67 (17)	101 (50)	139 (83)	169 (117)

TABLE III
RESPONSE TIME (IN MILLISECONDS) OVER THE 5-NODE BUS NETWORK

delay (milliseconds)	15	25
$\lambda_1 = 1$, equal ranges	152 (101)	218 (167)
$\lambda_1 = 1$, eq/ing ranges	159 (102)	221 (170)
$\lambda_2 = 1$, equal ranges	135 (82)	185 (135)
$\lambda_2 = 1$, eq/ing ranges	144 (84)	194 (140)
$\lambda_3 = 1$, equal ranges	130 (75)	182 (123)
$\lambda_3 = 1$, eq/ing ranges	130 (78)	189 (130)

If only n_1 receives request ($\lambda_1 = 1$ and $\lambda_2 = \lambda_3 = 0$), under either equal or equalizing ranges for all nodes, the expected response time is $t^{\text{rep}} = t_1^{\text{rep}} = \pi_1 t_{11}^{\text{rep}} + \pi_2 t_{12}^{\text{rep}} + \pi_3 t_{13}^{\text{rep}} = 2d\pi_1 + 4d\pi_2 + 6d\pi_3 = 4d(\pi_1 + \pi_2 + \pi_3) = 4d$, since $\pi_1 = \pi_3$ in both cases. This is also verified by the first two table rows with $\lambda_1 = 1$, where the average response time is the same for using both equal and equalizing ranges, and it is actually 4 times the delay plus 50 to 60 milliseconds needed for processes not related to Raft. On the other hand, if only n_2 receives requests ($\lambda_2 = 1$ and $\lambda_1 = \lambda_3 = 0$), the expected response time is $t^{\text{rep}} = t_2^{\text{rep}} = \pi_1 t_{21}^{\text{rep}} + \pi_2 t_{22}^{\text{rep}} + \pi_3 t_{23}^{\text{rep}} = 4d\pi_1 + 2d\pi_2 + 4d\pi_3 = 2d(\pi_1+\pi_2+\pi_3)+2d(\pi_1+\pi_3) = 2d(1+2\pi_1)$, since $\pi_1 = \pi_3$, which turns to $2d(1 + 2/3) = 10d/3$ under equalizing ranges and $\pi_1 = 1/3$. Under equal ranges, using Equation (4), the expected response time is $2d(1+2/(3+4d-4d^2)) \approx 10d/3$ for all tested delays. The last two table rows with $\lambda_2 = 1$ confirm experimentally that the average response time is approximately $10/3$ times the delay, plus 50 to 60 milliseconds as before.

Response Time of the 5-Node Bus Network: Table III presents the average response times and their theoretically expected values over the 5-node bus network. For example, in the first row and first column of this table, where all requests go to n_1 ($\lambda_1 = 1$ and rest $\lambda_i = 0$) and all election timeout ranges are equal, the expected response time is $t^{\text{rep}} = t_1^{\text{rep}} = \pi_1 t_{11}^{\text{rep}} + \pi_2 t_{12}^{\text{rep}} + \pi_3 t_{13}^{\text{rep}} + \pi_4 t_{14}^{\text{rep}} + \pi_5 t_{15}^{\text{rep}} = 4d\pi_1 + 4d\pi_2 + 6d\pi_3 + 8d\pi_4 + 12d\pi_5 = 16d\pi_1 + 12d\pi_2 + 6d\pi_3 = 6d + 4d\pi_1$, which turns to $6 \cdot 15 + 4 \cdot 15 \cdot 18.4\% = 101$ milliseconds for delay equal to 15 milliseconds ($d = 0.015$). Probability $\pi_1 = 18.4\%$, as it is shown in Figure 6(b). The measured average response time is again increased by 50 to 60 milliseconds. This difference is almost fixed for all table cells, which validates that the time needed for the Raft operations is approximately the theoretically expected one.

VII. CONCLUSION & FUTURE WORK

In this paper, models are presented for estimating the leadership probabilities and the response time of a Raft-operated cluster over a network. The models are used in order for the ranges of the random election timeouts to be adjusted, such as the leadership probabilities and the expected response time to be the desired, e.g., the latter one to be minimized. The models are specified for a 3-node and a 5-node bus network, which both are simple cases pointing out the logic behind the models. The capabilities of manipulating the leadership probabilities, using these models, are presented by equalizing them over both bus networks. Over all tested cases, the aimed leadership probabilities are achieved, since the deviation between the experimentation data and the theoretical estimations is always less than the negligible 0.6%. As for the response time, the corresponding differences are negligible again, since the 50 to 60 milliseconds extra delay is not related to the Raft protocol.

A main issue that we need to consider in our future work is to evaluate the presented models over network topologies that are used in real distributed systems. Moreover, the effect of the limited processing capabilities of the system nodes, as well as the existence of various transmission rates and non-zero packet loss ratios, should be considered in the models formulation.

Appendix
The Transition Probabilities of Section IV-C

After the failure of leader n_1, n_5 is the successive leader, if it is voted at least by itself, n_3 and n_4, thus

$$\begin{aligned} p_{15} &= p_{15}^{\{n_3,n_4\}} \\ &= \frac{1}{\alpha_5}\int_0^{\alpha_5} \bar{F}\left(\frac{\tau_5+4d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_5+4d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_5+2d}{\alpha_4}\right)d\tau_5. \end{aligned}$$

Otherwise, n_4 **is the successive leader**, if it is voted at least by itself and either n_2, n_3 or n_3, n_5, thus

$$\begin{aligned} p_{14} &= p_{14}^{\{n_2,n_3\}} + p_{14}^{\{n_3,n_5\}} - p_{14}^{\{n_2,n_3,n_5\}} \\ &= \frac{1}{\alpha_4}\int_0^{\alpha_4} \bar{F}\left(\frac{\tau_4+4d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_4+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_4-2d}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_4+2d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_4+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_4}{\alpha_5}\right) \\ &\quad - \bar{F}\left(\frac{\tau_4+4d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_4+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_4}{\alpha_5}\right)d\tau_4. \end{aligned}$$

Similarly, n_3 **is the successive leader**, if it is voted at least by itself and either n_1, n_2 or n_4, n_5. The event that n_3 is voted at least by itself and n_2, n_4 is subset of the event that is voted at least by itself and n_1, n_2. This is true, since there is no way n_3 to be voted by n_2 and not be voted by n_1. If the vote request of n_3 reaches n_2 before the requests of the other potential candidates n_4 and n_5, then this request is also the first one received by n_1 that does not have enough time to transition to candidate after it has failed. Thus,

$$\begin{aligned} p_{13} &= p_{13}^{\{n_1,n_2\}} + p_{13}^{\{n_4,n_5\}} - p_{13}^{\{n_1,n_2,n_4,n_5\}} \\ &= \frac{1}{\alpha_3}\int_0^{\alpha_3} \bar{F}\left(\frac{\tau_3+2d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_3-2d}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3-4d}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_3}{\alpha_2}\right)\bar{F}\left(\frac{\tau_3}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3}{\alpha_5}\right) \\ &\quad - \bar{F}\left(\frac{\tau_3+2d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_3}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3}{\alpha_5}\right)d\tau_3. \end{aligned}$$

Finally, n_2 **is the successive leader**, if it is voted at least by itself, n_1 and n_3, thus

$$\begin{aligned} p_{12} &= p_{12}^{\{n_1,n_3\}} \\ &= \frac{1}{\alpha_2}\int_0^{\alpha_2} \bar{F}\left(\frac{\tau_2}{\alpha_3}\right)\bar{F}\left(\frac{\tau_2-2d}{\alpha_4}\right)\bar{F}\left(\frac{\tau_2-4d}{\alpha_5}\right)d\tau_2. \end{aligned}$$

After leader n_2 has failed, n_5 is the successive leader, if it is voted at least by itself, n_3 and n_4, thus

$$\begin{aligned} p_{25} &= p_{25}^{\{n_3,n_4\}} \\ &= \frac{1}{\alpha_5}\int_0^{\alpha_5} \bar{F}\left(\frac{\tau_5+2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_5+4d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_5+2d}{\alpha_4}\right)d\tau_5. \end{aligned}$$

Otherwise, n_4 **is the successive leader**, if it is voted at least by itself and either n_2, n_3 or n_3, n_5, thus

$$\begin{aligned} p_{24} &= p_{24}^{\{n_2,n_3\}} + p_{24}^{\{n_3,n_5\}} - p_{24}^{\{n_2,n_3,n_5\}} \\ &= \frac{1}{\alpha_4}\int_0^{\alpha_4} \bar{F}\left(\frac{\tau_4}{\alpha_1}\right)\bar{F}\left(\frac{\tau_4+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_4}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_4+2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_4+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_4-2d}{\alpha_5}\right) \\ &\quad - \bar{F}\left(\frac{\tau_4+2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_4+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_4}{\alpha_5}\right)d\tau_4. \end{aligned}$$

Similarly, n_3 **is the successive leader**, if it is voted at least by itself and either n_1, n_2 or n_2, n_4 or n_4, n_5, thus

$$\begin{aligned} p_{23} &= p_{23}^{\{n_1,n_2\}} + p_{23}^{\{n_2,n_4\}} + p_{23}^{\{n_4,n_5\}} - p_{23}^{\{n_1,n_2,n_4\}} \\ &\quad - p_{23}^{\{n_1,n_2,n_4,n_5\}} - p_{23}^{\{n_2,n_4,n_5\}} + p_{23}^{\{n_1,n_2,n_4,n_5\}} \\ &= p_{23}^{\{n_1,n_2\}} + p_{23}^{\{n_2,n_4\}} + p_{23}^{\{n_4,n_5\}} - p_{23}^{\{n_1,n_2,n_4\}} \\ &\quad - p_{23}^{\{n_2,n_4,n_5\}} \\ &= \frac{1}{\alpha_3}\int_0^{\alpha_3} \bar{F}\left(\frac{\tau_3+2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_3-2d}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3-4d}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_3}{\alpha_1}\right)\bar{F}\left(\frac{\tau_3}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3-2d}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_3-2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_3}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3}{\alpha_5}\right) \\ &\quad - \bar{F}\left(\frac{\tau_3+2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_3}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3-2d}{\alpha_5}\right) \\ &\quad - \bar{F}\left(\frac{\tau_3}{\alpha_1}\right)\bar{F}\left(\frac{\tau_3}{\alpha_4}\right)\bar{F}\left(\frac{\tau_3}{\alpha_5}\right)d\tau_3. \end{aligned}$$

Finally, n_1 **is the successive leader**, if it is voted at least by itself, n_2 and n_3, thus

$$\begin{aligned} p_{21} &= p_{21}^{\{n_2,n_3\}} \\ &= \frac{1}{\alpha_1}\int_0^{\alpha_1} \bar{F}\left(\frac{\tau_1+2d}{\alpha_3}\right)\bar{F}\left(\frac{\tau_1}{\alpha_4}\right)\bar{F}\left(\frac{\tau_1-2d}{\alpha_5}\right)d\tau_1. \end{aligned}$$

After n_3 has failed, n_1 is the successive leader, if it is voted at least by itself, n_2 and n_3, thus

$$\begin{aligned} p_{31} &= p_{31}^{\{n_2,n_3\}} \\ &= \frac{1}{\alpha_1}\int_0^{\alpha_1} \bar{F}\left(\frac{\tau_1+2d}{\alpha_2}\right)\bar{F}\left(\frac{\tau_1+2d}{\alpha_4}\right)\bar{F}\left(\frac{\tau_1}{\alpha_5}\right)d\tau_1. \end{aligned}$$

Otherwise, n_2 **is the successive leader**, if it is voted at least by itself and either n_1, n_3 or n_3, n_4, thus

$$\begin{aligned} p_{32} &= p_{32}^{\{n_1,n_3\}} + p_{32}^{\{n_3,n_4\}} - p_{32}^{\{n_1,n_3,n_4\}} \\ &= \frac{1}{\alpha_2}\int_0^{\alpha_2} \bar{F}\left(\frac{\tau_2}{\alpha_1}\right)\bar{F}\left(\frac{\tau_2}{\alpha_4}\right)\bar{F}\left(\frac{\tau_2-2d}{\alpha_5}\right) \\ &\quad + \bar{F}\left(\frac{\tau_2-2d}{\alpha_1}\right)\bar{F}\left(\frac{\tau_2+2d}{\alpha_4}\right)\bar{F}\left(\frac{\tau_2}{\alpha_5}\right) \\ &\quad - \bar{F}\left(\frac{\tau_2}{\alpha_1}\right)\bar{F}\left(\frac{\tau_2+2d}{\alpha_4}\right)\bar{F}\left(\frac{\tau_2}{\alpha_5}\right) d\tau_2. \end{aligned}$$

Obviously, due to the bus symmetry, $p_{ij} = p_{ji}, \forall n_i, n_j \in \mathcal{N}$.

Acknowledgment

The European Union and its agencies are not liable or otherwise responsible for the contents of this document; its content reflects the view of its authors only.

References

[1] K. Choumas and T. Korakis, "When raft meets SDN: How to elect a leader over a network," in *Proc. IEEE NetSoft*, Ghent, Belgium, 2020, pp. 140–141.

[2] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 333–354, 1st Quart., 2018.

[3] J. Medved, R. Varga, A. Tkacik, and K. Gray, "OpenDaylight: Towards a model-driven SDN controller architecture," in *Proc. IEEE WoWMoM*, Sydney, NSW, Australia, 2014, pp. 1–6.

[4] P. Berde *et al.*, "ONOS: Towards an open, distributed SDN OS," in *Proc. HotSDN*, 2014, pp. 1–6.

[5] F. Esposito *et al.*, "Necklace: An architecture for distributed and robust service function chains with guarantees," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 1, pp. 152–166, Mar. 2021.

[6] B. Burns, B. Grant, D. Oppenheimer, E. Brewer, and J. Wilkes, "Borg, omega, and Kubernetes: Lessons learned from three container-management systems over a decade," *ACM Queue*, vol. 14, no. 1, pp. 70–93, 2016.

[7] O. Sefraoui, M. Aissaoui, and M. Eleuldj, "OpenStack: Toward an open-source solution for cloud computing," *Int. J. Comput. Appl.*, vol. 55, no. 3, pp. 38–42, 2012.

[8] "Hyperledger Fabric: Distributed Ledger Software." Accessed: Feb. 3, 2022. [Online]. Available: https://www.hyperledger.org/use/fabric

[9] "etcd: Distributed, Reliable Key-Value Store for the Most Critical Data of a Distributed System." Accessed: Feb. 3, 2022. [Online]. Available: https://etcd.io/

[10] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX ATC*, 2014, pp. 305–320.

[11] "The Raft Consensus Algorithm." Accessed: Feb. 3, 2022. [Online]. Available: https://raft.github.io/

[12] H. Howard, M. Schwarzkopf, A. Madhavapeddy, and J. Crowcroft, "Raft refloated: Do we have consensus?" *ACM SIGOPS Oper. Syst. Rev.*, vol. 49, no. 1, pp. 12–21, 2015.

[13] K. Choumas, D. Giatsios, P. Flegkas, and T. Korakis, "The SDN control plane challenge for minimum control traffic: Distributed or centralized?" in *Proc. IEEE CCNC*, Las Vegas, NV, USA, 2019, pp. 1–7.

[14] M. Karatisoglou, K. Choumas, and T. Korakis, "Controller placement for minimum control traffic in OpenDaylight clustering," in *Proc. IEEE WF-5G*, Dresden, Germany, 2019, pp. 353–358.

[15] Y. Zhang, B. Han, Z.-L. Zhang, and V. Gopalakrishnan, "Network-assisted raft consensus algorithm," in *Proc. SIGCOMM Posters Demos*, 2017, pp. 94–96.

[16] E. Sakic and W. Kellerer, "Response time and availability study of RAFT consensus in distributed SDN control plane," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 304–318, Mar. 2018.

[17] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "Efficient controller placement and reelection mechanism in distributed control system for software defined wireless sensor networks," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 6, p. e3588, 2019.

[18] R. Hanmer, L. Jagadeesan, V. Mendiratta, and H. Zhang, "Friend or foe: Strong consistency vs. overload in high-availability distributed systems and SDN," in *Proc. ISSREW*, Memphis, TN, USA, 2018, pp. 59–64.

[19] C. Fluri, D. Melnyk, and R. Wattenhofer, "Improving raft when there are failures," in *Proc. LADC*, 2018, pp. 167–170.

[20] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the raft consensus algorithm for private blockchains," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 1, pp. 172–181, Jan. 2020.

[21] "Network Implementation Testbed using Open Source platforms (NITOS)." Accessed: Feb. 3, 2022. [Online]. Available: https://nitlab.inf.uth.gr/NITlab/nitos

[22] "Postman: API Platform." Accessed: Feb. 3, 2022. [Online]. Available: https://www.postman.com/

Kostas Choumas received the Diploma and M.S. degrees in electrical and computer engineering from the University of Thessaly, Volos, Greece, in 2007 and 2008, respectively, and the Ph.D. degree from the University of Thessaly in 2015, under the supervision of Asst. Prof. T. Korakis and Prof. L. Tassiulas. Since 2015, he has been a Postdoctoral Associate with the University of Thessaly. His research interests include the design and implementation of enhanced algorithms on wireless and software defined networks, as well as the experimentation on these algorithms performance using testbed platforms.

Thanasis Korakis (Member, IEEE) received the B.S. and M.S. degrees in computer science from the University of Athens in 1994 and 1997, respectively, and the Ph.D. degree in electrical engineering from the University of Thessaly, Greece, in 2005. In Summer of 2004, he was a Visiting Researcher with the CSE Department, University of California at Riverside. From 2005 to 2006, he was a Research Scientist with the ECE Department, Polytechnic University, NY, USA, where he was a Research Assistant Professor from 2006 to 2012. He is currently an Associate Professor with the ECE Department, University of Thessaly. His interests are in the networking field with an emphasis on access layer protocols, cooperative networks, quality-of-service provisioning, network management, and experimental platforms. He has received several awards, including the Best Paper Awards in WiNTECH 2013, GREE 2013, and CloudComp 2015. From 2007 to 2012, he was a Voting Member of the IEEE 802.16 Standardization Group.