Measuring Robustness of SDN Control Layers

Yury Jiménez, Juan Antonio Cordero, and Cristina Cervelló-Pastor

***Abstract*—The controller placement problem remains a key aspect of Software Defined Networking (SDN). The selection of a suboptimal controller may impact severely the performance of the control layer and, in consequence, cause a substantial degradation in the data layer. Different approaches for controller placement have been proposed with various objectives in mind, but most of them do not consider the characteristics of the resulting control layer in terms of robustness. In this paper we propose and formalize a complete metric for estimating robustness of a SDN control layer, and we evaluate two heuristics for controller selection and control layer construction: *Fast Failover* and a simplified version of *k-Critical*. The results of the performed evaluation indicate that the control layer topology induced by *k-Critical* is less prone to failures, more robust and more homogenous than those computed by *Fast Failover*.**

***Index Terms*—Software Defined Networking (SDN), controller placement, data layer, control layer, robustness.**

I. Introduction

Software Defined Networking (SDN) emerges as a solution for simplifying management and operation in large or complex networks, by decoupling control and data planes in the network, typically implemented in *routers* in traditional networks. Physical separation between the control layer (handled by *controllers*) and the data layer (implemented by *nodes* or *switches*) in SDNs implies that reliability of the communication between devices working at each layer needs to be addressed specifically, as any network failure may cause a disconnection in the forwarding layer between a controller and a node, and thus lead to a severe data packet loss and performance degradation [1] [2]. Therefore, one of the most important aspects when building a control layer is protection against failures.

In a SDN environment, the performance and characteristics of the control layer are affected by both the controller placement and the underlying network connectivity. Given that, in essence, a control layer is a shortest path tree rooted at the controller, the position of the controller(s) is the main optimization variable for a given SDN – in particular, in terms of control layer robustness as it can affect the SDN network ability to respond rapidly to network events, limiting the availability, convergence time and protection against failures [3] [4] [5] [6]. Through the delay-constrained shortest paths, henceforth called control channels, the controllers collect network state information from nodes, and decide and distribute the forwarding decisions to them [7].

Heller *et al.* [8] evaluate the controller placement for WANs, where latency dominates. Different controller placement are compared in terms of node-to-controller latency, for the fundamental limits imposed on reaction delay, fault discovery, and event propagation efficiency. Authors conclude that a single controller is often sufficient to meet existing reaction-time requirements in WANs. This analysis is extended by Hock *et al.* [3], who present a framework for evaluation of different resilience metrics (*e.g.*, node-to-controller latency, failure tolerance, load balancing, inter-controller latency), in order to enable selection of the optimal controller placement; authors conclude that optimal values for the examined resilience metrics are often impossible to achieve simultaneously. The objective of *Fast Failover* [4] is to minimize the data loss caused by a node or link failure in the control layer by considering a notion of local node protection (against failures in parents). The node that maximizes the number of locally-protected nodes is thus selected as controller (see section III-A). *K-Critical* [5] explores a controller placement heuristic that balances controller-node communication latency and path length minimization objectives, and selects k controllers under some given requirements, *e.g.*, maximum delay between the controller and nodes. Resulting trees tend to be wide near the controllers and narrow further away from them, where paths have as few hops as possible in order to reduce the data loss when nodes or links fail. As shown in these works [2-6], the controller selection has a deep impact in both cases, in the robustness of the control layer topology (*e.g.*, robustness) and therefore the resilience of the underlying SDN.

In this paper we propose a robustness metric and evaluate the robustness of the control layers built with *Fast Failover* and *k-Critical* controller placement heuristics. Section II describes the network model and introduces the main elements of the notation used throughout the paper. Section III discusses the notion of robustness, defines different parameters to measure the robustness of a control layer in a SDN and details the procedure to compute them. Section IV focuses on the evaluation of both algorithms. Section V concludes the paper.

II. Network model and notation

Consider a SDN modeled by a graph $G = (V, E, \delta)$, where $V = \{1, \ldots, n\}$ is the set of vertices in the graph (set of nodes in the network), E is the set of edges in the graph (links between nodes in the network), and $\delta : E \longrightarrow \mathbb{R}^+$ maps a link $e \in E$ with $\delta(e)$, the delay associated to link e. Let N be the number of nodes in the network ($N = |V|$). The control layer or management tree induced by the selection of a

Yury Jiménez and Cristina Cervelló-Pastor are with the Department of Telematics Engineering, Universitat Politècnica de Catalunya (UPC), C/ Esteve Terradas, 7, 08860, Castelldefels, Spain, e-mails: {yury.jimenez;cristina}@entel.upc.edu.

Juan Antonio Cordero is with ICTEAM, Université catholique de Louvain (UCLouvain), Place Sainte Barbe, 2, B-1348, Louvain-la-Neuve, Belgium, e-mail: juan.cordero@uclouvain.be.

This work has been supported by the Government of Catalonia through a predoctoral FI scholarship, the Government of Spain through project TEC2013-47960-C4-1-P; and the Belgian Science Policy Office (BSPO) through the project IAP BESTCOM.

controller $n \in V$ in a given SDN (with graph G) is a subgraph $T_n \subset G$; its set of vertices and edges is denoted by V_{T_n} and E_{T_n}, respectively. Given a node $x \in V$ and a controller n, $T_x \subset T_n$ will denote the subtree of T_n rooted at node x. We assume that nodes keep track of their set of neighbors by periodically exchanging $HELLO$ messages. Moreover, for a management tree T_n, each node $s \in V_{T_n}$ also knows its parent node in the tree, denoted by $p_{T_n}(s)$, and the adjacent nodes attached to it in the control layer (children).

III. Robustness metric

Intuitively, a network is robust when communication between two nodes is not severely affected by –or can efficiently recover from– network failures, thus minimizing data loss in these cases. In terms of network paths, this notion of robustness implies that (1) the probability of failures along existing paths in the control layer of a given SDN is low, meaning that the number of involved nodes in a control channel is small, and (2) in case of failure in a control layer, it is possible for affected nodes to reestablish communication by using additional, unaffected paths in the tree. For evaluating the robustness provided by a controller placement, we assume that such a reconnection algorithm is available in the network and makes every node able to detect failures and reconnect dynamically to any existing path at distance 1.

To evaluate the robustness of a control layer, we first use a resilience metric (section III-A) based on the fraction of locally protected nodes in the control layer [4]. While this is a valid first-order estimation of robustness, it has some drawbacks that derive from their implicit assumptions, as discussed in section III-B. We thus propose (section III-C) a new, more complete robustness metric, which takes into account the fraction of nodes potentially affected by network failures, and their ability to recover.

A. Robustness as 1-hop local node protection

In a first approximation, the robustness of a control layer T_n rooted at a given controller n can be measured as the expected fraction of *protected* (unaffected) nodes within the set of downstream nodes when one of them fail [4]. The underlying notion of node protection, hereafter denoted *1-hop local node protection*, can be formalized as follows. Let $p \equiv p_{T_n}(s)$ denote the parent of node $s \in V_{T_n}$ and T_p be the subtree rooted at node p, where $T_p \subset T_n$. Then, the 1-hop local protection of the node s is defined in Equation (1).

$$P(s) = \begin{cases} 0 & , \exists s' \in V, s' \notin T_p : (s, s') \in E \\ 1 & , \text{ otherwise} \end{cases} \quad (1)$$

According to this expression, s is considered *protected* if $P(s) = 0$, unprotected otherwise. Note that this protection metric is only effective, in practice, against failures in the parent node; but the fact that a node is *protected* according to this definition does not imply that it is resilient with respect to further failures in the corresponding tree, *e.g.* if the failure occurs at the 2-hop parent. Therefore, this protection metric is not sufficient to determine the network robustness with respect to failures. This metric provides an idea about how many nodes have a certain local (1-hop) protection, but is not informative about the effective level of protection and, hence, cannot be used to directly infer the number of nodes that can reestablish communication with the controller in case of a failure.

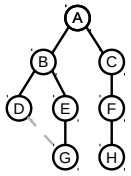

Fig. 1. Local protection in a control layer over a SDN.

B. Towards a network robustness metric

When there is a failure in a SDN, the number of affected nodes does not only depend on their local protection on the first hop (with respect to their parents), but on their (inner) protection with respect to the failing node and also on the protection provided by upper nodes affected by the failure. Consider the situation in Fig. 1. Note that G is a local 1-hop protected node, as defined in section III-A: if E fails, it can still get reconnected to the control layer via D, which is a neighbor of G in the network. However, G is not protected against a failure in B, given that its possible path through D also involves B. Note also that G would be, in practice, "protected" against a B failure if there was a link in the network between E and F (not shown in Figure 1) and its parent E could hence reconnect to the control layer via $\{F, C, A\}$ (A assumed to be controller).

C. Robustness as generalized node protection

In order to define a more accurate metric for control layer robustness, we introduce the notion of *generalized local node protection*, formalized in Equation (2). Given a controller n, a node $s \in T_x \subset T_n$ has *generalized local protection* against a failure in x if it has a connection with a node s' in the direction of the controller that is not in T_x.

$$P'(s) = \begin{cases} 0 & , \exists s' \in V, s' \notin T_x : (s, s') \in E \\ 1 & , \text{ otherwise} \end{cases} \quad (2)$$

In this context, given a control layer T_n (rooted at controller n), the nodes primarily affected by a failure in node x are the downstream nodes of x, that is, $|T_x|$. The fraction of these nodes that are locally protected against this failure is then:

$$\frac{1}{|T_x|} \sum_{s \in T_x} P'(s) \quad (3)$$

Given a controller n, any node s depending on a locally-protected node z against a failure in x ($z \in p(x, s)$) is also protected, as the reconnection of z allows it to communicate to the controller by way of the path $p(y, z) \cup p^*(z, n)$, where $p^*(z, n)$ is the new path from z towards the controller n, after x's failure and z's reconnection to the control layer. The binary function:

$$F(s) = \prod_{z \in p(s,x)} P'(z) = \{0, 1\} \quad (4)$$

indicates whether a node $s \in T_x$ is protected against the failure of x (value 0) or not (value 1). Note that, for a node $s \in T_x$ to be protected, it is only necessary that one of the ascendant nodes (denoted by z) of s in the path towards x is locally protected, or s is locally protected itself (Eq.2). In other words, a locally protected node induces protection for any downstream node. Taking all these cases into account, the fraction of protected nodes (locally protected or indirectly protected by upper or upstream nodes) against the failure of x in T_n can be computed as follows:

$$R_{x,n} = \frac{1}{|T_x|} \sum_{s \in T_x} F(s) = \frac{1}{|T_x|} \sum_{s \in T_x} \prod_{z \in p(s,x)} P'(z) \quad (5)$$

In order to discover the robustness in a control layer $T_n \subset G$, rooted at controller n, we compute, for each possible node failure in the network, the number of affected nodes and the number of affected (unprotected) nodes, that is, those which can not reestablish communication with the controller n. Algorithm 1 describes the computation of the control layer robustness parameter.

Algorithm 1 Evaluating Control Layer Robustness

Require: $G = (V, E, \delta), n \in V, T_n \subset G$
  for each node-to-fail $x \neq n, x \in V_{T_n}$ **do**
    $T_x \longleftarrow$ subtree rooted at x
    for each path $p(x, y) \subset T_x$, where y is a leaf node **do**
      for each node $s \in p(x, y), s \neq x$ **do**
        for each node $z \in p(x, s)$, where $z \neq x$ **do**
          $P'(z) \longleftarrow$ Eq. (2)
        end for
        $F(s) \longleftarrow \prod_{z \in p(x,s)} P'(z)$ (Eq. (4))
      end for
    end for
    $R_{x,n} \longleftarrow \frac{1}{|T_x|} \sum_{s \in T_x} F(s)$ (Eq. (5))
  end for
  return $\mathbb{E}_n\{R_{x,n}\}$

IV. Simulation and Results

A. Setup

Both algorithms for controller selection and control layer construction (*Fast Failover* and simplified *k-Critical*, for $k = 1$) are implemented in MATLAB, and their performance is evaluated over a set of randomly generated graphs of fixed size (100 nodes) with three levels of node connectivity: sparsely, medium and strongly connected graphs. Network graphs were generated by using Gephi [9]. Depending on the value of the network wiring probability, Gephi generates networks with close-to-Gaussian distribution of node degrees. For sparse graphs a wiring probability of 0.036 was used, the resulting networks have an average node degree within $[1-5]$. Medium (in density) and dense graphs have an average node degree within $[1-10]$ and $[1-50]$ respectively; graphs were generated with a wiring probability of 0.050 and 0.18. For each node connectivity level, results are averaged over 100 graph samples. For each edge, a random link length is assigned following a uniform distribution within $[1, 10]$ km. Delays associated to network links depend on their length, assuming typical optical fiber rates and using this expression: $Delay = \frac{\mathcal{D}_{link}(m)}{\mathcal{V}_{light}(m/sec)}$. Both algorithms (*Fast Failover* and *k-Critical* for $k = 1$) are executed over these graphs: controllers are selected and the corresponding delay-based, controller-rooted shortest path trees are computed. This setup allows to compare the properties of both algorithms by observing the structure and impact of controller selection over the induced control layer. The delay value ($\mathcal{D}_{req} = 0.20sec$) was determined so that *k-Critical* selected a single controller in all network categories.

B. Results

Performed simulations have been examined in terms of delay, topology structure and control layer robustness.

1) Delay: Fig. 2 displays the average delay between controllers and nodes in the generated control layers from controllers selected by *k-Critical* and *Fast Failover*. It can be observed that *k-Critical* produces substantially "faster" trees that *Fast Failover* for all network categories. This effect, which is particularly relevant in sparse SDNs, implies that the dissemination of control information throughout the SDN is performed, both for an average node and for the most "distant" node from the controller, within shorter time intervals.

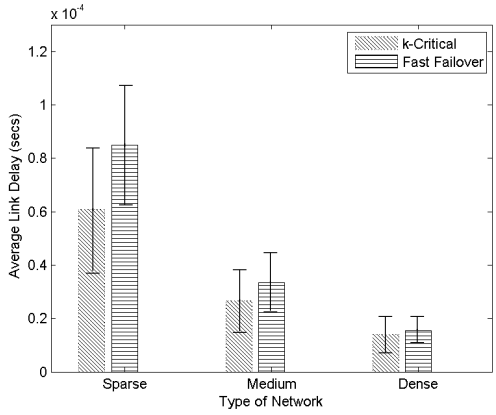

Fig. 2. Average link delay.

2) Topology: k-Critical also produces shorter trees (in hops) than *Fast Failover*, that is, trees for which the longest branch (Fig. 3) has fewer hops than the trees rooted at the controller selected by *Fast Failover*. Although the advantage of *k-Critical* over *Fast Failover* naturally decreases with node density (as, in a full mesh SDN, all trees would have length 1), it is still observable for dense SDNs. The fact that tree branches are shorter in *k-Critical* than in *Fast Failover* also implies that a random node x has, in average, fewer downstream nodes (*i.e.*, nodes for which the control channel towards the controller traverses x) in the first case (*k-Critical*) than in the second. These downstream nodes are the nodes affected by the failure of any node x (including those that may recover due to some sort of protection).

3) Robustness: Fig. 4 shows the percentage of "locally 1-hop protected nodes" in control layer trees generated by *k-Critical* and *Fast Failover*. This is, as detailed in section III-A, the fraction of nodes that could reconnect with the control

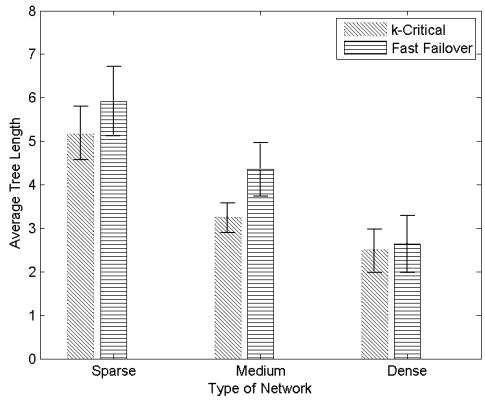

Fig. 3. Average tree length. *Average on average branch length.*

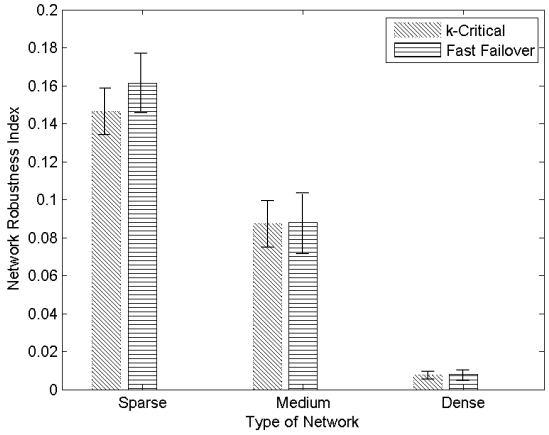

Fig. 5. Network robustness index. *0=robust, 1=non-robust.*

layer in case of failure in their upstream link/parent. Although *Fast Failover* is designed to optimize this specific metric, its performance in this aspect is very similar to the one achieved by *k-Critical*. As argued in section III-B, the metric of "locally

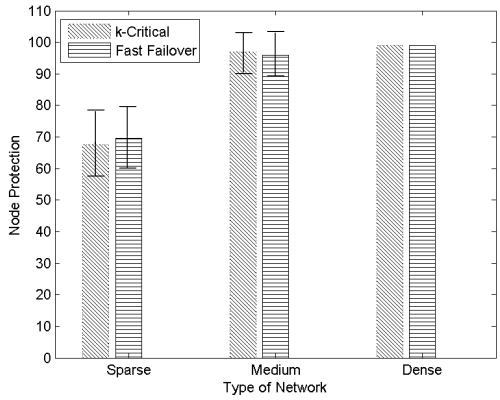

Fig. 4. Nodes locally protected against failures in immediate upstream link/parent.

1-hop protected nodes" is not sufficient to evaluate the robustness of a control layer; a generalization of this metric is thus proposed in section III-C. Fig. 5 shows the complementary of this metric, that is, the average (computed over all possible node failures) of the fraction of downstream nodes that *cannot* recover from a failure, in *k-Critical* and *Fast Failover* control layers. Note that this metric behaves as expected when the network density increases – SDNs with higher connectivity are, in general, more robust than those with low connectivity, due to the fact that more links are available, and thus more redundant paths can be leveraged in case of node failures. It can be observed that, in the light of this generalized robustness metric, *k-Critical* ($k = 1$) controllers produce more robust control trees than *Fast Failover*, in particular in sparse SDNs – that is, in networks where algorithm robustness is more necessary. As the SDN node density increases, both algorithms tend to provide a similar level of control layer robustness.

V. Conclusions

The problem of controller placement in Software Defined Networks remains open. This paper proposes and formalizes a metric to quantify the control layer robustness, defined as the fraction of nodes that have an backup path to the controller in case of a node or link failure. It was evaluated over control layer created from controllers found by a simplified version of *k-Critical* and *Fast Failover* heuristics. The latter is aligned with the fast failover mechanisms included in the specification of OpenFlow. Although *k-Critical* has not been explicitly designed to optimize any robustness metric, it shows a very similar performance to *Fast Failover* in terms of the underlying robustness metric. Moreover, *k-Critical* has a lower complexity ($O(n)$: it grows linearly with the size of the network) than *Fast Failover* ($O(n^2)$). Further work in the direction of control layer robustness should focus on the formalization of the desirable properties, the theoretical analysis of the generated trees and the evaluation on realistic topologies.

References

[1] S. Hassas, A. Tootoonchian, Y. Ganjali. "On Scalability of Software-Defined Networking", *Communications Magazine, IEEE.*, vol. 51, no.2, ISBN 0163-6804, 2013.

[2] B. Nunes, M. Mendonca, X. Nguyen, *et al.* "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *Communications Surveys & Tutorials, IEEE*, vol. 16, no.3, pp. 1-18, 2014.

[3] D. Hock, M. Hartmann, S. Gebert, *et al.* "Pareto-optimal resilient controller placement in SDN-based core networks", *Teletraffic Congress (ITC), 2013 25th International.*, pp. 1-9, Shanghai, 2013.

[4] M. Tatipamula, N. Beheshti-Zavareh, Y. Zhang. "Controller Placement For Fast Failover in the Split Architecture", U.S. Patent 0028073 A1, January 31, 2013.

[5] Y. Jiménez, C. Cervelló-Pastor, A. García, "Defining a Network Management Architecture", The PhD Forum of the 21st IEEE International Conference on Network Protocols, Germany, 2013.

[6] Y. Jiménez, C. Cervelló-Pastor, and A. García, "On the controller placement for designing a distributed SDN control layer", *IFIP Networking 2014 Conference*, Norway, ISBN 978-3-901882-58-6, 2014.

[7] N. McKeown, T. Anderson, H. Balakrishnan *et al.* " OpenFlow: enabling innovation in campus networks", *ACM SIGCOMM CCR*, Vol. 38, no.2, pp. 69-74, 2008.

[8] B. Heller, R. Sherwood, N. McKeown. "The Control Placement Problem", *ACM SIGCOMM.*, New York, ISBN: 978-1-4503-1477-0,2012.

[9] M. Bastian, S. Heymann, M. Jacomy, "Gephi: an open source software for exploring and manipulating networks", *International AAAI Conference on Weblogs and Social Media*, 2009.