# *POCO-PLC*: Enabling Dynamic Pareto-Optimal Resilient Controller Placement in SDN Networks

David Hock, Matthias Hartmann, Steffen Gebert, Thomas Zinner, Phuoc Tran-Gia
University of Würzburg, Institute of Computer Science
{hock,hartmann,gebert,zinner,trangia}@informatik.uni-wuerzburg.de

## I. Introduction and Problem Description

Recently, Software Defined Networking (SDN) has gained a lot of attention. The paradigm shift towards centralized architectures with a separation of control plane and data plane is expected for several use cases, including amongst others core communication networks, data center networks, or Network Function Virtualization (NFV). An important issue during SDN deployment is the placement of controllers in the network. The first to address this topic were Heller et al. [3] followed by other researchers who added different resilience issues into their observations. An overview on relevant publications can be found in our previous work [1], where we also introduced our POCO toolset.

Most of the research effort done in the area of SDN controller placement so far concentrated on a rather static analysis of a network, i.e., static latencies between the nodes were assumed. Furthermore, the individual load caused by the different nodes attached to a controller was not regarded in detail. In our previous publication, we only used values like city populations to assign different weights to the nodes in the network, and considered that dynamic traffic matrices as offered e.g. by the SNDlib library could be used for analyzing dynamic controller placements. Bari et al. [4] were the first to focus an entire paper on the dynamic provisioning of controllers in SDN networks. Based on heuristics they repeatedly calculate the best placement according to a predefined metric and assign this placement to the network.

Here, we provide and present our powerful and user-friendly *POCO-PLC* toolset that facilitates the analysis and optimization of the controller placement in SDN networks under dynamic conditions. We give a general overview on the functionality and architecture of *POCO-PLC*. Then, we indicate possible use cases enabled by *POCO-PLC*. Finally, we briefly address the planned demo presentation.

## II. *POCO* and *POCO-PLC*

In previous work, we showed that an exhaustive evaluation of multiple control plane resilience metrics for all possible placements is computationally feasible for realistic network sizes [1]. The *POCO*-toolset used to produce the presented results was made available online [5]. We also developed a Graphical User Interface for *POCO* [2]. Altogether, *POCO* provides a simple and convenient way to calculate, illustrate and analyze different resilience issues for the controller placement process in SDN networks. The scope of *POCO-PLC* goes beyond the existing implementation of *POCO*. It combines the powerful MATLAB analysis tool with a distributed application deployed across the PlanetLab research network. This enables an analysis and optimization of the controller placement based on monitoring data obtained live from a realistic, world-wide distributed network. Moreover, it allows the real-time computation and performance analysis of new controller placements.

Figure 1 illustrates the architecture of *POCO-PLC*. It can be divided into three logical parts: the *POCO* tool running on a local machine, a dedicated aggregation node running e.g. on one of the PlanetLab nodes, and an arbitrary set of PlanetLab nodes. Each of these nodes can either have the role of a simple switch that requires a controller to function properly, or it can be a controller itself. The *POCO* GUI used for *POCO-PLC* consists of the normal *POCO* GUI as one part (not shown in the Figure) as well as additional capabilities to interact with PlanetLab. This interaction is realized via a dedicated aggregation node. The aggregation node collects statistics from the different PlanetLab nodes and also configures and deploys the current output of *POCO* to the network. Each of the PlanetLab nodes used in the measurement is equipped with a *POCO-PLC* agent implemented in Python. The functionality of the agent includes for example latency measurements to the other nodes and local measurements on the node such as CPU load measurements. PlanetLab nodes that are used as switches dynamically choose the controller that is closest to the node, i.e., the controller that provides the shortest round-trip time. Furthermore, the agent can be extended by more complex tasks such as a complete emulation of switch and controller functions and a communication between the different nodes.

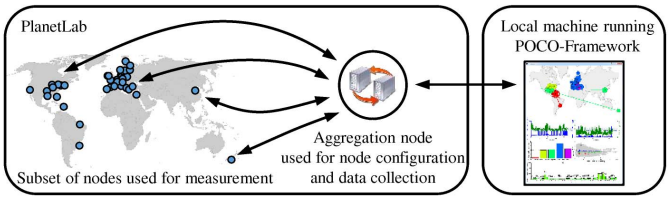


Fig. 1. *POCO-PLC*: interconnection between *POCO* and PlanetLab.

To explain the extensions of *POCO-PLC* to *POCO* in more detail, a screenshot of the *POCO-PLC* overview GUI running on the local machine is shown in Figure 2.

This work has been performed in the framework of the CELTIC EUREKA project SASER-SIEGFRIED (Project ID CPP2011/2-5), and it is partly funded by the BMBF (Project ID 16BP12308). The authors alone are responsible for the content of the paper.

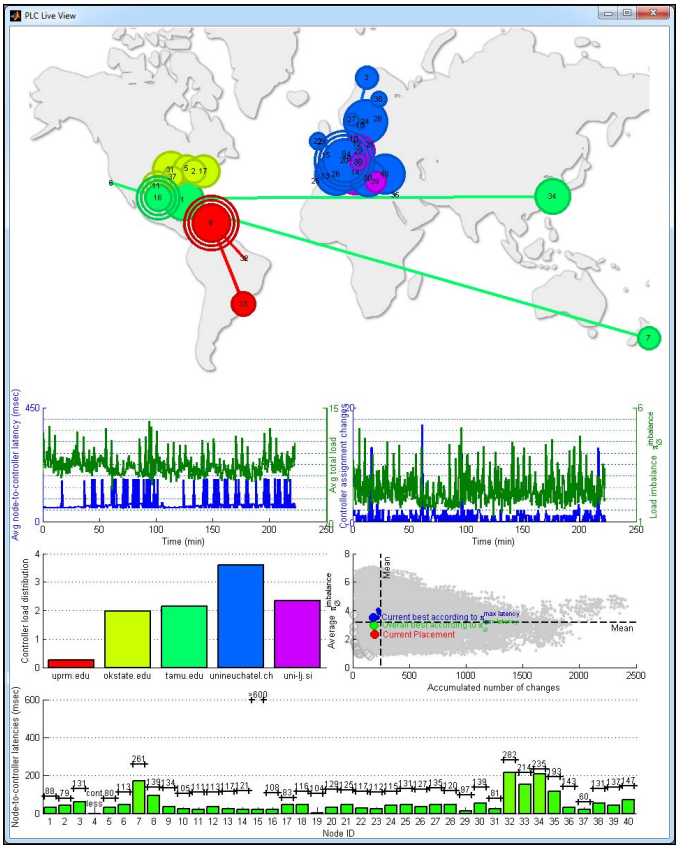


Fig. 2. Screenshot of the *POCO-PLC* overview GUI.

In the upper part of the GUI, the current network situation and placement of controllers is illustrated using the global map. Similar to our previous work [1], various visualization and coloring options can be chosen to indicate the node-to-controller assignment, the latencies, or the load balancing. In the displayed example, the node sizes indicate the current CPU load on the different nodes. The colors and links between the nodes indicate which node is assigned to which controller. The overview GUI is updating periodically every few seconds. This provides a live view on the current network situation.

More detailed real-time data and time series statistics are shown below: The full-width bar plot in the bottom of the GUI window shows the current round-trip delay between each node and its currently selected controller. In addition, it shows the average of the round-trip delays to all available controllers. The bar plot above indicates the current load on the different controllers. Time series of different values that can be used to determine the performance of a controller placement are shown in the two charts directly below the world map. The chart on the top left shows the average node-to-controller latencies as well as the total network load, i.e., currently, the sum of all CPU loads in the PlanetLab network. The chart on the right side shows the load imbalance, as defined in [1]. In addition, it shows how often the assignment of nodes to controllers changes over time. This is a measure for the stability of a controller placement with the given dynamic controller selection.

In addition to the evaluation of the currently selected controller placement, the fast exhaustive evaluation of the *POCO* framework offers the possibility to perform a live evaluation of all other possible controller placements. *POCO-PLC* calculates and accumulates these values and shows the performance of each possible placement in the solution space in the plot in the middle on the right side. The performance of the best placements regarding imbalance and assignment stability are the gray dots on the very left and bottom of the plot. The performance of the currently chosen placement and best solutions regarding other metrics are also displayed.

## III. Research Perspectives Enabled by *POCO-PLC*

In the following, we briefly discuss different research perspectives that are enabled using *POCO-PLC*. Current research on controller placement in SDN networks mostly concentrated on static issues. *POCO-PLC* enables an integration of dynamics into the research on controller placements, which is especially achieved by the integration of the PlanetLab network. Due to its distributed testbed nature, PlanetLab can be used as a large realistic unpredicable source for dynamic input parameters. These parameters can be dynamic regarding changing latencies in the network or regarding changing loads of the distributed controllers. Both types of dynamic input can be included separately. This allows to analyze different use cases: (1) use cases with only dynamic node loads but static latencies or node-to-controller assignment, (2) use cases with constant node weights but dynamic assignment, and (3) use cases with dynamic assignment and dynamic node weights.

The *POCO-PLC* framework is not limited to the functionality described here. There are various options for extensions: (1) different values could be measured at each node and used as input parameter, (2) additional functions could be implemented in the nodes such as an emulation of SDN control traffic. Furthermore, (3) different controller architectures including hierarchical approaches could be implemented and their performance could be compared using *POCO-PLC*.

## IV. Scenario and Demo Presentation

The demo presentation at Infocom shows the functionality of the *POCO-PLC* framework. As a particular scenario, we illustrate how *POCO-PLC* enables a live comparison of different placements including previously collected time series of metrics such as the average load imbalance over time or the number of assignment changes for different placements. The scenario shows how the consideration of knowledge from previous evaluations can improve the optimization for future network situations. The source code of the extension of *POCO-PLC* is available online [5].

## References

[1] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, "Pareto-Optimal Resilient Controller Placement in SDN-based Core Networks," in *ITC*, Shanghai, China, 2013.

[2] D. Hock, S. Gebert, M. Hartmann, T. Zinner, and P. Tran-Gia, "POCO: A Framework for the Pareto-Optimal Resilient Controller Placement in SDN-based Core Networks," in *NOMS*, Krakow, Poland, 2014.

[3] B. Heller, R. Sherwood, and N. McKeown, "The Controller Placement Problem," in *ACM SIGCOMM HotSDN*, Helsinki, Finland, 2012.

[4] M. F. Bari, A. R. Roy, S. R. Chowdhury, Q. Zhang, M. F. Zhani, R. Ahmed, and R. Boutaba, "Dynamic Controller Provisioning in Software Defined Networks," in *CNSM*, Zürich, Switzerland, 2013.

[5] http://www3.informatik.uni-wuerzburg.de/poco.