Decentralized Solutions for Monitoring Large-Scale Software-Defined Networks

G. Tangari, M. Charalambides, D. Tuncer, and G. Pavlou

University College London, London, UK

Abstract. Software-Defined Networking (SDN) technologies offer the possibility to automatically and frequently reconfigure the network resources by enabling simple and flexible network programmability. One of the key challenges to address when developing a new SDN-based solution is the design of a monitoring framework that can provide frequent and consistent updates to heterogeneous management applications. To cope with the requirements of large-scale networks (i.e. large number of geographically dispersed devices), a distributed monitoring approach is required. This PhD aims at investigating decentralized solutions for resource monitoring in SDN. The research will focus on the design of monitoring entities for the collection and processing of information at different network locations and will investigate how these can efficiently share their knowledge in a distributed management environment.

1 Introduction

Effective resource monitoring is a fundamental requirement for the management of large-scale networks. Monitoring systems are designed to collect measurements from the network resources, process the acquired data and expose the resulting knowledge, in order to provide a reliable representation of the network state. Today, network monitoring mainly operates on long timescales producing periodic reports, which are mostly used for manual and infrequent network reconfigurations. However, with the advent of Software-Defined Networking (SDN) technologies, the monitoring requirements are changing since network operators will be able to reconfigure their resources at a faster pace to automatically react to emerging conditions.

By decoupling the control logic from the forwarding hardware, SDN enables easy and flexible network programmability. This key feature allows the implementation of complex high-level network policies [1][2][3] and the design of applications that reconfigure the network automatically and frequently [4][5][6]. These novel capabilities pose new monitoring requirements, which cannot be met by currently deployed approaches. As such, the overarching objective of this PhD is to design a monitoring framework that can provide frequent and consistent network state updates to complex and heterogeneous applications, thus enabling fast and effective reconfigurations.

To address these requirements, the monitoring system has to be *scalable*, *flexible* toward heterogeneous applications and *adaptive* with respect to the network conditions. Our research starts from these considerations and focuses on a distributed monitoring system for large-scale software defined networks. The choice of a decentralized approach has its roots in recent research [7][8] which has demonstrated the weakness of centralized SDN proposals. First, using a central controller poses scalability limitations in terms of the number of switches and traffic flows. Second, as the network diameter grows, the associated latencies can become considerable, penalizing the responsiveness of the management operations. In contrast, decentralized solutions can cope with a large number of devices, distributed over wide geographic areas, and facilitate fast reconfigurations since monitoring information is acquired closer to the source.

2 Requirements and Research Challenges

This PhD aims to investigate decentralized solutions for resource monitoring in software-defined networks. The research will be based on the SDN framework proposed in [9] which introduces a clear separation between management and control functionality. A set of *managers*, distributed over the network, hosts various management applications that implement the necessary logic to decide on network (re)configurations, while corresponding *controllers* are in charge of enforcing those decisions. To communicate with the forwarding hardware, the controllers rely on OpenFlow[10], the *de facto* standard for the southbound communication in SDN. As described in [9], monitoring is part of the distributed management functionality with the objective to gather information from network devices and subsequently support applications for their decisions. Although of paramount importance, the authors did not elaborate on the technical challenges associated with collecting and disseminating information in a distributed environment. The objective of this PhD is to address this gap by investigating the challenges described below.

Design considerations. Effective design of distributed monitoring functionality has to take into account a few key issues. The monitoring operations, if very intrusive, could adversely affect the network performance. At the same time, monitoring operations need to be frequent and fast to enable management applications to operate on short timescales. In addition, they should provide precise and high-granularity information to support accurate decisions. When the management system operates inside a large-scale SDN, the impact of these issues is amplified since the decisions might be taken far away from the locations where monitoring is performed. We identify below the two main features that need to be provided by monitoring entities.

1. **Programmability**: the frequency and the granularity level of the measurements have to be highly configurable based on the requirements of heterogeneous management applications.
2. **Adaptability**: the measurement rate should be frequently reconfigured, in such a way to follow the rate of change of the measured values. Adaptability, as argued in [11][12], enables a reduction of the monitoring overhead, while ensuring acceptable levels of accuracy.

We have identified the main operations composing the monitoring entity function in a distributed management framework. The first one is the analysis of the application requirements. By exposing a high-level API to the management applications, the monitoring entity receives a large variety of requirements. It interprets such requirements and translates them into sets of monitoring specifications, indicating the type of measurements to be performed and the targets of such measurements. Then, it efficiently schedules the individual monitoring operations, in such a way to limit the overhead and avoid the creation of bottlenecks[13]. For example, to gather statistics from an OpenFlow-enabled switch, the monitoring entity instructs the corresponding controller to query, through the OpenFlow interface, the target switches, or to configure them in order to push statistics upon flow expirations. Acquired statistics are finally processed and subsequently exposed to the management applications.

Distributing the monitoring function. In a distributed SDN environment, management applications operating at different locations may need statistics gathered from outside their local scope. In such a setting, the monitoring entities will need to share their local knowledge. Ensuring the right tradeoff in state distribution is an open issue, as argued in [8]. In our case, the distribution of the monitoring functionality poses a significant challenge: how can the monitoring entities frequently access reliable and detailed *knowledge* concerning remote resources at an acceptable *cost*? A solution where every local manager shares all the acquired statistics with every other manager is unlikely to scale due to the explosion of overhead and synchronization times. This would prevent decision-making entities to quickly react to emerging conditions and could therefore adversely affect the usage of resources. Part of our work will investigate the possible techniques to achieve the right balance between accuracy and overhead for a wide range of applications. Striking the right tradeoff in statistics aggregation, before the *knowledge* is exchanged, is a possible way to address the issue since this influences the overhead associated with state distribution. Another research challenge concerns the intelligent selection of managers between which knowledge should be shared according to the application requirements. Based on this, a suitable paradigm will be chosen for selective knowledge exchange.

3 Use Case: ISP-Managed Content Distribution

To experiment with the capabilities of the proposed approach, we will consider a distributed SDN network environment, where an ISP operates a content distribution service. In this scenario, a set of content items is cached within the ISP. The management system of the network periodically updates (*e.g.* in order of hours) the bandwidth allocation and the content placement, but also reconfigures in real-time the request routing (*i.e.* the routing of users' requests) by programming the underlying forwarding hardware.

Figure 1 depicts a simplified representation of this use case. Here, the network is divided in two partitions, each under the control of a local manager (*i.e., Mgr1/2*). In addition, a controller is assigned to each partition (*i.e., Ctrl1/2*) to interact with the network devices in the corresponding area.

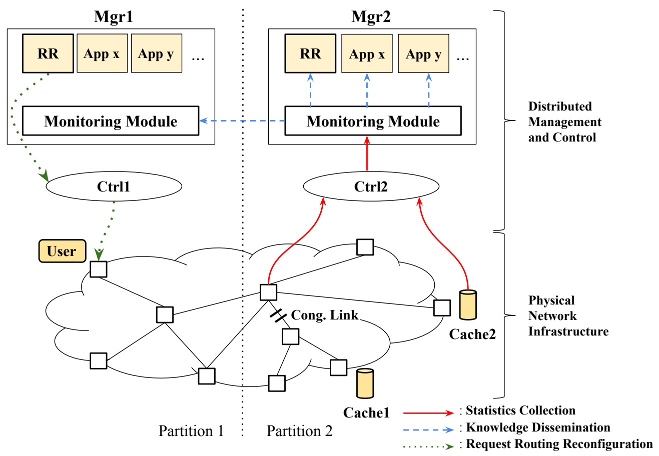

Fig. 1. Example representation of the use case.

Each manager implements an instance of a distributed Request Routing (RR) reconfiguration application which interacts with the monitoring modules to check the link utilization and the server load in the relevant partition. To collect network statistics, each monitoring module schedules periodic switch polls to acquire the values of the switch-port counters, together with periodic checks of the servers' CPUs and outbound traffic. The acquired data is then *processed* (filtered and aggregated) to form *knowledge* which is *disseminated* both locally (*i.e.* to the local applications) and to remote monitoring modules. We consider a simple example for this use case. Let us assume that, by default, all the requests received at a given *user* location are served by *Cache1* and that congestion occurs on a link located in partition 2. In this case the knowledge obtained by *Mgr2*, notifying the congestion, is exchanged with the monitoring module on *Mgr1*. This enables the *RR* application on *Mgr1* to modify the request-routing configurations by assigning part of the new requests to *Cache2*, thus reducing the utilization of the congested link. The performance of the reconfiguration performed by *Mgr1* depends on the timeliness, accuracy and level of granularity of the information shared between the two monitoring modules.

4 Summary and Future Work

We have presented the main research challenges associated with the design of a scalable monitoring framework that can provide heterogeneous management applications, distributed over a large-scale SDN, with reliable and accurate knowledge. In next steps of this research, we plan to demonstrate the capabilities of our approach by investigating 1) the type of information which can be extracted from the network infrastructure (not only the forwarding hardware), 2) how information can be efficiently exchanged, and 3) the possible implications in terms of delay and cost for the network. We will use the use-case described in the previous section as a baseline for the evaluation of the proposed approach in terms of generated overhead and synchronization time.

Acknowledgement

This research was funded by the UK EPSRC KCN project (EP/L026120/1) and by the Flamingo Network of Excellence project (318488) of the EU Seventh Framework Programme.

References

1. H. Kim and N. Feamster. Improving Network Management with Software Defined Networking. In IEEE Communication Magazine, February 2013.
2. Y. Yuan, R. Alur, and B. T. Loo. NetEgg: Programming network policies by examples. In Workshop on Hot Topics in Networks (Hotnets), 2014.
3. S. Shin et al. FRESCO: Modular Composable Security Services for Software-Defined Networks. In ISOC NDSS, February 2013.
4. M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat. Hedera: dynamic flow scheduling for data center networks. In Proc. NSDI'10.
5. B. Heller et al. ElasticTree: Saving Energy in Data Center Networks. In Proc NSDI'10.
6. S. Agarwal, M. Kodialam, and T. Lakshman. Traffic engineering in software defined networks. In Proc. INFOCOM 2013.
7. A. Tootoonchian and Y. Ganjali. HyperFlow: A distributed control plane for OpenFlow. In Proceedings of the 2010 internet network management conference on Research on enterprise networking, 2010.
8. D. Levin et al. Logically centralized?: state distribution trade-offs in software defined networks. In HotSDN 2012.
9. D. Tuncer, M. Charalambides, S. Clayman, and G. Pavlou. Adaptive resource management and control in software defined networks. In IEEE TNSM, Mar. 2015.
10. N. McKeown et al. OpenFlow: Enabling Innovation in Campus Networks. In ACM SIGCOMM Computer Communication, 2008.
11. S. Chowdhury, M. Bari, R. Ahmed, and R. Boutaba. Payless: A low cost network monitoring framework for software defined networks. In Proc. NOMS, 2014.
12. M. Moshref, M. Yu, R. Govindan, and A. Vahdat. Dream: dynamic resource allocation for software-defined measurement. In SIGCOMM, 2014.
13. J. C. Mogul et al. Devoflow: Cost-Effective Flow Management for High Performance Enterprise Networks. In Proc. Hotnets 2010.