This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2015.2480398, IEEE Transactions on Wireless Communications

Alleviating Hidden and Exposed Nodes in High-Throughput Wireless Mesh Networks

Sandip Chakraborty, *Member, IEEE*, Sukumar Nandi, *Senior Member, IEEE*, Subhrendu Chattopadhyay

***Abstract*—This paper proposes an opportunistic approach to mitigate the hidden and exposed node problem in a high-throughput mesh network, by exploiting the frame aggregation and block acknowledgment (BACK) capabilities of IEEE 802.11n/ac wireless networking standard. Hidden nodes significantly drop down the throughput of a wireless mesh network by increasing data loss due to collision, whereas exposed nodes cause under-utilization of the achievable network capacity. The problem becomes worse in IEEE 802.11n/ac supported high-throughput mesh networks, due to the large physical layer frame size and prolonged channel reservation from frame aggregation. The proposed approach uses the standard 'Carrier Sense Multiple Access' (CSMA) technology along with an 'Opportunistic Collision Avoidance' (OCA) method that blocks the communication for hidden nodes and opportunistically allows exposed nodes to communicate with the peers. The performance of the proposed CSMA/OCA mechanism for high throughput mesh networks is studied using the results from an IEEE 802.11n+s wireless mesh networking testbed, and the scalability of the scheme has been analyzed using simulation results.**

***Keywords*-IEEE 802.11n; Frame Aggregation; Hidden and Exposed Nodes; Spatial Reuse**

I. Introduction

Wireless Mesh Networking [1] technology provides a key milestone to the next generation backbone access network, where a 'mesh' of wireless routers supports the backbone connectivity to the end-users through multi-hop communications. To assist commercial, enterprise and community mesh backbone technology, IEEE 802.11s [2] is gaining significant attention among the developers for its compatibility with commercially successful IEEE 802.11 wireless networking standard. The IEEE 802.11s amendment to the wireless networking standard contributes to the medium access control (MAC) specifications for the wireless mesh access, that can operate along with any physical layer technology supported by the IEEE 802.11 task groups. The recent developments over IEEE 802.11 technology enhances wireless communications for high data rate supports, up to 600 Mbps with IEEE 802.11n, and up to 6.77 Gbps with IEEE 802.11ac [3]. Therefore the mesh networking standard along with high data rate support has exorbitant potentials to provide high-throughput backbone network connectivity to the end users, that effectively directs towards the design of 'wireless world' [4].

The IEEE 802.11 coordination function for wireless channel access relies on 'Listen before Talk' which is known as 'Carrier Sense Multiple Access with Collision Avoidance' (CSMA/CA). The clear channel assessment (CCA) in CSMA/CA uses two listen or carrier sensing (CS) mechanism - physical CS (PCS) and virtual CS (VCS). Extensive studies on IEEE 802.11 CCA technologies have revealed that PCS fails to detect nodes which are beyond the CS range (known as hidden nodes), whereas VCS reduces spatial reuse by blocking the communications which are not susceptible to interference (known as exposed nodes) [5]. The performance impacts of hidden and exposed nodes in a network is characterized by the CS range and the interference range. In PCS, a node can initiate a new communication only if all other nodes in its CS range are inactive. However, the communication becomes successful only if the receiver is not within the interference range of any other active nodes in the network. As CCA is performed at the transmitter, whereas the receiver is susceptible to interference, there is a high probability of hidden nodes in a mesh network. VCS [6] is proposed to solve the hidden node problem through a handshaking between the transmitter and the receiver before the actual communication takes place, through a pair of messages termed as 'Request to Send' (RTS) and 'Clear to Send' (CTS). The transmitter sends a RTS message only if the CCA reports no active nodes in the CS range of the transmitter. The receiver replies back with a CTS message only if there are no active nodes within the CS range of the receiver. On overhearing the RTS and the CTS messages, other nodes within the CS range of transmitter and receiver block their communication for the time duration mentioned within the RTS and the CTS messages. As a consequence, VCS introduces the problem of exposed nodes which are outside the receiver's interference range, but within the CS range of the transmitter. Though the exposed nodes are harmless for an ongoing communication, they are blocked as a result of VCS and therefore cause under-utilization of the effective network capacity.

The performance issues experienced by hidden and exposed nodes in a multi-hop and mesh network have been well studied in the literature. Most of the existing approaches design mechanisms to eliminate either the hidden nodes or the exposed nodes, such as busy tone signaling [7], use of separate control channels [8], adaptation in CS threshold and temporary disabling the CS mechanism [9] etc. However, none of these approaches can eliminate hidden and exposed nodes simultaneously. In fact, existing research [9] has shown that it

Copyright (c) 2013 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Sandip Chakraborty is with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur 721302, INDIA (e-mail: sandipc@cse.iitkgp.ernet.in)

Sukumar Nandi and Subhrendu Chattopadhyay are with the Department Computer Science and Engineering, Indian Institute of Technology Guwahati, Guwahati 781039. INDIA (e-mail: {sukumar,subhrendu}@iitg.ernet.in)

1536-1276 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2020.3008426, IEEE Transactions on Network and Service Management

Aloe: Fault-Tolerant Network Management and Orchestration Framework for IoT Applications

Subhrendu Chattopadhyay, Soumyajit Chatterjee, Sukumar Nandi, Sandip Chakraborty

***Abstract*—Internet of Things (IoT) platforms use a large number of low-cost resource constrained devices and generates millions of short-flows. In-network processing is gaining popularity day by day to handle IoT applications and services. However, traditional software-defined networking (SDN) based management systems are not suitable to handle the plug and play nature of such systems. In this paper, we propose Aloe, an auto-scalable SDN orchestration framework. Aloe exploits in-network processing framework by using multiple lightweight controller instances in place of service grade SDN controller applications. The proposed framework ensures the availability and significant reduction in flow-setup delay by deploying instances in the vicinity the resource constraint IoT devices dynamically. Aloe supports fault-tolerance with recovery from network partitioning by employing self-stabilizing placement of migration capable controller instances. Aloe also provides resource reservation for micro-controllers so that they can ensure the quality of services (QoS). The performance of the proposed system is measured by using an in-house testbed along with a large scale deployment in Amazon web services (AWS) cloud platform. The experimental results from these two testbeds show significant improvement in response time for standard IoT based services. This improvement of performance is due to the reduction in flow-setup time. We found that Aloe can improve flow-setup time by around** $10\% - 30\%$ **in comparison to one of the states of the art orchestration framework.**

***Index Terms*—Orchestration Framework, Fault-tolerance, Programmable network, IoT, In-network processing, SDN**

I. Introduction

The rapid proliferation and deployments of large-scale Internet-of-Things (IoT) applications have made the network architecture complicated from the perspective of end-user service management. Simultaneously, with the advancement of edge-computing, in-network processing and platform-as-a-service technologies, end-users consider the network as a platform for the deployment and execution of myriads of diverse applications dynamically and seamlessly. Consequently, network management has become increasingly difficult in today's world with this complex service-oriented platform overlay on top of the inherently distributed TCP/IP network architecture. The concept of software-defined networking (SDN) [1] has gained popularity over the last decade to make the network management simple, cost-effective and logically centralized, where a network manager can monitor, control and deploy new network services through a central controller. Nevertheless, edge and in-network processing over an IoT platform is still challenging even with an SDN based architecture [2].

S. Chattopadhyay and S. Nandi are with the Department of CSE, Indian Institute of Technology Guwahati, India 781039. email:{subhrendu, sukumar}@iitg.ac.in. S. Chatterjee and S. Chakraborty are with Department of CSE, Indian Institute of Technology Kharagpur, India, 721302. email: {soumyachat@iitkgp.ac.in, sandipc@cse.iitkgp.ac.in}

Motivation: The primary motivation for supporting a distributed network management and flow steering framework coupled with edge and in-network processing over an IoT platform are as follows: (1) The platform should be agile to support rapid deployment of applications without incurring additional overhead, ensuring the scalability of the system [3], [4]. (2) With micro-service architectures [5], in-network processing requires dividing a service into multiple micro-services and deploying them at different network nodes for reducing the application response time with parallel computations. However, such micro-services may need to communicate with each other, and therefore the flow-setup delay from the in-network nodes need to be very low to ensure near real-time processing. (3) The percentage of short-lived flows are high for IoT based networks [6]. This also escalates the requirement for reducing flow-setup delay in the network. (4) Failure rates of IoT nodes are in-general high [7]. Therefore, the system should support a fault-tolerant or fault-resilient architecture to ensure liveness.

Limitations of existing approaches: Although SDN supported edge computing and in-network processing have been widely studied in the literature for the last few years [2], [8], [9] as a promising technology to solve many of the network management problems, they have certain limitations. First, the SDN controller is a single-point bottleneck. Every flow initiation requires communication between switches and the controller; therefore, the performance depends on the switch-controller delay. With a single controller bottleneck, the delay between a switch and the controller increases, which affects the flow-setup performance. As we mentioned earlier, the majority of the flows in an IoT network are short-lived, the impact of switch-controller delay is more severe on the performance of short-lived flows. To solve this issue, researchers have explored distributed SDN architecture with multiple controllers deployed over the network [10]. However, with a distributed SDN architecture, the question arises about how many controllers to deploy and where to deploy those controllers. Static controller deployments may not alleviate this problem, as IoT networks are mostly dynamic with plug-and-play deployments of devices. Dynamic controller deployment requires hosting the controller software over commercially-off-the-shelf (COTS) devices and designing methodologies for controller coordination, which is a challenging task [11]. The problem is escalated with the objective of developing a fault-tolerant or fault-resilient architecture in a network where the majority of the flows are short-lived flows.

Authorized licensed use limited to: University of Exeter. Downloaded on July 15,2020 at 10:40:30 UTC from IEEE Xplore. Restrictions apply.

The Journal of Supercomputing
https://doi.org/10.1007/s11227-021-04135-2

Containerized deployment of micro-services in fog devices: a reinforcement learning-based approach

Shubha Brata Nath[1] · Subhrendu Chattopadhyay[2] · Raja Karmakar[3] · Sourav Kanti Addya[4] · Sandip Chakraborty[1] · Soumya K Ghosh[1]

Accepted: 8 October 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

The real power of fog computing comes when deployed under a smart environment, where the raw data sensed by the Internet of Things (IoT) devices should not cross the data boundary to preserve the privacy of the environment, yet a fast computation and the processing of the data is required. Devices like home network gateway, WiFi access points or core network switches can work as a fog device in such scenarios as its computing resources can be leveraged by the applications for data processing. However, these devices have their primary workload (like packet forwarding in a router/switch) that is time-varying and often generates spikes in the resource demand when bandwidth-hungry end-user applications, are started. In this paper, we propose *pick–test–choose*, a dynamic micro-service deployment and execution model that considers such time-varying primary workloads and workload spikes in the fog nodes. The proposed mechanism uses a reinforcement learning mechanism, *Bayesian optimization*, to decide the target fog node for an application micro-service based on its prior observation of the system's states. We implement PTC in a testbed setup and evaluate its performance. We observe that PTC performs better than four other baseline models for micro-service offloading in a fog computing framework. In the experiment with an optical character recognition service, the proposed PTC gives average response time in the range of 9.71 sec–50 sec, which is better than Foglets (24.21 sec–80.35 sec), first-fit (16.74 sec–88 sec), best-fit (11.48 sec–57.39 sec) and mobility-based method (12 sec–53 sec). A further scalability study with an emulated setup over Amazon EC2 further confirms the superiority of PTC over other baselines.

Keywords Fog computing · Container deployment · Bayesian optimization · Internet of things · Reinforcement learning

✉ Sandip Chakraborty
sandipc@cse.iitkgp.ac.in

Extended author information available on the last page of the article