Status-aware and SLA-aware QoS Routing Model for SDN Transport Network

Atefeh Meshinchi
Polytechnique Montréal
Montreal, Canada
email: atefeh.meshinchi@polymtl.ca

Ranwa Al Mallah
Ryerson University
Montreal, Canada
email: ranwa.almallah@ryerson.ca

Alejandro Quintero
Polytechnique Montréal
Montreal, Canada
email: alejandro.quintero@polymtl.ca

***Abstract*—Internet of Things (IoT) is a key technological enabler to create smart environments and provide various benefits. In the context of a smart city, a huge number of IoT applications are being developed for emergency management operation and city traffic congestion management. These applications require fast system reaction to get the valuable data and make appropriate decisions. Therefore, it is essential to design and develop a service model that ensures an appropriate level of Quality of Service (QoS) for such applications. In this paper, we take advantage of Software-Defined Networking (SDN) technology integrated into the IoT system to propose a new QoS routing model for core transport SDN. In the model, the application QoS preferences and network elements status are directly considered in the resource allocation process aiming to satisfy the application expectation while maximizing network performance. We modeled a status-aware and Service-Level-Agreement-aware (SLA-aware) routing mechanism and implemented multi-path and load-balancing approaches in the model to enhance the network throughput and increase system availability.**

Keywords–Internet of Things; Routing; Quality of service; Software-Defined Networking.

I. INTRODUCTION

The proliferation of devices in a communicating–actuating network creates the Internet of Things (IoT). It is a radical evolution of the current Internet into a network of interconnected objects. In IoT, smart objects are able to sense, communicate, compute, analyze, and make nonhuman-intervention decisions using locally- or globally-gathered data. Diverse technologies and techniques in the field of devices, data, and communication are contributed to the fulfillment of the IoT services.

In the multi-layered IoT reference architecture [1], application and service support layers, also called middle-ware layer, interact directly with user requirements while the network layer transmits the data to the upper layer. The device layer provides the data collection capability. Quality of Service (QoS) management in IoT systems is a very complex task because of the extremely large variety of devices and services, as well as the technologies and techniques involved in the systems architecture.

Applications are fundamental to the IoT. They provide the presentation layer for the data captured from the billions of devices around the world. IoT application could be classified from different perspectives such as the type of information they manage, the type of recipient (person or system oriented), and their criticality. In general, IoT applications are classified into three different types: (1) control applications, also called mission-critical applications,such as city-traffic management and emergency management, which need very fast response with as less error as possible, (2) monitoring applications, such as intelligent security surveillance tasks, which are fed by cameras and need more throughput, and (3) analysis and inquiry applications, such as the inquiry into the transported item state in the intelligent logistics, which are throughput and delay tolerant. Although other metrics enforced from device layer like quality of information and data sampling rate are important in the service quality, the productivity and performance of an IoT application is mainly impacted by the performance of the communication network.

The Internet as a large-scale networking system has had great success in the interconnection of computer networks and with the creation of IPv6 it extends the TCP/IP address spaces to provide identifiers for the large number of connected devices. However, the legacy computer network and the Internet still face some limitations. On one hand, the control intelligence, which is implemented by various routing and management protocols, is embedded in every network elements and It is difficult to change. Vendor-dependent platforms and interfaces make the Internet evolution complex and slow.

On the other hand, the Internet provides best-effort services and it may not meet the more specific requirements of applications in terms of service quality [2] [3]. Two main standardized QoS implementation models in classical IP networks are Integrated Services (IntServ) [4], with the idea of per-flow resource reservation, and Differentiated Services (DiffServ) [5], with the idea of traffic classification and prioritization. While both models are offering advanced features in traffic engineering, the scalability and robustness of the IntServ approach and lack of end-to-end connection guarantee and per-flow QoS setup in DiffServ are the drawbacks of the two approaches.

IPv6 as the most recent version of the Internet protocol (IP), resolve IPv4 depletion problem and bring more scalability for IP based solutions, particularly in the IoT domain where a large number of the devices/things interconnect with each other. IPv6 provides other technical benefits in terms of security and mobility in addition to the larger addressing space. Also, IPv6 design improves network performance and reduces routing time by having a fixed header size. The DiffServ mechanism performance is boosted by IPv6 design because it leverages the Flow Label field in the IPv6 header. The edge router reads the required field values from the same layer and header during the packet classification process and it does not need to go to the transport layer data. However, still, stream assignment to the class of service, and setting the preferences within the class is a manual setup and is a static operation. All those limitations are not suitable for the IoT

 ISBN: 978-1-61208-811-2

system since the Internet of Things is placing new demands on network infrastructure due to the diverse application domains (e.g., smart health, smart city). Under different contexts and domains, IoT applications could adopt different classification methods and QoS policies compared with the current web applications. Moreover, the number of connected devices and the amount of generated data will become an increasing stress on the Internet network. Although there is an option for companies to pay a certain price to make their services reliable and to improve performance that user experienced, the Internet standard still enforces limitations and challenges such as number of QoS traffic classes, complexity, and cost of deployment, and lack of scalability.

Software Defined Networking (SDN) is a potential solution to the problems faced by traditional computer networks like the Internet. Unlike traditional networking system where the traffic control-plane and forwarding-plane are installed and tied together in one single element, SDN technology brings the capability to decouple the control-plane from data-plane to enhance the network evolution, interoperability, and scalability. It allows network owners and administrators to programmatically initialize, control, and manage network behavior by decoupling the control plane from the data-plane.

SDN improves the interoperability between multi-vendor products and removes vendor lock-in state which are common drawbacks in the traditional network. It decreases the overhead of network element configuration and troubleshooting, leading to the optimized capital and operational expenditures for IT enterprises. Additionally, software-driven networks enable the possibility of network control by a software application and make the network management more efficient, quick and flexible. Any network operator could develop customized solutions based on the business need and enforce it through a centralized controller. Beside the flexible and scalable QoS management, the network provider could leverage SDN technology to manage other aspects of the network like security and resource provisioning and configuration, and network monitoring in a more flexible and efficient way [6]–[9]. SDN technology could be deployed either in IPv4- or IPv6-based network, so both IPv6 and SDN could coexist in the network, each bringing different feature for network efficiency.

The main question addressed in this work is how resource allocation can be matched to the IoT application QoS needs while considering the resource capabilities and limitations. Therefore, the goal is to build an adaptive and flexible QoS model which could keep pace with dynamic business and application requirements. To do so, we propose a status-aware and SLA-aware routing mechanism across software-defined communication networks (internet or private network). Unlike similar works, we implement multi-path and load-balancing approaches in the model to enhance network throughput and system availability, along with user experience. It is worth noting that SLA is a formal negotiated agreement between service providers and customers and it can cover many aspects of their relationship such as the performance of services, customer care, billing, and provisioning.

In the classical network, there is no way to fetch the status information directly from the network elements in a centralized and real-time way, and consequently the network topology and the QoS parameters. Therefore, current routing path mechanisms do not consider the current network status e.g., packet loss, delay, or available bandwidth in the path calculation process. Besides, link cost metrics used in the routing mechanism are same for all the application type. But, our model assigns different path for the same data dynamically depending on the current network status and link cost metrics are diverse depending on the application type.

The rest of the paper is organized as follows. In Section II, we provide a review of related studies. In Section III, we present the routing model and provide the mathematical formulation of the problem. Section IV details the experiments and results. We then provide a detailed analysis of the results and finally conclude our work in Section V.

II. PREVIOUS WORKS

Internet of Things continues to evolve and expand in terms of domains, interconnected-devices, data, and applications, and as a result, IoT specifics challenges are getting more bigger and complex [10]. Both research communities and industrial enterprises are working on IoT-specific challenges to solve technological barriers which slow down the evolution of IoT. QoS management as one of the critical subsystem in IoT framework attracted the attentions in research institute. To provide QoS in the IoT, it is necessary to ensure suitable mechanisms at each layer of the IoT since various applications could have the dependency on the different QoS attributes, which must be provided by a specific IoT layer. Various research communities have attempted to define high-level QoS schemes taking into account the multi-layer IoT architecture, service components, enabling technologies, and data classification. Others are proposing solutions in the aspects of the resource identification, routing protocols, clustering, and topology update within any leveraged networking technologies in IoT system.

Wireless Sensor Networks (WSNs), widely used in IoT infrastructure, are comprised of hundreds or thousands of low-ends battery-powered devices. A large number of the studies investigate on resource management within such an constrained environment. Proposed solutions differ on the design methodology and details of the quality factors. Energy and bandwidth efficiency, storage and coverage optimization, or data accuracy enhancement are mostly targeted in those researches [11] [12]. For instance, in [13], the authors present the computational QoS model for WSN routing using the directed graph theory and considering the response time, reliability, and availability as QoS factors. Mostafaei [14] developed the Reliable Routing Distributed Learning Automaton (RRDLA) algorithm to streamline the performance of the wireless sensor network, and therefore reduce energy consumption within it, by means of the modeling and simulation. End-to-end delay, packet delivery rate, network lifetime, and the number of times are parameters modeled in RRDLA.

Middleware-based approach solution [15] is one of the promising approaches to manage QoS within heterogeneous IoT environment. Within proposed model by Heinzelman [15], applications sends QoS requirement to the middleware and then middleware configures networks devices to meet the application expectations. This approach enables the adaptation of the code allocation on the basis of the current application requirements. If application QoS requirements from application are not feasible to fulfill by network resources, middleware

 ISBN: 978-1-61208-811-2

negotiates a new QoS guarantee with both application and network.

Other IoT enabling technologies like the Internet and the cellular access networks (e.g.3G, LTE) have the evolved QoS functions within their closed systems, although their integration into IoT bring issues and limitations. With the realization of the 5G technology, the barriers of the access networks are rectified, since 5G provides less-delay high-bandwidth access network for the IoT system. the continuous researches are ongoing to enhancement QoS within 5G network leveraging mostly learning-based techniques for to adapt system design with IoT use-cases [16]. The QoS magnification within Internet and computer network are involving new mechanisms like Diffserve, InterSev, and MPLS, and technologies like IPv6. As described in previous sections, The Internet still imposes complications in terms of QoS deployment in IoT space and since static QoS differentiation mechanism could not meet the requirements of dynamic and data-centric applications.

The idea of SDN presented new research topics in the literature, not only in computer networks, but also in other networking technology like cellular network and Wireless Sensor Network in more recent years and it is getting more promising vision by the appearance of IoT and also SDN success stories. It led to the widespread adoption of Software-Defined WSNs (SDWSNs)and Software Defined Wireless Network (SDWN) [17]–[19]. As a more recent field, research domain SD-IoT aims to integrate SDN into the IoT framework to improve the system control and management. For instance, the SD-IoT framework model in [6] offers as centralized control system over security services (SDSec), storage services (SDS) and infrastructure resources. Few studies [8] attempt to improve resource utilization in terms of data acquisition, transmission, and processing within the IoT framework. For instance, Ubi-Flow offered in [9] proposes an efficient flow control and mobility management framework in urban multi-network environment using distributed SDN controllers. In [20], the authors propose a traffic-aware quality-of-service routing scheme in Software-Defined Internet of Things network. They exploit features such as flow-based nature, and network flexibility, in order to fulfill QoS requirements of each flow in the network. The authors in [21] propose an application-aware QoS routing algorithm for SDN-based IoT networking to guarantee multiple QoS requirements of high-priority IoT applications and to adapt to the current network status for better routing paths. Although these efforts argue that software-defined technology can facilitate IoT system and resource management, most proposals target high-level architectural and framework enhancement and are more like conceptual and analytical models, and need to be implemented and assessed.

The concept of SD-IoT is in its infancy and standardization efforts in terms of framework, protocol and software-defined applications are still underway [22]. We realized a lack of study in the field of performance management in which heterogeneous devices, network resources, and application needs could be managed in an easy and flexible way. The currently designed solutions are very high-level and mainly focused on improving one or multiple QoS factors in a closed subsystem of IoT. Most works lack of flexibility and scalability considering IoT heterogeneity and dynamic natures in terms of applications and services.

We are working on an middleware-based QoS management framework for IoT application, to control and allocate IoT infrastructure resources within a multi-platform environment including transport and sensing network. Our model takes advantages of SDN characteristics to takes into account the application QoS preferences and network elements status to allocate resources effectively, guaranteeing application productivity and maximizing network performance.

As part of the end to end framework, in the next section, we propose a routing algorithm to determine the best possible path for IoT applications' traffic across the software-driven network. The proposed algorithm is presented in a mathematical model for the route optimization problem between two connected things. The model would be developed as an customized application on top of the SDN controller, and it takes advantage of the SDN technology to consider the network resource status in the route calculation process.

III. ROUTING MODEL

SDN architecture [23] is made of three logical layers. The data plane layer is composed of physical devices that forwards traffic packets, The control plane includes the centralized networking controller, supervises all network traffic and makes decisions about where the traffic must be forwarded. The application layer represents the services that interact with the controller to specify the networking needs of the applications in terms of security, configuration, and management. The communication between the forwarding devices and controller is done through Southbound interfaces. A controller exercises direct control over the states of the data-plane devices via well-defined Application Programming Interfaces (APIs). OpenFlow [24] is the first standard Southbound interface. On the other hand, the Northbound interfaces enable the programmable network functions that tell the controller how to manage the network. Thus, the value of the Northbound interfaces is tied to the innovative and adaptive network services aligned with business and users needs. The customized network services are softwares and could be implemented as plug-ins to the centralized controller or any other standalone environment interacting with the controller through APIs. Supporting APIs in SDN hide the complexity and heterogeneity of the physical infrastructure. East/westbound interfaces are a special case of interfaces required by distributed controllers. They are used to interconnect the SDN architecture with external SDN-based network architectures or legacy networks.

In this work, we assume that SDN technology has been integrated into IoT framework, so that underlying network resources are SDN-enabled and the SDN controller resides in the IoT middleware layer. This schema would take advantages of northbound interfaces to design a centralized routing algorithm to manage IoT specific application needs and accordingly infrastructure resources including network elements and IoT gateways as interface toward low-end sensing devices. The quality of performance metrics taken into account are packet loss rate, delay, and bandwidth as they represent the main QoS metrics.

The proposed routing model is based on the well-known Network Design Problems (NDPs): Multi-Commodity Flow Problem (MCFP) and Constrained-Based Routing (CBR). The term multi-commodity (opposed to a single-commodity) is related to the fact that multiple demands could simultaneously arrive in the system and ask for routing resources in the

 ISBN: 978-1-61208-811-2

network, which is very common in the communication and computer networks, as well as in IoT systems. The objective of the MCFP problem is to flow the different traffic demands from various sources to the distinct destinations through the network at minimum cost without exceeding the network link capacities.

Constraint-based routing denotes a class of routing algorithms where path selection decisions are made based on a set of requirements or constraints, in addition to the destination. These constraints can be imposed either by administrative policies or QoS requirements. We consider the network QoS status as well as application QoS constraints in our formulation. The objective is a minimum-cost feasible solution for the constraint-based routing problem to find the cheapest possible way of sending a certain amount of flow through the network. The decision making framework and QoS routing algorithm are illustrated in Figure 1 and Algorithm 1, respectively. Operation mode of algorithm and element details are provided as we describe the mathematical model.

We suppose a network of interconnected nodes where each link has a dedicated capacity. We also assume that all network nodes are OpenFlow-enabled and connected to one centralized SDN controller [25]. The SDN-based network can be represented by a strongly connected graph G = (V,E) where V = {1, 2, ..., v} denotes the set of nodes (OpenFlow-enabled network elements) and E =$\{(i,j) : i,j \in V, i \neq j\}$ denotes the set of edges, also referred to the bi-directional links between OpenFlow network elements. Each link (i, j) has the associated maximum bandwidth B_{ij}, available bandwidth b_{ij}, delay d_{ij}, and packet loss ratio pl_{ij}. In our context, the delay represents the total link delay consisting of processing, propagation, transmission, and queueing delay. *Network Topology and Link Status* refers to those parameters, and could be stored in centralized database, called *Topology Database*.

On the other hand, K = {1, 2, ..., k} and $\mid K \mid = k$, represent the set of different commodities, also determined as *IoT application demands* in decision-making framework, to be routed on the network graph. For each demand $k \in K$, three parameters are given: S^k as the source of the demand, T^k as the destination of the demand, and F^k as the positive demand volume. Demand volume represents either the traffic volume or the required bandwidth between a pair of nodes and the unit of the demand volume needs to be consistent with the unit of link capacities. *SLA-based Application QoS Database* elements refer to D^k_{SLA}, PL^k_{SLA}, and B^k_{SLA} representing the acceptable values of delay, packet loss ratio, and average required bandwidth respectively, which are agreed in the application-service provider SLA for IoT service *k*. The parameters $S^k, T^k, F^k, D^k_{SLA}, PL^k_{SLA}$, and B^k_{SLA} are considered as the input to the routing path calculation algorithm. The algorithm takes the information about each demand as well as the network link information (b_{ij}, d_{ij}, and pl_{ij}) and calculates the best possible path p^k for each new-arrival demand *k* between the source and destination across the SDN network with minimum cost flow. Network topology and link status information are regularly gathered and updated by the SDN controller. According to the OpenFlow specification v1.0, Topology Discovery function as the defacto standard function is implemented in all controllers. This function enables the controller to discover a network topology of the entire SDN infrastructure. To calculate Link QoS status, we can take advantage of the implemented counters in the OpenFlow-enabled network element. Those counters are stored packet processing/statistical records in particular tables in flow pr port basis. The system could provide multiple paths for any demand *k* aiming not to violate the accepted QoS level by any demands. All determined paths for service *k* are from source S^k to destination T^k so that each one routes a portion of the whole demand volume F^k. We assume that there exists no pair of flows with the same origin and destination.

Objective function: The objective is to route application flows in the network with minimum cost in respect to the particular cost metrics for each application. Equation (1) represents the objective function where C_{ij} is the unit cost of link (i, j) and X^k_{ij} as a variable represents the amount of volume corresponding to the demand *k* to be routed on the link (i, j).

$$Minimize \sum_{(i,j)\in E} \sum_{k} C_{ij} X^k_{ij} \tag{1}$$

The link cost metric in our model is represented as a weighted sum of the available link bandwidth, packet loss ratio, and delay, as per (2), where C_{ij} expresses the cost of link (i, j), metrics b_{ij}, pl_{ij} , and d_{ij} refer to the available link bandwidth, packet loss ratio, and delay in link (i, j), respectively; all dynamically calculated based on the current network status monitored by the controller.

$$C_{ij} = \alpha \times b_{ij} + \beta \times pl_{ij} + \gamma \times d_{ij} \tag{2}$$

The coefficient α, β, and γ as scaling factors have the relation expressed in (3). So, each metric can have different weight to give a priority to a particular one. Regarding our application classification based on traffic sensitivity to the delay and bandwidth, we could define a diverse weight for each metric in each particular application class. *Application classifier* within decision-making framework could provide dynamic input to differentiate application classes and enforce the result in traffic prioritization and Queuing mechanisms.

$$\alpha + \beta + \gamma = 1, 0 \leq \alpha, \beta, \gamma \leq 1 \tag{3}$$

Among the quality-related metrics used for link cost formulation, bandwidth is positive so that the higher value means the higher service quality. Delay and packet loss ratio are negative, meaning that the higher the value, the lower the service quality. Additionally, each metric has a different unit. Delay unit is in *seconds*, the bandwidth unit is *bps*, and loss ratio is a digit represented in *percentage*. To express a weighted sum of independent metrics, values of those metrics need to be adjusted theoretically to a common scale. We use the feature scaling method [26] to normalize the range of those independent metrics. This method re-scales the range of all values and brings them all into the range [0, 1]. Equations (4) to (6) present the normalization formulas for each metric. To set a boundary for each variable, maximum and minimum ranges could either have a constant value or adjust dynamically based on the network topology information at any given time.

$$b'_{ij} = \frac{b_{max} - b_{ij}}{b_{max} - b_{min}}, b_{min} \leq b_{ij} \leq b_{max} \tag{4}$$

 ISBN: 978-1-61208-811-2

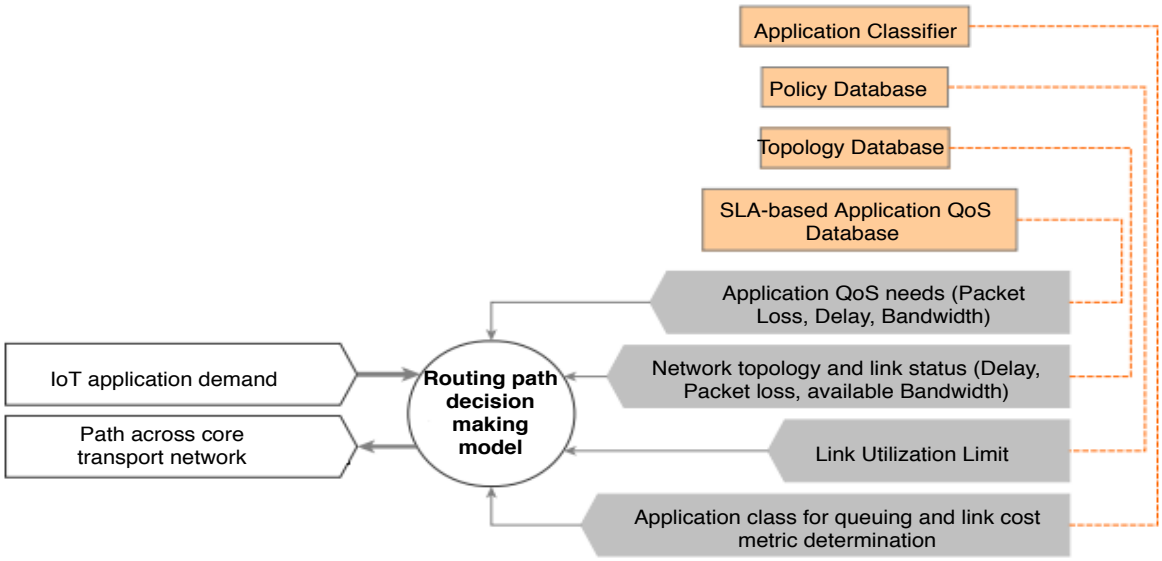

Figure 1. Routing path decision making framework - Parameters (gray boxes) and components (orange boxes) which define the input parameters.

Input: $G = (V, E)$ as network topology: $V = \{1, 2, ..., v\}$ as nodes and $E = \{(i, j) : i, j \in V, i \neq j\}$ as bidirectional links
Input: (S^k,T^k, F^k) per each demand k in K: $S^k \in V$ as Source of demand, $T^k \in V$ as Destination of demand k, $F^k \geq 0$ as Total demand volume, $[Mbps]$
Output: P^k: Routing path across network for demand k
for (i, j) in E **do**
Get network link QoS parameters ($b_{ij} \geq 0$, $pl_{ij} \geq 0$, $d_{ij} \geq 0$, $B_{ij} > 0$;
Calculate the current link utilization rate ;
Get the link utilization limit $u_{Threshold}$;
if Link utilization rate $\geq u_{Threshold}$ **then** ;
Exclude this link from the logical network topology to calculate paths for current active demands ;
end;
end ;
for k in K **do**
Get Max acceptable delay D^k_{SLA}, Max acceptable packet loss PL^k_{SLA}, and Min required bandwidth B^k_{SLA} for each demand k in K from SLA-DB ;
Set the link cost metrics depending on the application class, if any ;
end;
for k in K **do**
Calculate the best-fit path per each demand k in K based on the proposed mathematical model;
end;

Figure 2. Proposed QoS support routing algorithm

$$pl'_{ij} = \frac{pl_{ij} - pl_{min}}{pl_{max} - pl_{min}}, pl_{min} \leq pl_{ij} \leq pl_{max} \quad (5)$$

$$d'_{ij} = \frac{d_{ij} - d_{min}}{d_{max} - d_{min}}, d_{min} \leq d_{ij} \leq d_{max} \quad (6)$$

Constraint function: we introduce the several types of conditions which are imposed by the network and application.

• Path delay constraint for each demand is defined in (7), where d^k_p is the end-to-end delay for the routing path p^k determined for demand k, and D^k_{SLA} is the maximum acceptable delay for demand k agreed in SLA.

$$d^k_p \leq D^k_{SLA} \quad (7)$$

Delay metric is an additive metric. So, total delay p_p in the path p from one source to a destination is calculated by the summing of the delay on each link (i, j) across the path, formulated in (8).

$$d_p = \sum_{(i,j) \in p} d_{ij} \quad (8)$$

So, (7) can be replaced by Equation (9):

$$\sum_{(i,j) \in E, P^k} d_{ij} \leq D^k_{SLA}, \forall k \in K \quad (9)$$

• Constraint for path packet loss ratio in each demand is defined in (10), where pl^k_p is the total packet loss ratio for the routing path p^k determined for demand k and PL^k_{SLA} is the maximum acceptable packet loss ratio for demand k agreed in SLA.

$$pl^k_p \leq PL^k_{SLA} \quad (10)$$

 ISBN: 978-1-61208-811-2

Packet loss is a multiplicative metrics. Packet loss ratio along the path is determined in (11), where pl_p refers to the packet loss ratio for path p and pl_{ij} refers to the packet loss ratio for a particular link (i, j) across the path p:

$$pl_p = 1 - \prod_{(i,j)\in p} (1 - pl_{ij}) \tag{11}$$

If the packet loss ratio in the network link is very small and close to zero, packet loss measure could be considered as an additive measure and can be approximately simplified by (12).

$$pl_p = \sum_{(i,j)\in p} pl_{ij} \tag{12}$$

So, (10) can be expressed by (13):

$$\sum_{(i,j)\in E, P^k} pl_{ij} \leq PL^k_{SLA}, \forall k \in K \tag{13}$$

• Link capacity constraint is formulated in (14). In multi-commodity environment, each link could be part of multiple routing paths used by different commodities. Then, the summation of the volume of the different commodity in any link (i, j) must be less than the available link bandwidth b_{ij}, which is measurable in the SDN transport network. This constraint could fulfill the context of the congestion management in the network.

$$\sum_{k\in K} X^k_{ij} \leq b_{ij}, \forall (i,j) \in E \tag{14}$$

• Link utilization/load balancing constraint. To choose the optimized and well-fit path for a given traffic, we consider the link utilization constraint in the path allocation process. Since link bandwidth as one of link status information is monitored and stored by controller in network topology database, link utilization rate on any given link (i, j) can be measured by dividing the current link bandwidth with maximum link capacity, expressed in (15) in a unit of percentage.

$$Utilization\ Ratio\ (i,j) = \frac{B_{ij} - b_{ij}}{B_{ij}} \tag{15}$$

Generally, the link with utilization rate 70% 80% is susceptible to congestion [27]. To balance traffic across links and to avoid congestion, we define a limit for the link utilization rate so that the proposed routing algorithm excludes links with link utilization rate higher than this limit from the path calculation scenario. The link utilization limit can be provided in a policy database accessed by the controller. This limit can be defined either as a constant value given by network providers or as a dynamically adjustable value. In the later approach, the controller applies a developed logic considering the traffic volume, demand arrival rate, and network status at any given time. We formulate the load balancing constraint with (16), where $\cup_{Threshold}$ is the link utilization limit:

$$\sum_{k\in K} X^k_{ij} \leq b_{ij} + (\cup_{Threshold} - 1) * B_{ij}, \forall (i,j) \in E \tag{16}$$

Regarding the delay/bandwidth threshold, unlike IT applications which are categorized based on the type of traffic, IoT application could be classified from different perspectives such as the type of information they manage, the type of recipient (person or system oriented), and the criticality level of services they provide. Currently, IoT systems still suffer from the lack of standardized mechanisms to represent the diverse application requirements, also as reference for our proposed model. To setup the reasonable thresholds for QoS metrics, we plan to do experiments with multiple applications within end to end deployed framework, also to gather data from production applications, and to calculate the average value for those metrics.

• The flow conservation law is formulated with (17). It states that the total incoming flow into each node in the network equals the total outgoing flow from that node, except for the source and destination nodes of flow. It also guarantees the same bandwidth in all links across the determined path for a particular demand.

$$\sum_{(i,j)\in E} X^k_{ij} - \sum_{(j,i)\in E} X^k_{ji} = \begin{cases} F^k, i = S^k \\ -F^k, i = T^k \\ 0, i \neq S^k \& T^k \end{cases} \tag{17}$$

In the offered mathematical formulation, the number of variables is $|E||K|$ and the number of constraints is $|E||K| + |K| + |E|$. Since the number of the QoS parameters used in our model is more than one, it is proven to be *N*ρ-complete [28] as the complex problem. The complexity of standard Dijkstra algorithm is O(v^2), and with more efficient implementation in link-state protocols like Open Shortest Path First (OSPF) will be O($vlogv$).

Compared with OSPF in which computational requirements of calculating link state information rise rapidly exponentially/logarithmically as the size and complexity of the network increase, the complexity of our proposed algorithm is driven by the multiplication of number of links and nodes. Theoretically, it can be concluded that the proposed algorithm could provide less convergence time as could be the most-fitted solution for mission-critical IoT applications. However practical analysis would be needed to provide more insight about the improvement level.

IV. Experiments and results

To investigate the feasibility and the performance of the proposed model, we implemented it in A Mathematical Programming Language (*AMPL*). According to our mathematical formulation, the number of variables is $|E|$ $|K|$ and the number of constraints is $|E|$ $|K|$ + $|K|$+$|E|$. This problem is $N\rho - complete$ due to the number of QoS parameters [61]. Since the objective function and all constraints are in the linear format and only the values of the variable X could be discrete, this model is classified as Mixed Integer Programming (MIP) problem. Thus, we paired *AMPL* with the *CPLEX* solver to solve the problem. *CPLEX* uses branch-and-bound algorithm to find the optimal solution for Mixed Integer Programming problem.

Bellman-Ford and *Dijkstra* are the two main algorithms used in a weighted graph to compute the shortest paths from a single source to a destination. In the current packet switching

network, Bellman-Ford algorithm has enabled the development of distance-vector routing protocols while the Dijkstra algorithm introduces link state routing protocols. In both, each router learns about remote networks from the neighbor routers or the configuration to build the routing table. Link state routing protocols enable a router to build and track a full map of all network links while distance vector protocols work with less information about the network area. Since our routing model is contextually similar to link-state protocols, we aim to evaluate the performance of the proposed routing model with the Open Shortest-Path First (OSPF), which is one of the well-established and widely-adopted link-state routing protocols. OSPF considers link bandwidth as the link cost metric to calculate the routing. The link cost calculation formula in OSPF is determined by (19):

$$Interface\ cost = \frac{Reference\ bandwidth}{Interface\ bandwidth} \tag{18}$$

In Cisco products, the default reference bandwidth value in OSPF is 100 Mbps (108bps). Hence, we have the following equation as the cost of link (i, j):

$$Cost_{ij} = \frac{100}{B_{ij}} \tag{19}$$

The QoS requirements of IoT applications are not clearly defined because of the data-oriented and diverse needs. To facilitate the implementation of our experiment environment, we map the IoT application classes onto the IoT data delivery model as in Table II and we assign dynamic cost metrics for the different application classes of Table I.

In Table I, we present three different queues in each port of the OpenFlow-enabled network elements: (1) Priority Queue (PQ) as the most prioritized queue includes mission-critical applications with intensive delay sensitivity. If the delay requirement of application is less than a pre-defined threshold, it is marked as a high-prioritized demand and it is inserted in PQ of egress ports of network elements. (2) Q1 represents the data-centric application with bandwidth sensitivity, and less delay-sensitive compared to the pre-defined threshold. (3) Q2 contains applications with no strict QoS requirements, also called the best-effort application.

For the delay-sensitive application, we set both packet loss and delay as the metrics and for the bandwidth-sensitive application, packet loss and bandwidth are considered as link cost metrics. For the Best Effort (BE) application, either the combination of packet loss, delay, and bandwidth would be used in our routing model or the traditional less-complex best-effort routing algorithm could be applied.

In the experiment, we characterize the application from different classes (delay-centric and BW-centric) with different quality requirements for each network performance metric and we calculate delay, packet loss, and link utilization rate of the paths determined by our model. The results are compared with the characteristics of the calculated paths by OSPF routing model. The experiment scenario is visualized in Figure 3.

Multiple network topologies (Topology A, B, C, and D-elaborated in Tables III, IV, V, and VI, respectively) designed to run the test ensure the path diversity between any pair of nodes. We define the network topology and assign the

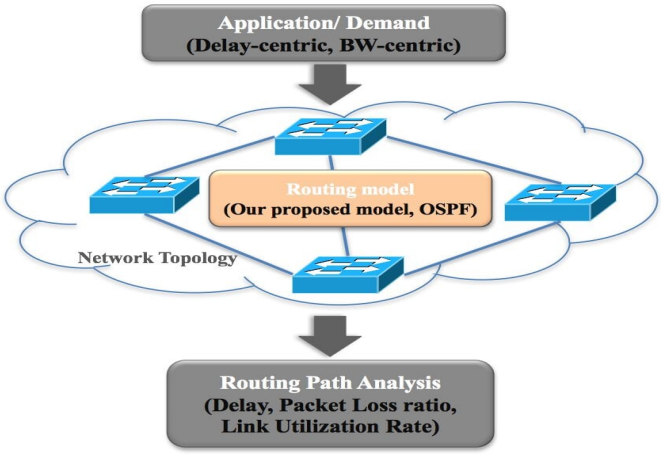

Figure 3. Experiment and performance analysis scenario.

maximum capacity, available bandwidth, delay, and packet loss ratio for network links. Also, we present the service demands specifying the source, destination, and volume as well as the QoS requirements in terms of delay, packet loss, and minimum bandwidth. The simulated demands are directed towards the bottlenecks to investigate delay and throughput of the Delay-centric and Bandwidth-centric traffic, respectively. Both single-commodity and multi-commodity scenarios are investigated under the same network situation. In the single demand scenario, we define an individual demand, while in the multi-demand scenario, we create more stress on the network by defining multiple simultaneous demands.

1) Result analysis - Delay: The path delay and packet loss ratio for all delay-centric demands characterized in the different topologies are demonstrated in Table VII and Table VIII for single-commodity and multi-demand scenario, respectively. Referring to the results, we observe that our model finds the optimized routing paths in terms of the delay and packet loss for the delay-centric demand compared to the OSPF. Our model can enhance the performance and efficiency of delay-centric applications. Event-driven IoT applications in smart cities are often mission-critical and delay intolerant, such as the emergency signals and safety related applications. To be effective, the information should be transmitted in a limited time frame.

On the other hand, BW-centric demands are not concerned about the delay and our model looks for the links with the optimized high available bandwidth and lower packet loss rate. The results depicted in Tables IX and X show the delay associated with the paths calculated by our model. It is almost lower than the delay associated with the paths calculated by OSPF for BW-centric applications, in both single and multi-commodity environment. Since we set the delay constraint in our mathematical model for all application classes, our model aims to find the best-fit path with the acceptable level of delay for BW-centric application depending on the network status at any given time.

SDN-based middleware in the networking layer makes our model dynamically aware of the network status and have access to the SLA-related application QoS databases. Besides,

 ISBN: 978-1-61208-811-2

TABLE I. APPLICATION CLASSIFICATION AND QUEUEING POLICY IN OPENFLOW NETWORK ELEMENT: D^k_{max} AS MAXIMUM ACCEPTABLE DELAY FOR SERVICE *k*, BWk_{min} AS THE MINIMUM REQUIRED BW FOR SERVICE *k*, $D_{Threshold}$ AS THE DELAY THRESHOLD, $BW_{Threshold}$ AS THE BW THRESHOLD

Application Class	QoS attributes	Priority	Type of queue	Traffic Class mapped with Cisco classification
Delay-Centric (Mission Critical)	$D^k_{max} \leq D_{Threshold}$	1	PQ (Priority Queue)	EF (Expedited Forwarding)
Bandwidth-Centric (Multimedia application)	$D^k_{max} \geq D_{Threshold}$, $BW^k_{min} \geq BW_{Threshold}$	2	Q1	AF (Assured Forwarding)
General (Non-Real time analytic application)	No strict QoS needs	3	Q2	BE (Best Effort)

TABLE II. MAPPING IoT APPLICATION CLASSIFICATION AND LINK COST METRICS

IoT Application	Application class	Link cost metric
Mission-critical, Event-related application	Delay-centric	Delay and Packet-loss rate
Continuous application (Query-driven, Real-time monitoring)	Bandwidth-centric	Bandwidth and Packet-loss rate
General application (Non-real time monitoring)	BE	All three metrics

TABLE III. TOPOLOGY-A LINK CONFIGURATION

Link	Max BW(Mbps)	PacketLoss	Delay(ms)	Available BW(Mbps)	Link	Max BW(Mbps)	PacketLoss(%)	Delay(ms)	Available BW(Mbps)
(1,2)	300	1%	0.01	300	(2,6)	400	2%	0.01	200
(1,4)	400	1%	0.02	400	(3,6)	600	2%	0.05	400
(1,5)	200	2%	0.01	200	(4,5)	400	1%	0.01	300
(2,3)	600	2%	0.02	300	(5,6)	300	1%	0.01	300
(2,5)	600	2%	0.05	200					

TABLE IV. TOPOLOGY-B LINK CONFIGURATION

Link	Max BW(Mbps)	PacketLoss(%)	Delay(ms)	Available BW(Mbps)	Link	Max BW(Mbps)	PacketLoss(%)	Delay(ms)	Available BW(Mbps)
(1,2)	400	1	0.01	200	(4,7)	600	1	0.01	200
(1,4)	600	2	0.02	300	(5,6)	400	1	0.01	300
(2,3)	300	1	0.02	300	(5,7)	300	1	0.01	200
(2,3)	200	2	0.01	150	(5,8)	200	2	0.02	100
(2,5)	300	1	0.02	300	(6,8)	300	1	0.01	200
(3,5)	400	1	0.05	200	(6,9)	300	2	0.02	1500
(3,6)	600	2	0.03	400	(7,8)	400	2	0.03	400
(4,5)	200	2	0.01	100	(8,9)	600	2	0.02	400

TABLE V. TOPOLOGY-C LINK CONFIGURATION

Link	Link BW (Mbps)	Packet loss(%)	Delay(ms)	Available BW(Mbps)	Link	Link BW(Mbps)	Packet loss(%)	Delay(ms)	Available BW(Mbps)
(1 2)	500	1	0.01	300	(5 6)	600	1.5	0.01	400
(1 4)	600	2	0.02	300	(5 12)	300	0	0.01	300
(1 5)	200	1.5	0.05	100	(6 10)	500	1.5	0.05	300
(2 3)	600	1.5	0.02	300	(7 8)	600	1.5	0.02	150
(2 5)	300	1.5	0.05	150	(7 10)	400	2	0.05	200
(2 6)	400	1.5	0.01	200	(8 9)	400	1.5	0.01	200
(2 9)	400	0	0.01	200	(8 11)	600	1.5	0.01	300
(3 6)	600	1.5	0.05	500	(10 11)	400	1.5	0.01	200
(3 7)	600	2	0.05	500	(11 12)	300	1	0.02	200
(4 5)	400	0	0.01	300					

this information is directly applied to the resource allocation process. Application sensitivity to delay or bandwidth leads to having different link cost metrics. For the mission-critical application, it seeks for the delay-less and loss-less paths, while meeting the capacity constraints, and the outcome is the least cost (least-delay) and SLA-respected paths. Also, the framework structure allocate dynamic paths regarding the current network link conditions, in case of any change in the network link status, the model will be notified and new QoS-respected paths are calculated for the flows based on

TABLE VI. TOPOLOGY-D LINK CONFIGURATION

Link	Link BW (Mbps)	Packet loss(%)	Delay(ms)	Available BW(Mbps)	Link	Link BW(Mbps)	Packet loss(%)	Delay(ms)	Available BW(Mbps)
(1 2)	400	1	0.01	200	(7 10)	600	1	0.01	400
(1 4)	600	2	0.02	300	(8 9)	300	2	0.02	200
(2 3)	300	1	0.02	300	(8 10)	200	1	0.01	200
(2 4)	200	2	0.01	150	(8 11)	300	2	0.01	100
(2 5)	300	1	0.02	300	(9 11)	400	1	0.01	200
(3 5)	400	1	0.05	200	(9 12)	600	2	0.02	400
(3 6)	600	2	0.01	400	(10 11)	600	1	0.01	400
(4 5)	600	2	0.03	400	(10 13)	300	2	0.02	250
(4 7)	200	1	0.01	100	(11 12)	200	1	0.01	100
(5 6)	600	2	0.01	400	(11 13)	200	2	0.02	200
(5 7)	300	1	0.01	200	(11 14)	300	1	0.01	200
(5 8)	200	2	0.02	100	(12 14)	200	2	0.02	100
(6 8)	300	1	0.01	200	(12 15)	600	1	0.01	200
(6 9)	600	2	0.02	400	(13 14)	300	2	0.01	200
(7 8)	400	2	0.02	400	(13 15)	300	2	0.02	100

TABLE VII. DELAY AND PACKET LOSS RATE: DELAY-CENTRIC APPLICATION IN SINGLE-DEMAND SCENARIO

		Single-Demand			
		Our model		OSPF	
		Delay (ms)	Loss rate (%)	Delay (ms)	Loss rate (%)
Topology-A	Delay-centric 1	0.06	3	0.07	4
	Delay-centric 2	0.03	2	0.06	3
	Delay-centric 3	0.05	4	0.08	5
Topology-B	Delay-centric 1	0.04	5	0.06	5
	Delay-centric 2	0.02	2	0.04	3
	Delay-centric 2	0.06	5	0.08	7
Topology-C	Delay-centric 1	0.03	2	0.1	6
	Delay-centric 2	0.05	4	0.11	4
	Delay-centric 3	0.03	1	0.07	3
	Delay-centric 4	0.03	3	0.08	5

TABLE VIII. DELAY AND PACKET LOSS RATE: DELAY-CENTRIC APPLICATION IN MULTI-DEMAND SCENARIO

			Multiple-Demand			
			Our model		OSPF	
			Delay (ms)	Loss rate (%)	Delay (ms)	Loss rate (%)
Topology A	Test1	Delay-centric 1	0.03	2	0.06	3
	Test2	Delay-centric 1	0.03	2	0.06	3
		Delay-centric 2	0.05	4	0.08	5
	Test3	Delay-centric 1	0.03	2	0.06	3
Topology B	Test 1	Delay-centric 1	0.04	5	0.06	6
		Delay-centric 1	0.03	2	0.05	3
	Test2	Delay-centric 1	0.04	5	0.06	6
		Delay-centric 2	0.03	2	0.05	3
	Test3	Delay-centric 1	0.04	5	0.06	6
		Delay-centric 2	0.03	2	0.05	3
Topology D	Test1	Delay-centric 1	0.04	4	0.08	8
		Delay-centric 2	0.04	4	0.06	6
	Test2	Delay-centric 1	0.04	4	0.08	8
		Delay-centric 2	0.04	4	0.06	6
		Delay-centric 3	0.02	2	0.04	4
	Test3	Delay-centric 1	0.04	4	0.08	8
		Delay-centric 2	0.04	4	0.07	7
		Delay-centric 3	0.02	2	0.04	4

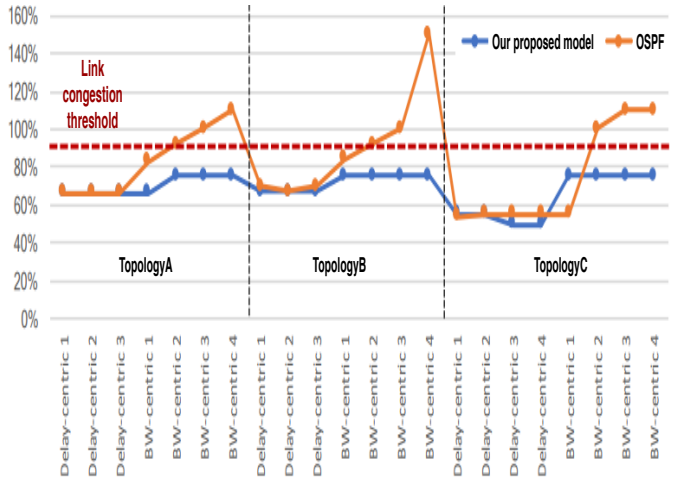

Figure 4. Maximum link utilization across network links in single-demand scenario.

TABLE IX. DELAY AND PACKET LOSS RATE: BANDWIDTH-CENTRIC APPLICATION IN SINGLE-DEMAND SCENARIO

		Single-Demand			
		Our model		OSPF	
		Delay	Packet lost rate(%)	Delay	Packet loss rate(%)
Topology A	BW-centric1	0.05	4	0.08	5
	BW-centric2	0.5	4	0.08	5
	BW-centric3	0.07	4	0.08	5
	BW-centric4	0.07	4	0.08	5
Topology B	BW-centric1	0.04	3	0.06	4
	BW-centric2	0.06	3	0.07	3
	BW-centric3	0.06	3.5	0.07	3
	BW-centric4	0.06	3.5	0.06	6
Topology C	BW-centric1	0.06	3	0.06	3
	BW-centric2	0.03	2.5	0.1	6
	BW-centric3	0.08	2.8	0.1	6
	BW-centric4	0.08	3.5	0.1	6

new network situation. Thus, we can state that the application effectiveness and also customer satisfaction are reinforced in the offered QoS routing. In OSPF, the routing approach is to forward the demand through the links with the highest maximum capacity. The high-bandwidth links could not be always considered as the less-delay links since link delay is affected by other factors such as queueing and congestion.

2) Result analysis - throughput: To assess the network throughput in our model, we measure the maximum link utilization rate across the network after allocating the paths for the demands. Figures 4 and 5 demonstrate the maximum link utilization rate in all experiments for single and multi-demand scenarios, respectively.

 ISBN: 978-1-61208-811-2

TABLE X. DELAY AND PACKET LOSS RATE: BANDWIDTH-CENTRIC APPLICATION IN MULTI-DEMAND SCENARIO

			Multiple-Demand			
			Our model		OSPF	
			Delay (ms)	Loss rate (%)	Delay (ms)	Loss rate (%)
Topology A	Test1	BW-centric1	0.05	4	0.08	5
	Test2	BW-centric1	0.05	4	0.08	5
		BW-centric2	0.06	3	0.07	4
	Test3	BW-centric1	0.06	3	0.07	4
Topology B	Test 1	BW-centric1	0.03	3	0.07	4
	Test2	BW-centric1	0.03	3	0.07	4
		BW-centric2	0.07	6	0.08	8
	Test3	BW-centric1	0.03	3	0.07	4
		BW-centric2	0.07	6	0.08	8
		BW-centric3	0.02	2	0.02	2
Topology D	Test1	BW-centric1	0.06	5	0.07	7
	Test2	BW-centric1	0.06	5	0.07	7
		BW-centric2	0.08	4	0.08	8
	Test3	BW-centric1	0.06	5	0.07	7
		BW-centric2	0.08	4	0.08	8
		BW-centric3	0.04	5	0.02	3

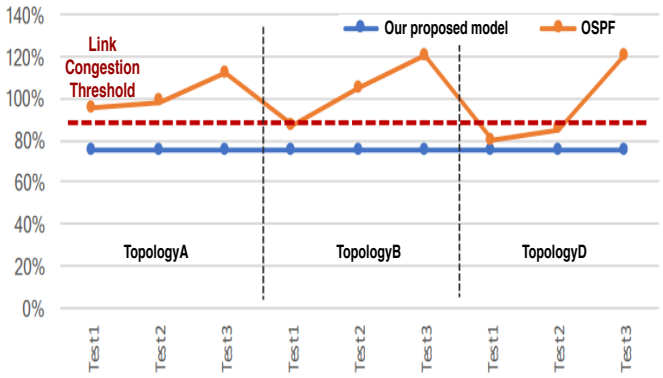

Figure 5. Maximum link utilization across network links in multi-demand scenario.

For the link utilization limit in our load balancing constraint, we set the predefined limit as 75%. Therefore, in the graphs, it can be seen that our model keeps the link utilization rate stable with the maximum 75% while respecting to the other constraints. On the contrary, we could see that the maximum link utilization rate across the network with the same configuration exceeds 100% when running the OSPF routing protocol. When link utilization exceeds 100% in theory (and about 85% in practice), the link is considered congested. Consequently, the demands passing through the congested link could not be served as desired and they may suffer from more delay and loss. Obviously, the rate of the congestion is increased in the multi-commodity environment with the arrival of more bandwidth-intensive demands. The congestion costs a lot for the delay sensitive applications and they might not fulfill their missions depending on the extent they are impacted and delayed. Besides, our model is aware of the current link utilization rate and it excludes the links with the load more than the predefined limit from the path calculation process. This can be considered a congestion prevention method so that the links with the highest available bandwidth and less utilization rate are discovered by our model to avoid the congestion and balance the load across the network links. Our model could be designed in the way to apply different link utilization limits depending on the network status and demand arrival rate to meet application QoS needs.

In our model, when multiple BW-intensive demand requests for data transfer service, the multipath approach is applied if one path could not provide the requested bandwidth. The BW-intensive applications are delay-tolerant compared to the delay-centric applications. Since the proposed model is aware of the currently available link bandwidth, it seeks to direct the flow toward the links with the higher available bandwidth which cost less.

Differently, OSPF does not have access to the currently available bandwidth and the utilization rate. It directs the demand flows toward the high capacity links and the same path could be assigned for a particular demand, independent of the current network status. So, it could cause congestion and failure in the high-load links. Consequently, the demand performance through the failed links are impacted in terms of the delay and loss. The effectiveness of the delay-sensitive demands passing through the congested link might be degraded by undesired delay, or even the transferred data could be useless because of its late arrival. To make it clear, we demonstrate the details of the paths calculated for experiment Test3-Topology B in which we characterize multiple demands (two delay-centric and three BW-centric) arriving the network. It can be seen that we have congestion in two links (4-7) and (5-7) which transfer data for four demands. Two of them are delay-centric demands and they could be impacted by the link congestion. Noting that to have the multi-path approach in OSPF networks, we need to implement load-balancers across the network. Also, to avoid the congestion across the OSPF-based network, the QoS mechanisms such as queueing and congestion control methods should be implemented in all the network elements. Though the queueing and priority scheduling policies could impact the behavior of the system in order to determine what demands to be removed in the congestion situation, the active demands and the network situation have the major impacts on the consequences. In general, the implementation of QoS mechanisms across a large network is resource-intensive, time intensive, and a complex task. It is error-prone because of the technical resource involvement. Moreover, in OSPF, the speed of convergence and the system adaptability to application needs and network changes are low.

V. CONCLUSION

The large number of IoT devices enable a wide variety of services in many different application domains such as smart city, smart transport, smart home, and smart health. There must be QoS approaches at every layer of the IoT architecture to ensure an acceptable level of performance especially for safety applications.

The main contribution of this work is the proposition of a flexible and programmable control layer to provide customized QoS support services for IoT applications. It is achieved by the integration of SDN technology into IoT system architecture. This scheme overcomes the challenge of the dynamic and diverse definition of the SLA and application QoS in the IoT. We proposed a status-aware and SLA-aware routing mechanism across the core transport software-defined network. SDN technology enables, in one hand, the possibility of real time

 ISBN: 978-1-61208-811-2

monitoring of network statistical information and calculating network status, and in the other hand, to get application requirements dynamically to determine the best-fit routing path for the data transmission. Multi-path and load-balancing approaches implemented in the model lead to enhancing the network throughput and increasing system availability, which is crucial for mission-critical applications. All those features in addition to the capability of enforcing different cost metrics for different application types, our model seeks to find the less-delay and less-loss rate paths not only for delay-centric applications but also for bandwidth-centric applications. From the scalability and flexibility aspects, the system could be evolved interfacing with different routing algorithms to be applied to different application types in different network situations.

REFERENCES

[1] T. Kurakova, "Overview of the internet of things," *Proceedings of the Internet of things and its enablers (INTHITEN)*, pp. 82–94, 2013.

[2] F. Y. Okay and S. Ozdemir, "Routing in fog-enabled iot platforms: A survey and an sdn-based solution," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4871–4889, 2018.

[3] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[4] R. Braden, D. Clark, and S. Shenker, "Rfc1633: Integrated services in the internet architecture: an overview," pp. 1–33, 1994.

[5] S. Blake *et al.*, "An architecture for differentiated services," Tech. Rep., 1998.

[6] Y. Jararweh *et al.*, "Sdiot: a software defined based internet of things framework," *Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 4, pp. 453–461, 2015.

[7] J. Liu, Y. Li, M. Chen, W. Dong, and D. Jin, "Software-defined internet of things for smart urban sensing," *IEEE communications magazine*, vol. 53, no. 9, pp. 55–63, 2015.

[8] N. Bizanis and F. A. Kuipers, "Sdn and virtualization solutions for the internet of things: A survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

[9] D. Wu, D. I. Arkhipov, E. Asmare, Z. Qin, and J. A. McCann, "Ubiflow: Mobility management in urban-scale software defined iot," in *Computer Communications (INFOCOM), 2015 IEEE Conference on*. IEEE, 2015, pp. 208–216.

[10] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[11] B. Bhushan and G. Sahoo, "Routing protocols in wireless sensor networks," in *Computational intelligence in sensor networks*. Springer, 2019, pp. 215–248.

[12] I. Snigdh and N. Gupta, "Quality of service metrics in wireless sensor networks: A survey," *Journal of The Institution of Engineers (India): Series B*, vol. 97, no. 1, pp. 91–96, 2016.

[13] Z. Ming and M. Yan, "A modeling and computational method for qos in iot," in *2012 IEEE International Conference on Computer Science and Automation Engineering*. IEEE, 2012, pp. 275–279.

[14] H. Mostafaei, "Energy-efficient algorithm for reliable routing of wireless sensor networks," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 7, pp. 5567–5575, 2018.

[15] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for internet of things: a survey," *IEEE Internet of things journal*, vol. 3, no. 1, pp. 70–95, 2015.

[16] S. Zafar, S. Jangsher, O. Bouachir, M. Aloqaily, and J. B. Othman, "Qos enhancement with deep learning-based interference prediction in mobile iot," *Computer Communications*, vol. 148, pp. 86–97, 2019.

[17] J. Ordonez-Lucena *et al.*, "Network slicing for 5g with sdn/nfv: Concepts, architectures, and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, 2017.

[18] E. Coronado, S. N. Khan, and R. Riggio, "5g-empower: A software-defined networking platform for 5g radio access networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 715–728, 2019.

[19] H. Mostafaei and M. Menth, "Software-defined wireless sensor networks: A survey," *Journal of Network and Computer Applications*, vol. 119, pp. 42–56, 2018.

[20] N. Saha, S. Bera, and S. Misra, "Sway: Traffic-aware qos routing in software-defined iot," *IEEE Transactions on Emerging Topics in Computing*, p. 1, 2018.

[21] G.-C. Deng and K. Wang, "An application-aware qos routing algorithm for sdn-based iot networking," in *2018 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2018, pp. 00 186–00 191.

[22] S. K. Tayyaba, M. A. Shah, O. A. Khan, and A. W. Ahmed, "Software defined network (sdn) based internet of things (iot): A road ahead," in *Proceedings of the International Conference on Future Networks and Distributed Systems*. ACM, 2017, p. 15.

[23] ONF, "Software-defined networking: The new norm for networks," Open Networking Foundation, Tech. Rep., April 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[24] N. McKeown *et al.*, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[25] A. Shalimov *et al.*, "Advanced study of sdn/openflow controllers," in *Proceedings of the 9th central & eastern european software engineering conference in russia*. ACM, 2013, p. 1.

[26] G. Joel, "Data science from scratch," 2015.

[27] S. Song, J. Lee, K. Son, H. Jung, and J. Lee, "A congestion avoidance algorithm in sdn environment," in *2016 International Conference on Information Networking (ICOIN)*. IEEE, 2016, pp. 420–423.

[28] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*. Springer, 1972, pp. 85–103.

 ISBN: 978-1-61208-811-2