Available online at www.sciencedirect.com

ScienceDirect

Cognitive Systems Research 64 (2020) 146–163

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

Segment routing based energy aware routing for software defined data center

B. Balakiruthiga [a,*], P. Deepalakshmi [a], Sachi Nandan Mohanty [b], Deepak Gupta [c], P. Pavan Kumar [b], K. Shankar [d]

[a] *Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education (KARE), Virudhunagar, Tamil Nadu, India*
[b] *Department of Computer Science and Engineering, FST, ICFAI Foundation for Higher Education, Hyderabad, India*
[c] *Maharaja Agrasen Institute of Technology, Delhi, India*
[d] *Department of Computer Applications, Alagappa University, Karaikudi, India*

Received 15 February 2020; received in revised form 28 May 2020; accepted 25 August 2020
Available online 2 September 2020

Abstract

Despite the fact that most of the data centers are software-defined, the multifaceted network architecture and increase in network traffic make data centers to suffer from overhead. Multipath TCP supports multiple paths for a single routing session and ensures proper utilization of bandwidth over all available links. As rise in number of nodes in data center is frequent and drastic, scalability issue limits the performance of many existing techniques. Segment Routing is vibrant in reducing scalability disputes and routing overhead. Segment routing approach combined with MPTCP traffic result in efficient routing approach. The downfall of the link capacity due to drastic incoming traffic remains as a major concern in data center network which enforces preventing link energy depletion due to high network traffic. Our proposed work, segment routing based energy aware routing approach for software defined data center aims to achieve throughput maximization through preserving link residual capacity and proper utilization of links. As well, our approach shows a decrease in length of segment label stack with respect to maximum segment label depth. Analysis is done by comparing the executions of other existing approaches in a single-controller environment with our energy-aware routing approach in a distributed environment. Distributed controller setup prevents network from single point of failure. It helps to prevent controller overhead and provides improved network performance through throughput.

Keywords: Software Defined Data Center (SDDC); Multipath TCP (MPTCP); Segment Routing (SR); Data Center Network (DCN); Software Defined Network (SDN); Segment Label Stack (SLS); Open Daylight (ODL)

1. Introduction

With the SDN-enabled data center network, integration of various functionalities is contented for functions like load balancing, routing, resource distribution, and guaranteed quality of service, (QOS) that become sequential (Xia, Zhao, Wen, & Xie, 2017). Formation of a communication path between two hosts from thousands of connection existing in DCN is a difficult task (Oktian, Lee, Lee, &

* Corresponding author at: Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education (KARE), Virudhunagar, Tamil Nadu, India.
E-mail addresses: b.balakiruthiga@klu.ac.in (B. Balakiruthiga), deepa.kumar@klu.ac.in (P. Deepalakshmi), deepakgupta@mait.ac.in (D. Gupta), drkshankar@ieee.org (K. Shankar).

https://doi.org/10.1016/j.cogsys.2020.08.009
1389-0417/© 2020 Elsevier B.V. All rights reserved.

Lam, 2017). Hence, the major reason for implementing SDN in DCN is to make routing more flexible and efficient. It is possible to avoid self-learning of network switches about the network topology since controller maintains network information. Several existing mechanisms suffer due to delay in packet transmission. Many reasons cause delay including time taken by the routing mechanism for path computation, link establishment, congestion, and inefficient utilization of links. All these issues will ultimately lead to a reduction in throughput. Establishment of proper communication channel with minimum number of hops and available capacity is mandatory for an efficient communication process. Moreover, a routing approach should be able to adapt scaling of hosts in the data center. Applying SR -mechanism in SDDC provides flexible routing system. Dugeon, Guedrez, Lahoud, & Texier (2017) have demonstrated the working mechanism of SR by combining the capabilities of an SDN controller and a path generation engine (PCE) to reduce the size of SLS to generate SR paths. The major objective of their work is to reduce the stack size compared to the prior applications. In our proposed approach also, we selected SR as the routing methodology since it can achieve less cost and is easily programmable compared to ordinary MPLS routing schemes.

Our traffic routing model implements SR in MPTCP traffic. Due to the coupling of all data plane devices to a controller where the routing protocol is available, the controller manages forwarding devices based on the routing protocol. MPTCP is a progression of TCP, which can effectively use multiple paths within a single transport connection (Sandri, Silva, Rocha, & Verdi, 2015; Li, Hu, Liu, Fu, Chen, & Zhang, 2015). With MPTCP, the incoming traffic can be partitioned in to multiple paths i.e., sub paths. That is, multiple paths between the hosts can propagate a single flow simultaneously. In case of link failure, alternative path for traffic routing is applicable. Thus, MPTCP increases throughput and reliability in packet transmission. MPTCP is the most used transport approach in DCNs (Detal et al., 2013) principally when we want to improve network performance (Paasch, Khalili, & Bonaventure, 2013). Modern data centers suffer from huge traffic which leads to congestion (Govindarajan, Meng, & Ong, 2013). There are scenarios where there is lack of effective utilization of network resources such as computation speed and storage space across multiple servers. Considering several DCN topologies, MPTCP provides better load balancing and traffic management strategies in case of congestion (Jouet, Perkins, & Pezaros, 2016; Zannettou, Sirivianos, & Papadopoulos, 2016; Lu & Zhu, 2015). DCN may no longer provide increased network performance and long lifetime by adopting an effective routing mechanism alone. The notable challenging factor affecting DCN's performance is energy whose utilization is based on several factors like load, delay and capacity of network resources such as data plane devices, controllers and links.

Many research works are proposed and some under progress regarding the energy issue. Frequent transmissions with peak traffic volumes through a network link leads to link failure. If user utilizes the entire capacity of a link and makes the queue to be full all the time, it is inefficient to guarantee reliable transmission.

Furthermore, when load increases drastically, it will lead to link down. Thus, it is obligatory to avoid complete utilization of a link's capacity. To gain energy, it is important to reserve some portion of link bandwidth. Hence, data centers need an on-demand requirement of energy efficient routing approach which can maximize number of successful transmission with proper link utilization. As a result, we contribute in this paper an energy efficient routing approach for SDDC using SR through MPTCP traffic.

1.1. Network model

A software defined data center network is modeled as a directed graph G = (V, L, Con) where 'V' is the set of nodes and 'L' is the set of links/edges and 'Con' is set of controllers, where Con $\in$ V. Since the network permits bidirectional traffic, V (V-1)/2 demand pairs are possible. Each link connecting the vertices possesses capacity 'C_l' to route the demand volume 'h'. Group of interconnecting devices is defined as, $I = \{v \mid v \in V \bigwedge v \notin Con\}$. Table 1 shows the definitions for various parameters employed in our approach.

1.2. Problem statement

The objective function for maximizing throughput is given in Eq.(1) with the constraint that total demand volume for all candidate paths running between every demand pair utilizing the link must be less than the link capacity as in Eq. (2). The total throughput 'TP' for all paths running between every demand pair (u, v) in the network is summation ofTP_p.

$$Maximize \sum_{p \in P_z} TP_p \tag{1}$$

$$\text{Subjectto} \sum_{z=I}^{z} \sum_{P=1}^{P_z} \delta_{zpl} x_{zp} \le C_l, l = 1, 2 \cdots l \tag{2}$$

Definition 1: Entire set of candidate paths (shortest paths) between all demand pairs is $P_z = \{P_1, P_2, P_3 .... P_z\}$.

Definition 2: 'x_{zp}' denotes the link – path formulation index, where 'z' represents the demand index and 'p' represents the path index.

Definition 3: Link-path identifier, δ_{zpl} – Set to 1 if path 'p' for demand pair use link 'l'. Otherwise, it will be set to 0.

Utilization of links 'l_u' as given in Eq. (3) signifies the ratio of total traffic 'y_l' over the path to the path capacity. The maximum utilization of links is directly proportional to the throughput

$$l_u = y_l / C_l, l = 1, 2, \cdots l \tag{3}$$

$$\sum_{z=I}^{z} \sum_{p=1}^{P_z} \delta_{zpl} x_{zp} = y_l, l = 1, 2 \cdots l$$

Table 1
Parameters used.

Parameter	Definitions
Demand volume 'h' (Traffic volume)	Measure of traffic transmitted over a link
Demand pair (Traffic pair)	Any pair of nodes (u, v) with, 'u' as source node and 'v' as destination between the traffic flows
Path	A link or set of links which connect the demand pair (u, v)
Path flow	Measure of demand volume transmitted on a particular path
Link flow	Amount of flow on a specific link irrespective of sink location that carries the demand volume
Demand index 'z'	Numbering the demand pair, which has positive demand volume to be carried on it
Path index 'p'	labeling of paths
Link index	Denotes the particular link in the path

2. Background

2.1. Segment routing (SR) in software defined data center (SDDC)

SR, a source-driven routing mechanism partitions the network into segments. Each node and link targets a Segment Identifier (SID), a flat unsigned 32-bit integer advertised by each node using standard routing protocols (ISIS/OSPF or BGP). This eliminates the need to run additional label distribution protocols (LDP). The source node selects a path and appends SID in every packet header as an ordered list of segments. Each segment is recognized by the Segment ID (SID) consisting of a Segment instruction

With SR (github.com; mininet.org), the network no longer requires to maintain a per-application or a per-flow state. Instead, it follows the forwarding instructions provided in packet itself. SR depends on some protocol extensions such as IS-IS and OSPF. SR can operate either on MPLS data plane or on IPv6 data plane which can integrate with all capabilities of MPLS, including Layer 3 VPN (L3VPN), Virtual Private Wire Service (VPWS), Virtual Private LAN Service (VPLS), and Ethernet VPN (EVPN).

The Multiprotocol Label Switching (MPLS) architecture directly applies SR strategy without any change in the forwarding plane. SR utilizes network bandwidth more effectively than traditional MPLS networks and also offers lower latency. An SR segment is programmed as an MPLS label. Further, the ordered list of SR segments is kept as a stack of labels. The segment to be processed next is on top of the stack. The related label pops out from stack after processing a segment. Therefore, it is better to facilitate these advantages of SR-MPLS strategy in proposed routing model for SDN implementation instead of SR-IPV6 implementation. The SDN control plane structures supported by SR architecture are distributed control plane, centralized control plane or hybrid control plane. In a distributed control plane architecture, IS-IS or OSPF or BGP allocates and signals the segments. In the proposed work, through distributed SDN control plane architecture, we prevent single point of failure and increase scale adaptability. Some of the important notations used in our paper for SR are as follows,

Segment: A segment is an instruction that a node process on the ingress packet. According to the shortest path identified, this instruction helps intermediate nodes to direct packets to a specific host. A Segment in SR is identified with Segment Identifier (SID) and in MPLS environment (suitable for SDN enabled networks) SID is encoded in 32 bits MPLS label.

Segment Routing Global Block (SRGB): In SR, a node broadcasts an SR Global Block (SRGB) that contains a range of labels allocated for SR. Example range of labels are 1000, 2000 (Davoli, Veltri, Ventre, Siracusano, & Salsano, 2015).

A Segment is classifiable as global or local segment to the node that advertises it.

Local Segments: An individual node advertises local segments. A local Segment Identifier (SID) takes its value outside the SRGB such as 3000 or 3020 and so on. Though SR domain advertises this local segment, only the node advertising it holds all related forwarding instructions.

*Global Segments: A*ll SR nodes in the domain advertise Global segments. A global Segment Identifier (SID) takes its value within SRGB such as 1100 or 1200 and so on. All the SR nodes deploy forwarding instructions for each of the global SIDs.

Adjacency SID (Adj-SID): Adj-SID is a label attached to an IGP adjacency, an interface to reach the neighbor router. The Adjacency SID implements packet forwarding through an exact exit interface. An Adj-SID gets advertised as a local segment by default. If needed, it can also reach global dimensions though advertisement. A router can maintain Adj-SIDs only for its neighbors. Typically, a router allocates Adj-SID value dynamically.

Node-SID: This is a prefix-SID, which is a segment denoting a specific network prefix and it is always global within an IGP domain. Moreover, it is a label, associated with the specific SR node, attached to the loopback address. When a packet with a Node-SID arrives at SR ingress node as a top label, the node forwards the packet to node which owns the Node-SID. The receiving node inspects the next label in the stack N. Label stack in our traffic model do consider both Node-SID and Adj-SID. There can be any number of intermediate nodes between Node SIDs. Therefore, by using Node-SID we can reduce the label length. Adj-SID is applicable when connecting adjacent node (via interface) and changes dynamically. Therefore, Adj-SID is applicable whenever there is a possibility to link two adjacent nodes.

2.1.1. Segment routing (SR) operations

The two normal label stack lists used are MPLS label stack and list of IPV6 addresses. It is purely applicable in the data plane used during implementation setup. In our proposed work, we have applied the MPLS label stack which is more suitable for the SDN Controller setup (Davoli et al., 2015). Following is the list of operations during packet transmission via SR path from source to endpoint.

- PUSH – The SR node performs PUSH operation to inject a segment label into the SLS when a packet arrives at ingress router. The segment then appears on top of the label list.
- NEXT – Router performs POP segment label when the packet reaches next intermediate node based on the segment label.
- Continue –The router performs this when packet reaches the next label switch. The CONTINUE operation proceeds when current active segment instruction is in process but not yet finished.

Pang, Xu, and Fu (2017) proposed a collaborative traffic management mechanism with SR and MPTCP in SDN based DCN. Considering the three layers (Control layer, Data layer and Host layer) of SDN, implementation happened in the NS-3 simulator. To reduce the storage consumption in forwarding switches which uses ternary content addressable memory (TCAM), this collaborative approach is recommendable. Over the DCN physical topology, MPTCP traffic flow is significant before SR paths appear for efficient traffic management. This implementation achieves reduction in flow table size, network overhead caused by SR Segment labels in packet header in case of single controller environment. Our approach also implements SR with MPTCP traffic. We have deployed this mechanism in distributed controller environment whereas authors of (Pang et al., 2017) used centralized controller architecture and hedera for traffic scheduling. We also adopted label generation algorithm to reduce segment label size. Additionally, our approach considers link capacity as an important routing metric to reroute traffic in congestion state.

Lee and Sheu (2016) have proposed a routing algorithm for SDN with SR strategy. It is a traffic management implementation to reduce the extra cost experienced by packet header size in a network. The authors have developed a bandwidth-satisfying path between two hosts. Their main aim is to increase the throughput and decrease the network congestion. They have analyzed the performance of routing algorithm using in terms of throughput and average link utilization. This approach is also implemented in single controller environment. They have not considered multi path delivery between the hosts. By using source routing mechanism, they have implemented the bandwidth satisfying path. In our approach, we use both SR and MPTCP which provides increased throughput and finds alternate path with suitable link capacity factor.

(Paasch et al., 2013) have applied two variants of SR label generation algorithm using Node Segment Identifier (Node-SID) and Adjacent Segment Identifier (Adj-SID) for SR path generation and routing. The major objective of this work is to overcome some of the existing issues of SR over SDN such as increase in controller overhead and segment label length compared to Maximum SID depth (MSD). Using PCE as a centralized entity, they have considered their implementation under various real time topologies. Results appear based on the label stack size with respect to the various topologies used. They used two variants of algorithm, Adj-SID as local segments and Adj-SID as global segments. Our proposed routing model developed a label generation algorithm using both Node-SID as well as Adj-SID. SR advertises Adj-SID as global for the entire communication process. If controller commands a node to install forwarding instruction, even if the node is not the one which advertised the SID, it can fix. We adopt the same idea for distributed controller whereas they (Paasch et al., 2013) proposed for centralized entity.

Davoli et al. (2015) have proposed a traffic management model of SR for SDN to achieve better scalability and efficient flow allocation during high traffic conditions. Using Mininet emulator, they have implemented this model and analyzed the performance based on flows with respect to number of packets transmitted and considered flow completion time for both traffic organization and SR module.

The existing research works do not have routing algorithm for the SDN controller to configure the routes in edge nodes for SR implementation. Traffic Engineering in SR includes distribution of network load to complete network and reduces network congestion. Therefore, it is important to avoid overloading of networks. At the same time, it is relevant to prevent under-utilization of network links. Existing research works lack with poor performance due to above mentioned constraints based on the parameters like end-to-end delay, guaranteed bandwidth, and throughput. It also experiences more network cost due to more number of hop counts. Thus increase in utilization of network resources leads to decrease in network throughput. Hence, it has become necessary to propose an effective routing method to control the SLS and to support scalability demands using multi controller environment compared to single controller setting.

2.2. MultipathTCP (MPTCP) in software defined data center (SDDC)

Zannettou et al. (2016) proposed an MPTCP-aware SDN controller which performs routing operations to regulate MPTCP sub flows. The controller computes various groups of paths between the hosts to allow path diversity in existing DC topology. Controller takes care of different sub flow allocation to paths. In this paper, the authors demonstrated that multiple sub flows per pair of IP addresses could improve network performance. Users can analyze performance based on average throughput gained over all MPTCP connections in the network and distribution of MPTCP with respect to link capacity. This work reduced the number of installed rules at open flow switches compared to previous works.

(Sheu, Liu, Jagadeesha, & Chang, 2016) proposed an efficient algorithm for k max-min bandwidth disjoint paths for MPTCP. It computes the set of candidate paths from source to destination in polynomial time. It outperforms the earlier approached in terms of throughput attained.

Hyunwoo, Calin, and Schulzrinne (2016) proposed the SDN implementation in MPTCP, which dynamically adds or removes MPTCP paths. It helps to improve the poor performance caused by large number of out-of order packets experienced when the paths have different bandwidths and delays. Under various network conditions, the proposed system monitors the available capacity of all connected paths and chooses the most appropriate path. Mininet is applicable for implementation. The results have indicated that it would not guarantee maximum throughput utilization and it is cost ineffective. No congestion control strategy is considered during application (Hyunwoo et al., 2016). The major limitation of this work is introducing substantial overhead on the controller and the hosts due to adjusting the number of sub flows. In addition, it requires more number of supplementary connections between the controller and the hosts.

Duan, Wang, and Wu (2015) proposed an MPTCP system to resolve the limitations such as, inefficient routing, and fixed number of sub-flows. This work employs an SDN controller to compute sub-flow route calculation. Moreover, each server is deployable with a monitor for adjusting the number of sub-flows. The simulation for the proposed implementation is executable with the use of the DC architecture. Our energy efficient routing mechanism places link capacity as a factor for routing through SR label stack. This reduces the overhead induced by the number of sub flows.

Raiciu et al. (2011) validated that the implementation of MPTCP in Data Centers is advantageous regarding the performance and robustness. They have used flow-based ECMP approach for routing MPTCP sub flows. Our routing model have implemented energy based routing approach.

Detal et al. (2013) proposed a routing technique which allows the end hosts to choose packet header values for selecting a specific path. To enable this approach the authors have implemented a new path selection technique on switches. In our approach, path generation is done by SR whereas path selection is based on energy factor among the paths created.

(Deebak and Al-Turjman, 2020) proposed a hybrid routing scheme with two multipath routing protocols. Through these protocols, dynamic selection of sensor monitoring nodes are possible. With the help of modified two-fish algorithm, the proposed mechanism identifies the optimal routing paths based on nodes mobility, frequent link interruptions, repeated update rate and energy level of sensor nodes. The authors claimed that this mechanism achieves secure data transmission in adhoc sensor networks.

Most of the MPTCP implementations (Lei, 2015) available are cost ineffective, since deployment cost is high. Increase in number of forwarding rules leads to huge storage consumption as switches consume more TCAM (Ternary Content Addressable memory) space. Number of sub flows in flow allocation mechanism of MPTCP is static irrespective of actual traffic conditions. Appending label list to every packet header increases the entire network overhead. ECMP-based hashing is vital for MPTCP routing (Jarraya, Madi, & Debbabi, 2014) which leads to failure of server and network resources. The available flow allocation mechanisms of MPTCP consider both long lived and short lived packets with respect to the flow completion time. Existing mechanisms experience lack of better performance in case of congestion control and buffer space (Wischik, Raiciu, Greenhalgh, & Handley, 2011). So, implementation of proactive SDN environment instead of reactive SDN environment has to be encouraged. In addition, they are not maintaining the knowledge of entire network during the implementation of load balancing techniques and hence lack in optimal routing. Here multipath forwarding wants the sub-flows needed for identification and localization function through a separate path. This separate path generation is non-executable with the application of regular load balancing mechanism. No proper solution for MPTCP mechanism has been proposed regarding efficient path allocation and number of sub flows. In MPTCP, path choices are static, not adaptive. Therefore, we cannot expect any significant optimal performance. The existing routing techniques with MPTCP mechanisms have a drawback of dropping of packets since one can select all paths at the establishment phase of the network. But, this does not get updated upon network changes. Adaptive control mechanisms for deciding the number of sub-flows were not recommendable with the existing techniques. Increased overhead is experienced on the SDN controller due to frequent modifications of number of sub flows. Therefore, one cannot guarantee maximum throughput utilization with existing systems of MPTCP. Through our proposed energy efficient routing model, we could control congestion with the help of rerouting of packets over alternate path between source and destination hosts.

2.3. Energy in data center

Optimization of energy in data centers has become mandate for traffic management. Because a traffic approach is efficient when the network resources are properly utilized, in particular network bandwidth. Appropriate network bandwidth utilization guarantees maximum throughput and reliability in transmission.

Dynamic topology change, buffer overflow, and network scale depletes the bandwidth in both data plane devices and links.

(Al-Turjman, Deebak, & Mostarda, 2019) proposed a multihop routing methodology to increase energy efficiency in device to device communication. This methodology uses combinatorial optimization problem which optimizes energy efficiency of a cellular network. It examines the metrics such as throughput, packet delivery ratio, resource utilization and energy efficiency.

(Al-Turjman & Kilic, 2018) analyzed the features of various existing routing mechanisms for wireless nano sensor networks and developed an energy aware routing mechanism. This routing technique is based on backward- learning paradigm. Results shows that proposed routing protocol is energy as well as transmission efficient by achieving less computational cost.

Yuan, Jay Kuo, and Ahmad (2010) explored the various aspects of energy efficiency in cloud based multimedia services in data centers and future directions for research. They also dealt with the challenges in implementation of green data center.

carpa, gluck, lefevre, & Avalon, 2015) have proposed a framework called SR based Energy Efficient Traffic Engineering that dynamically adapts the number of powered-on links to the traffic load. The proposed algorithm is applicable in OMNET++ framework to test the number of links turned off to reduce bandwidth consumption and thus improving network throughput. The results show that nearly 44% of the links got turned off in DCN architecture. Our proposed algorithm reroute the traffic and preserves the link bandwidth and also prevents link failure due to over utilization of links.

Xu, Dai, Huang, and Yang (2015) proposed an energy efficient routing algorithm. OmNET++ is the simulator used for performance analysis in fat tree and bcube topologies. BEERS algorithm is used to schedule the queues to handle traffic on link. This helps to minimize the energy in data center traffic.

In this paper, the proposed energy efficient routing mechanism with SR through MPTCP traffic preserves link bandwidth and prevents network from link failures. It also ensures maximum link utilization with increased throughput. Our proposed energy efficient routing model is established over jellyfish topology with ODL controller arrangement. Table 2 shows the analysis of some existing research works regarding the proposed approaches and solutions.

2.4. Jellyfish DCN topology

A jellyfish DCN (Singla, Hong, Popa, & Godfrey, 2012) topology has been chosen as an underlying physical topology for the formation of data plane and host cluster layer for our proposed work. Jellyfish topology is more suitable for massive data centers with lower cabling cost and attains higher capacity. It can accommodate 25% of more servers than fat tree topology and hence network scalability is possible in terms of switches. Its rack-rack model reduces the traffic complexity and enhances network load balancing feature. It provides fault tolerant capability among the nodes and offers good traffic control. This is an expandable topology based on random graph and is highly flexible for DCN.

Krishnan and Figueira (2015) analyzes hardware underlay and software overlay of SDN based DCN architecture and presented some innovative ideas for exploiting the value of the underlay in a Commercial-off-the-shelf (COTS) ASICs setting.

Chen et al. (2011) have presented a comparative study on various DCN architectures and routing mechanisms. Considering the performance metrics like reliability, scalability, and robustness, they analyzed the performance of various routing architectures like BCube, DCell, Portland (Niranjan Mysore et al., 2009), VL2, Helios, and c-Through implementations.

2.5. SDN controllers

There are two applications in the SDN controller architecture: single and distributed SDN controllers. In single SDN controller architecture, only one SDN controller is present in the network. It manages switches in a centralized manner. Main concerns of the single controller architecture are:

Scalability: Once the network scales, the number of requests to the SDN controller increases, which give rise to scalability problems.

Robustness: The complete network will break due to a single point of failure problem with the single controller architecture. Switches will fail to forward packets to the controller leading to loss of packets. This also decreases the network throughput.

In distributed controller (Blial, Ben Mamoun, & Benaini, 2016) environment, multiple controllers are applicable according to network density. A control plane setup containing multiple controllers has one component that acts as a root controller. A centralized controller will take the charge and distribute it among several controllers assigned for segmentation in network. This approach is referred as Master/ Slave approach where a root controller is considerable as a master and other controllers present in the control plane as slaves. A logically centralized control plane structure is applicable when providing more reliability and scalability. Physically distributed control plane components organize this element (Oktian et al., 2017). The interface among those elements is possible through the east and west bound interface.

Through this interface, the network structure can achieve better scalability. Thus, it is possible to avoid a Single Point of Failure (SPOF) and performance degradation. Member controllers distribute the load among them and the root controller manages all member controllers. It also adapts dynamic topology change in the network easily

Table 2
Survey on existing research works of SR, MPTCP and Energy based routing in SDN.

Reference	Year of publication	Mechanism Proposed	Parameters Considered	Experimental setup
Pang et al. (2017)	2017	Traffic management mechanism with SR and MPTCP in SDN based DCN	Storage consumption (TCAM) in forwarding switches, flow table size, network overhead	Centralized controller,NS-3.26,Fat tree DCN
Dugeon et al. (2017)	2017	Path segmentation technique to reduce maximum stack depth (MSD)	Segment label stack size for SR paths	SND lib network
Lee and Sheu (2016)	2016	Traffic management routing method for SDN with SR through building bandwidth satisfying path	Request rejection rate, Cost due to packet header size, throughput, average link utilization and network congestion	Centralized controller , Java, Waxman topologies
Guedrez et al. (2016)	2016	Label generation algorithm with Node-SID and Adj-SID	Controller overhead, segment label length compared with Maximum SID depth (MSD).	SND lib network
Duan et al. (2015)	2015	SR assignment algorithm	Flow allocation (Flow completion time) during huge traffic conditions.	Centralized controller , MininetColt telecom topology of zoo datasets
Zannettou et al. (2016)	2016	MPTCP-aware SDN controller using long-lived MPTCP flows in reactive SDN environment	MPTCP subflows, link capacity, Number of rules in openflow switch	Centralized controller, Fat tree
Sheu et al. (2016)	2019	Generating multiple candidate paths for k max–min bandwidth disjoint paths for MPTCP	Average throughput, average hop count	Centralized controller, Mininet, Ryu, Waxman topologies
Hyunwoo et al. (2016)	2016	Dynamic addition and removal of packets with MPTCP	Out-of order delivery of packets , available capacity of path	Mininet, POX controller, wifi networks
Duan et al. (2015)	2015	MPTCP routing scheme with a monitor deployed in each server for subflow check	Fixed number of subflows,	Centralized controller, NS3,Fat tree
Al-Turjman et al. (2019)	2019	Energy efficient multihop routing methodology through smart edge device	Energy efficiency, throughput, resource utilization, packet delivery ratio	NS3 simulator
Al-Turjman et al. (2019)	2019	Centralized routing and scheduling algorithm	Data size, link load and link capacity, Arrival time, Throughput, end-end delay	Java
Al-Turjman and Kilic (2018)	2018	Energy aware routing protocol for adhoc wireless nano sensor networks, last good neighbor (LaGOON)	Energy usage , Packet statistics	NS3 simulator
Deebak and Al-Turjman (2020)	2019	Hybrid multipath delivery routing scheme	Better monitor and detection ratio	NS-2.34 Simulator

without causing an overhead to the root controller. Therefore, we choose Distributed controller, ODL for our process application.

2.5.1. Open daylight (ODL) controller

ODL has a flat architecture, represented as a horizontal partitioning of network in to multiple parts where each segment is applicable by a single controller managing the connected SDN switches. The major advantages of this flat architecture are reduced latency and enhanced resiliency. Controllers present in same level have equal responsibilities and partial view of network at a particular time. This architecture is more resilient to failure. ODL Controller builds global network state by sharing local network state information to others. Users can share any data among the controllers. Controllers share no restriction about the network information. Some important information exchanged among the controllers are local network state information (Static or Dynamic), cross controller event information and inventory.

Oktian et al. (2017) have conducted a survey on various design choices on distributed SDN controller setup. Controllers analyze design based on the performance criteria's such as scalability, consistency, failure, robustness, and privacy. They did a deep study on various distributed controllers like floodlight, NOX, Beacon, RYU, ODL and so on. Experts completely examine the pros and cons of all the controllers and explained about selection of a proper design.

Blial et al. (2016) have explained the differences between various types of multi controller architectures regarding distribution methods and the communication system. They also provided performance analysis of already existing research works describing multi controller architectures design and communication procedure.

Considering the existing research, we set following as the major objectives of our energy efficient routing model.

- Maximize throughput with increased link utilization
- Prevent fast depletion of link capacity
- Instead of single SDN controller setup, physically distributed and logically centralized SDN controller environment is used which overcomes single point of failure (SPOF)

– Reduce the size of SLS in packet header by using both Node-SID and Adj-SID

3. Energy efficient routing model

The framework for our proposed approach is shown in Fig. 1. Data plane contains host cluster which comprises of switches and host. We use jellyfish DCN topology for underlying physical infrastructure. MPTCP traffic is considered among the data plane devices. ODL controller is used in control plane.

Communication between data plane and control plane is possible through southbound interface. It defines the communication protocol between data plane and control plane. Here OpenFlow is used as SI protocol. The data plane devices are assigned to controller. For every packet request at a switch, the controller assigns instructions to the switches for packet forwarding. Control plane maintains network state information and necessary mechanisms installed for proposed algorithm. Also, it issues commands to the data plane devices based on the mechanism. For our approach, SR routing methodology is installed to generate labels for packet transmission. Northbound API is the interface to connect control plane to the upper application plane. All control logics and policy respect to the mechanism used is maintained by application plane. In our approach, it is routing and traffic management. West/east bound interface is used for communication among the controllers in control plane. Opendaylight as a distributed controller environment, facilitates this interface function to the network.

3.1. Algorithm design

We portrayed proposed algorithm structure in Fig. 2 and explained the same in following subsections.

3.1.1. Candidate path generation

The proposed schema starts with computation of shortest paths for the considered demand pair (u, v).

OSPF protocol is applicable as a routing protocol for SR operations in our approach. Therefore, this approach calculates the link cost. We referred Dijkstra's algorithm (www.elsevier.com) to compute shortest paths with respect to the link cost.

As in the preliminary computation, a shortest path between 'u' and 'v' containing some set of vertices, v ∈ V and edges, l ∈ L is generated as its basic logic. As multiple candidate paths needed between demand pair (u, v), it is necessary to repeat the computation until the entire possible shortest path for the corresponding demand pair (u, v) are traversed. For achieving this, the edges and vertices present in the already computed candidate path turns to be inactive for the successive computations of algorithm. Therefore, this generates all the possible candidate paths visiting all edges and vertices in the network.

The below section represents algorithm 1 used to generate the multiple paths.

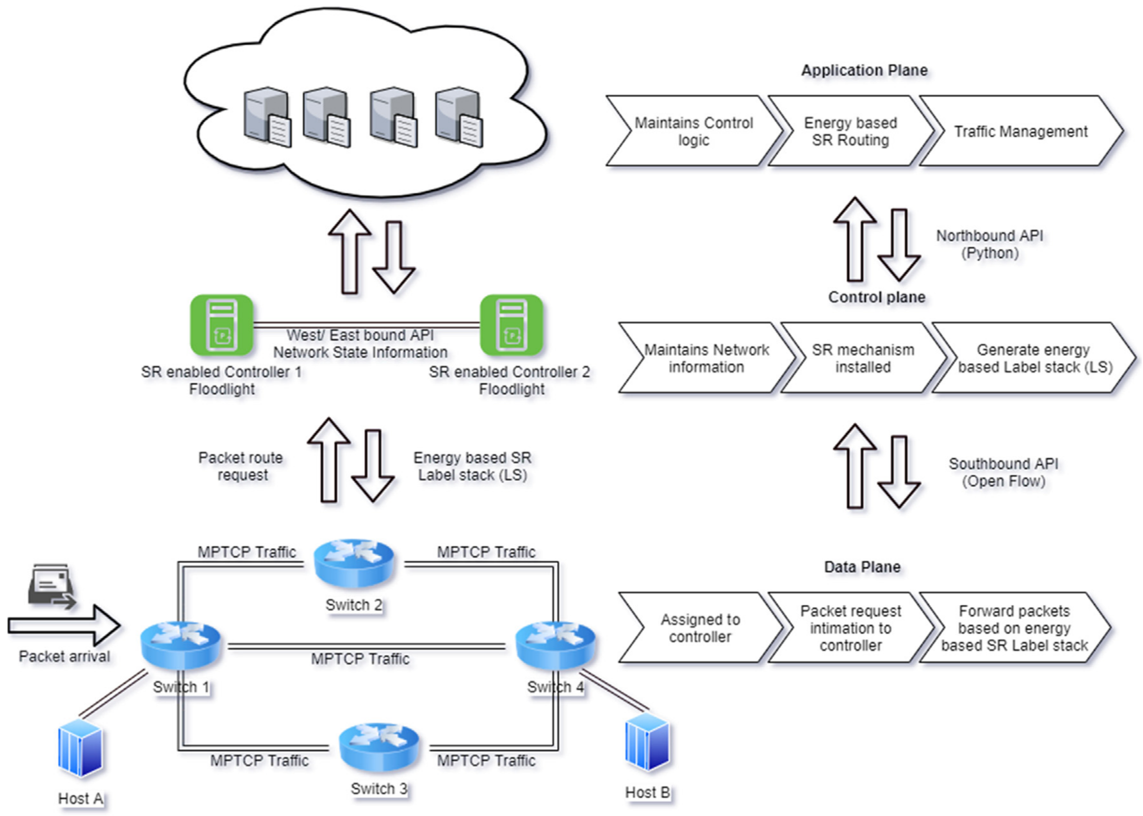

Fig. 1. Framework for proposed routing model.

Algorithm 1 (*Candidate path generation*).

```
Input: Graph (V, L)
Output: Possible shortest paths between demand pair (u,
  v), u, v $\in$ V
u- Source node
v- Destination node
k, m- Intermediate nodes
P- Permanent list holds vertices already considered for
  computation
T- Temporary list holds vertices not considered so far
  for computation
$D_{uv}$ (t)- Minimum distance from node 'u' to node 'v'
$d_{uv}$ (t) – Link cost between node 'u' and 'v'
$d^{u}_{km}$ (t)- cost of link k to m and it is known to node 'u' at
  time 't'
$f_{uv}$ – Next hop for 'u' with minimum distance
Initialize: P = {u}, T = N\ {u}
$D_{uv}$ (t) [] = $d^{u}_{uv}$ (t); $f_{uv}$ [] = v, for all v $\in$ T
While (T $\neq$ NULL) do
  $D_{temp} = \infty$;
  For (m in T) do
    If ($D_{um}$ (t) < $D_{temp}$) then
      $D_{temp}$ = $D_{um}$ (t);
      k = m;
    End if
  End for
$P = P \cup \{k\}$
$T = T \backslash \{k\}$
  For (v $\in N_k \cap$ T) do
    If ($D_{uv}$ (t) > $D_{uk}$ (t) + $d^{u}_{kv}$ (t)) then
      $D_{uv}$ (t) [ ] = $D_{uk}$ (t) + $d^{u}_{kv}$ (t);
      $f_{uv}$ [ ] = $f_{uk}$;
  End if
  End for
Return$D_{uv}$ (t) [ ];
Return$f_{uv}$ [ ];
End while
T = N\{$f_{uv}$ [ ] }
P = {$u \in N \cap f_{uv}$ [ ] }
While (T $\neq$ NULL) do
  Goto step 2
End while
```

Fig. 3 depicts the hypothetical scenario of the network topology with 20 nodes. Let the source node be 'A' and destination node be 'T'. Assume that source host attaches to node 'A' and destination host to node 'T'. In addition, an ODL Controller is responsible for control and data plane activities. It is a sample scenario to understand the mechanisms proposed in this paper. Fig. 4 shows the first and second computation of algorithm. The former gives the candidate path from source 'A' to destination 'T' with the link cost of 9 and with the Path I:$P_1 = \{A, B, D, G, I, S, T\}$. While the latter yields the candidate path from source 'A' to destination 'T' with link cost of 13. Computation happens with the remaining set of vertices present in the topology excluding the intermediate nodes present in path P_1 giving the path as Path II: $P_2 = \{A, L, K, N, Q, T\}$. Dashed circle represents the nodes contained in candidate path. The process terminates computation at this stage as possible intermediate nodes connecting the destination 'T' become inactive because it is included in the generated candidate paths.

3.1.2. Path selection

A set of candidate paths P_Z *[ ]* = {$P_{1,}$ P_2. . .P_Z} between 'u' and 'v' has been computed. Moreover, this is considerable as a MPTCP multiple paths for a single transmission session for demand pair (u, v). Now among the generated paths, we need to choose a preliminary path to route the packet. This selection is executable based on two parameters; capacity of the path 'C_P' and demand volume of the packet arrived 'h'. The basic criteria is, a link can able to accommodate a traffic volume if it is less than or equal to the capacity of the link. Otherwise, traffic overflow occurs and leads to loss of packet. Sum of capacity over all edges connecting the source 'u' and destination 'v' for every path in P_Z*[ ]* is calculated. In addition, users chose a path like the one below.

Definition 1: Total capacity of a path:

We can calculate the total capacity of a path using expression Eqn 4,

$$\textit{Total capacity of a path} - |C_p| = \sum_{\forall l \in P} \mathrm{Ac}(P) \quad (4)$$

The Sum of available by evaluating the capacity 'Ac' over all links 'l' of path 'P' connecting the source 'u' and destination 'v'.

Definition 2: Available Capacity of a link, A_C (l):

Unused capacity of the link is termed as available capacity Eqn 5. Deducting the present amount of traffic 'r' from the total link capacity 'C_l' at time 't' represents the available capacity.

$$A_C(l) = 1/T(r - C_l) \quad (5)$$

where Available Capacity of a link, A_C (l), Data rate, r and capacity of the link, C_l

The below section includes an algorithm to select the path.

Algorithm 2 (*Path Selection*).

```
Input: Candidate paths between demand pair (u, v), $P_z$=
  {$P_1, P_2, P_3 .... P_z$}
Output: Path, P
For all Pin $P_{z\ do}$
  If ($|C_p|$ < h) then
    P = P + 1;
  Else
    Generate LS;
    Route pkt;
  End if
End for
```

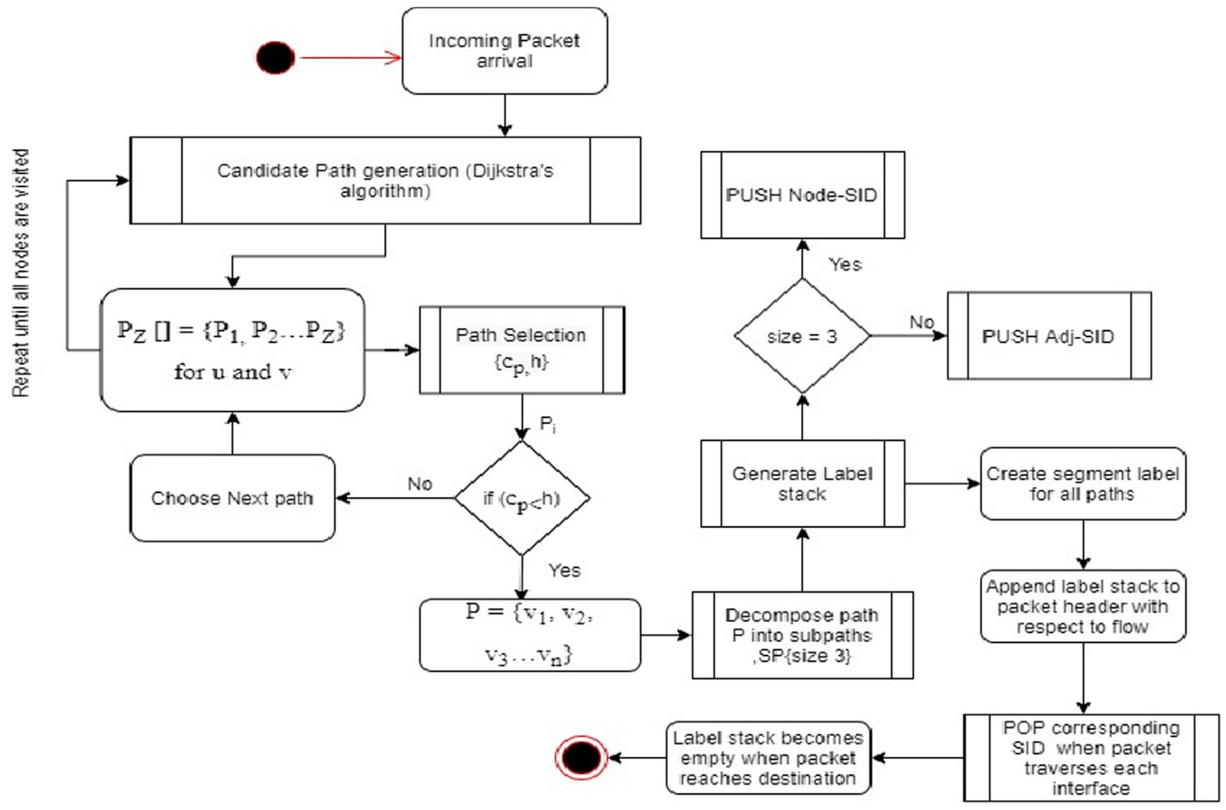
Incoming Packet arrival
Candidate Path generation (Dijkstra's algorithm)
Repeat until all nodes are visited
P_Z [] = {P_1, P_2…P_Z} for u and v
Path Selection {c_p,h}
P_i
if (c_p<h)
No
Choose Next path
Yes
P = {v_1, v_2, v_3…v_n}
Decompose path P into subpaths ,SP{size 3}
Generate Label stack
size = 3
Yes
PUSH Node-SID
No
PUSH Adj-SID
Create segment label for all paths
Append label stack to packet header with respect to flow
POP corresponding SID when packet traverses each interface
Label stack becomes empty when packet reaches destination

Fig. 2. Proposed Algorithm model.

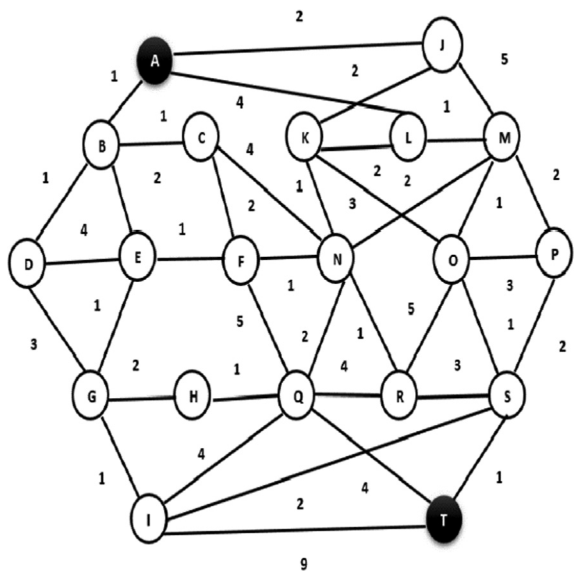
Fig. 3. 20 node network topology.

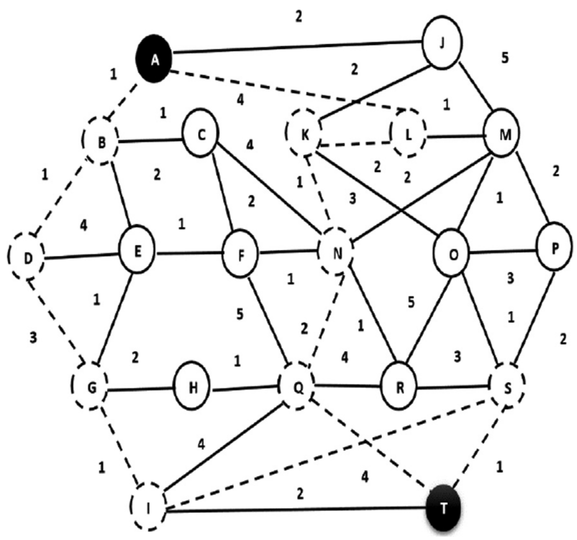
Fig. 4. Two generated candidate paths P_1 and P_2.

3.1.3. Decomposing the path in to sub paths

Next, the sub paths partition all paths. Each sub path treats the paths as a segment containing either nodes or edges. The notion of treating the sub path as segment is, because it is a segment routing enabled network. A network path is divisible in to any number of segment areas with respect to the number of intermediate nodes. Hence, according to the segments, it is possible to generate the SLS. Here the size of sub path is set as 3. Therefore, the path groups consecutive 3 nodes into a sub path.

The equation below shows an algorithm to decompose the path into sections.

Algorithm 3 (*Decomposing path to sub paths*).

```
Input: P = {$v_1$, $v_2$, $v_3$...$v_n$}   //Selected path with
  vertices
Output: $v_1$ [] = {$sp_1$, sp2...$sp_n$}/   /Sub path list
Initialize P [] = [], i = 0;
while (i < P.length)
  sp.push (P.slice (i, 3 + i));
  i+=2;
  Return sp;
End while
```

3.1.4. Building SLS

Next, it is vital to generate the segment routing label stack to route the packet. If the sub path is having 3 nodes, then push the end Node-SID into the label stack. The controllers should push the Adj-SID, if the sub path is having less than 3 nodes. When the packet and the label stack on top of it reaches each exit interface or the node along the path to its destination, the corresponding SID's pops out of the stack. Label stack becomes empty once the packet reaches the destination host. The section below represents an algorithm to build the label stack.

Algorithm 4 (*Building label stack*).

```
Input: P [] = {$sp_1$, sp2...$sp_n$}
Output: Label stack, LS
if (sp[i].length==3) do
  PUSH (LS, Node-SID (sp[i]end))
else
  PUSH (LS, Adj-SID (sp[i]end))
  sp[i] ++;
End if
```

Consider the path, $P_1 = \{A, B, D, G, I, S, T\}$from the topology represented in Fig. 4. The path 1 decomposes into 3 sub paths. Sub path 1 is having 3 nodes {A,B,D}, sub path 2 is having 3 nodes {D,G,I} and sub path 3 is having {I,S,T}. As we mentioned earlier, a path decomposes into any number of segments / sub paths. Here we kept the size of sub path as 3. If the destination node is in the next hop neighbor of previous sub path's end node, then Adj-SID of destination node pushes to the label stack otherwise the Node-SID is pushed. Replace the sub path 1 with end node D's Node-SID (it contains 3 nodes) 1037. Replace the sub path 2 with end node I's Node-SID (it contains 3 nodes) 1051. Replace the sub path 3 with end node T's Node-SID (it contains 3 nodes) 1010. Now the label stack, LS for path 1 is {1037, 1051, and 1010}. In case the controller selects path 2 to route the packet, then the LS for path 2 is applicable using the above process. The path 2 decomposes into 3 sub paths. Sub path 1 is having 3 nodes {A, L, K}, sub path 2 is having 3 nodes {K, N, Q} and sub path 3 is having 2 nodes {Q, T}. Replace the sub path 1 with end node K's Node-SID (it contains 3 nodes) 1016. Replace the sub path 2 with end node Q's Node-SID (it contains 3 nodes) 1021. Replace the sub path 3 with end node T's Adj-SID (it contains 2 nodes) 5011. Now the label stack, LS for path 2 is {1016, 1021, and 5011}. Figs. 5 and 6 represents the segment and label distribution of paths P_1 and P2.

4. Performance analysis

Simulation of the proposed routing model is applicable using Mininet, a virtual network framework (Opendaylight Controller) that acts an efficient SDN performance analysis framework. The other network emulators used with exist-

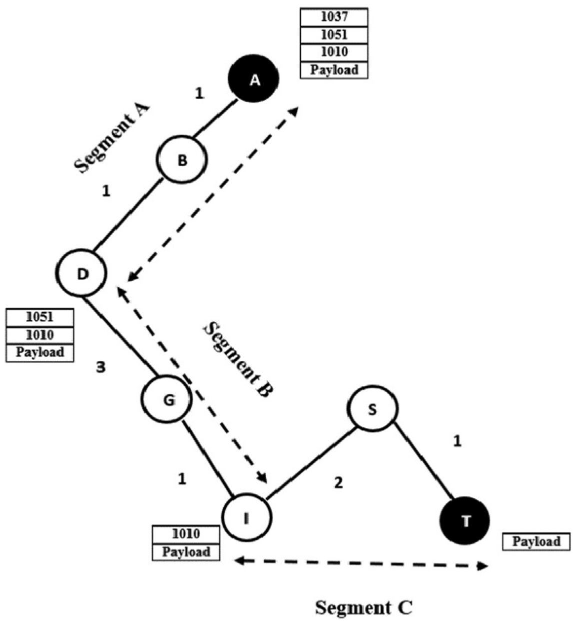

Fig. 5. Label Stack for Path, P_1.

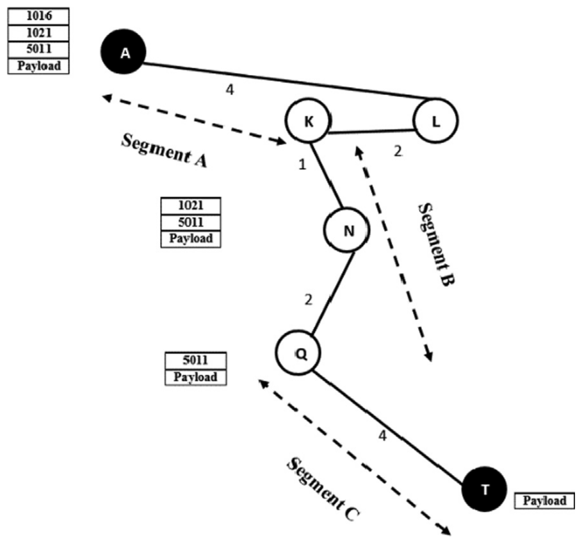

Fig. 6. Label Stack for Path, P_2.

ing research works for SDN implementation can be found in (Roy, Bari, Zhani, Ahmed, & Boutaba, 2014; ns-3 project, 2013; Kim et al., 2012). The simulated network traces for data transfer is acquired through wireshark (Wireshark). Openflow switch 1.3 has an inbuilt meter component which helps to retrieve the network statistics like data rate for incoming and outgoing flows at switch, link traffic and bandwidth utilization. Using sFlow (Sflow) the link utilization for each and every path can be measured. To analyze the proposed traffic model the following metrics are used,

- Network throughput – Measures the network efficiency and reliability (Drastic traffic condition)
- Link utilization – Measures the network efficiency (network scale)
- Packet header overhead – Measures SLS size
- Energy – Measures network lifetime (bandwidth consumption in link)

A simulation analysis is performed using the following approaches:

Multipath TCP (MPTCP): This approach allows routing in MPTCP Session-enabled network with candidate paths.

Segment Routing (SR): This approach allows routing in TCP-enabled networking through SR label stack.

MPTCP-SR: This approach allows combined implementation of SR and MPTCP traffic in single controller environment.

Energy aware MPTCP-SR: This approach allows proposed energy aware routing model using SR through MPTCP traffic in distributed controller setup.

The Table 3 shows the simulation setup for proposed energy aware routing mechanism in distributed environment.

4.1. Network throughput

The graph in Fig. 7 shows the accomplished throughput with respect to the number of nodes. Due to energy aware MPTCP-SR routing approach, there is an increase in throughput of about 5 to 10% in all node ranges. Throughput is measured in bits/sec. The graph in Fig. 8 shows the throughput attained in terms of different demand volume for a 150-node network. Demand volumes used are 250, 500, 750, 850 and 952 mbps respectively. When demand volume is greater than the available link capacity, Round Trip Time (RTT) for transmission will also be increased. Throughput is inversely proportional to RTT. Throughput experiences gradual degradation when RTT is increased.

Using our energy aware approach with distributed controller setup, degradation of throughput is prevented up to some extent. The graph in Fig. 9 shows the RTT experienced during packet transmission for the various demand volume distributions. RTT (Round trip time) for a particular transmission is identified using the ping test. When the volume increases, the RTT experienced by energy aware approach is low compared to the remaining techniques. While implementing with 450 mbps, our approach experiences 38 ms(milliseconds) of RTT, which is quiet low, where MPTCP approach experiences 62 ms, SR approach 60 ms and MPTCP-SR experiences 49 ms. Round Trip Time(RTT) is measured in Milliseconds.

Table 3
Simulation setup.

Parameters	Significance
Controller	2 - 3
Open daylight (ODL)	
DCN Topology	50 to 250 Switches
Jellyfish topology	
Switch	Openflow Switch 1.3
Traffic	Bidirectional MPTCP
Link Capacity	1000Mbps
Demand Volume	Random
Demand Pair	Random

4.2. Consequences of link utilization

This section has identified the probable candidate paths between the demand pair. The demand volume transmitted over the link for the particular path should be less than the total link capacity. If so, the current path is expected to have low congestion factor. Then there is no need of traffic rerouting from a path to other path.

If there is a rise in traffic experienced over the link then the congestion factor becomes high and so rerouting the traffic using the next path is needed. High link utilization tends to increase in throughput as well as increase in congestion factor. Distributed controller setup with MPTCP-SR approach overcomes this particular limitation up to some extent compared to the other approaches. Generally, more number of links is utilized if the throughput achieved is high. Because lower throughput is a cause of less number of packets transmitted to the destination, which means due to congestion situation some number of packets, got dropped and most of the routes are underutilized. Due to proposed distributed controller setup and enhanced MPTCP-SR routing strategy, there is an increase in throughput of about 5 to 10% in all node ranges as shown in Fig. 5 and also at most, more number of links are utilized in all node ranges in case of proposed implementation. The graph in Fig. 10 shows the Link utilization scenario with the network setup in Table 4.

Results clearly represents the energy ware approach having maximum number of paths with proper link utilization. The Congestion index for each link can be represented as in expression (6).

$$Congestion\ \ index,\ \ C_i = \begin{cases} 1, \sum_{z=1}^{z} C_l > L_{rc} \\ 0, \sum_{z=1}^{z} C_l \leq L_{rc} \end{cases} \tag{6}$$

where

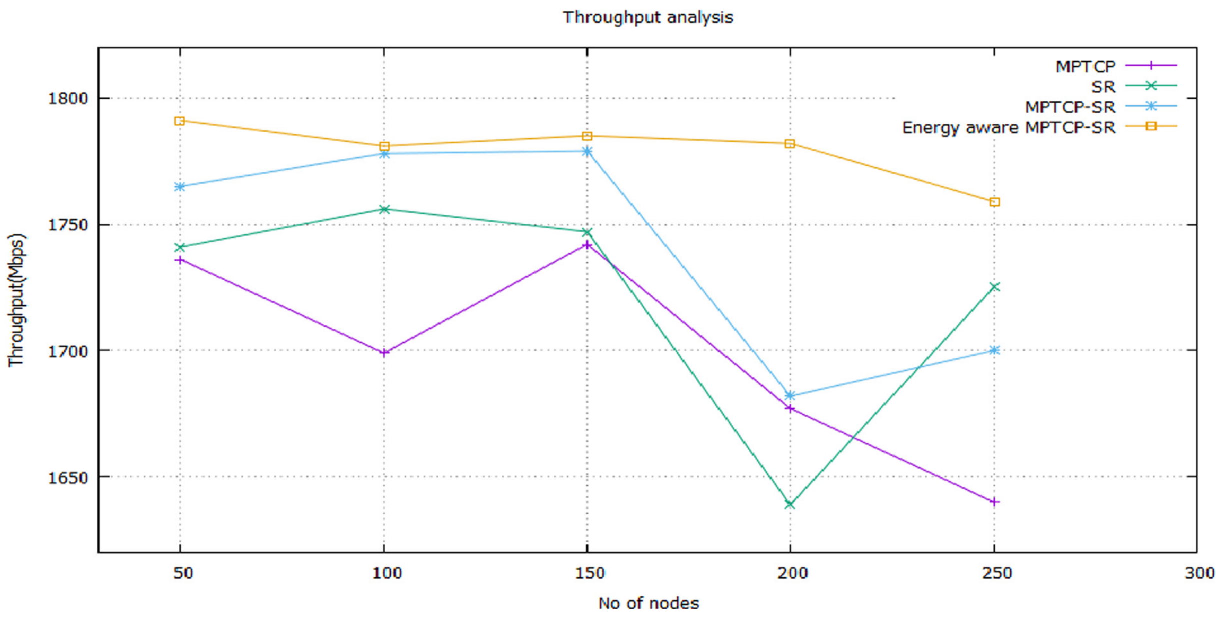

Fig. 7. Throughput attained in terms of no of nodes.

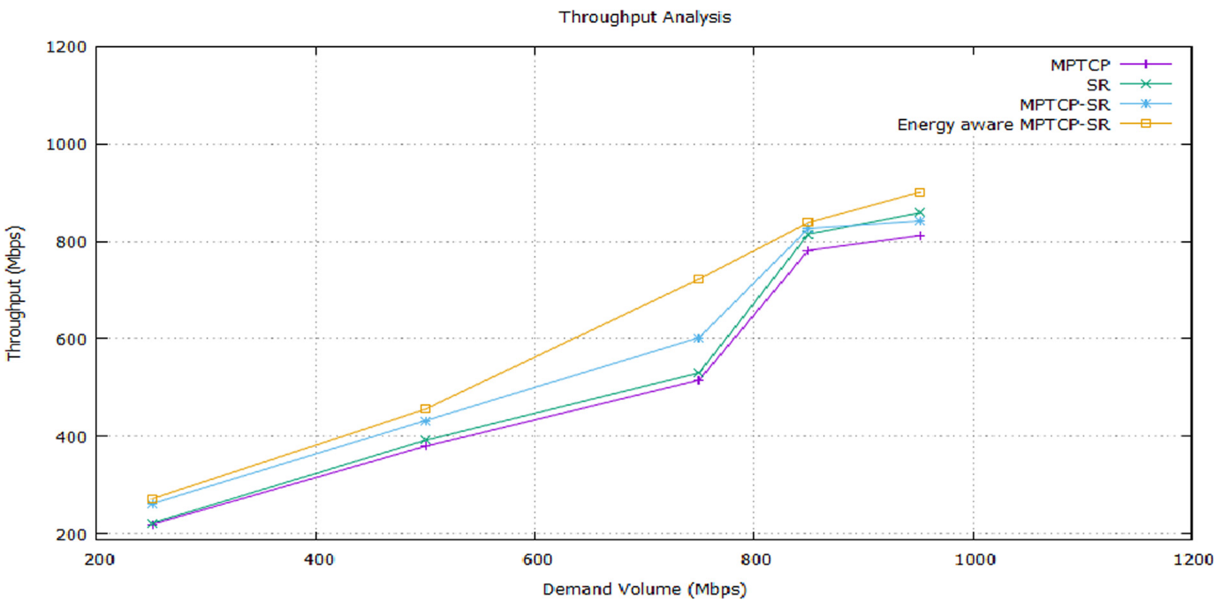

Fig. 8. Throughput attained in terms of demand volume (Mbps).

C_l = Capacity of the link, L_{rc} = Residual capacity of link

It is clearly understood that the congestion index based on the number of nodes using our proposed approach is very low from the results of throughput and link utilization. To identify the congestion index of the particular link, the above consideration is used. If the link capacity of a particular link is more than the residual capacity, then the congestion index is set as 1 and traffic is rerouted through other path. Otherwise, it is set as 0 and no traffic reroute decision is taken. The graph in Fig. 11 shows the average congestion experienced by the network in all the approaches based on the demand volume arrived at ingress node. Our proposed Energy aware MPTCP – SR approach shows less congestion when compared to other methods.

4.3. Size of SLS

Maximum SID Depth (MSD) is the maximum number of labels injected by a node to a packet header. Number of labels used in the packet header gives the Maximum SID depth for MPLS-SR traffic. We have considered the maximum depth of label list as 5 based on the hardware used as a node machine. Most of the existing approaches are not producing less number of labels compared to

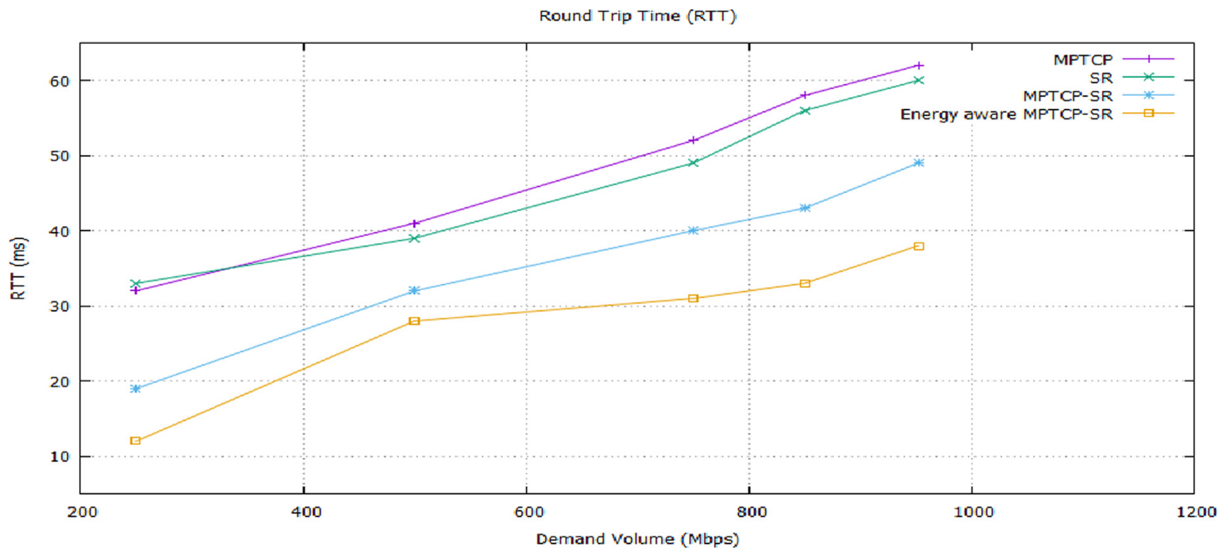

Fig. 9. RTT (mbps).

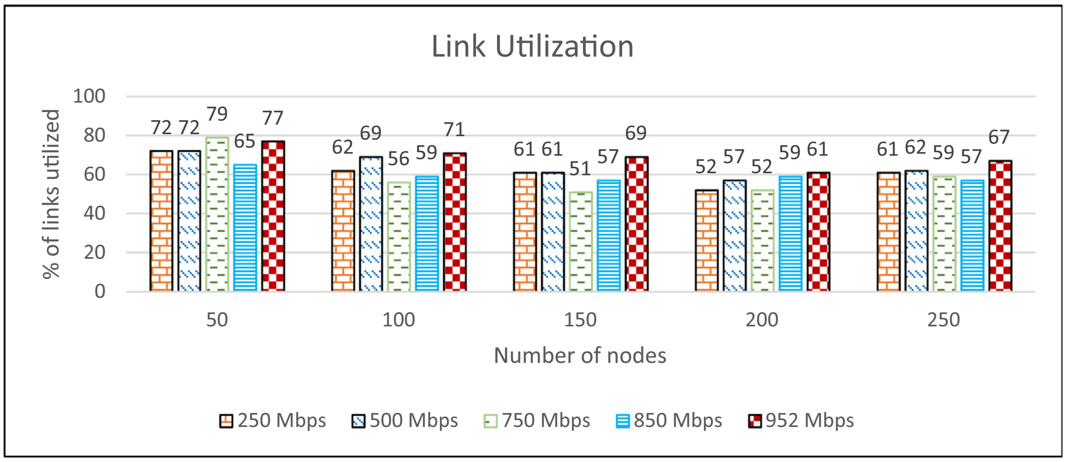

Fig. 10. Link utilization in terms of number of nodes.

Table 4
Link Utilization – Network setup.

Parameters	Significance
Node distribution	50, 100, 150, 200, 250
Demand volume, h	250, 500, 750, 850 and 952 mbps
Link capacity, C_l	1000 mbps

Maximum depth of label list. The maximum number of path possess label stack comes under MSD in energy aware routing approach. The remaining two implementations produce only less number of paths under MSD. Consider the sample topology in Fig. 3. For source ‘A’ to the destinations ‘Q’,’S’ and ‘I’, for the demand pair (A,Q), 2 paths are generated, for the demand pair (A,S), 2 paths are generated, then for (A, I) 2 paths are generated. Table 7 shows the various segment list for the paths generated between source ‘A’ and destinations ‘Q’,’S’ and ‘I’. The length of label list generated for all the above-mentioned demand pairs using energy aware approach is less than 5.

Whereas, while using MPTCP-SR approach which involves both Node-SID and Adj-SID advertisement, out of 7 paths, only 5 paths have less than Maximum SID depth. Among the 3 paths, the SR approach has only 1 path, which is less than Maximum SID depth. The graph in Fig. 12 represents the percentage of paths having label stack length less than MSD.

4.4. Consequences in link energy level

Energy of the link: Energy of the link (7) can be calculated as ratio of the residual capacity available to its actual capacity.

$$\Delta_l^{Energy} = \frac{L_{rc}}{C_l} \tag{7}$$

Overloading of links due to drastic demand volume at the time of transmission depletes link energy. If the same link is engaged for several frequent transmissions, it

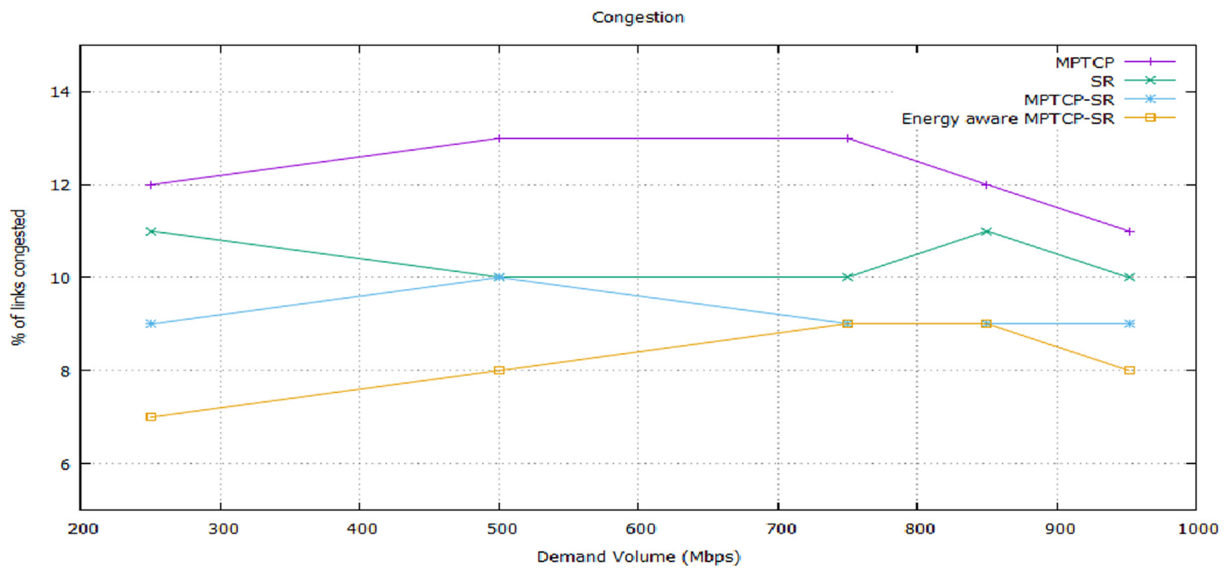

Fig. 11. %of links congested in terms of demand volume.

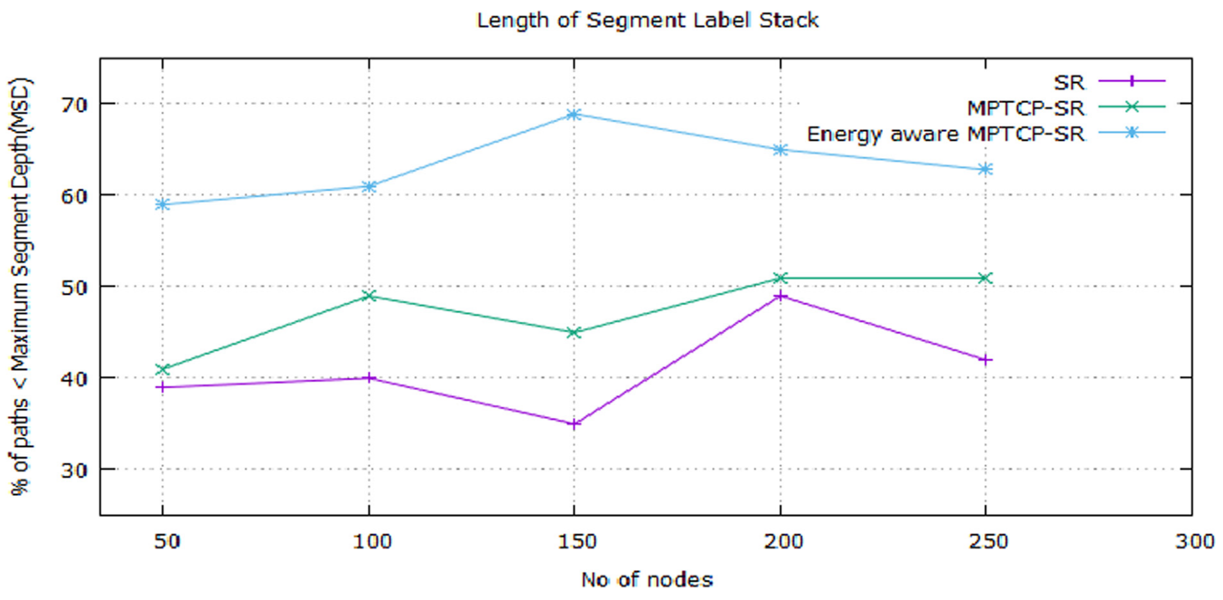

Fig. 12. Label Stack length.

becomes more prone to failure. Hence, the node connecting the link becomes inactive and the transmission is stopped at this stage. A sudden failure of the link during the run-time causes potential delays and packet loss. This obviously leads to loss of reliability and throughput at the receiving host. Energy of a particular link needs to be preserved to sudden link failure. In the research work (Al-Turjman, Mostarda, Ever, Darwish, & Khalil, 2019), the authors have proposed an efficient scheduling and routing approach with respect to link capacity, link load, delay and data size. This facilitates the better user experience by providing increased transmission speed in Internet of Things (IOT) data exchange. Using our approach, possible set of candidate paths are generated between a demand pair. Every path is formulated with disjoint links and nodes. So if a link is not ideal and already utilized with many packets in queue, next path can be chosen. Therefore, severe energy depletion is prevented to avoid link failure. To analyze the energy efficiency of the network with our proposed approach, two different node distributions

Table 5
Link energy level – Network setup I.

Parameters	Significance
Number of nodes	50
Number of transmissions	250, 500, 1000
Demand volume, h	750
Link capacity, C_l	1000 mbps
Demand pairs, Z	2
Candidate paths, P_z	3 for each z

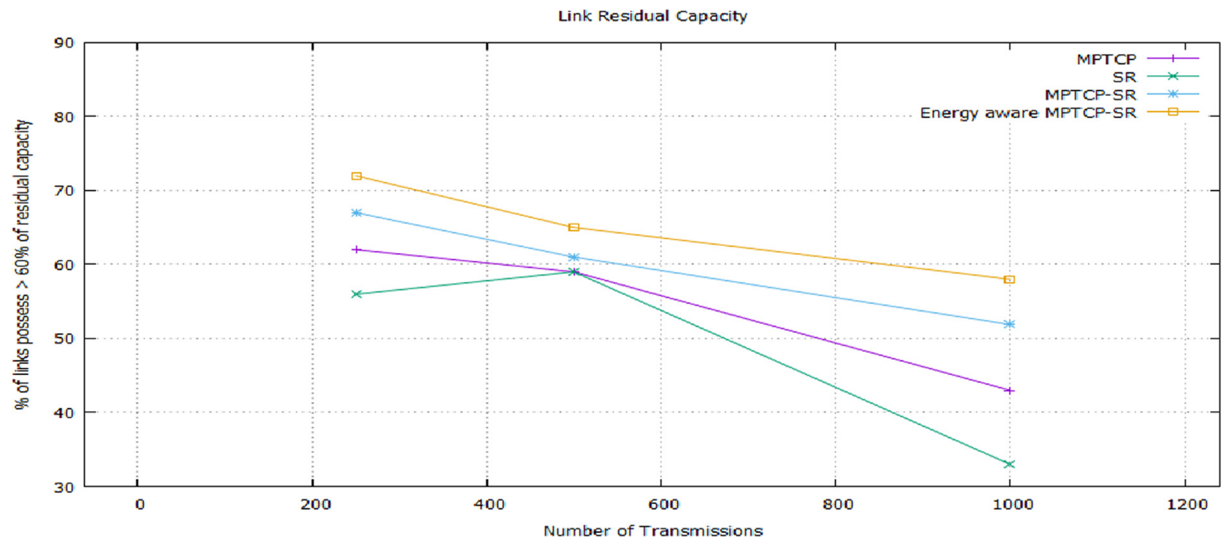

Fig. 13. Link Residual Capacity – 50-node setup.

Table 6
Link energy level – Network setup II.

Parameters	Significance
Number of nodes	100
Number of transmissions	250, 500, 1000
Demand volume, h	850
Link capacity, C_l	1000 mbps
Demand pairs, Z	3
Candidate paths, P_z	2, 3, 4

are used. The network setup shown in Table 5 is considered for the former. Two demand pairs are considered in the network. Three set of candidate paths are generated for each pair. Link residual capacity is tested in three different time intervals, after 250 transmissions, after 500 transmissions and at the end of 1000 transmissions. Graph in Fig. 13 shows the percentage of links having more than 60% of residual capacity. With our energy aware approach, more number of links is satisfying the mentioned criteria

Table 7
Candidate path and Label stack generation using Energy aware MPTCP – SR routing approach.

Source	Destination	Candidate paths	Sub paths	Node-SID and Adj – SID	Label stack Length
A	Q	{A,B,C,N,Q}	{(A,B,C),(C,N,Q)}	{1007,1021}	2
		{A,L,M,O,R,Q}	{(A,L,M),(M,O,R),(R,Q)}	{1009,1054,5111}	3
	S	{A,L,M,P,S}	{(A,L,M),(M,P,S)}	{1009,1019}	2
		{A,B,C,N,R,S}	{(A,B,C),(C,N,R,),(R,S)}	{1007,1022,5032}	3
	I	{A,B,D,G,I}	{(A,B,D),(D,G,I)}	{1010,1023}	2
		{A,L,K,N,Q,I}	{(A,L,K),(K,N,Q),(Q,I)}	{1011,1024,5052}	3
		{A,J,M,P,S,I}	{(A,J,M),(M,P,S),(S,I)}	{1061,1019,5018}	3

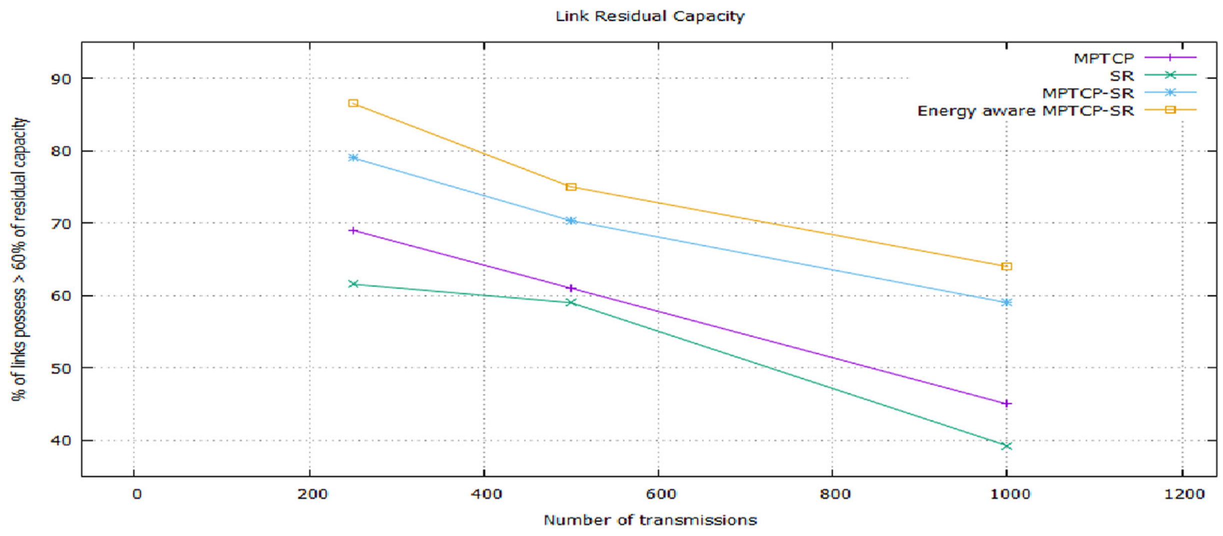

Fig. 14. Link Residual Capacity – 100 node setup.

compared to the other approaches. In the end of all transmission, average of 62% of links is satisfying this constraint. In addition, for the latter, the network setup is shown in Table 6. Three demand pairs are considered in this network. Number of candidate paths generated is varied for each pair. Graph in Fig. 14 shows the percentage of links having more than 60% of residual capacity with this setup. With our energy aware approach, more number of links is satisfying the mentioned criteria compared to the other approaches. In the end of all transmission, nearly an average of 59% of links is satisfying this constraint.

5. Conclusion and future work

In this paper, we have proposed an energy-efficient routing methodology using MPTCP-SR approach in a physically distributed and logically centralized ODL controller environment. Using the Mininet virtual network framework, this article has analyzed the performance of the proposed approach. For every demand pair, it is possible to generate a set of candidate paths and treat them as MPTCP paths. The Segment Routing strategy decomposes the paths into segments. Therefore, the SLS generated for the chosen path routes the packets according to the capacity of the path. The results show that the approach outperforms by achieving throughput with increased link utilization. In addition, capacity of links is preserved from complete exhaustion. Approximately 60% of link's residual capacity is preserved through our routing approach. Size of SLS is achieved under the MSD as five for most of the demand pairs using our routing scheme. Even though this approach achieves better results, mechanisms that are still more efficient is needed for less energy consumption in data centers. We have considered link capacity as a network performance energy metric. Energy consumption by switches and controllers for the entire network communication remains a major challenge for network lifetime. Achieving efficient bandwidth utilization of links through sleep and active state becomes more vital in data centers communication. Our future work aims at designing an energy efficient routing approach based on hop count and link cost affording link capacity in large-scale networks. Scheduling of transmissions during extreme traffic arrival with dynamic topology changes like switch migration and controller placement will also be another part of our work.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Al-Turjman, F., Deebak, B. D., & Mostarda, L. (2019). Energy aware resource allocation in multi-hop multimedia routing via the smart edge device. *IEEE Access, 7*, 151203–151214.

Al-Turjman, F., & Kilic, K. I. (2018). Lagoon: A simple energy-aware routing protocol for wireless nano-sensor networks. *IET Wireless Sensor Systems, 9*(3), 110–118.

Al-Turjman, F., Mostarda, L., Ever, E., Darwish, A., & Khalil, N. S. (2019). Network experience scheduling and routing approach for big data transmission in the Internet of Things. *IEEE Access, 7*, 14501–14512.

Blial, O., Ben Mamoun, M., & Benaini, R. (2016). An overview on SDN architectures with multiple controllers. *Journal of Computer Networks and Communications, 2016*, 1–8. https://doi.org/10.1155/2016/9396525.

carpa, R., gluck, O., lefevre, L., Avalon, I. (2015). Segment routing based traffic engineering for energy efficient backbone networks. In 2014 IEEE international conference on advanced networks and telecommuncations systems (ANTS), 12 Mar 2015. https://doi.org/10.1109/ANTS.2014.7057272. ISBN 978-1-4799-5868-9. ISSN 2153-1676.

Chen, K., Hu, C., Zhang, X., Zheng, K., Chen, Y., & Vasilakos, A. (2011). Survey on routing in data centers: Insights and future directions. *IEEE Network, 25*(4), 6–10. https://doi.org/10.1109/MNET.2011.5958002.

Davoli, L., Veltri, L., Ventre, P. L., Siracusano, G., & Salsano, S. (2015). Traffic engineering with segment routing: SDN-based architectural design and open source implementation. In 2015 fourth European workshop on software defined networks (pp. 111–112). IEEE.

Deebak, B. D., & Al-Turjman, F. (2020). A hybrid secure routing and monitoring mechanism in IoT-based wireless sensor networks. *Ad Hoc Networks, 97*.

Detal, G., Paasch, C., van der Linden, S., Mrindol, P., Avoine, G., & Bonaventure, O. (2013). Revisiting flow-based load balancing: stateless path selection in data center networks. *Computer Networks, 57*(5), 1204–1216.

Duan, J., Wang, Z., Wu, C. (2015). Responsive multipath TCP in SDN-based datacenters. In: 10 Sep 2015, IEEE international conference on communications (ICC), London, UK. https://doi.org/10.1109/ICC.2015.7249165. ISBN 978-1-4673-6432-4, ISSN 1550-3607.

Dugeon, O., Guedrez, R., Lahoud, S., & Texier, G. (2017). Demonstration of segment routing with SDN based label stack optimization. In 2017 20th conference on innovations in clouds, internet and networks (ICIN) (pp. 143–145). IEEE.

https://github.com/perimore/Segment-Routing-Demo.

Govindarajan, K., Meng, K. C., & Ong, H. (2013). A literature review on software-defined networking (SDN) research topics, challenges and solutions. In 2013 fifth international conference on advanced computing (ICoAC) (pp. 293–299). IEEE.

Guedrez, R., Dugeon, O., Lahoudy, S., Texier, G. (2016). Label encoding algorithm for MPLS segment routing. In 2016 IEEE, IEEE 15th international symposium on network computing and applications (NCA). ISBN 978-1-5090-3216-7. https://doi.org/10.1109/NCA.2016.7778603.

Hyunwoo, N., Calin, D., Schulzrinne, H. (2016). Towards dynamic MPTCP path control using SDN. In 2016 IEEE netsoft conference and workshops (NetSoft), 04 July 2016. https://doi.org/10.1109/NETSOFT.2016.7502424. ISBN 978-1-4673-9486-4.

Jarraya, Y., Madi, T., & Debbabi, M. (2014). A survey and a layered taxonomy of software-defined networking. *IEEE Communications Surveys Tutorials, 99*, 1.

Jouet, S., Perkins, C., & Pezaros, D. (2016). OTCP: SDN-managed congestion control for data center networks. In NOMS 2016-2016 IEEE/IFIP network operations and management symposium (pp. 171–179). IEEE.

Kim, H., Schlansker, M., Santos, J. R., Tourrilhes, J., Turner, Y., & Feamster, N. (2012). Coronet: Fault tolerance for software defined networks. In 2012 20th IEEE international conference on network protocols (ICNP) (pp. 1–2). IEEE.

Krishnan, R., Figueira, N. (2015). Analysis of data center SDN controller architectures: technology and business impacts. In 2015 international conference on computing, networking and communications (ICNC), 30 March 2015. https://doi.org/10.1109/ICCNC.2015.7069324. ISBN 978-1-4799-6959-3.

Lee, M.-C., & Sheu, J.-P. (2016). An efficient routing algorithm based on segment routing in software-defined networking. *Science Direct, Computer Networks, 103*(5), 44–55.

Lei, Y.-c., wang, K., hsu, y.h. (2015). Multipath routing in SDN-based data center networks. In 2015 European conference on networks and communications (EuCNC), 13 Aug 2015. https://doi.org/10.1109/EuCNC.2015.7194100. ISBN. 978-1-4673-7359-3.

Li, L., Hu, N., Liu, K., Fu, B., Chen, M., & Zhang, L. (2015). Amtcp: an adaptive multi-path transmission control protocol. In Proceedings of the 12th ACM international conference on computing frontiers (pp. 1–8).

Lu, Y., & Zhu, S. (2015). SDN-based TCP congestion control in data center networks. In 2015 IEEE 34th international performance computing and communications conference (IPCCC) (pp. 1–7). IEEE.

http://mininet.org/.

Niranjan Mysore, R., Pamboris, A., Farrington, N., Huang, N., Miri, P., Radhakrishnan, S., ... & Vahdat, A. (2009, August). Portland: a scalable fault-tolerant layer 2 data center network fabric. In Proceedings of the ACM SIGCOMM 2009 conference on Data communication (pp. 39–50).

ns-3 project (2013). ns-3: OpenFlow switch support. http://www.nsnam.org/docs/release/3.13/models/html/openflowswitch.html.

Oktian, Y. E., Lee, S. G., Lee, H. J., & Lam, J. H. (2017). Distributed SDN controller system: A survey on design choice. *Science Direct, Computer Networks, 121*, 100–111.

Opendaylight Controller [Online] Available at: https://www.odl.org/.

Paasch, C., Khalili, R., & Bonaventure, O. (2013). On the benefits of applying experimental design to improve multipath TCP. In Proceedings of the ninth ACM conference on Emerging networking experiments and technologies (pp. 393–398).

Pang, J., Xu, G., & Fu, X. (2017). SDN-based data center networking with collaboration of multipath TCP and segment routing. *IEEE Access, 5*, 9764–9773. https://doi.org/10.1109/ACCESS.2017.2700867.

Raiciu, C., Barre, S., Pluntke, C., Greenhalgh, A., Wischik, D., Handley, M., (2011). Improving datacenter performance and robustness with multipath TCP. In SIGCOMM '11 proceedings of the ACM SIGCOMM 2011 conference (pp. 266–277). https://doi.org/10.1145/2018436.2018467. ISBN 978-1-4503-0797-0.

Roy, A. R., Bari, M. F., Zhani, M. F., Ahmed, R., & Boutaba, R. (2014). Design and management of dot: A distributed openflow testbed. In 2014 IEEE network operations and management symposium (NOMS) (pp. 1–9). IEEE.

Sandri, M., Silva, A., Rocha, L. A., & Verdi, F. L. (2015). On the benefits of using multipath tcp and openflow in shared bottlenecks. In 2015 IEEE 29th international conference on advanced information networking and applications (pp. 9–16). IEEE.

Sflow. [Online] Available at: https://github.com/sflow-rt/mininet-dashboard.

Sheu, J. P., Liu, L. W., Jagadeesha, R. B., & Chang, Y. C. (2016). An efficient multipath routing algorithm for multipath TCP in software-defined networks. In 2016 European conference on networks and communications (EuCNC) (pp. 371–376). IEEE.

Singla, A., Hong, C.-Y., Popa, L., Godfrey, P.B. (2012). Jellyfish: Networking data centers randomly. In NSDI.

Wireshark. [Online] Available at: http://mininet.org/walkthrough/

Wischik, D., Raiciu, C., Greenhalgh, A., & Handley, M. (2011). Design, implementation and evaluation of congestion control for multipath TCP. In NSDI (Vol. 11, pp. 8–8).

https://www.elsevier.com/books/network-routing/medhi/978-0-12-800737-2.

Xia, W., Zhao, P., Wen, Y., & Xie, H. (2017). A survey on data center networking (DCN): infrastructure and operations. *IEEE Communications Surveys & Tutorials, 19*(1). https://doi.org/10.1109/COMST.2016.2626784, ISSN 553-877X.

Xu, G., Dai, B., Huang, B., Yang, J. (2015). Bandwidth aware energy efficient routing with SDN in data center networks. In 2015 IEEE 17th international conference on high performance computing and communications, 2015 IEEE 7th international symposium on cyberspace safety and security, and 2015 IEEE 12th international conference on embedded software and systems, 30 Nov 2015. https://doi.org/10.1109/HPCC-CSS-ICESS.2015.12. ISBN 978-1-4799-8937-9.

Yuan, H., Jay Kuo, C.-C., Ahmad, I., 2010. Energy efficiency in data centers and cloud-based multimedia services: An overview and future directions. In 2010 international conference on green computing (pp. 375–382).

Zannettou, S., Sirivianos, M., & Papadopoulos, F. (2016). Exploiting path diversity in datacenters using MPTCP-aware SDN. In 2016 IEEE symposium on computers and communication (ISCC) (pp. 539–546). IEEE.