Opposition Report

Karthika and Giarré - Group 2

In general terms, the report is quite good, and I enjoyed reading it. It is obvious at first sight that you have done a great job so far. I liked the formulation of the problem. It is not ready to be delivered as the final version, although it is not far from it. All I can say is great job, but do not become complacent.

Some of the key aspects to improve:

- Explain some acronyms and their meanings.
- In the literature review, explain why all those papers are important for the research.
- You have room for improvement and further analysis in the final part. It is interesting, but it feels too short.
- Define clearly who is your target and why is it interesting to them. Try to give some key takeaways from this report. For instance, if it is addressed to a network operator engineer, they may want to explain in 2 to 3 things why ECMP or Top-K is better in one network to its boss.

A final remark that I would like to see and that makes your research even more accessible and appealing is to share the code in a GitHub repository or the appendix, for example. This ensures you are fully transparent about your results.

Effectiveness of Flow Migration Strategies in an SD-WAN Environment

SATHEESH KARTHIKA GIARRÈ FEDERICO
satheesh | fgiarre @kth.se

November 27, 2023

Software Defined Networking (SDN) has revolutionized the management of network infrastructures in the era of 5G and beyond, allowing multiple tenants to programmatically optimize shared physical resources to meet diverse service requirements. Within this domain, Software-Defined Wide Area Networks (SD-WANs) have emerged as a critical technology, enabling companies to interconnect branch networks with greater flexibility and efficiency. This report presents a comprehensive analysis of contemporary strategies for resource placement and traffic migration within SD-WANs, focusing on their efficacy and economic viability. Through discrete-time simulations, we compare heuristic and AI-enabled approaches, examining their performance in ensuring optimal Quality of Service (QoS) and resource utilization. Our findings elucidate the operational trade-offs inherent in each method, providing valuable insights for the strategic deployment of SD-WANs. The research highlights the benefits and limitations of each strategy, thus guiding network administrators in making informed decisions to enhance network performance and service delivery.

Contents

1 Introduction

This research explores the application of Software Defined Networking (SDN) and Software-Defined Wide Area Networks (SD-WAN) technologies for optimizing path placement and migration in geographically distributed service networks. Software Defined Networking (SDN) is a novel paradigm in the networking field that allows changing the network behaviours programmatically [1]. While SDN has proven effective for large-scale data centres and Internet Service Providers, its architecture is less suited for small-scale operations [1]. This led to companies adopting SD-WAN infrastructure for its programmable networking capabilities [2, 3]. In SD-WAN, companies share a common infrastructure, choosing paths for their data that facilitate operations requiring inter-branch cooperation. This makes it a very appealing approach for small/medium-scale companies.

One common use case of SDN is to reserve resources on demand, migrating network traffic to provide full capacity to a requesting service. SD-WAN offers fewer methods for achieving this, but path placement and migration can still be used to reserve certain paths for requesting service. This study aims to address a critical gap: optimizing network resource allocation in SD-WAN environments, particularly for companies requiring cooperation between branches.

In this context, our research focuses on a key question: "How do different path placement and migration methods, including heuristic and AI-enabled solutions, compare in terms of efficiency and effectiveness in SD-WAN environments for ensuring optimal Quality of Service (QoS) and Load Balancing?" This question guides our investigation into the effectiveness of various strategies in optimising network resource allocation in SD-WAN environments.

In this report, we started with an extensive literature review in Section 2 on how to develop a taxonomy of the most prevalent techniques for addressing Quality of Service (QoS) and Load Balancing in the context of path placement and migration. The formalization of the research problem and the hypothesis are articulated in Section 3. Following this, Section 4 outlines the modelling and description of our discrete-time test bed, utilised to test some of the methods identified in the literature review. The analysis and interpretation of the results obtained are presented in Section 5. The report concludes with final observations and considerations in Section 6.

2 Literature Study

In the survey about the use of SDN for load balancing from Hamdan et al. [4], a complete taxonomy of all the state-of-the-art methods of load balancing using SDN is found. Hamdan et al. describe the space of link load balancing solutions to be currently implemented in 3 ways: *i)* Meta-heuristic, *ii)* Machine Learning, and *iii)* Other methods. The aim of using these methods is usually to choose a way of placing paths in order to avoid link congestion by distributing the flows in different paths while trying to deliver good Quality of Service (QoS).

We can follow the survey from Hamdam et al. [4] in order to find significant examples for implementations of link load balancing.

2.1 Meta-heuristics

A heuristic is a technique aimed to solve a problem faster when traditional techniques are too slow. A metaheuristic is a higher-level technique or heuristic that seeks, generates, or selects a heuristic that may provide a sufficiently good solution to an optimization problem [5].

2.1.1 ECMP

A classical implementation for solving the problem of load balancing is using ECMP [6]. A hash value is computed based on the packet fields, and the resulting hash is used in order to take different paths in the network. ECMP paths of different flows may have some sections in common, and while this is not an issue for Mice Flows (MFs), it can create bottlenecks when Elephant Flows (EFs) needs to be placed. In order to address this problem, a solution is to try and understand whether a flow is actually an MF or EF in order to avoid the overlapping of two EFs [7].

2.1.2 Top-K paths

In order to avoid random paths being too long and not viable for good QoS, the authors in [8] compute the Top-K paths from source to destination and then pick one of them randomly. For computing the Top-K paths, the authors use shortest paths algorithms initially but switch to a Fuzzy Synthetic Evaluation Mechanism (FSEM) once the network gets loaded. When using FSEM, the Top-K paths will be computed by considering weights based on the number of hops and link loads.

2.1.3 Colonies

Authors in [9, 10] solve the problem by implementing colonies of artificial animal-behaving agents (e.g. ants, bees) in order to solve the path routing problem by taking as inspiration the behavior of the animal in the real world. This leads to good results when it comes to multi-constrained min-maxing problems, such as optimizing the paths by constrained like energy efficiency, delay, loss, etc.

2.1.4 Genetic Algorithms

Genetic Algorithms [11] are heuristics that keep improving their results until they reach a good enough solution. This usually translates to running a fast heuristic several times to reduce the solution space until only one result, hopefully the optimal one, remains like in the case of [12]. We can also find in the literature the combination of colonies algorithms and genetic algorithms like in [13] in order to enhance the quality of dynamic migration of EFs.

2.2 Machine Learning

These methods exploit the centralized and global view of SDN controllers combined with the learning of Machine Learning (ML) models in order to take decisions.

2.2.1 Neural Networks approaches

A Neural Network called BPANN was proposed by Chen et al. in [14], optimizing the flow placement and migration to achieve low latencies. Unfortunately, the results show that while the model is capable of diminishing the latency in communications, all the other metrics where left behind. In general, other Neural Networks described in the survey [4] suffer from the same results.

2.2.2 Reinforcement Learning and Deep Reinforcement Learning approaches

Reinforcement Learning (RL) aims to create an intelligent agent that learns the best actions to take in a certain environment, in order to get better with time. Zhang et al. implemented a RL model that performed well in migrating flows while retaining almost optimal efficiency in small networks [15]. In fact, one limitation of RL models is that the action space should be limited, while in larger networks the possibilities and scenarios can explode in size. To tackle this problem, more generalized approaches are obtainable thanks to Deep Reinforcement Learning (DRL). An implementation of a DRL model used for data plane load balancing can be found in [16].

2.3 Other methods

Other methods comprehend simple naive flow migration between shortest paths, discarding paths with bottlenecks or saturated links like in the case of [17, 18] in order to assure quality of service.

3 Research Question and Hypothesis

The extensive literature on SDN and SD-WAN highlights various methods for path placement and migration, yet there is a significant research gap in the comparative analysis of these methods. This leads us to our central research question:

Research Question: ”How do different path placement and migration methods, including heuristic and AI-enabled solutions, compare in terms of efficiency and effectiveness in SD-WAN environments for ensuring optimal Quality of Service (QoS) and Load Balancing?”

This question is pivotal for understanding the comparative advantages and limitations of different strategies in the specific context of SD-WAN environments. We aim to compare the different strategies proposed in the literature (on a common testbed when possible) in order to assess the advantages and disadvantages when it comes to picking one strategy over the others.

We hypothesize that while Machine Learning (ML) methods may excel in specific tasks, they may not generalize well enough to adapt to changing network conditions. Conversely, meta-heuristic approaches could offer a more balanced trade-off between time complexity and performance outcomes. This hypothesis will guide our exploration in the later sections, where we evaluate these strategies using a discrete-time simulation framework.

4 Research Methods

Our research adopts a rigorous quantitative methodology, with discrete-time simulations serving as the primary tool for data collection and analysis. This approach is chosen for its precision in capturing the dynamic interactions within a network and providing measurable outputs that enable a robust comparison of different network strategies.

4.1 Methodological Framework

The methodological framework of our research is grounded in the principles of network theory and optimisation algorithms. These provide a robust theoretical and philosophical foundation, offering guidelines for solving the research problem and establishing correct procedures for finding solutions. It ensures that our approach is aligned with established scientific practices.

4.2 Simulation Methods

The core activity of our research is the execution of a series of discrete-time simulations to assess the performances of different strategies. As the simulation runs, new requests are created for resource allocation (in this specific scenario, bandwidth request) and link creation, for services initiated in different nodes. A virtual link composed of a series of physical links is created to satisfy this request, also called a Flow. Occasionally, Flows can request the priority usage of links, requesting the migration of other Flows to other links if possible. At this point, the different strategies can be applied and the results assessed.

Our simulation methods are detailed as follows:

1. **Initialization:** We configure the simulation environment using SimPy and establish parameters based on the ”Garr2011” topology, which reflects the infrastructure connecting major Italian cities.

2. **Execution:** Simulations run with varying network loads and flow requests. Flows are initiated and terminated at random intervals, modelled by a Poisson distribution to reflect real-world network traffic dynamics.

3. **Data Collection:** We record key performance metrics at each simulation interval, including Path Efficiency, Saturation, Link Load, and Unused Links.

4. **Analysis:** Statistical methods are applied to the data to evaluate the performance of each network strategy, with a focus on understanding the efficacy of path placement and migration.

In the simulator, Flows enter the network at random intervals, following a distribution $A \sim Poisson(\text{MTBA})$ where MTBA is the Mean Time Between Arrivals (MTBA), a parameter set for the whole simulation. Flows also exit the network when their task is finished: following the same reasoning used for the arrivals. Flows duration in the network is sampled from a distribution $E \sim Poisson(\text{MTBE})$, where MTBE is the Mean Time Between Exits (MTBE). In order to model the migration event happening, the Mean Time Between Migrations (MTBM) is used as the rate of distribution. The topology used to test the strategies is "Garr2011"[*]. The choice of this topology was beneficial in two different ways. Firstly, the size and number of nodes are adequate to simulate different companies having multiple branches, connected via a shared SD-WAN infrastructure. Additionally, the multigraph nature of the topology allows us to grasp the nature of SD-WAN where different links connecting the same nodes represent different infrastructures (e.g. radio, copper, fiber). The simulation parameters used are:

Parameter	Value / Range
MTBA	5 ticks
MTBE	50 ticks
MTBM	10 ticks
Link Capacity	[34 Mbps, 20 Gbps]
Flow requested bandwidth	[10 Mbps, 1 Gbps]

Table 1: Parameters used in the simulation

Whenever a new flow arrives in the system, a simple Dijkstra path is computed between its source and destination. Given p the path returned by the Dijkstra shortest path, b_r the flow's bandwidth request, and l_{e^t} the load of edge e at time t, when a flow is placed on the network every load of edges used in the path between source and destination is updated using the following equation:

$$\forall l \in p \quad l_{e^{t+1}} = l_{e^t} + b_r \tag{1}$$

Given b_e the bandwidth available at the edge e, we incur in the saturation of a link when $l_{e^t} > b_e$.

4.3 Strategies Tested

In our simulation environment, we rigorously test two main strategies to evaluate their performance in terms of path placement and migration.

4.3.1 Equal-Cost Multipath (ECMP)

The Equal-Cost Multipath (ECMP) is a routing strategy that distributes the traffic among different paths having the same costs. We implemented this concept by generating all paths that have the same cost (in this case, the length of the path) of the shortest path between the source and the destination of the Flow. We then instantiate the Flow on the path that has the least saturated edges. This method has some advantages and disadvantages: The main advantage is that since the number of edges used is always equal to the ones of the shortest path, the lowest cost in terms of hops possible is assured. However, since only the shortest paths are used, some important links may get congested, while others may not be used at all.

[*] Taken from http://www.topology-zoo.org/

4.3.2 Top-K paths

Contrasting with ECMP, in the Top-K paths strategy, we consider the k-shortest paths that connect the source and the sink of the flow, and one of these is picked as the path to use. This method expands the potential routes that traffic can take, distributing the load more evenly across the network at the cost of potentially longer paths. This strategy may improve overall network performance by reducing the likelihood of congestion on any single link. The implication of this behaviour will be discussed in the Results section.

4.4 Strategy Assessment Metrics

To assess the efficiency of the placement and migration strategies, we designed the following assessment metrics: *Path efficiency* (P), *Saturation* (S), *Link Load* (L), *Unused Links* (U). These metrics provide a comprehensive view of network performance, resource utilisation and strategy effectiveness, allowing us to compare the efficacy of different path placement and migration methods in SD-WAN environments.

4.4.1 Path efficiency

This metric is used to assess the optimality of the paths computed. Since resources are often expensive and latency is key to today's operations, we want to avoid strategies that use paths that are too long. Given $|p^*|$ as the length of the shortest path connecting the source and destination and $|p|$ as the length of the path used by the flow, we use the following formula to calculate the mean value:

$$P = \frac{|p|}{|p^*|} \tag{2}$$

In the next sections, we are going to use the mean of the value P. However, since the variation of P can signify both an improvement or degradation with respect to previous placements, we also record the standard deviation of it σP in order to show the window of variation that the value P is subject to. The narrower the σP the more deterministic and reliable the strategy can be.

4.4.2 Saturation

This metric is used to measure the saturation level of the network, returning the ratio of links that are saturated over the network. To assess this metric, given E the set of all the edges present in the network, we used the following formula:

$$S = \frac{1}{|E|}\sum_{e}^{E} \rho \tag{3}$$

Where ρ is defined as:

$$\rho = \begin{cases} 1 & \text{if} \quad l_e > b_e \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

4.4.3 Link Load

This metric is used to assess the mean load of every (used) link. This can be useful when addressing how well the load balancing of the strategy is performing. Given A the set of edges in the network that are active at the moment of the computation (when $l_e > 0$), the value of L can be computed as:

$$L = \frac{1}{|A|}\sum_{a}^{A} \frac{l_a}{b_a} \tag{5}$$

4.4.4 Unused Links

This metric can be used to check the amount of links that the strategy is currently using. If a strategy is always trying to target the same edges, most of the links will remain unused while the used ones will become saturated. Given I the set of edges that are inactive (when $l_e = 0$), the ratio of unused links over the total links can be computed as:

$$U = \frac{|I|}{|E|} \tag{6}$$

5 Results and Analysis

By running the simulations, we were able to extract some meaningful results from the metrics described in Section 4.2.

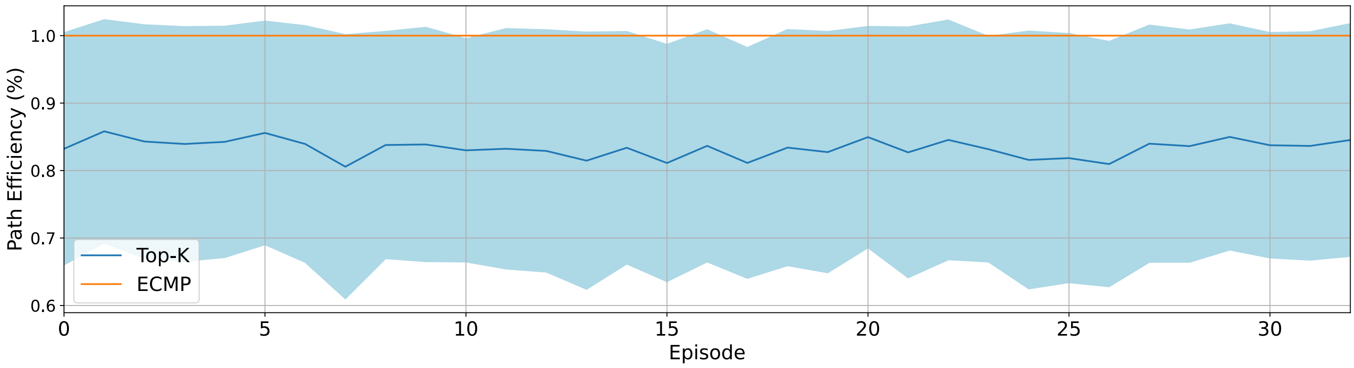

Figure 1: Mean Path Efficiency of the Strategies

5.1 Path Efficiency

Figire 1 plots the Mean Path Efficiency of the strategies. As observed in the figure, ECMP always has a value of 1, since the paths chosen are of the same length as the shortest path. On the contrary, the efficiency of Top-K spans between 0.8 and 0.9, though the standard deviation indicates that for each migration this value can go as high as 1 while it could also deteriorate and go as low as 0.6. This variation is because the Top-K can find solutions that are near the optimal but can also choose paths with up to 2 more hops with respect to the optimal path.

5.2 Saturation

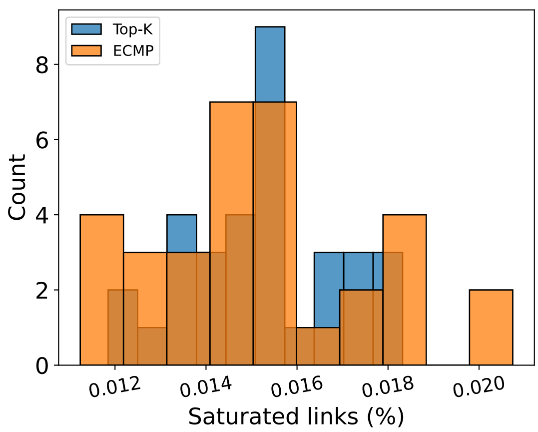

Figure 2: Ratio of saturated links in the network

Figure 2 shows the distribution of the mean saturation S during the simulated episodes. One thing that is noticeable, is that ECMP values are more spread in the scale, while Top-K is more concentrated in the middle values.

5.3 Link Usage

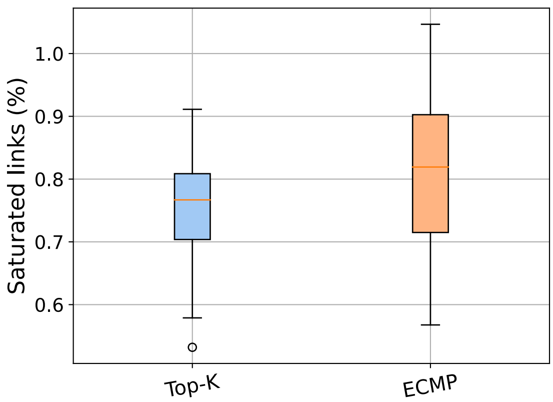

Figure 3: Distribution of mean Link Usage

In Figure 3 boxplots are plotted in order to show the distribution of the mean values of L during the various episodes. We observe that the Top-K has a lower median than ECMP. This is because with Top-K takes into consideration links that are not usually taken into consideration by ECMP, distributing the load more efficiently in the network.

5.4 Unused Links

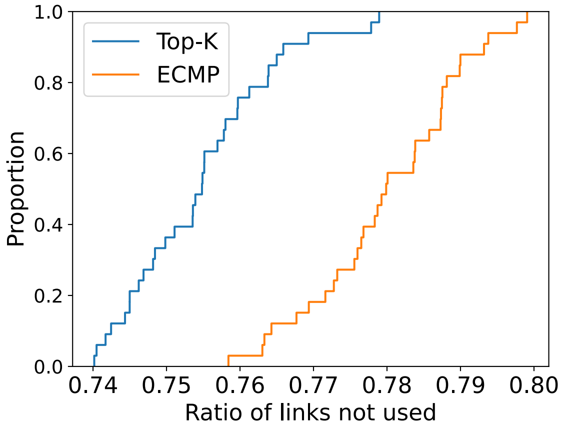

Figure 4: CDF for the distribution of unused link ratio

As seen in Figure 4, the ratio of unused links is 3-4% lower when using the Top-K strategy, confirming the assumption made in the previous plots about the load being distributed in multiple links.

5.5 Implications for SD-WAN Environments

The findings from our simulations provide actionable insights into the management of SD-WAN environments. With a focus on Quality of Service and Load Balancing, the results elucidate the practical trade-offs associated with the ECMP and Top-K path strategies. These insights are instrumental for network administrators, informing the strategic decision-making process for efficient network traffic management and guiding infrastructure investment planning.

6 Discussion

In our discussion, we reflect on the broader significance of our results, their reliability, and their applicability to real-world network management.

After the results of the simulation, we can address the strengths and weaknesses of both Top-K and ECMP: *i)* Top-K achieves a better spread of the load and less saturation than ECMP, at the cost of a lower path efficiency; *ii)* ECMP achieves perfect efficiency, at the cost of congestion in the fewer links that are considered.

When it comes to other methods such as colonies and genetic algorithms, the literature review reveals that these methods are not suited for dynamic problems. This is because the computation needs to be performed from scratch at each network change, and this can pose a problem in a real scenario where a multitude of factors can impact the system.

Machine Learning methods (Deep Reinforcement Learning (DRL), more precisely) has already been applied to the problem in the paper [19], obtaining high results when trying to optimize the policies for single parameters. However, RL and DRL methods work poorly when the environment is dynamic, due to the policies being built on top of observation of the environment itself. A promising approach to solve this problem is to use Graph Neural Networks [20] in order to add an additional layer that returns a view of the environment that is always the same for the DRL agent, even if the network underneath is changing.

6.1 Trustworthiness of Results

To ensure the trustworthiness of our results, we incorporate the following practices:

- **Validation of Models:** Our simulation models are validated against known benchmarks and published results to ensure they accurately reflect network behaviour.
- **Replicability:** All methods are documented in detail to allow for replicability by independent researchers.
- **Transparency:** We maintain transparency by making our data and simulation code available for review.
- **Statistical Rigour:** Statistical analysis is performed to validate the significance of our results, using confidence intervals and hypothesis testing where appropriate.

List of Acronyms and Abbreviations

DRL Deep Reinforcement Learning

ECMP Equal-Cost Multipath

EF Elephant Flow

FSEM Fuzzy Synthetic Evaluation Mechanism

MF Mice Flow

ML Machine Learning

MTBA Mean Time Between Arrivals

MTBE Mean Time Between Exits

MTBM Mean Time Between Migrations

QoS Quality of Service

RL Reinforcement Learning

SDN Software Defined Networking

References

[1] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 27–51, 2014.

[2] T. Jackisch, "Sdn vs traditional network," 04 2022.

[3] K. G. Yalda, D. J. Hamad, and N. Ţăpuş, "A survey on software-defined wide area network (sd-wan) architectures," in *2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. IEEE, 2022, pp. 1–5.

[4] M. Hamdan, E. Hassan, A. Abdelaziz, A. Elhigazi, B. Mohammed, S. Khan, A. V. Vasilakos, and M. Marsono, "A comprehensive survey of load balancing techniques in software-defined network," *Journal of Network and Computer Applications*, vol. 174, p. 102856, 2021. doi: https://doi.org/10.1016/j.jnca.2020.102856. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520303222

[5] A. A. Bara'a, A. D. Abbood, A. A. Hasan, C. Pizzuti, M. Al-Ani, S. Özdemir, and R. D. Al-Dabbagh, "A review of heuristics and metaheuristics for community detection in complex networks: Current usage, emerging development and future directions," *Swarm and Evolutionary Computation*, vol. 63, p. 100885, 2021.

[6] C. Hopps, "Analysis of an equal-cost multi-path algorithm," Tech. Rep., 2000.

[7] A. R. Curtis, W. Kim, and P. Yalagandula, "Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 1629–1637.

[8] J. Li, X. Chang, Y. Ren, Z. Zhang, and G. Wang, "An effective path load balancing mechanism based on sdn," in *2014 IEEE 13th international conference on trust, security and privacy in computing and communications*. IEEE, 2014, pp. 527–533.

[9] O. Dobrijevic, M. Santl, and M. Matijasevic, "Ant colony optimization for qoe-centric flow routing in software-defined networks," in *2015 11th International Conference on Network and Service Management (CNSM)*, 2015. doi: 10.1109/CNSM.2015.7367371 pp. 274–278.

[10] A. Fawaz, I. Mougharbel, and H. Y. Kanaan, "New routing application using bees colony for energy internet," in *2022 3rd International Conference on Smart Grid and Renewable Energy (SGRE)*, 2022. doi: 10.1109/SGRE53517.2022.9774114 pp. 1–8.

[11] M. Srinivas and L. Patnaik, "Genetic algorithms: a survey," *Computer*, vol. 27, no. 6, pp. 17–26, 1994. doi: 10.1109/2.294849

[12] S. Jamali, A. Badirzadeh, and M. S. Siapoush, "On the use of the genetic programming for balanced load distribution in software-defined networks," *Digital Communications and Networks*, vol. 5, no. 4, pp. 288–296, 2019.

[13] H. Li, H. Lu, and X. Fu, "An optimal and dynamic elephant flow scheduling for sdn-based data center networks," *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 1, pp. 247–255, 2020.

[14] C. Chen-Xiao and X. Ya-Bin, "Research on load balance method in sdn," *International Journal of Grid and Distributed Computing*, vol. 9, no. 1, pp. 25–36, 2016.

[15] J. Zhang, M. Ye, Z. Guo, C.-Y. Yen, and H. J. Chao, "Cfr-rl: Traffic engineering with reinforcement learning in sdn," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2249–2259, 2020.

[16] C. Yu, J. Lan, Z. Guo, and Y. Hu, "Drom: Optimizing the routing in software-defined networks with deep reinforcement learning," *IEEE Access*, vol. 6, pp. 64 533–64 539, 2018.

[17] L. S. da Silva, C. R. Storck, and F. de LP Duarte-Figueiredo, "A dynamic load balancing algorithm for data plane traffic." in *LANOMS*, 2019.

[18] Z. Guo, S. Hui, Y. Xu, and H. J. Chao, "Dynamic flow scheduling for power-efficient data center networks," in *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*. IEEE, 2016, pp. 1–10.

[19] X. Xiao, "Dvne-drl: Dynamic virtual network embedding algorithm based on deep reinforcement learning," 2023.

[20] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, 2020.