applied sciences

MDPI

Article

A Novel Strategy for Computing Routing Paths for Software-Defined Networks Based on MOCell Optimization

Jose E. Gonzalez-Trejo [1,*], Raul Rivera-Rodriguez [2,*], Andrei Tchernykh [3], Jose E. Lozano-Rizk [2], Salvador Villarreal-Reyes [1], Alejandro Galaviz-Mosqueda [4] and Jose L. Gonzalez Compean [5]

1 Centro de Investigacion Cientifica y de Educacion Superior de Ensenada, Electronics and Telecommunications Department, Ensenada 22860, Mexico
2 Centro de Investigacion Cientifica y de Educacion Superior de Ensenada, Telematics Division, Ensenada 22860, Mexico
3 Centro de Investigacion Cientifica y de Educacion Superior de Ensenada, Computer Science Department, Ensenada 22860, Mexico
4 Monterrey Unit, Centro de Investigacion Cientifica y de Educacion Superior de Ensenada, Apocada 66629, Mexico
5 Centro de Investigación y de Estudios Avanzados del I.P.N. (Cinvestav Tamaulipas), Victoria City 87130, Mexico
* Correspondence: gtjose@cicese.edu.mx (J.E.G.-T.); rrivera@cicese.edu.mx (R.R.-R.)

Abstract: Software-defined networking (SDN) is the fastest growing and most widely deployed network infrastructure due to its adaptability to new networking technologies and intelligent applications. SDN simplifies network management and control by separating the control plane from the data plane. The SDN controller performs the routing process using the traditional shortest path approach to obtain end-to-end paths. This process usually does not consider the nodes' capacity and may cause network congestion and delays, affecting flow performance. Therefore, we evaluate the most conventional routing criteria in the SDN scenario based on Dijkstra's algorithm and compare the found paths with our proposal based on a cellular genetic algorithm for multi-objective optimization (MOCell). We compare our proposal with another multi-objective evolutionary algorithm based on decomposition (MOEA/D) for benchmark purposes. We evaluate various network parameters such as bandwidth, delay, and packet loss to find the optimal end-to-end path. We consider a large-scale inter-domain SDN scenario. The simulation results show that our proposed method can improve the performance of data streams with TCP traffic by up to 54% over the traditional routing method of the shortest path and by 33% for the highest bandwidth path. When transmitting a constant data stream using the UDP protocol, the throughput of the MOCell method is more than 1.65% and 9.77% for the respective paths.

Keywords: quality of service; routing; software-defined network; genetic algorithms

Citation: Gonzalez-Trejo, J.E.; Rivera-Rodriguez, R.; Tchernykh, A.; Lozano-Rizk, J.E.; Villarreal-Reyes, S.; Galaviz-Mosqueda, A.; Gonzalez Compean, J.L. A Novel Strategy for Computing Routing Paths for Software-Defined Networks Based on MOCell Optimization. *Appl. Sci.* **2022**, *12*, 11590. https://doi.org/10.3390/app122211590

Academic Editors: Mirosław Klinkowski, Francesco Palmieri and Davide Careglio

Received: 1 October 2022
Accepted: 11 November 2022
Published: 15 November 2022

1. Introduction

In recent years, there has been a rapid expansion of the Internet and the continuous development of information and communication technologies such as 5G [1], cloud computing [2], Internet of Things (IoT) [2,3], and Big Data [4]. However, the evolution of these technologies requires network operators and Internet service providers (ISPs) to adopt new strategies and technologies to meet the emerging traffic demands that cannot be met by traditional networks.

The traditional network design originated as a simple distributed system to provide best-effort packet communication services in a reliable environment using network devices to forward packets and make routing decisions by implementing packet routing rules. The traffic accounts for routing tables on devices such as switches/routers.

However, with the rapid evolution of networks, the constant increase in devices and the excessive consumption of bandwidth lead to specific performance problems such as

a higher rate of packet loss, a more significant increase in end-to-end transmission delay, and, in the worst case, network collapse [5].

Software-defined networking (SDN) is an emerging centralized network architecture technology that differs from traditional networks by decoupling the control plane from the data plane [6]. The data plane is responsible only for forward and switch data and consists of the network's physical infrastructure, such as switches and routers, both physical and virtual, accessible through an open interface. The control layer is the essential layer of the architecture, consisting of SDN controllers that centralize the intelligence and management of the network and provide a global view of the network [7,8]. SDN has a third layer, the application layer, which provides users with services such as traffic engineering, quality of service, and network security.

The SDN architecture also has two communication interfaces that allow the controllers to interact with the other layers:

- Southbound interface: Provides a communication environment between the controller and communication devices. Installs the appropriate flow rules in the device forwarding table. OpenFlow [9] is the open-source community's most widely implemented Southbound interface standard.
- Northbound interface: Provides communication between the SDN controller and network applications running on the application plane. This communication is critical, as each network application's requirements can be very different. Applications can communicate with the SDN using various APIs, such as ad hoc and REST APIs, to request network resources or services according to their needs.

SDN promises to solve several problems in traditional networks due to its flexibility. However, there are still some open problems and challenges, such as the design of routing algorithms that guarantee QoS. The SDN controller typically employs the shortest path method when an application requires the transfer of a flow between end-to-end networks [10]. As the variety of services increases, so do the requirements for service improvement, which include latency, bandwidth, packet loss rate, and network stability. The conventional best-effort routing algorithm results in the network architecture having certain constraints in order to optimize the performance parameters for the diversity of demands [11]. Figure 1 shows an example of a data flow that requires a path with a bandwidth of more than 80 Mbps and a delay of less than 7 ms. In the case of the traditional routing algorithm based on the shortest path, the traffic flows along the path R1,R2,R3,R7, which has the following values: bandwidth = 70 Mbps and delay = 8 ms. This does not meet the criteria for bandwidth service. This path may result in poor service performance and cause problems such as congestion. Therefore, it is helpful to consider different criteria for selecting a good path. For example, using the highest bandwidth path (R1,R2,R5,R6,R7) results in a bandwidth of 90 and a delay of 6, which meets the data flow requirements.

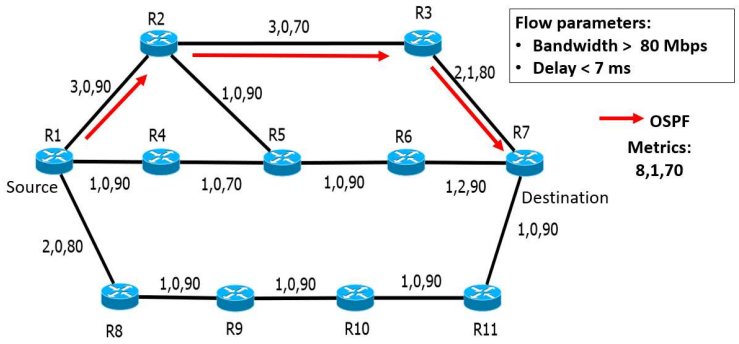

Figure 1. Traditional routing. Metrics: delay (ms), packet loss (%), and bandwidth (Mbps).

These problems can be avoided by effectively routing flows while considering available network parameters and QoS requirements. The selection of paths in the routing process must evaluate the network conditions through the multiple available metrics, which can be

represented as objectives. The goals may conflict in this type of problem, and each purpose has a different solution. With more than one optimization criterion, this routing process is known as a multi-objective optimization problem (MOP) [12].

Discovering paths that provide QoS in a network according to different data flow requirements is a difficult problem, which is usually solved optimally using heuristic algorithms [13], especially those based on integer linear programming, game theory, and evolutionary algorithms (EA). Algorithms based on game theory and approximation are usually computationally intensive. Therefore, these two methods are usually not suitable for providing acceptable solutions. On the other hand, EAs search for global optima by developing an iterative search within a finite time. They have proven to provide high-quality solutions to the problem within a limited computation time [14]. Genetic algorithms (GA) are a family of models of EAs. They have emerged as a method for optimizing difficult search problems based on the principle of genetic selection, such as inheritance mutation, selection, and crossover. Their advantage is that they can handle large search spaces and generate multiple solutions in a single pass, which is suitable for this type of QoS routing optimization for SDN [15].

Our proposal (Figure 2) resides in the application plane and communicates with the SDN controller using the REST API provided by the Northbound interface. We use these SDN capabilities to create, query, and delete flow rules on OpenFlow (OF) switches, in order to forward data flow according to the path established by our path selection proposal or optimization criteria, based on a cellular genetic algorithm for multi-objective optimization (MOCell).

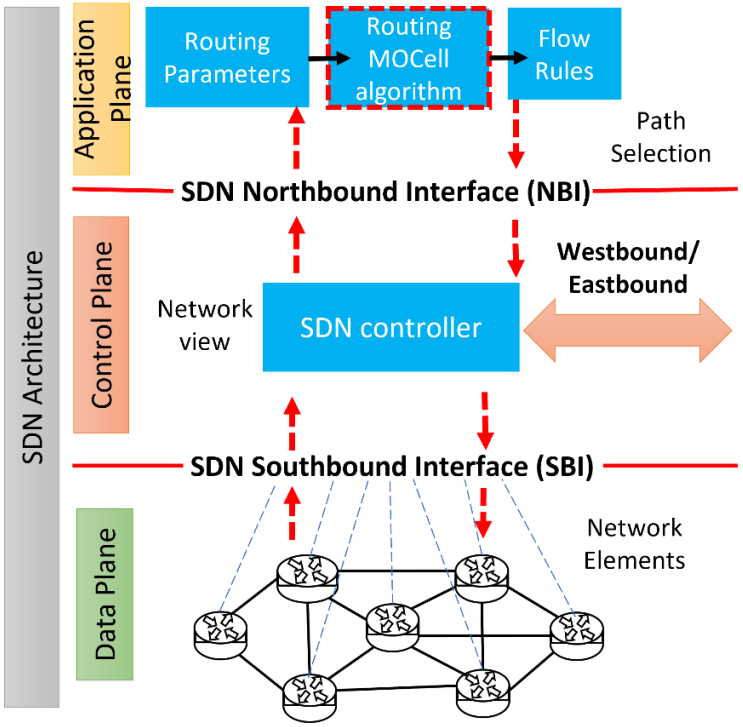

Figure 2. Routing architecture for SDN.

The main contributions of our proposal are as follows:

- An SDN strategy for routing data flows based on QoS that considers an inter-domain approach and compares its performance with traditional routing algorithms in the context of SDN.
- Using MOCell, the routing strategy considers some of the most critical network parameters: available bandwidth, delay, and packet loss.

The MOCell algorithm differs from conventional methods in evaluating multiple possible solutions from the first iteration. Its population is structured into small neighborhoods

that generate high-quality solutions. They are fed back with the non-dominant solutions stored in each iteration. This algorithm has shown better performance in extensive testing than the *NSGA-II* and *SPEA2* algorithms, which are widely recognized and used in practice [16].

Alongside this research, we set aside for future work the analysis of time complexity and computational delay. The fastest output results and least computational effort will likely be the result of Dijkstra's algorithm, rather than the MOCell method. However, the network performance results of this work presented in Section 4 show how the MOCell method obtains better results than Dijkstra's algorithm and the multi-objective evolutionary algorithm based on decomposition (MOEA/D). Additionally, the stationary state of the SDN network connection processes allows the implementation and evaluation of the MOCell method for a new inter-domain routing proposal.

The rest of the article is organized as follows: Related work is presented in Section 2. Section 3 presents our proposed system and describes the experimental setup. In Section 4, the performance evolution is presented, and the results are discussed. Finally, Section 5 describes the conclusions of the work and future directions.

2. Related Work

This section describes the research works on routing proposals to improve the quality of service (QoS) implemented in the SDN architecture. We describe the works according to the network parameters considered for QoS improvement, the use of SDN architecture resources, and their contribution depending on a single domain or multiple.

In OpenQoS [17], the authors developed an architecture that classifies incoming data streams into multimedia or data streams and provides quality service support for the different multimedia streams, considering the application's requirements. Their proposed algorithm focuses on the shortest path according to the delay as a constraint, while for data streams, it provides the best possible path according to the controller (hops).

In CECT [18], a network congestion avoidance scheme was proposed in which resources are reallocated based on data flow requirements. It should be noted that the routing algorithm uses a secondary genetic algorithm when there is a congestion problem. Therefore, the data streams are reallocated according to the minimum bandwidth required by the application. Jungmin et al. [19] proposed a method based on distributed data centers in an SDN cloud that guarantees bandwidth allocation for critical applications through priority queues in each network device. In AmoebaNet [20], a service is proposed that guarantees the QoS of services on campus or local area networks based on SDN and facilitates the transfer of large-scale scientific data (Big Data) depending on the application. The algorithm is based on the bandwidth constraint to calculate the end-to-end network path. Bastam et al. [21] proposed a dynamic method for traffic engineering in SDN data centers based on a linear propagation model and decomposition techniques to partition the flow allocation problem by determining the available capacity of paths for the redirection of the flows. In VSDN [22], the authors proposed an architecture within a domain for video transmission over SDN. This architecture includes a routing module and a network topology monitor that verifies network conditions. The authors monitored bandwidth, jitter, and link delay network parameters to forward data packets using a constraint-based routing algorithm. In PRIME-Q [23], a multi-domain routing method with end-to-end quality of service is presented, which considers privacy in a multi-domain SDN network. The algorithm can classify path flows according to three categories of QoS, bandwidth, delay, and the combination of both parameters in an optimization function.

In QosComm [24], the authors designed a strategy for routing data flows among an SDN-based distributed data center according to the application's QoS requirements. The algorithm selects the path with the minimum delay that satisfies the application requirements based on bandwidth, packet loss rate, and jitter. Li et al. [25] proposed a QoS-compatible routing method based on an evolutionary algorithm for multi-objective optimization (NSGA-II). This method considers the bandwidth threshold required by a

flow and minimizes the parameter delay and packet loss. These parameters are determined by monitoring the network regularly to avoid bottlenecks, unlike most previous studies based on the assumption of the same domain.

In VQoSRR [26], a video streaming adaptive QoS-based routing and resource reservation scheme is proposed that shifts traffic to an alternate path when the QoS is violated, which satisfies video demands and enhances user experience more effectively than best-effort networks. In Civanlar et al. [27], the authors proposed a multi-domain design where each domain has an SDN controller, which shares the summarized topology of their domain. Each controller has end-to-end decision-making capabilities, and the flow decision selection is based on the effective bandwidth theory of the martingale process.

In MCEAACO-QSRP [28], the authors designed a multi-objective optimization strategy based on adaptive ant colonies to guarantee the quality of service in the industrial Internet of Things (IoT), considering the QoS constraints of delay, security, and power consumption.

Li-Sheng et al. [13] proposed a routing algorithm based on a multi-objective genetic algorithm for the QoS of the optical burst communication network. The algorithm takes the parameters of the minimization of delay, bandwidth, and packet loss.

As a summary of the reviewed works (Table 1), we can argue that the most implemented routing model to guarantee QoS is focused on the available bandwidth parameter [18–21,26,27]. VSDN [22] and PRIME-Q [23] classified flows according to the multimedia flows' required bandwidth and delay parameters. PRIME-Q has the particularity of offering three options: the first is path selection with maximum available bandwidth; the second is path selection with the minimum delay; and the third is both network parameters as restrictions. Another approach is to reallocate resources when a congestion problem occurs [18,19,21,27], where the parameter to consider when improving the QoS is the bandwidth available in the network, without considering the delay and packet loss that may occur on the links, except for OpenQoS [17], which obtains a better performance by giving higher priority to delay than bandwidth. On the other hand, multi-objective techniques have been implemented to meet the requirements of multiple QoS parameters such as bandwidth, delay, packet loss, and jitter [24,25,28,29]. These methods focus on optimizing one objective function, while the other parameters satisfy the minimum requirements of the data streams. Our routing proposal differs mainly by considering the minimization of these parameters, which gives us a set of feasible and balanced solutions since they are non-dominated solutions.

Table 1. Comparison of QoS routing methods.

Reference	Network Parameters	Path Selection	Domain
[17]	Bandwidth, Delay, Jitter	Min delay and bandwidth as constraints	Single
[18]	Bandwidth	Min bandwidth	Single
[19]	Bandwidth	Max bandwidth	Inter-domain
[20]	Bandwidth	Max bandwidth	Single
[21]	Bandwidth	Max bandwidth	Single
[22]	Bandwidth, Delay, Jitter	Max bandwidth and delay as constraints	Single
[23]	Bandwidth, Delay	(1) Max bandwidth, (2) min delay, and (3) max bandwidth and delay as constraints	Inter-domain
[24]	Bandwidth, Delay, Jitter, Packet Loss	Application requirements as constraint	Inter-domain
[25]	Bandwidth, Delay, Jitter, Packet Loss	Application requirements as constraint	Single
[26]	Bandwidth	Bandwidth as constraint	Single
[27]	Bandwidth	Bandwidth as constraint	Inter-domain
[28]	Delay, Security, Power Consumption	Application requirements as constraint	Single
[13]	Bandwidth, Delay, Packet Loss	Min delay and application requirements as constraints	Single
MOCell Routing	Bandwidth, Delay, Packet Loss	Min (bandwidth, delay, and packet loss)	Inter-domain

3. Materials and Methods

3.1. Problem Formalization

Generally, a network topology can be represented as a directed graph $G(V, E)$, where V represents all nodes in the network and E represents all of the connections between the nodes. $n = |V|$ is the number of nodes and $m = |E|$ is the number of edges in the network. Each connection $e_{(i,j)} \epsilon E$ has associated properties, such as λ_e, the available bandwidth, μ_e, the delay, and ρ_e, the packet loss. The status and measurements of the network are updated when a routing request is made, in order to obtain the better accuracy of the available resources and to choose a path that satisfies the QoS parameters [29,30].

The path search problem must solve a path P from a source node s to a destination node d that satisfies the values of different network parameters. We define the following function for a path P:

Hops path computes the number of links between s and d node. For this case, the cost of each link is assigned to 1.

$$h(P) = \sum_{u,v \in P} e \tag{1}$$

The bandwidth path is the minimum available bandwidth between s to d node, and the cost metric of each link is set to be inversely proportional to the bandwidth capacity of that link.

$$\lambda(P) = min_{\forall e \in P} \left\{ \frac{1}{\lambda_e} \right\} \tag{2}$$

The delay path is the total path delay.

$$\mu(P) = \sum_{e \epsilon P} e_\mu \tag{3}$$

The packet loss path is the percentage of packets lost in comparison with the number of packages sent (0 means no packet loss).

$$\rho(P) = 1 - \prod_{e \epsilon P} (1 - e_\rho) \tag{4}$$

The delay function is additive, while the bandwidth function is concave, and the packet loss function is multiplicative but can be converted to an additive function by taking the logarithm of the ratio; with this parameter, we can provide network QoS.

Table 2 shows the network parameters described as variables.

Table 2. Description of the network parameters used.

Variable	Description
$G\ (V,E)$	Represents communication network topology.
V	Sets all nodes in the network.
E	Sets all links between nodes.
$n = \|V\|$	Number of nodes.
$m = \|E\|$	Number of edges in the network.
λ_e	The available bandwidth on a link.
μ_e	The delay on a link.
ρ_e	The packet loss on a link.
s	Source node.
d	Destination node.
P	Path from a source node to a destination node.
$h(P)$	Computes the number of links for a path.
$\lambda(P)$	Computes the minimum available bandwidth link on a path.
$\mu(P)$	Computes the total path delay.
$\rho(P)$	Computes the percentage of packets lost concerning the number of packages sent.

To evaluate path performance, we consider throughput [31] (Equation (5)) as a performance measure that refers to the successful delivery of data $E_{s,t}$ over a communication channel; in our case, a path where $E_{s,t}$ is the received data d_{rx} from source s_{tx} excluding retransmitted data r_{data} during the entire observation time T_t.

$$t_h = \frac{E_{s,t}}{T_t} \tag{5}$$

where

$$E_{s,t} = \frac{s_{tx} - (d_{rx} - r_{data})}{s_{tx}} \tag{6}$$

3.2. QoS Routing

The shortest path algorithm determines the path with the lowest connection cost, specified by a value inversely proportional to the available bandwidth of each connection. Dijkstra's algorithm is usually used for this type of search. In our case, we create variations of the shortest path using Dijkstra's algorithm to consider the path metrics with the lowest bandwidth cost and minimum hops, delay, and packet loss. [32].

- The minimum hops path computes the shortest path with the minimum number of links between the source and destination node.

$$H = min\ (h(P)) \tag{7}$$

- Minimum cost of available bandwidth path:

$$B = min(\lambda(P)) \tag{8}$$

- Minimum delay path:

$$D = min(\mu(P)) \tag{9}$$

- Minimum packet loss path:

$$PL = min(\rho(P)) \tag{10}$$

3.3. Dijkstra's Algorithm

Dijkstra's algorithm (Algorithm 1) is widely used in the shortest path routing algorithm. It is based on a set of candidate neighbor nodes, and the calculation to identify the shortest path between a source s and a destination d; in other words, it computes the shortest paths to all destinations [33].

For the path calculation with Dijkstra's algorithm, we need to calculate the cost between two nodes i, j and the path with the least cost to node j, where j $(j \neq i)$. We can represent this as two parameters:

$$dt_{i,j} = link\ cost\ between\ node\ i\ and\ node\ j \tag{11}$$

$$Dt_{i,j} = cost\ of\ the\ minimum\ cost\ pathbetween\ node\ i\ and\ node\ j \tag{12}$$

Dijkstra's algorithm uses two lists to specify the minimum cost of nodes, one containing the computed nodes of the permanent list S and another for the nodes still missing from the provisional list S'. At each iteration, the algorithm computes the shortest path from a neighboring node k to node i with the lowest cost to node i. The list S is incremented by adding the computed node. In contrast, the list S' is decremented by removing the nodes that are added to S.

Our work evaluated the shortest path using different QoS parameters (B, D, PL, Equations (8)–(10)). We use Dijkstra's algorithm, modifying the minimum cost of the links with the metric we want to evaluate, and select the path that corresponds to the source and destination of the data flow.

Algorithm 1. Dijkstra's shortest path algorithm (centralized approach)

1: S = {i} //permanent list; start with source node i
2: S′ = N\{i} // tentative list (of the rest of the nodes)
3: *Find* $k \epsilon S'$ such that $D_{ik} = min_{m \epsilon S'} D_{im}$ //identify a neighboring node k not in the list S with the minimum cost path from i.
4: S = S U {K} //Add k to the permanent list S
5: S′ = S′\{K} //Drop k from the tentative list S′
6: if S′ = ø //if S′ is empty stop
7: end
8: For j ϵ $N_k \cap$ S′ do //consider the list of neighboring N_k, of the intermediary k.
9: $Dt_{i,j}$= min (Dt_{ij}, $Dt_{ik} + dt_{kj}$) //check for improvement in the minimum cost path
10: return to step 3,

3.4. Multi-Constrained Optimal Path (MCOP)

The problem in this work considers three conflicting objectives that must be optimized simultaneously. The general formula of a MOP (multi-objective problem) [34] is as follows:

Find a vector $X^* = \left[x_1^*, x_2^*, \ldots, x_n^*\right]^T$, where m inequality constraints $g_i(x) \geq 0$, $i = 1, 2, \ldots, m$, p equation constraints $h_i(x) = 0, i = 1, 2, \ldots, p$, and minimizing the function $f(x) = [f_1(x), f_2(x), \ldots f_k(x)]^T$ are satisfied, and where $x = [x_1, x_2, \ldots x_n]^T$ is the decision variable vector.

The solution for an MOP may not be clear-cut because the objectives often conflict and cannot be met simultaneously, so no solution minimizes all objectives. The key is to construct the evaluation function for the objective problem. Two main approaches are usually used to formulate these functions. The first is to combine all of the objectives into a single objective using a feasible method or to select a particular objective and consider the other objectives as constraints. The second is to determine a representative set for all non-dominated solutions, called the Pareto front. For this case study, we consider approaching the problem using both approaches. We use the weighted sum for the first method and, for the second, the MOCell genetic algorithm. The forms are described in more detail below:

1. The weighted sum method [35] is the sum of the values corresponding to the different objectives, each multiplied by a weighted coefficient. This method provides a weight for the various network parameters that can be used as a cost for path computation. We use it as a fitness function for our proposed routing method.

$$w_{B,D,PL} = min_{x \in X} \sum_{i=1}^{k} w_i f_i(x) = min_{e \in p} \sum_{(u,v) \epsilon p} w_k(u,v) \tag{13}$$

2. The method based on Pareto dominance determines all non-dominated solutions that satisfy the conditions defined for the problem S (feasible region). Any point $x \epsilon S$ is a feasible solution. The feasible region is defined as follows [36]: Given 3 vectors $u = (u_i, \ldots, u_k)$, $v = (v_i, \ldots, v_k)$, and $w = (w, \ldots, w_k)$, we say that u dominates v and v dominates w (denoted by $u < v$ *and* $u < w$), if and only if u is less than v and w.

Each vector in the Pareto set has some correspondence in objective function space, obtaining the so-called Pareto front. Formally:

Pareto optimality: A solution $X^* \epsilon S$ is Pareto optimal if and only if it is non-dominated by any other solution $X' \epsilon S$.

Pareto optimal set: For a given MOP, $f(x)$ is the Pareto optimal set, and p_s is defined as follows:

$$p_S = \left\{x \in S \middle| \nexists x' \in S f\left(x'\right) \prec f\left(x\right)\right\} \tag{14}$$

Pareto front: For a given MOP, $f(x)$, and Pareto optimal set, p_S, the Pareto front, p_F, is defined as follows:

$$p_F = \left\{f(x) \epsilon \mathbb{R}^k \middle| x \in p^*\right\} \tag{15}$$

MOPs can have a Pareto front composed of a possibly infinite number of solutions. The algorithms that use stochastic techniques, such as metaheuristics, aim to find an approximation p_F, which represents the actual Pareto front.

To solve our problem with our proposed method, we need to find the path with the minimum end-to-end delay (D), packet loss (PL), and minimum cost of available bandwidth (B), as represented in Equation (16). The values of the functions are normalized to the minimum cost of delay and bandwidth for the QoS of the application.

$$Min_{e \in p} \ (B, D, PL) \tag{16}$$

3.5. MOCell-Based QoS Optimal Path

In this work, we consider the implementation of cellular GA for path selection considering the QoS of flows, particularly MOCell. The main feature of this algorithm is the distribution of its population in small neighborhoods, where each individual (possible solution to the problem) can be recombined only in the surrounding cells (neighborhood cells). Neighborhoods can be grouped with 4, 8, and 16 neighbors, with eight neighbors being the default configuration in the MOCell algorithm [37]. The population is represented on a two-dimensional toroidal grid so that everyone has the same number of neighbors. The main idea is to explore the search space further since overlapping neighborhoods cause a slow spread of solutions in the population. At the same time, genetic operators exploit each neighborhood (see Figure 3). MOCell maintains a file that stores the non-dominated solutions found in each iteration; the grouping distance of NSGA-II [38] is used to compute them. MOCell differs from other genetic algorithms by maintaining a diverse solution set and a feedback mechanism of non-dominant solutions to replace individuals in the population after each generation.

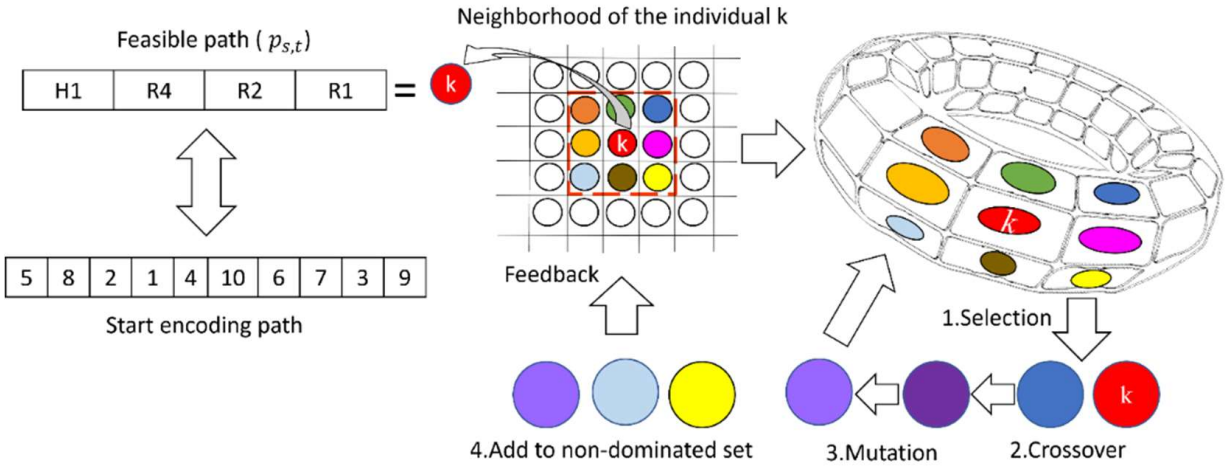

Figure 3. MOCell algorithm.

The genetic algorithms [39] start from a set (population) of potential solutions (individuals). The best solutions are selected according to a fitness function through a selection process to assign the parents of each neighborhood and carry out the crossover and mutation processes in order to find better solutions to the problem (Algorithm 2 introduces the MOCell pseudocode).

Algorithm 2. Pseudocode of metaheuristic based on MOCell

```
data = parameters    //initial parameter MOCell
pop = solution       // Create an initial population
grip = (pop)         //distribute the population in a toroidal grip.
P_F = []   //create pareto front = empty
While (terminal Condition = = false) do
    For individual∈pop do
        neighborhood = (individual) //everyone individual builds a neighborhood
        parents = (neighborhood)    //selection parents
        offspring = recombination (data, parents) //crossover
        offspring = mutation (data, offspring)
        pop = replacement ([parents], offspring)
        P_F = insert (Offspring)
    End
pop = feedback (ParetoFront)
End
```

One of the simplest methods for generating routing paths is random search. This involves using arrays of random integers to symbolize different paths. The population is generated by randomly assigning different priority IDs for each device, considering the size of the topology, as well as validating the path with the decoding process mentioned above. All individuals are distributed in a toroidal grid.

For the generation of an encoded path, a chain is created containing the total number of randomly ordered network devices. The positions of the numbers represent the genes (priority IDs) that make up a chromosome (pathway). The first gene represents the source path, and to determine the path of a chromosome is necessary to decode the sequence using the adjacency matrix of the topology. For example, in the coding process of a routing path between H1 and R1 of the network topology, as shown in Figure 4, the following procedure is defined:

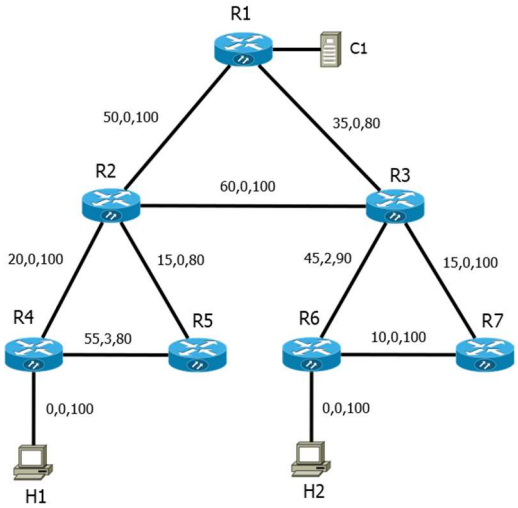

Figure 4. Example topology. Metrics: delay (ms), packet loss rate (%), and bandwidth (Mbps).

Coding:

- Generated a chain with 10 elements corresponding to network nodes.
- Filled the string with random numbers from 1 to 10.
- Created a chromosome: 5 8 2 1 4 10 6 7 3 9.

Decoding:

Table 3 shows the network adjacency matrix used to perform this procedure. The genes' value is defined from highest to lowest to determine the order of the path, as shown below:

(1) The path is initialized on H1 towards R4, as it is the only adjacent node.

(2) R4 has two adjacent nodes: R2 with an ID = 8 and R5 with an ID = 4. The selected node has the highest ID—in this case, R2.

(3) In R2, R1 is chosen because it has the highest value ID.

(4) The decoding process is completed; the path reaches the proposed destination node (R1).

(5) Path: H1, R4, R2, R1.

Table 3. Example topology adjacencies matrix for coding representation. The path source: H1 and destination: R1. The hops of H1-R1 are highlighted.

Chromosome	**5**	**8**	**2**	**1**	**4**	**10**	**6**	**7**	**3**	**9**
Nodes	R1	R2	R3	R4	R5	R6	R7	H1	H2	H3
R1	1	1	0	0	0	0	0	0	0	0
R2	1	0	1	1	0	0	0	0	0	0
R3	1	1	0	0	0	1	1	0	0	0
R4	0	1	0	0	1	0	0	1	0	0
R5	0	1	0	1	0	0	0	0	0	0
R6	0	0	1	0	0	0	1	0	1	0
R7	0	0	1	0	0	1	0	0	0	1
H1	0	0	0	1	0	0	0	0	0	0
H2	0	0	0	0	0	1	0	0	0	0
H3	0	0	0	0	0	0	1	0	0	0

In genetic algorithms, to improve the characteristics of solutions (chromosomes), crossover operators are used in each generation. In the MOCell algorithm, a single-point crossover *(SPX)* between two parents without the point from the origin and destination was used to conserve the path [40]. For this method, a cut-off point between both parents is selected, and an element of the solution vector is selected randomly to assign it as a cut-off point. The genes of both parents are exchanged between the parents' genes, providing two new descendants with information about parental genetics.

In genetic algorithms, an additional operator is used to avoid falling local optima and maintain genetic diversity from one population to another, called the mutation operator. We employ the swap *mutation* operator, which proposes the random selection of a pair of genes in the chromosome and the exchange of values between them [41]. Figure 5 shows the operators used.

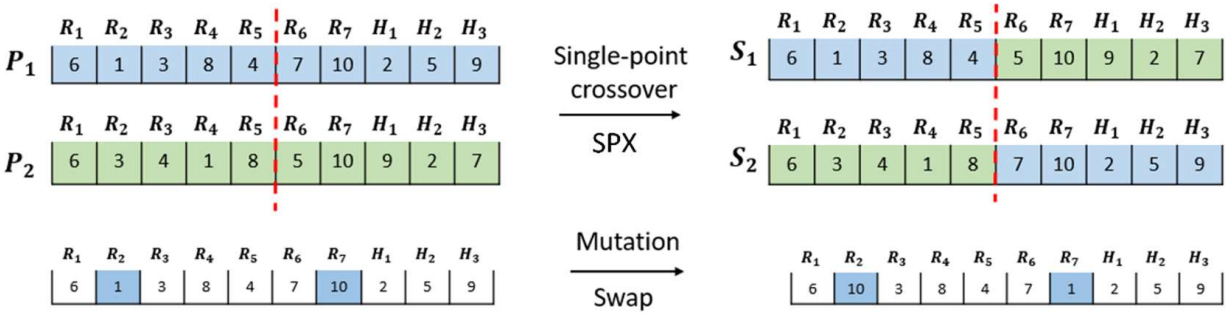

Figure 5. Genetic operators.

Fitness function representation: The optimization problem is formulated with three objective functions, f_1, f_2, and f_3, represented by QoS parameters such as delay, packet loss, and available bandwidth, respectively. The purpose of the first is to minimize the cost of delay and latency covered by a specific path. The purpose of the second is to minimize the loss of data, and that of the third is to minimize the loss of quality of service when the demand for bandwidth consumption is not satisfactory for the correct functionality of the flow.

The f_1 function (Equation (9)) consists of the summation of all delay costs for the path. The QoS parameters, such as delay, have an additive metric composition rule for the end-to-end path.

The f_2 function (Equation (10)) indicates the packet loss on the path, represented as the sum of the data losses in the links.

The f_3 function (Equation (8)) consists of the bandwidth parameter; the end-to-end path selects the minimum capacity bandwidth of the network links.

We consider a standard configuration for the algorithm parameters: a population size of 100 for domains A and B and 400 for domain C, considering twice the number of individuals concerning the number of nodes in each domain; a neighborhood size of 8; and the tournament selection method with a crossover probability pc of 0.6 and operator SPX and a mutation pm of 0.6 with operator swap.

3.6. MOEA/D QoS Optimal Path

The multi-objective evolutionary algorithm based on decomposition (MOEA/D) [42] decomposes a MOP into a set of scalar optimization subproblems using N uniformly distributed weight vectors, w_i, each of which is iteratively optimized using recombination operators (crossover and mutation). Each normalized weight vector is uniformly distributed in T regions, called neighboring subproblems, and the objective is to find a Pareto-optimal solution along each reference neighborhood. The child solutions replace the parent solutions of the subproblems if their cost, w, is better than the previous cost. An important aspect of the algorithm is the choice of the decomposition method. According to Zhang and Li [43], three common decomposition methods are implemented: weighted sum, Tchebysheff, and penalty-based boundary interaction. For simplicity, in this paper, we focus only on the weighted sum approach according to Equation (13). The problem is divided into three weight vectors using the weighted sum method, where each neighborhood is assigned a network parameter priority using weight vectors, w (Algorithm 3, introduces MOEA/D pseudocode).

Algorithm 3. Pseudocode of metaheuristics based on MOEA/D

```
data = parameters        // Initial parameter MOEA/D
pop = solution           // Create an initial population
W (t = 0)                // Initialize Weight vectors
T = w_i ϵ W(t)           //Calculate neighborhoods by the weight vectors
P_F = [] //create pareto front = empty
While (terminal Condition==false) do
    For neighborhood ϵpop do   //everyone Weight vector builds a neighborhood
        For individualϵneighborhood do
            parents = (neighborhood)    //selection parents by neighborhood
            offspring = recombination (data, parents)    //crossover
            offspring = mutation (data, offspring)
            pop = replacement (parents, offspring)
            P_F = Insert (Offspring)
        End
    End
pop = feedback (Pareto Front)
End
```

3.7. Simulation Model

The simulation model implements a network topology that considers the transfer of data flows from host to host among SDN-based distributed clouds. In our research, the experimental configuration considers an extreme network topology where a very dense cloud is considered for communication between two data centers, presenting a three-tier data center topology. This topology consists of the access layer to connect hosts, an aggregation layer to connect access switches, and a core layer at the root where the network controller is located. At the same time, the intermediate cloud presents a topology in the form of a mesh to provide a challenge for the appropriate path selection process due to the

multiple alternatives that can arise in scenarios containing networks with a large number, such as IoT networks.

Figure 6 shows the topology used in the simulation model. It is based on three domains, where two data centers are connected by a cloud, defined as Domain A, Domain B, and Domain C (cloud). Each domain has an SDN network controller. Domains A and B comprise a distributed topology with 29 nodes, while Domain C represents a topology with 196 nodes. The link values define a particular state that an ISP might have with bandwidth, delay, and packet loss metrics.

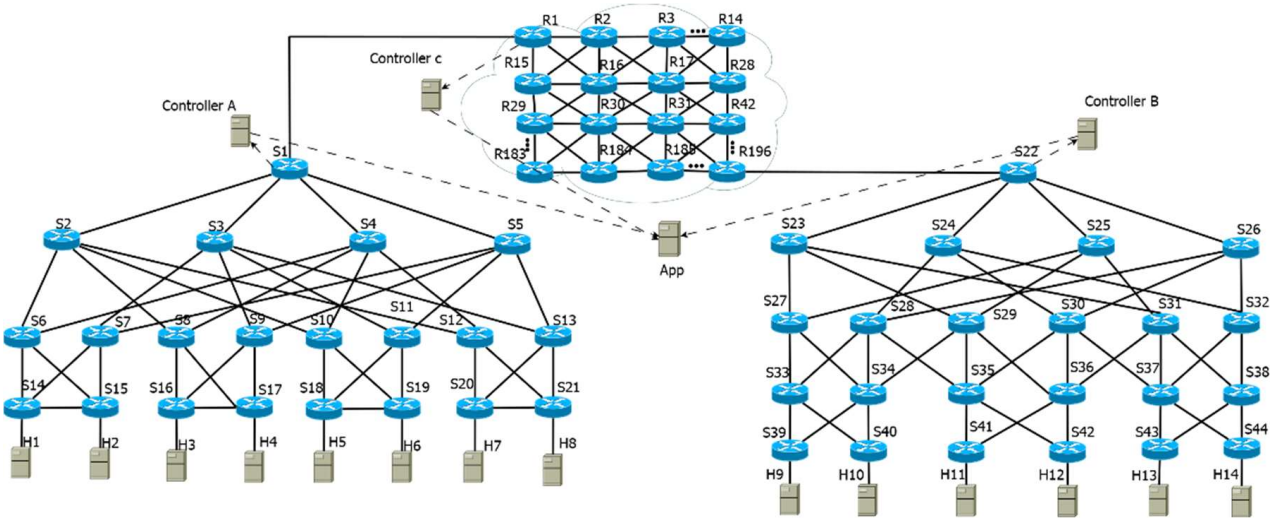

Figure 6. Experimental topology.

3.8. Experiment Setup

The path selection process and the MOCell algorithm implementation were developed in the MATLAB programming language. We used the OpenDayLight (ODL) controller with the RESTCONF module. The distributed data topology was designed using the Mininet simulator [20]. Mininet is one of the most used platforms in the literature for SDN emulation due to its compatibility and flexibility with other applications and controllers. For the simulation environment, we configured a virtual machine with a Linux Debian 9 operating system with 12 CPUs, 16 RAM, and 100 GB of disk space (NLSAS disks). Below is a list of the software specifications and tools used:

- Linux Debian 9;
- Mininet 2.3;
- OpenDayLight (ODL) SDN Controller;
- MATLAB 2021 for MOCell-based code;
- Python 3 for REST API communication scripts;
- Iperf for data stream transfers performance.

The ODL controllers for Domains A, B, and C were configured according to the simulated network topology (Figure 6) and the link parameters (bandwidth, packet loss, and delay) corresponding to the values in Appendix A. We simulated each domain along with its SDN controller, and they were configured to connect with the Mininet simulator.

The network topology was divided into three domains. Controllers A, B, and C managed the domains, respectively. Since the domains were separate, each controller had its own domain status and network information. Therefore, it was necessary to implement an application that coordinates the controllers to establish the routing process that allows communication among them.

4. Evaluation

In contrast to problems where we only try to satisfy a simple objective and where optimality is given by a single minimal solution, in the multi-objective problem, we strive to find the set of non-dominated solutions that satisfy the constraints associated with path selection according to QoS criteria and simultaneously optimize two or more features. For each domain, 30 independent runs of the genetic algorithm were performed with a breakpoint of 500 generations. Figure 7 shows the initial and final populations for each implementation of the MOCell in the three independent domains. The values are normalized to the maximum value found for the scores of each parameter and domain. A significant decrease was observed in the space of better solutions representing higher QoS parameters. As for the path selection of the non-dominated solutions, it is possible to consider a priority order of the functions according to the flow requirements. In this case, we choose the delay first, packet loss second, and bandwidth third, according to the performance of the different algorithms evaluated in Table 4 and their flows, as shown in Figures 8 and 9.

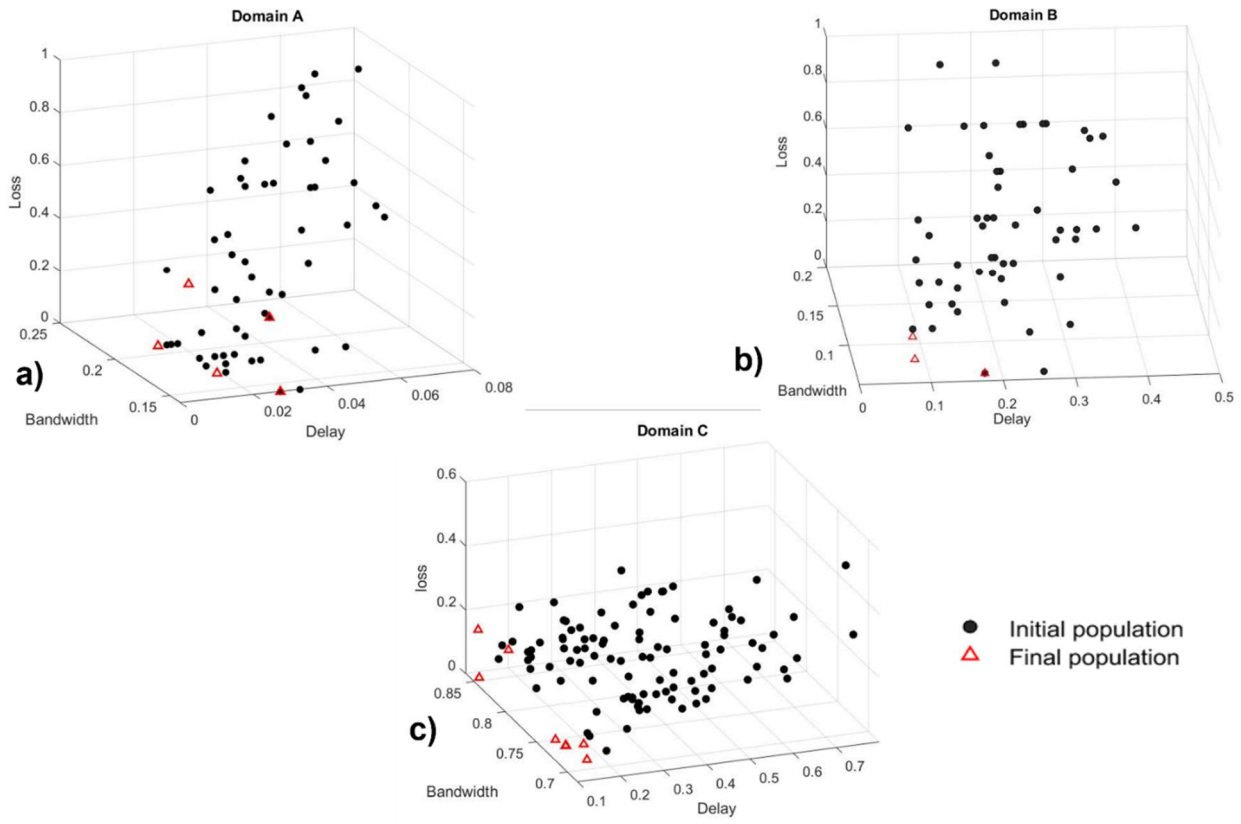

Figure 7. Set of non-dominated solutions obtained by the multi-objective cellular genetic algorithm MOCell. (**a**) Domain A, (**b**) Domain B, and (**c**) Domain C.

The experiment measured data transmission between the end-to-end hosts of data centers H1 and H14. We compared traditional data forwarding based on the shortest path algorithm and the highest available bandwidth. We also considered the path with the lowest delay, the lowest packet loss, a weighting criterion, and our MOCell routing proposal. We used the IPERF [44] application to test network performance with the connection-oriented protocol (TCP) and connectionless protocol (UDP). Our method queries the SDN controller to obtain the network's topology and compute the paths from one end to the other. Table 4 shows the network paths for hosts from source to destination according to their optimization value and all QoS parameters used in this work.

Table 4. Results of routing paths and network parameters.

Metric	Path	H (Hops)	B (Mbps)	D (ms)	PL (0–1)	Data Rate (Mbps)
H	H1-S14-S6-S2-S1-R1-R16-R31-R46-R61-R76-R91-R106-R121-R136-R151-R166-R181-R196-S22-S26-S32-S38-S44-H14	24	78	130	5.5×10^{-3}	23.8
B	H1-S14-S6-S2-S1-R1-R16-R31-R45-R46-R59-R72-R73-R86-R101-R114-R129-R142-R155-R169-R156-R171-R172-R187-R174-R161-R147-R162-R163-R164-R178-R193-R180-R181-R196-S22-S24-S32-S37-R31-R38-S44-H14	42	86	166	3.7×10^{-3}	38
D	H1-S14-S7-S3-S1-R1-R16-R31-R46-R33-R47-R62-R77-R91-R106-R120-R135-R150-R164-R179-R180-R195-R196-S22-S24-S30-S37-S44-H14	28	79	58	2.6×10^{-3}	53.5
PL	H1-S14-S7-S5-S1-R1-R2-R3-R17-R30-R43-R58-57-R72-R85-R99-R114-R129-R144-R145-R160-R174-R189-R190-R191-R192-R193-R194-R195-R196-S22-S26-S32-S37-S44-H14	35	74	138	0	63.1
$w_{B,D,PL}$	H1-S14-S7-S5-S1-R1-R2-R17-R32-R47-R48-R62-R77-R90-R104-R103-R116-R131-R146-R147-R162-R177-R192-R193-R194-R195-R196-S22-S24-S28-S35-S30-S37-S44-H14	34	79	85	3×10^{-3}	64.4
MOCell ($Minf_1, f_2, f_3$)	H1-S14-S7-S5-S1-R1-R15-R29-R43-R57-R72-R86-R85-R100-R115-R130-R117-R118-R133-R148-R149-R164-R179-R194-R195-R196-S22-S24-S28-S35-S30-S37-S44-H14	33	79	75	2×10^{-3}	66.9
MOEA/D	H1-S14-S7-S5-S1-R1-R2-R3-R18-R33-R48-R63-R98-R92-R107-R121-R134-R149-R164-R178-R193-R194-R195-R196-S22-S24-S28-S35-S30-S37-S44-H14	31	79	80	2×10^{-3}	66.37

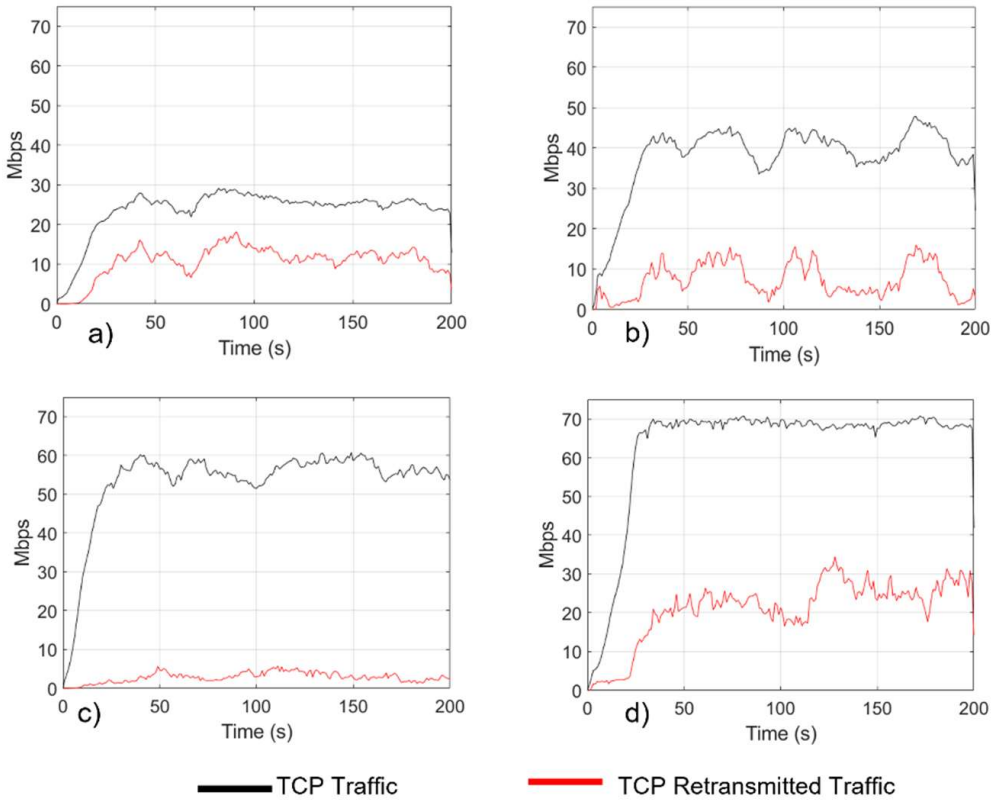

Figure 8. Evaluation of the path capacity using TCP traffic. The paths were selected according to the following methods: (**a**) H (hops), (**b**) B (bandwidth), (**c**) D (delay), and (**d**) PL (packet loss).

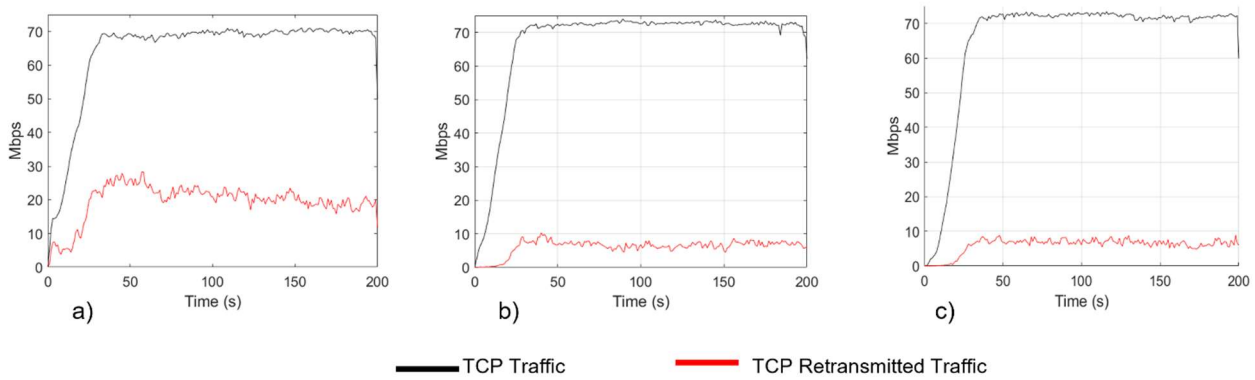

Figure 9. Evaluation of the path capacity using TCP traffic. The paths were selected according to the following multi-objective methods: (**a**) $w_{B,D,PL}$ (weighted sum), (**b**) Min_{f_1,f_2,f_3} (MOCell), and (**c**) MOEA/D.

Each test was performed under the same conditions as the experiment, where the total availability of the network was used for path selection according to the parameters in Tables A1–A3 from Appendix A.

The following steps were defined for the execution of the test:

- Design and execute the network topology with three domains (A, B, and C) and an external controller per domain.
- Execute network performance tests between the end nodes with IPERF according to the optimization parameter.
- Transfer the traffic during the specified period.
- Repeat the experiment using data stream forwarding according to the optimization parameter.
- Obtain results.

An inter-domain approach was used to evaluate the data flow performance between two hosts. IPERF was used to generate TCP and UDP traffic between hosts H1 and H14 to simulate a scenario where an application transfers data among hosts.

The test used several optimization criteria to select the path (these are described in Table 4). Paths shared by host-to-host routing methods have different network metrics. Our proposal computes and configures the path considering all available QoS parameters in the network. The algorithm can be configured depending on the requirements of a particular application, but in this case, the path minimizes the parameters considered.

Network Performance

We use the IPERF tool to evaluate the performance of different routing methods between the extreme nodes of our network topology, from H1 to H14. The values of the IPERF tool were configured with standard TCP traffic and UDP parameters and a time interval of 200 s for each data transmission. We executed the experiment 30 times.

Figures 8 and 9 show the first experiment where the traffic of the paths was evaluated with the IPERF tool and the Wireshark tool [45] based on the maximum possible data transfer rate for TCP traffic. The graphs show in black the total traffic transmitted in the experiment, and the traffic in red represents retransmitted packets that affect the network performance, such as data packets that arrive in a different order from which they were sent or the re-sending of packets that were damaged or lost during their initial transmission. We obtained the best performance with our proposed method since it minimized all of the parameters represented in Equation (16). For evaluating path selection based on MOCell, we considered the multi-objective functions in order of priority as $f_1 < f_2 < f_3$, as described in Equations (8)–(10).

This evaluated comparison between algorithms shows the capacity of the selected path for each. Next, we discuss each algorithm regarding the chosen path and how it affects the network performance results.

As seen in Table 4, the average data transfer speeds of the traditional methods are slower (minimum hops 23.8 Mbps and maximum available bandwidth 38 Mbps) than the path selected by our method (66.9 Mbps), as were the paths with the lowest delay (53.4 Mbps) and packet loss (63.1 Mbps). For the test, we used the exact Dijkstra search algorithm to show the efficiency of different paths with optimization criteria according to the parameters usually implemented in traditional routing: fewer hops and higher bandwidth. We also added the path with the minimum delay and data loss.

The path with the *H* (fewest hops, most frequent routing) had the lowest average data transfer rate because it had little effective traffic. A significant portion of its traffic had to be retransmitted because its path went through several nodes where there would have been significant losses, making it the worst path in this test.

The path with *B* (the highest available bandwidth) had the highest number of hops and a high probability of packet loss and delay because it had to go through many nodes to reach its destination.

The path with *PL* (the least packet loss) had a higher traffic transfer rate than the previous paths. However, it was the path with the most packet retransmissions due to packet delays, significantly affecting performance.

The path with $w_{B,D,PL}$ (weighted sum), considered the weight by the available metrics, achieved better performance than the traditional methods.

From the obtained results, we can see that an evolutionary algorithm (MOCell and MOEA/D) can improve the routing-related aspects by evaluating multiple available paths for establishing routing flows and evaluating their QoS parameters.

The path performance is determined by the amount of data successfully transmitted during the cycle period (Equation (5)). We can evaluate the throughput in terms of the maximum data transmission capacity in the network to normalize the throughput size of each path and calculate the area enclosed by the throughput function (Figure 10) using the Riemann sum [46]. We find that the path proposed by Min_{f_1,f_2,f_3} (MOCell) can exploit 71.08% of the network throughput on average, and MOEA/D can exploit 69.23%, while the paths selected by the other techniques had values below 55.99% (see Figure 10). The MOCell-based routing path improves by more than 54% on average compared with the traditional routing method with fewer hops *(H)* and by 35.44% for the path with the highest available bandwidth (*B*), in the case of the network topology presented in this paper. Finally, in the case of the evolutive algorithms, MOCell had a 1.85% higher throughput than the MOEA/D method when using the same parameters and metrics for mutations, generations, and the method of selecting individuals by tournament; in this way, a fair evaluation was obtained.

The path chosen by the MOCell method shows slightly better performance than the MOEA/D method, especially in the first 30 s of transmission, since it converges earlier. After this period, both methods show similar performance.

Figure 11 and Table 5 show the values of the throughput area (normalized by the maximum capacity) of 30 independent runs of the Mininet experiment for each path selected by the algorithms. This graph shows that the $w_{B,D,PL}$ (weighted sum) and *PL* (packet loss) paths have a larger standard deviation in their performance than the other paths. The selected path using the minimum delay algorithm (*D*) leads to a higher median without considering the paths of the MOCell and MOEA/D methods.

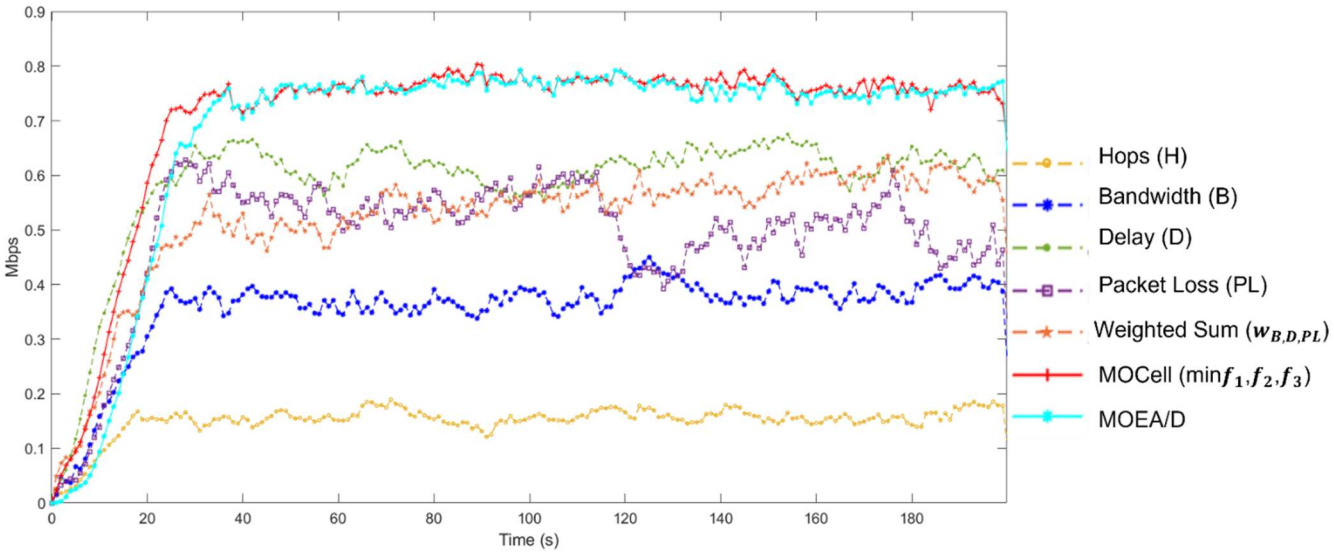

Figure 10. TCP throughput (t_h) comparison of routing paths.

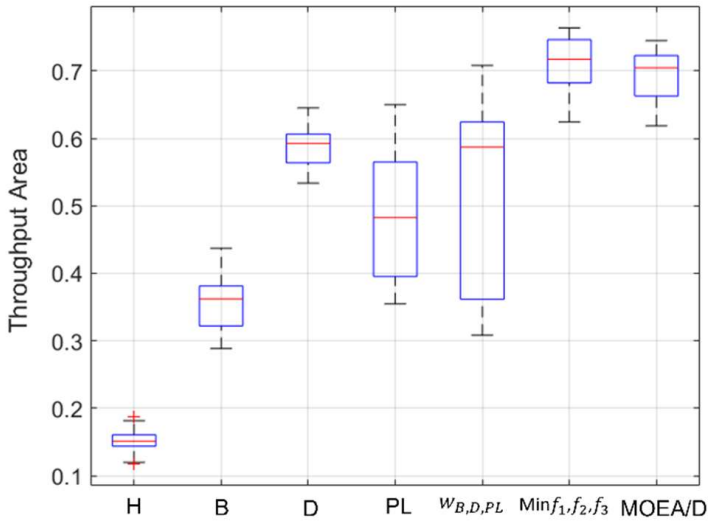

Figure 11. TCP performance of paths. Time interval 200 s. STDEV: H = 0.01847, B = 0.04224, D = 0.02953, PL = 0.09682, $w_{B,D,PL}$ = 0.13832, $Minf_1, f_2, f_3$ = 0.03777, and MOEA/D = 0.04104.

Table 5. Routing Paths Throughput Areas.

Algorithm	Median Area
H (Hops)	0.1511
B (Bandwidth)	0.3622
D (Delay)	0.5924
PL (Packet Loss)	0.4825
$w_{B,D,PL}$ (Weighted sum)	0.5872
$Minf_1,f_2,f_3$ (MOCell)	0.7174
MOEA/D	0.7046

From the results, it can be concluded that both packet loss and delay are related to not only the available bandwidth but also the low speed of the network, as is usually considered when assigning paths and QoS for the data streams. Retransmissions of packets in transit lead to poor or slow network performance, and low throughput indicates problems such as increased latency and possible congestion.

As mentioned earlier, minimizing all of the factors that affect end-to-end link performance is essential to improve the speed and efficiency of data transfer. It is critical to leverage the various parameters in the network to improve network performance and provide a better quality of service. Our proposal can improve the routing-related aspects by evaluating multiple available paths to set up routing flows and evaluate their QoS requirements.

The second experiment was performed with UDP traffic between nodes at the ends of topology H1 to H14. As mentioned above, the performance of the paths selected by the different algorithms presented is evaluated in this experiment. In this case, the data transmission consisted of a constant flow of UDP traffic with a bandwidth of 45 Mbps for 200 s and the standard configuration of the IPERF tool. The network state corresponded to the first experiment's conditions, and the initial conditions were restarted for each evaluated case in the Mininet simulator.

Table 4 shows that the capacities of the paths selected by each of the evaluated algorithms perfectly covered the required bandwidth per UDP flow; however, the performance of each path varied due to the other network parameters involved. Table 6 shows the average values for the performance of the paths, and Figure 12 presents a graph of the lost packets.

Table 6. Performance summary of UDP flow between H1 and H14.

Algorithm	Data Rate (Mbps)	Jitter (ms)	Packet Loss (%)	Throughput
H	43.83	1.081	0.025	0.9749
B	40.56	1.958	0.1067	0.8932
D	43.10	0.952	0.0410	0.9589
PL	42.58	1.71	0.0550	0.9449
$w_{B,D,PL}$	43.6	1.325	0.4519	0.9548
$Min f_1, f_2, f_3$	44.56	0.986	0.0090	0.9909
MOEA/D	44.48	0.994	0.0097	0.9902

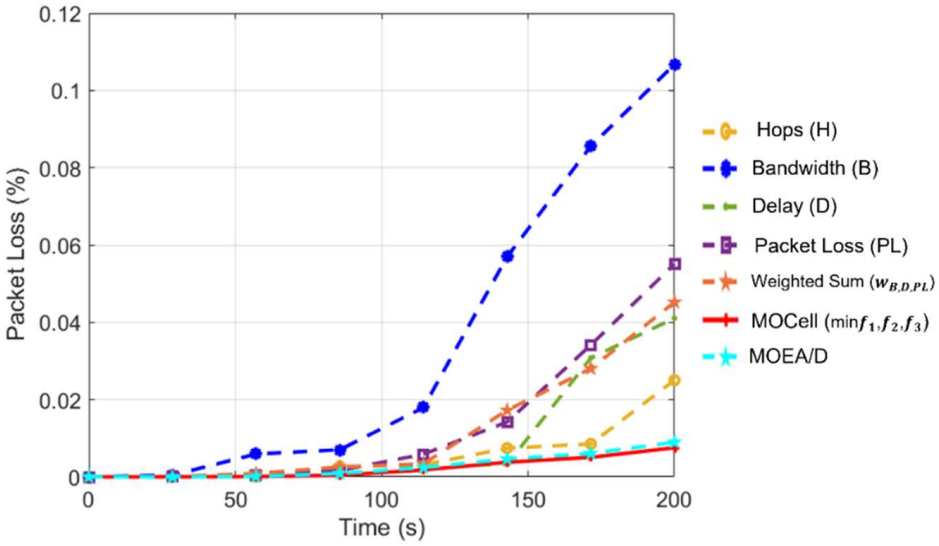

Figure 12. UDP packet loss, STDEV: H = 0.00834, B = 0.03683, D = 0.01754, PL = 0.01709, $w_{B,D,PL}$ = 0.01343, Min_{f_1,f_2,f_3} = 0.00258, and MOEA/D = 0.0021.

The most critical path is chosen by the widest path, B, although it has the highest bandwidth capacity to divert flows, followed by the path PL. However, this path consists

of the links with the lowest losses. This severe drawback is a result of this path selecting connections with high jitter values. It should be remembered that a packet that takes longer than the time specified by the waiting window is usually considered to be a lost packet, as well as packets that arrive out of order.

The weighted sum method has the same effect. This method states that even if we consider different parameters for path selection, it is necessary to properly determine the weights of the metrics to obtain a balanced path. We can solve these optimization problems using evolutive algorithms, which provide a set of non-dominated paths and consider multiple metrics. The MOCell method reaches a throughput of 99.09%, which is only 0.07% better than MOEA/D and, compared with the other algorithms, 1.6% higher than path H, 3.2% higher than path D, 3.6% higher than path $w_{B,D,PL}$, 4.6% higher than path PL, and 9.77% higher than path B. Link capacity is a necessary network parameter to guarantee QoS. However, this is not sufficient to provide adequate service in all cases. Therefore, it is advisable to never lose sight of the other parameters available in the network and to use more than one criterion in the routing process to improve the quality of service and enhance the quality of experience.

5. Conclusions and Future Work

This document provides an overview of the routing algorithms used in the SDN network controller, including their quantitative performance characteristics, such as the number of hops, available bandwidth, delay and packet loss, and their throughput, considering a large-scale multi-domain network topology. This work suggests that the proposal of a genetic algorithm in the routing process can significantly improve network performance. Minimizing the cost of network metrics and reducing the likelihood of selecting connections with few resources or poor link performance can significantly reduce data flow performance or, in the worst case scenario, cause congestion. The experiments in this work show that our proposal based on the MOCell algorithm provides optimal routing by having a global view of the network and abstracting its QoS metrics. The proposed MOCell method adapts to any routing constraint, while the standard algorithms in the controller do not consider any of these essential properties to improve the routing service.

Managing a centralized environment as in SDN can significantly enhance the routing process by improving path selection and providing better service. Evolutionary algorithms and their variants take advantage of this problem because they are scalable and can find reasonable solutions during the first run, which helps networks to meet QoS requirements and improves network performance.

In future work, we will consider optimizing the algorithm for different population sizes, as well as different crossover and mutation methods. Similarly, we will evaluate the performance of the MOCell algorithm in more detail compared with other evolutionary algorithms such as NSGA-II and MOEA/D, considering convergence time and diversity of solutions. In the evaluation of other evolutionary algorithms, both their convergence to the Pareto front and the dispersion of the set of found solutions are evaluated, while in this paper we focus exclusively on the selection of a path from the solutions found by the methods. Therefore, we consider it appropriate to use other metrics such as cardinality, accuracy, and diversity to evaluate the performance of evolutionary multi-objective algorithms.

Author Contributions: Conceptualization, J.E.G.-T., R.R.-R., A.T. and J.E.L.-R.; methodology, J.E.G.-T., A.T. and R.R.-R.; software, J.E.L.-R.; validation, R.R.-R. and A.G.-M.; formal analysis, A.T.; investigation, J.E.L.-R., A.T., R.R.-R. and J.E.L.-R.; resources, R.R.-R.; data curation, J.E.G.-T. and R.R.-R.; writing—original draft preparation, J.E.G.-T. and R.R.-R.; writing—review and editing, R.R.-R., J.E.L.-R. and S.V.-R.; visualization, J.E.L.-R.; supervision, R.R.-R., J.E.L.-R. and J.L.G.C.; project administration, R.R.-R.; funding acquisition, R.R.-R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by "CICESE Research Center" and "The Council for Science and Technology of Mexico" (CONACYT), No. CVU: 765196.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to the large size of the dataset required to maintain a public repository.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Controller A link metrics in the simulation model. Metrics: delay (ms), packet loss (%), and bandwidth (Mbps).

Link (S)	Metrics	Link (S)	Metrics	Link (S)	Metrics	Link (S)	Metrics
1,2	10,0,97	3,13	16,0.003,82	6,15	3,0,81	11,18	22,0,80
1,3	5,0,94	4,6	13,0,88	7,14	3,0,79	11,19	23,0,76
1,4	8,0,99	4,8	7,0,97	7,15	2,0,85	12,13	7,0,80
1,5	3,0,93	4,10	16,0,90	8,l9	9,0,78	12,20	21,0.003,100
2,6	18,0.002,93	4,12	18,0.001,96	8,16	18,0.002,96	12,21	12,0,77
2,8	12,0,90	5,7	17,0,84	8,17	12,0,87	13,20	6,0,80
2,10	24,0.001,98	5,9	9,0,92	9,16	7,0,76	13,21	21,0.001,79
2,12	8,0,91	5,11	21,0.002,94	9,17	14,0,83	14,15	8,0,83
3,7	11,0,86	5,13	15,0,97	10,11	11,0,80	16,17	7,0,80
3,9	21,0.004,97	6,7	8,0,82	10,18	13,0,83	18,19	6,0,80
3,11	9,0,95	6,14	23,0,86	10,19	9,0,82	20,21	9,0,79
H1-H8	0,0,100						

Table A2. Controller B link metrics in the simulation model. Metrics: delay (ms), packet loss (%), and bandwidth (Mbps).

Link (S)	Metrics	Link (S)	Metrics	Link (S)	Metrics	Link (S)	Metrics
22,23	2,0,94	25,29	8,0,93	29,35	18,0,82	33,40	7,0,92
22,24	4,0,96	25,31	13,0.001,91	29,36	12,0,87	34,39	5,0,83
22,25	2,0,98	26,28	9,0,92	30,35	5,0,94	34,40	12,0,95
22,26	16,0,99	26,30	12,0.001,89	30,36	1,0.001,86	35,41	5,0.002,84
23,27	8,0.001,91	26,32	18,0,94	30,37	3,0,97	35,42	4,0,96
23,29	15,0,93	27,33	7,0.002,93	31,36	4,0.001,93	36,41	2,0,87
23,31	12,0.001,95	27,34	12,0,87	31,37	6,0,96	36,42	7,0,93
24,28	2,0,92	28,33	11,0,88	31,38	2,0,99	37,43	6,0,92
24,l0	7,0.001,94	28,34	6,0,93	32,37	20,0,94	37,44	3,0,94
24,32	12,0,97	28,35	2,0,97	32,38	5,0.001,86	38,43	5,0,94
25,27	12,0,98	29,34	15,0.002,97	33,39	3,0.002,82	38,44	3,0.001,95
H9-H14	0,0,100						

Table A3. Controller C link metrics in the simulation model. Metrics: delay (ms), packet loss (%), and bandwidth (Mbps).

Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics
1,2	2,0,89	47,48	2,0,89	46,60	2,0,97	92,106	3,0.001,93
1,15	2,0.001,86	47,60	1,0,85	46,61	3,0,86	137,152	3,0,86
1,16	1,0,91	47,61	3,0,93	92,107	1,0,98	38,139	3,0,100
2,3	3,0,97	47,62	1,0,85	93,94	3,0,93	138,151	3,0,86
2,15	1,0,97	48,49	2,0.001,94	93,106	1,0,100	138,152	4,0.001,85
2,16	2,0.001,94	48,61	2,0,99	93,107	2,0.001,92	138,153	2,0,86
2,17	4,0.001,91	48,62	3,0,97	93,108	2,0,96	139,140	4,0.001,90
3,4	1,0.001,98	48,63	1,0,97	94,95	1,0.001,97	139,152	3,0,91
3,16	4,0,87	49,50	2,0,89	94,107	2,0,96	139,153	1,0,97
3,17	4,0,95	49,62	1,0,93	94,108	2,0.001,94	139,154	1,0,97
3,18	3,0,86	49,63	1,0.001,96	94,109	4,0,90	140,153	1,0,97
4,5	1,0.001,85	49,64	2,0,86	95,96	3,0,87	140,154	3,0,85
4,17	1,0,91	50,51	1,0,97	95,108	2,0,96	141,142	2,0.001,87
4,18	1,0,84	50,63	1,0,94	95,109	2,0,98	141,166	3,0,91
4,19	4,0.001,87	50,64	2,0.001,95	95,110	1,0,95	141,156	3,0.001,71
5,6	4,0.001,82	50,65	1,0,95	96,97	1,0.001,93	142,143	2,0.001,86
5,18	2,0,82	51,52	1,0,98	96,109	4,0,85	142,155	3,0,98
5,19	4,0,90	51,64	3,0,100	96,110	4,0.001,93	142,156	2,0,91
5,20	3,0.001,94	51,65	3,0.001,91	96,111	2,0,88	142,157	2,0,91
6,7	2,0,99	51,66	1,0,86	97,98	4,0,95	143,144	2,0,89
6,19	4,0,95	52,53	4,0.001,85	97,110	3,0,87	143,156	3,0,95
6,20	2,0,94	52,65	3,0,96	97,111	2,0,88	143,157	4,0.001,96
6,21	2,0,83	52,66	1,0.001,90	97,112	3,0,100	143,158	4,0,90
7,8	3,0,96	52,67	4,0,85	98,111	1,0,98	144,145	4,0,94
7.20	1,0,86	53,54	4,0,96	98,112	1,0,91	144,157	3,0,89
7,21	4,0,95	53,66	2,0,96	99,100	2,0.001,83	144,158	4,0,85
7,22	2,0,88	53,67	2,0.001,96	99,113	1,0,99	144,159	3,0,93
8,9	2,0.001,89	53,68	1,0,89	99,114	3,0,76	145,146	4,0.001,96
8,21	2,0,96	54,55	4,0.001,85	100,101	3,0,96	145,158	1,0,96
8,22	1,0,84	54,67	1,0,93	100,113	2,0,87	145,159	3,0,98
8,23	3,0,84	54,68	4,0,95	100,114	3,0,100	145,160	4,0,89
9,10	1,0,83	54,69	3,0,85	100,115	3,0,85	146,147	1,0.001,96
9,22	3,0,87	55,56	3,0.001,100	101,102	2,0.001,86	146,159	1,0,95
9,23	4,0,95	55,68	2,0,92	101,114	1,0,99	146,160	1,0,100
9,24	4,0.001,85	55,69	4,0.001,86	101,115	2,0,94	146,161	3,0,92
10,11	3,0.001,88	55,70	1,0,91	101,116	1,0,92	147,148	3,0.001,96

Table A3. *Cont.*

Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics
10,23	1,0,93	56,69	3,0.001,87	102,103	1,0.001,90	147,160	3,0,85
10,24	4,0,80	56,70	3,0,85	102,115	1,0,89	147,161	1,0,100
10,25	1,0.001,80	57,58	3,0,88	102,116	3,0,92	147,162	3,0,100
11,12	4,0.001,88	57,71	2,0,78	102,117	3,0,100	148,149	3,0,94
11,24	3,0,88	57,72	1,0,80	103,104	3,0,91	148,161	3,0,97
11,25	2,0,86	58,59	3,0,94	103,116	3,0.001,97	148,162	2,0,92
11,26	1,0.001,90	58,71	2,0,87	103,117	3,0,90	148,163	3,0,91
12,13	2,0,89	58,72	3,0,88	103,118	2,0,96	149,150	1,0.001,96
12,25	3,0,81	58,73	4,0,97	104,105	3,0.001,87	149,162	2,0,87
12,26	2,0,96	59,60	1,0.001,87	104,117	1,0,92	149,163	1,0.001,90
12,27	3,0.001,95	59,72	3,0,96	104,118	4,0,90	149,164	1,0,97
13,14	4,0.001,91	59,73	1,0.001,95	104,119	2,0,91	150,151	1,0.001,100
13,26	1,0,81	59,74	1,0,90	105,106	4,0,89	150,163	2,0,95
13,27	1,0,82	60,61	2,0,86	105,118	2,0,97	150,164	1,0,87
13,28	1,0,98	60,73	4,0,93	105,119	3,0,100	150,165	2,0,89
14,27	3,0,86	60,74	3,0.001,93	105,120	2,0,91	151,152	1,0.001,91
14,28	2,0.001,88	60,75	1,0,85	106,107	1,0.001,94	151,164	1,0,91
15,16	2,0.001,86	61,62	4,0,96	106,119	2,0,98	151,165	4,0.001,99
15,29	3,0,89	61,74	1,0,92	106,120	1,0,94	151,166	4,0,95
15,30	2,0.001,72	61,75	2,0,88	106,121	3,0,92	152,153	1,0.001,92
16,17	4,0.001,93	61,76	3,0,91	107,108	4,0,95	152,165	4,0,95
16,29	2,0,86	62,63	1,0.001,97	107,120	2,0,90	152,166	3,0,93
16,30	4,0,92	62,75	1,0,91	107,121	4,0,100	152,167	4,0,100
16,31	3,0,93	62,76	2,0,92	107,122	3,0,94	153,154	1,0.001,92
17,18	3,0.001,91	62,77	1,0,89	108,109	4,0,99	153,166	2,0,93
17,30	4,0,91	63,64	3,0,98	108,121	1,0,95	153,167	3,0.001,94
17,31	4,0,92	63,76	4,0,97	108,122	2,0.001,85	153,168	2,0,93
17,32	2,0,94	63,77	3,0,100	108,123	2,0,90	154,167	4,0,98
18,19	2,0,95	63,78	4,0,97	109,110	1,0.001,100	154,168	4,0,100
18,31	4,0,86	64,65	2,0.001,86	109,122	1,0,100	155,156	2,0,72
18,32	3,0.001,90	64,77	3,0,99	109,123	3,0,90	155,169	1,0,98
18,33	4,0,90	64,78	4,0,87	109,124	4,0,85	155,170	1,0,70
19,20	1,0,98	64,79	3,0,91	110,111	2,0.001,98	156,157	2,0.001,86
19,32	3,0,96	65,66	3,0.001,93	110,123	2,0,96	156,169	3,0,98
19,33	4,0.001,89	65,78	4,0,90	110,124	4,0,95	156,170	2,0,95
19,34	2,0,100	65,79	3,0.001,97	110,125	3,0,91	156,171	3,0,100

Table A3. *Cont.*

Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics
20,21	2,0.001,86	65,80	4,0,92	111,112	1,0,91	157,158	4,0,94
20,33	2,0,97	66,67	3,0.001,87	111,124	4,0,94	157,170	2,0,88
20,34	4,0.001,98	66,79	1,0,99	111,125	3,0,89	157,171	3,0.001,94
20,35	4,0,93	66,80	4,0,88	111,126	2,0,89	157,172	4,0,91
21,22	4,0.001,97	66,81	1,0,100	112,125	4,0.001,88	158,159	4,0.001,99
21,34	4,0,93	67,68	2,0.001,92	112,126	3,0,88	158,171	2,0,95
21,35	3,0.001,95	67,80	2,0,86	113,114	1,0.001,81	158,172	2,0,86
21,36	3,0,97	67,81	3,0,89	113,127	1,0,81	158,173	2,0,94
22,23	1,0,90	67,82	4,0,93	113,128	3,0.001,87	159,160	4,0,88
22,35	4,0,92	68,69	4,0.001,91	114,115	3,0,95	159,172	2,0,99
22,36	3,0.001,100	68,81	4,0,87	114,127	4,0,86	159,173	1,0,94
22,37	4,0,100,87	68,82	3,0.001,100	114,128	3,0.001,98	159,174	2,0,91
23,24	4,0,86	68,83	4,0,99	114,129	1,0,98	160,161	2,0.001,98
23,36	3,0,99	69,70	2,0.001,88	115,116	3,0.001,92	160,173	2,0,93
23,37	3,0,89	69,82	2,0,100	115,128	4,0,92	160,174	4,0,95
23,38	2,0,100	69,83	2,0.001,92	115,129	2,0.001,85	160,175	4,0,92
24,25	3,0,100	69,84	2,0,94	115,130	1,0,95	161,162	3,0,95
24,37	1,0,95	70,85	1,0.001,94	116,117	4,0,92	161,174	1,0,96
24,38	3,0.001,98	70,84	2,0,96	116,129	2,0,94	161,175	4,0,88
24,39	2,0,86	71,72	2,0,70	116,130	3,0,91	161,176	3,0,86
25,26	3,0,98	71,85	2,0,74	116,131	1,0,95	162,163	4,0.001,99
25,38	2,0,95	71,86	3,0.001,95	117,118	1,0,89	162,175	3,0,94
25,39	4,0.001,96	72,73	3,0,98	117,130	1,0,92	162,176	3,0.001,85
25,40	4,0,91	72,85	1,0,96	117,131	3,0,90	162,177	1,0,94
26,27	4,0.001,100	72,86	2,0,93	117,132	3,0,90	163,164	3,0,98
26,37	3,0,90	72,87	3,0,94	118,119	4,0,87	163,176	4,0,88
26,40	4,0,90	73,74	2,0.001,94	118,131	2,0,94	163,177	2,0.001,97
26,4	4,0,91	73,86	1,0,98	118,132	3,0,96	163,178	3,0,92
27,28	3,0.001,97	73,87	2,0,86	118,133	1,0,94	164,165	3,0.001,97
27,20	3,0,96	73,88	3,0,92	119,120	2,0.001,90	164,177	1,0,91
27,41	4,0.001,99	74,75	1,0,100	119,132	2,0,86	164,178	1,0.001,99
27,42	2,0,91	74,87	3,0,86	119,133	4,0.001,87	164,179	1,0,95
28,41	4,0.001,97	74,88	4,0.001,88	119,134	1,0,88	165,166	1,0,90
28,42	1,0,96	74,89	4,0,97	120,121	4,0.001,91	165,178	1,0,91
29,30	3,0,96	75,76	3,0,94	120,133	3,0,93	165,179	4,0,97
29,43	1,0,84	75,88	4,0,95	120,134	4,0,91	165,180	3,0,87
29,44	2,0.001,81	75,89	2,0,91	120,135	1,0.001,89	166,167	1,0,89

Table A3. *Cont.*

Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics
30,31	3,0,90	75,90	3,0,90	121,122	3,0,89	166,179	1,0,85
30,43	2,0,88	76,77	1,0.001,91	121,134	1,0,87	166,180	2,0,87
30,44	4,0.001,99	76,89	3,0,98	121,135	4,0,88	166,181	3,0,91
30,45	1,0,85	76,90	4,0.001,92	121,136	4,0,100	167,168	3,0,88
31,32	2,0,91	76,91	4,0,94	122,123	3,0,86	167,180	4,0,95
31,44	2,0,95	77,78	4,0,97	122,135	2,0,88	167,181	4,0.001,85
31,45	1,0,95	77,90	2,0,92	122,136	3,0,91	167,182	4,0,94
31,46	1,0,89	77,91	1,0.001,85	122,137	1,0.001,91	168,181	3,0.001,88
32,33	4,0,85	77,92	4,0,99	123,124	2,0,98	168,182	3,0,93
32,45	1,0,88	78,79	3,0.001,96	123,136	4,0,92	169,170	1,0.001,87
32,46	1,0,94	78,91	2,0,89	123,137	3,0.001,99	169,183	2,0,99
32,47	2,0,97	78,92	1,0.001,98	132,138	3,0,88	169,184	2,0,70
33,34	3,0.001,89	78,93	1,0,99	124,125	2,0.001,98	170,171	1,0.001,94
33,46	1,0,93	79,80	3,0,97	124,137	3,0,100	170,183	4,0,96
33,47	1,0.001,98	79,92	1,0,89	124,138	4,0.001,85	170,184	3,0,100
33,48	3,0,89	79,93	3,0,88	124,139	1,0,100	170,185	1,0,86
34,35	2,0.001,89	79,94	2,0,85	125,126	1,0,97	171,172	4,0,97
34,47	4,0,90	80,81	4,0,94	125,138	2,0,100	171,184	1,0,93
34,48	4,0,86	80,93	4,0,100	125,139	4,0.001,94	171,185	1,0.001,99
34,49	1,0,97	80,94	1,0.001,85	125,140	4,89,89	171,186	1,0,96
35,36	4,0,86	80,95	4,0,98	126,139	1,94,84	172,173	1,0.001,94
35,48	1,0,89	81,82	4,0,85	126,140	1,95,85	172,185	3,0.001,91
35,49	3,0.001,95	81,94	2,0,93	127,128	3,0.001,91	172,186	1,0,97
35,50	1,0,85	81,95	4,0.001,100	127,141	1,0,76	172,187	1,0.001,100
36,37	3,0,88	81,96	2,0,93	127,142	2,0,85	173,174	2,0,92
36,49	1,0,99	82,83	2,0,99	128,129	3,0.001,91	173,186	3,0.001,99
36,50	4,0,96	82,95	2,0,93	128,141	4,0,96	173,187	2,0,89
36,51	4,0,88	82,96	2,0.001,97	128,142	2,0.001,96	173,188	1,0,98
37,38	2,0.001,100	82,97	1,0,86	128,143	3,0,98	174,175	2,0.001,99
37,50	4,0,90	83,84	4,0,99	129,130	3,0.001,89	174,187	2,0.001,99
37,51	1,0,90	83,96	1,0,85	129,142	4,0,99	174,188	4,0,90
37,52	2,0,92	83,97	3,0,91	129,143	3,0.001,98	174,189	4,0,97
38,39	2,0.001,89	83,98	3,0,97	129,144	4,0,85	175,176	4,0.001,97
28,51	2,0,97	84,97	1,0.001,88	130,131,	2,0.001,86	175,188	4,0,95
38,52	4,0,90	84,98	4,0,85	130,143	3,0,91	175,189	4,0,92
38,53	1,0,97	85,86	1,0,90	130,144	3,0.001,91	175,190	2,0.001,95
39,40	3,0,95	85,99	1,0,95	130,145	1,0,90	176,177	1,0.001,85

Table A3. *Cont.*

Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics	Link (R)	Metrics
39,52	1,0,94	85,100	2,0.001,91	131,132	4,0.001,95	176,189	1,0.001,87
39,53	4,0.001,90	86,87	2,0,92	131,144	1,0,88	176,190	1,0,86
39,54	1,0,87	86,99	2,0,93	131,145	4,0.001,94	176,191	2,0,85
40,41	3,0.001,100	86,100	3,0.001,98	131,146	2,0,100	177,178	4,0,90
40,53	2,0,98	86,101	4,0,98	132,133	3,0.001,99	177,190	1,0.001,100
40,54	1,0,89	87,88	2,0.001,94	132,145	2,0,86	177,191	3,0,93
40,55	4,0,87	87,100	1,0,93	132,146	2,0,96	177,192	1,0.001,94
41,42	4,0,91	87,101	4,0.001,93	132,147	1,0,84	178,179	3,0.001,94
41,54	4,0,92	87,102	1,0,87	133,134	3,0,98	178,191	1,0.001,85
41,55	4,0.001,89	88,89	1,0,97	133,146	4,0,88	178,192	4,0,100
41,56	1,0,98	88,101	4,0,92	133147	1,0,91	178,193	2,0,97
42,55	1,0,96	88,102	1,0.001,100	133,148	2,0,85	178,180	2,0,91
42,56	1,0,98	88,103	4,0,88	134,135	3,0,89	179,192	2,0.001,86
43,44	3,0,87	89,90	4,0,96	134,147	3,0,93	179,193	1,0,98
43,57	1,0,86	89,102	1,0,86	134,148	1,0.001,95	179,194	1,0.001,86
43,58	1,0,74	89,103	2,0,91	134,149	3,0,88	180,181	1,0.001,97
44,45	3,0.001,91	89,104	2,0,99	135,136	2,0,86	180,193	1,0.001,98
44,57	2,0,86	90,91	2,0,89	135,148	1,0,91	180,194	3,0,87
44,58	4,0,93	90,103	4,0,99	135,149	2,0,87	180,195	1,0.001,96
44,59	3,0,93	90,104	2,0,94	135,150	2,0,98	181,182	2,0,86
45,46	4,0.001,100	90,105	3,0,90	136,137	4,0.001,91	181,194	2,0.001,94
45,58	1,0,88	91,92	1,0,96	136,149	1,0,88	181,195	3,0,85
45,59	2,0,82	91,104	2,0,95	136,150	3,0.001,98	181,196	1,0,99
45,60	2,0,98	91,105	2,0,95	136,151	3,0,99	182,195	1,0.001,96
46,47	4,0,87	91,106	2,0,94	137,138	4,0,82	182,196	1,0,85
46,59	4,0,100	92,93	1,0.001,89	92,105	4,0,91	195,196	1,0,92

References

1. Xu, Y.; Gui, G.; Gacanin, H.; Adachi, F. A Survey on Resource Allocation for 5G Heterogeneous Networks: Current Research, Future Trends, and Challenges. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 668–695. [CrossRef]
2. García-Tadeo, D.A.; Peram, D.R.; Kumar, K.S.; Vives, L.; Sharma, T.; Manoharan, G. Comparing the impact of Internet of Things and cloud computing on organisational behavior: A survey. *Mater. Today Proc.* **2022**, *51*, 2281–2285. [CrossRef]
3. Laghari, A.A.; Wu, K.; Laghari, R.A.; Ali, M.; Khan, A.A. A Review and State of Art of Internet of Things (IoT). *Arch. Comput. Methods Eng.* **2022**, *29*, 1–19. [CrossRef]
4. Hajjaji, Y.; Boulila, W.; Farah, I.R.; Romdhani, I.; Hussain, A. Big data and IoT-based applications in smart environments: A systematic review. *Comput. Sci. Rev.* **2021**, *39*, 100318. [CrossRef]
5. Pavithra, H.; Srinivasan, G.N.; Swarnalatha, K.S. A Survey on Role of SDN in Implementing QoS in Routing in the Network. *Lect. Notes Electr. Eng.* **2022**, *789*, 361–366. [CrossRef]
6. Manguri, K.H.; Omer, S.M. SDN for IoT Environment: A Survey and Research Challenges. *ITM Web Conf.* **2022**, *42*, 01005. [CrossRef]
7. Amin, R.; Rojas, E.; Aqdus, A.; Ramzan, S.; Casillas-Perez, D.; Arco, J.M. A Survey on Machine Learning Techniques for Routing Optimization in SDN. *IEEE Access* **2021**, *9*, 104582–104611. [CrossRef]
8. Alsaeedi, M.; Mohamad, M.M.; Al-Roubaiey, A.A. Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey. *IEEE Access* **2019**, *7*, 107346–107379. [CrossRef]
9. Rudra, B.; Thanmayee, S. Architecture and Deployment Models-SDN Protocols, APIs, and Layers, Applications and Implementations. In *Software Defined Internet of Everything*; Springer: Cham, Switzerland, 2022; pp. 63–77. [CrossRef]

10. Akin, E.; Korkmaz, T. Comparison of Routing Algorithms with Static and Dynamic Link Cost in Software Defined Networking (SDN). *IEEE Access* **2019**, *7*, 148629–148644. [CrossRef]
11. Yang, S.; Tan, C.; Madsen, D.Ø.; Xiang, H.; Li, Y.; Khan, I.; Choi, B.J. Comparative Analysis of Routing Schemes Based on Machine Learning. *Mob. Inf. Syst.* **2022**, *2022*, 1–18. [CrossRef]
12. Khadir, K.; Guermouche, N.; Guittoum, A.; Monteil, T. A Genetic Algorithm-Based Approach for Fluctuating QoS Aware Selection of IoT Services. *IEEE Access* **2022**, *10*, 17946–17965. [CrossRef]
13. Cui, L.S.; Srivastava, G. QoS Routing Algorithm for OBS Networks Based on a Multi-Objective Genetic Algorithm. *IEEE Access* **2022**, *10*, 12047–12056. [CrossRef]
14. Unger, T. Genetic Algorithms: A Survey of some Mathematical Models-Part I. *Ir. Math. Soc. Bull.* **2021**, *0041*, 57–71. [CrossRef]
15. Zhang, D.; Shou, Y.; Xu, J. A mapreduce-based approach for shortest path problem in road networks. *J. Ambient. Intell. Humaniz. Computing* **2018**, *41*, 1–9. [CrossRef]
16. Pradhan, D.; Wang, S.; Ali, S.; Yue, T.; Liaaen, M. CBGA-ES+: A Cluster-Based Genetic Algorithm with Non-Dominated Elitist Selection for Supporting Multi-Objective Test Optimization. *IEEE Trans. Softw. Eng.* **2021**, *47*, 86–107. [CrossRef]
17. Egilmez, H.E.; Dane, S.T.; Bagci, K.T.; Tekalp, A.M. OpenQoS: An OpenFlow Controller Design for Multimedia Delivery with End-to-End Quality of Service over Software-Defined Networks. In Proceedings of the 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference, Hollywood, CA, USA, 3–6 December 2012; pp. 1–8.
18. Tajiki, M.M.; Akbari, B.; Shojafar, M.; Ghasemi, S.H.; Barazandeh, M.L.; Mokari, N.; Chiaraviglio, L.; Zink, M. CECT: Computationally efficient congestion-avoidance and traffic engineering in software-defined cloud data centers. *Clust. Comput.* **2018**, *21*, 1881–1897. [CrossRef]
19. Son, J.; Buyya, R. Priority-Aware VM Allocation and Network Bandwidth Provisioning in Software-Defined Networking (SDN)-Enabled Clouds. *IEEE Trans. Sustain. Comput.* **2019**, *4*, 17–28. [CrossRef]
20. Shah, S.A.R.; Noh, S.Y. A dynamic programmable network for large-scale scientific data transfer using AmoebaNet. *Appl. Sci.* **2019**, *9*, 4541. [CrossRef]
21. Bastam, M.; Sabaei, M.; Yousefpour, R. A scalable traffic engineering technique in an SDN-based data center network. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3268. [CrossRef]
22. Ejaz, S.; Iqbal, Z.; Shah, P.A.; Bukhari, B.H.; Ali, A.; Aadil, F. Traffic Load Balancing Using Software Defined Networking (SDN) Controller as Virtualized Network Function. *IEEE Access* **2019**, *7*, 46646–46658. [CrossRef]
23. Joshi, K.D.; Kataoka, K. PRIME-Q: Privacy Aware End-To-End QoS Framework in Multi-Domain SDN. In Proceedings of the 2019 IEEE Conference on Network Softwarization (NetSoft), Paris, France, 24–28 June 2019; pp. 169–177. [CrossRef]
24. Lozano-Rizk, J.E.; Nieto-Hipolito, J.I.; Rivera-Rodriguez, R.; Cosio-Leon, M.A.; Vazquez-Briseño, M.; Chimal-Eguia, J.C. QOSCOMM: A data flow allocation strategy among sdn-based data centers for iot big data analytics. *Appl. Sci.* **2020**, *10*, 7586. [CrossRef]
25. Li, D.; Wang, X.; Jin, Y.; Liu, H. Research on QoS routing method based on NSGAII in SDN. *J. Phys. Conf. Ser.* **2020**, *1656*, 012027. [CrossRef]
26. Elbasheer, M.O.; Aldegheishem, A.; Alrajeh, N.; Lloret, J. Video Streaming Adaptive QoS Routing with Resource Reservation (VQoSRR) Model for SDN Networks. *Electronics* **2022**, *11*, 1252. [CrossRef]
27. Lee, G.; Lee, C.; Roh, B. QoS Support Path Selection for Inter-Domain Flows Using Effective Delay and Directed Acyclic Graph in Multi-Domain SDN. *Electronics* **2022**, *11*, 2245. [CrossRef]
28. Li, C.; Liu, Y.; Xiao, J.; Zhou, J. MCEAACO-QSRP: A Novel QoS-Secure Routing Protocol for Industrial Internet of Things. *IEEE Internet Things J.* **2022**, *9*, 18760–18777. [CrossRef]
29. Guck, J.W.; van Bemten, A.; Reisslein, M.; Kellerer, W. Unicast QoS Routing Algorithms for SDN: A Comprehensive Survey and Performance Evaluation. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 388–415. [CrossRef]
30. Amalarethinam, D.I.G.; Mercy, P. An analysis on quality of service (Qos) based routing in internet of things (IOT). *Int. J. Adv. Sci. Technol.* **2020**, *29*, 139–163.
31. Naing, M.T.; Khaing, T.T.; Maw, A.H. Evaluation of TCP and UDP Traffic over Software-Defined Networking. In Proceedings of the 2019 International Conference on Advanced Information Technologies (ICAIT), Yangon, Myanmar, 6–7 November 2019. [CrossRef]
32. Ray, P.P. A survey on cognitive packet networks: Taxonomy, state-of-the-art, recurrent neural networks, and QoS metrics. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 5663–5683. [CrossRef]
33. Medhi, D.; Ramasamy, K. Network Flow Models. In *Network Routing*, 2nd ed.; Elsevier: Amsterdam, The Netherlands, 2018; pp. 114–157. [CrossRef]
34. Peña, D.; Tchernykh, A.; Nesmachnow, S.; Massobrio, R.; Feoktistov, A.; Bychkov, I.; Radchenko, G.; Drozdov, A.Y.; Garichev, S.N. Operating cost and quality of service optimization for multi-vehicle-type timetabling for urban bus systems. *J. Parallel Distrib. Comput.* **2019**, *133*, 272–285. [CrossRef]
35. Li, H.; Zhang, L. An efficient solution strategy for bilevel multiobjective optimization problems using multiobjective evolutionary algorithm. *Soft Comput.* **2021**, *25*, 8241–8261. [CrossRef]
36. Hua, Y.; Liu, Q.; Hao, K.; Jin, Y. A Survey of Evolutionary Algorithms for Multi-Objective Optimization Problems with Irregular Pareto Fronts. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 303–318. [CrossRef]

37. Nebro, A.J.; Durillo, J.J.; Luna, F.; Dorronsoro, B.; Alba, E. MOCell: A cellular genetic algorithm for multiobjective optimization. *Int. J. Intell. Syst.* **2009**, *24*, 726–746. [CrossRef]
38. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
39. Tian, Y.; Si, L.; Zhang, X.; Cheng, R.; He, C.; Tan, K.C.; Jin, Y. Evolutionary Large-Scale Multi-Objective Optimization: A Survey. *ACM Comput. Surv.* **2022**, *54*, 1–34. [CrossRef]
40. AEiben, E.; Smith, J.E. *Natural Computing Series Introduction to Evolutionary Computing*; Springer: Berlin/Heidelberg, Germany, 2015.
41. Harrison, K.R.; Garanovich, I.L.; Weir, T.; Boswell, S.G.; Elsayed, S.M.; Sarker, R.A. Evolutionary and Memetic Computing for Project Portfolio Selection and Scheduling: An Introduction. In *Adaptation, Learning, and Optimization*; Springer: Cham, Switzerland, 2022; Volume 26. [CrossRef]
42. Lavinas, Y.; Aranha, C.; Ochoa, G. Search Trajectories Networks of Multiobjective Evolutionary Algorithms. In *International Conference on the Applications of Evolutionary Computation, Madrid, Spain, 20–22 April 2022*; Springer: Cham, Switzerland, 2022; pp. 223–238. [CrossRef]
43. Zhang, Q.; Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]
44. Kumar, V.; Jangir, S.; Patanvariya, D.G. Traffic Load Balancing in SDN Using Round-Robin and Dijkstra Based Methodology. In Proceedings of the 2022 International Conference for Advancement in Technology (ICONAT), Goa, India, 21–22 January 2022. [CrossRef]
45. Tanner, N.H. Wireshark. In *Cybersecurity Blue Team Toolkit*; John Wiley & Sons: New York, NY, USA, 2019. [CrossRef]
46. Ergene, Ö.; Özdemir, A.Ş. Understanding the definite integral with the help of Riemann sums. *Particip. Educ. Res.* **2021**, *9*, 445–465. [CrossRef]