electronics

MDPI

Article

Packet Loss Optimization in Router Forwarding Tasks Based on the Particle Swarm Algorithm

Rana Fareed Ghani [1,*] and Laith Al-Jobouri [2]

[1] Department of Computer Science, University of Technology-Iraq, Baghdad 10066, Iraq
[2] School of Design and Informatics, Abertay University, Dundee DD1 1HG, UK
* Correspondence: rana.f.ghani@uotechnology.edu.iq

Abstract: Software-defined networks (SDNs) are computer networks where parameters and devices are configured by software. Recently, artificial intelligence aspects have been used for SDN programs for various applications, including packet classification and forwarding according to the quality of service (QoS) requirements. The main problem is that when packets from different applications pass through computer networks, they have different QoS criteria. To meet the requirements of packets, routers classify these packets, add them to multiple weighting queue systems, and forward them according to their priorities. Multiple queue systems in routers usually use a class-based weighted round-robin (CBWRR) scheduling algorithm with pre-configured fixed weights for each priority queue. The problem is that the intensity of traffic in general and of each packet class occasionally changes. Therefore, in this work, we suggest using the particle swarm optimization algorithm to find the optimal weights for the weighted fair round-robin algorithm (WFRR) by considering the variable densities of the traffic. This work presents a framework to simulate router operations by determining the weights and schedule packets and forwarding them. The proposed algorithm to optimize the weights is compared with the conventional WFRR algorithm, and the results show that the particle swarm optimization for the weighted round-robin algorithm is more efficient than WFRR, especially in high-intensity traffic. Moreover, the average packet-loss ratio does not exceed 7%, and the proposed algorithms are better than the conventional CBWRR algorithm and the related work results.

Keywords: computer network; software defined network; artificial intelligence; PSO algorithm

Citation: Ghani, R.F.; Al-Jobouri, L. Packet Loss Optimization in Router Forwarding Tasks Based on the Particle Swarm Algorithm. *Electronics* **2023**, *12*, 462. https://doi.org/10.3390/electronics12020462

Academic Editor: Juan-Carlos Cano

Received: 5 November 2022
Revised: 6 January 2023
Accepted: 11 January 2023
Published: 16 January 2023

1. Introduction

Communication networks consist of large sets of hardware devices with specific control logic or algorithms used to manually configure devices and operate different network functions. They comprise distributed control plane architectures that are difficult to integrate. However, the rapid increase in data traffic and the use of data centers and virtual machines require the automatic integration and configuration of network components. Therefore, the concept of a software-defined network (SDN) is proposed. SDNs use dedicated hardware devices to control network traffic using software-based controllers or application programming interfaces (APIs) to communicate with underlying hardware infrastructure and direct traffic on a network. The idea behind SDNs is to use a central controller for the control plane [1,2]. The SDN control plane can integrate a diverse set of devices and tune them at the run-time through programmable APIs, which are called southbound APIs. SDNs can also integrate information with applications through northbound APIs. Therefore, an SDN looks like a single logical network device [3]. These software functions increase control with greater speed and flexibility, allow network administrators to optimize the flow of data through the network, prioritize applications that require more availability, and increase security [4,5]. Adding artificial intelligence (AI) concepts to SDN functions shifts the computer network to an AI-based SDN. Current research efforts focus

on improving the task of the control plane in SDNs using the following AI sub fields: machine learning, optimization algorithms, and fuzzy inference systems. AI subfields are used to improve various functions in SDNs, such as network applications [6], load balancing [7], network security [8,9], routing [10,11], and wireless sensor networks [12].

The routers' tasks were also considered in AI-based SDN research efforts. One of these tasks involves managing packet forwarding. Routers use queues to keep the packets when the outgoing interface is not available [13]. Keeping packets in the queue has two main advantages. The first one is disciplining the flow of packets between networks which varies in their speed. The second advantage is to schedule the packets according to their priority. The queuing system within routers mostly uses multiple queues and schedulers. The scheduler is the key part of the queuing system because it prioritizes one queue over another according to one or more priority criteria. Cisco router devices use a variant of the round-robin scheduling algorithm, which is called a class-based weighted round-robin algorithm (CBWRR), as a schedular. In CBWRR, deciding the appropriate weights is crucial to increase the throughput of packet forwarding [14]. However, unappropriated weights results in starvation [15]. Starvation occurs when one or more packets has low priority; therefore, it is not forwarded by the router and is eventually dropped [16].

The significance of this research lies in its aim to automatically decide the appropriate weights of each priority queue to reduce the starvation problem. The proposed algorithm considers the quality of service (QoS) criteria to discriminate the priorities among packets from different applications and count the numbers of forwarded packets and dropped-out packets to decide the appropriate weights of the multiple priority queues. The particle swarm optimization (PSO) algorithm, which is a powerful stochastic and evolutionary optimization technique [17], is proposed as the AI technique in this research to optimize the weights of priority queues. As various research papers have shown that this algorithm is an effective optimization tool, it has been used extensively in many application fields [18–20].

This work proposes and implements a framework that uses the PSO algorithm to optimize and adjust weights of the multilevel queue system within routers periodically and automatically. Accordingly, the weights of queues become more flexible to accept variable traffic intensity and forward packets which will suffer less from packet dropping and starvation.

The main contribution of this work is improving the control plane in SDN using the PSO optimization algorithm, which is one of swarm intelligence algorithms, to configure the queue weights in the weighted round-robin scheduling approach dynamically and automatically in response to the QoS metrics and the number of packets of different QoS requirements. The aim is to investigate and achieve a better performance and reduce the starvation and packet loss problems. The results that we present in this paper show that using the PSO algorithm is promising to adjust and optimize the weights in multilevel queueing systems. Additionally, the results are compared to the results of the conventional CBWRR algorithm, which is implemented within the proposed framework, and the results show that the PSO-based algorithm is better to reduce the effect of starvation and packet-dropping problems.

This paper is divided into six parts. Section 2 reviews the related work in the research fields. Section 2 presents the theoretical background of the problem and the proposed system methodology, and Section 4 shows the results of the proposed system. Finally, Section 5 contains the conclusion and suggestions for future work.

2. Related Work

Research has suggested improving the work of SDN by using AI methods. Rezaee et al. [21] suggested improving the round-robin scheduling algorithm by adding a fuzzy inference system as an AI technique and QoS measurement component. The proposed algorithm uses a knowledge base to configure the queue weights dynamically and autonomously in response to recent QoS metrics to improve the performance and QoS. To evaluate the performance of the proposed algorithm, the loss ratio versus the workload for

the round robin and the proposed algorithm is measured. The result shows a lower loss ratio in the proposed fuzzy-based algorithm. Han et al. [22] investigated the problem of scheduling jobs under multiple factors. They applied their idea to the body and paint shop problem. In this problem, there could exist multiple waiting queues for multiple classes of jobs. Therefore, the main problem is optimizing the multi-queue limited buffer scheduling problems in a flexible flow shop with setup times. They applied a genetic algorithm (GA) to optimize this type of problem, and improve production efficiency and economic profit. The results of simulation experiments proved that combining GA with the proposed rules for local dispatching solved multi-queue scheduling problems in a flexible flow shop with reduced setup times.

Ahmed et al. [23] investigated the quality of service for real-time traffic to improve the performance of the end-to-end network. To achieve this aim, they proposed a low latency queue (LLQ) packet-scheduling algorithm. The improved algorithm used QoS, LLQ scheduling, and multiple queuing with optimized parameters. The performance of the LLQ scheduling algorithm was evaluated in different scenarios. The simulation results proved that this scheduling algorithm reduced the ratio of delay and packet loss and improves utilization. Although this work investigated the problem of scheduling packets from the multi-queue system, it did not use AI techniques.

The PSO algorithm has been used to solve scheduling algorithms in cloud environments. Rekha and Kalaiselvi [24] proposed an algorithm to schedule the requests of virtual machines for resources. The proposed algorithm uses a multilevel priority queue system in the cloud computing context and the particle swarm optimization (PSO) algorithm for splitting the ready queue into certain longer queues. The result shows that it has a lower cost, better throughput, and less delay than using only one queue or different algorithm. Khan et al. [25] investigated IT services provided to customers by cloud environments and the service level agreements (SLAs). They proposed an algorithm to provide cloud service with reliable Quality of service (QoS) and to maintain Service level agreements. It is important for cloud service providers to predict possible service violation before it happens to perform the required countermeasure for it. The major considered violation in this work are response time, accessibility, availability, and speed. In this paper, two variants of Particle swarm optimization (PSO) algorithm are used for the detection and predictions of QoS violations in terms of these concerns. The proposed methods were compared according to accuracy and speed measurements.

The main contribution of this work is that the PSO algorithm was proposed to optimize the queue weights in weighted round-robin scheduling in multi-queue systems for packet-scheduling within core routers in SDN. This approach dynamically and automatically changes the weights of the queues within routers in reaction to the QoS metrics and density of packets of different QoS requirements. The aim is to investigate and achieve a better performance and reduce starvation and packet loss.

3. Methods

3.1. Queuing and Priorities in Routers

DiffServ is an architecture classifying and marking packets as belonging to a specific class of service using 8 bits for the differentiated services (DS) field in the IP header [26]. This field is composed of two parts, and the six most significant bits identify the differentiated services code point (DSCP), while the two least significant bits define the explicit congestion notification (ESN). Classification is based only on the DSCP field. Each router is configured to differentiate traffic based on its set of classes [27,28].

Three types of DSCP values are used in DiffServ. The first is expedited forwarding (EF), which needs low latency (delay), low jitter, and low loss, the specific DSCP value (decimal 46). Often, QoS plans use EF to mark voice payload packets. The second is assured forwarding (AF), which defines a set of 12 DSCP values meant to be used in concert with each other. It defines the concept of four separate queues in a queuing system and three levels of drop priority within each queue for use with congestion avoidance tools. With

four queues and three drop priority classes per queue, 12 different DSCP markings are available one for each combination of queue and drop priority. The third is best effort (BE), which has a value of 0 for DSCP [5,29]. Table 1 summarizes the type of traffic classification according to the QoS requirements.

Table 1. Traffic types and parameters.

Traffic Type	DSCP Value	DiffServ DSCP	Size of IP Packet
Video traffic	10 12	AF11 AF12	160
Voice traffic	46	EF	120
FTP traffic	0	BE	480
HTTP traffic	18 20	AF21 AF22	240

The term "queuing" refers to the management of a set of queues that hold packets while they wait their turn to be forwarded. Most networking devices have a queuing system with multiple queues. To use multiple queues, the queuing system needs a classifier function to choose which packets are placed into which queue according to its DSCP value [28,29]. Furthermore, the queuing system needs a scheduler deciding which message to forward next when the interface becomes available. The scheduler is the most interesting part because it can perform prioritization, which refers here to the QoS of each packet within the traffic. One of the most common schedulers used by routers is the weighted round-robin algorithm. Round-robin scheduling includes the concept of weighting. The scheduler considers queues' priorities and weights to take and forward a different number of packets from each queue. Queuing systems uses a low-latency queue (LLQ) or priority queue to meet the QoS requirements for low delay, low jitter, and low loss of traffic in that queue by providing a high weight, which results in a greater time or bandwidth to this queue. However, if the traffic is mostly a type of voice traffic, then the scheduler never services the other queues, which is known as starvation and leads to packet loss when packets are not forwarded [5,30–32].

3.2. Class-Based Weighted Round-Robin Algorithm

One function of routers is classifying and scheduling packets according to their priorities. Routers mostly use round-robin scheduling with weights for each queue in a multilevel queueing system. The router interface forwards a dissimilar number of packets from each queue, prioritizing one queue over the others. According to Cisco, routers use class-based weighted fair queuing (CBWFQ) to provide the bandwidth amount to each class. CBWFQ uses a weighted round-robin scheduling algorithm configuring weightings as a percentage of the link bandwidth. For example, for three-level queuing system, weight may be given for each queue as 20, 30, and 50 percent of the specified bandwidth or time [5,33,34].

3.3. Particle Swarm Algorithm

Artificial intelligence is a collection of techniques inspired by nature. Particle swarm optimization (PSO) is inspired by birds [35]. The PSO algorithm, which has undergone various improvements since its founding in 1990, is used to solve a wide spectrum of problems. The superiority of PSO algorithm over other metaheuristic algorithms and ability to solve complex problems come from its many advantages [36]: easily implemented, few numbers of parameters, ability to run parallel computations, robustness, fast convergence, and low computational time. However, the PSO algorithm suffers from a few disadvantages [36]: control parameters are difficult to tune, and trapping into the local minima on solving high-dimensional problems.

3.4. The Proposed Framework of the PSO-Based Weighted Round-Robin Algorithm

This work implements a framework to represent and simulate the proposed scheduling algorithm (psoWRR). The framework consists of the following three main parts.

Packet generating: a packet is represented as a structure named packet consists of the following five fields: name of packet, packet size (in bytes), latency time, arrival time, and priority type. All these fields are integers for simplicity and are generated using the random function, except the names of the packets are a series of numbers. This information is used to decide which queue to send this packet to and prioritize one packet over another when forwarding. This step simulates receiving a packet at an edge router, which decides its DSCP value, marks the packet, and sends it to the core router.

Queueing: queues are represented as a structure consisting of the following three fields: an array of structure of type packet, front and rear values both are of integer type. The number of queues is decided and provided as the input to the algorithm. For the test and comparison, we have proposed a three-level queuing system. The highest queue is the EF queue, the second queue is the AF queue, and the lowest one is the BE queue. The multilevel architecture is represented as an array of queue types. In this work, we have tested the architecture of a multilevel queue with three queues. Packets are enqueued to one of the three levels according to their priority value, which simulates the DHCP marking value in real packets. The marked packets are queued in the EF, AF, and BE queues, respectively. At this point, the psoWRR algorithm is applied to optimize the weight of each queue to forward packets from this queue according to the available time quantum.

Packet-scheduling and forwarding: This part of the architecture represents dequeuing packets and forwarding them depending on the weight of the queue. Weights are used in this work as the time specified for each queue to forward packets. Forwarding begins with the non-empty highest priority queue and count down until the specified time of this queue equals zero. Each time a packet is compared with the remaining time to the packet at the front of the queue, if the remaining time is more than zero, then the packet is forwarded, otherwise it is dropped. The scheduler will dequeue packets from the first queue with the W1, which contains packets of voice traffic and DSCP mark is EF. Later, the scheduler will dequeue packets from the second queue with the W2 time, which contains packets with DSCP mark AF. After serving packets in the second queue, the scheduler will serve the packets with DSCP marks of type BE, with W3 time quantum.

Classification task in the router assigns each packet to one of the queue levels according to its DSCP field value. When the forwarding interface becomes available and the number of packets in the multi-queue is greater than 1, the packet that is currently being forwarded is the highest priority packet within queue level j, where j = 0 or $\sum_{j=0}^{j-1} Pj = 0$. Therefore, the packet (Pj) will be forwarded if it satisfies the following rule:

$$(Pj \mid (\max(pr(Qj)), Wj > 0)$$

where Pj is a packet allocated to queue j, pr(Qj) is the priority of queue j, and Wj is the remaining time of Qj.

Pj is forwarded when it reaches the front of the non-empty highest priority queue and the remaining time (weights) of queue j is greater than zero. If Qj is nonempty and Wj equals zero, then the packet that is in the front of the next priority queue is forwarded.

Wj in psoWRR are the dimensions of the particles which correspond to each priority queue as shown in Figure 1. Wj is decreased by subtracting the latency time of the forwarded packets from the weights of the queue until Wj becomes 0.

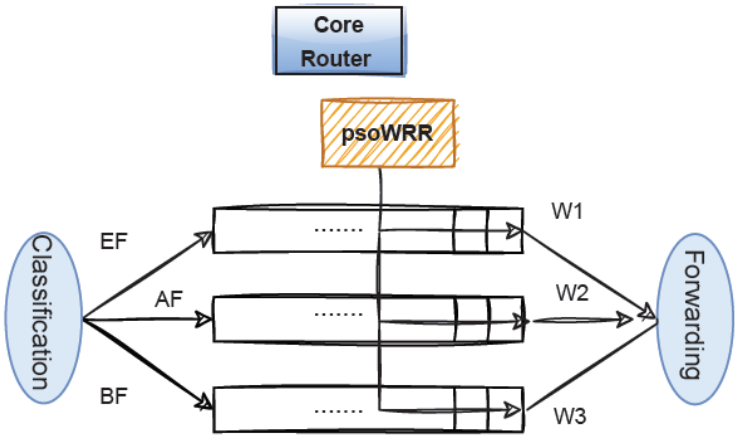

Figure 1. The proposed queuing architecture and psoWRR algorithm.

3.5. PSO-Based Scheduling Algorithm

In this work, we proposed using the PSO algorithm to find the appropriate weight to represent the inter-queues time quantum in the round-robin schedular that reduces the starvation problem and allows the dynamic and self-aware adjustment of these weights according to the changes in the intensity of traffic types. From now on, the proposed algorithm is called the particle swarm optimization for the weighted round-robin algorithm and abbreviated as psoWRR. The calculation of the time quantum of the queues occurs in the proposed system for the following reasons:

- Variations in the intensities of one or more traffic types;
- An interval time decided by the network administrator;
- Increments in the ratio of dropped out packets.

The proposed algorithm receives input parameters related to the problem in question, such as the number of queues and particles' maximum number of swarms during initialization. The general framework of the proposed psoWRR can be seen in the pseudo code in Algorithm 1. In the proposed algorithm, the global best particle is based on the best weights found in the swarm by following the PSO algorithm. The proposed algorithm consists of a PSO framework that allows it to be used to search for optimal queuing architectures, including the initialization of a swarm of particles, fitness evaluation of individual particles, velocity computation, and particle update. These operations are detailed in the following subsections.

Particle Swarm Initialization

Initializing the swarm is the first step in the proposed psoWRR. This function creates several swarms that are equal to the specified swarm size (sz) with random weights. The dimension of a particle is a vector of size (qn) equal to the number of queues. Each particle will have a random number of weight ranges [pmin–pmax].

Fitness Evaluation

The fitness evaluation algorithm is shown in Equation (1). This step of the algorithm includes calculating the number of packets of each queue, then according to the quantum time, the packet is removed from the corresponding queue. If the packets of the queue are all forwarded on time, then the forwarding interface is available directly to the next queue. If the weight is not enough to forward all packets in the corresponding queue, then the architecture moves the forwarding interface when the weight ends. When the weights of all the queues are in the architecture, the round restarts. If some packets are not forwarded within the required latency time, the system drops them. For each particle swarm of weights, the numbers of forwarded packets and dropped-out packets are calculated. The fitness function is evaluated according to the following equation:

$$Fitness = \begin{cases} FP + \frac{1}{DP} & if\ weight - sum \leq pmax \\ FP + \frac{1}{DP} - 1 & if\ weight - sum > pmax \end{cases} \quad (1)$$

where *FP* is the number of forwarded packets, and *DP* is the number of dropped packets. The value of the weight sum equals the sum of dimensions values within each particle. The particle in psoWRR consists of three dimensions. Therefore, the value of weight-sum is the sum of the values of the three dimensions.

Algorithm 1 psoWRR algorithm

Input:
Qn—Number of queues
Sz—Swarm size
itr—Maximum number of iterations
w—Control parameter value
D—Problem dimensionality
c1 = c2 = 2

Output: g^t best—the best position (solution) found so far

1: Start
2: Initialize the swarm randomly
3: for i = 1 to sz do
4: for j = 1 to qn do
5: $x^0_{ij} \leftarrow$ a random vector within [pmin, pmax] ^{D}
6: $v^0_{ij} \leftarrow$ a random vector within [vmin, vmax] ^{D}
7: end for
8: calculate f^0_i
9: p0 $best_i \leftarrow x^0_i$
10: end for
11: Calculate n the position of the greatest f^0_i
12: $g^0_{best} = p^0_{bestn}$
13: for t = 1 to itr do
14: for i = 1 to sz do
15: r1, r2 ← two independent values randomly generated from [0, 1]
16: $V^t_i = w\ V^{t-1}_i + c1\ r1\ (pbest^{t-1}_i - X^{t-1}_i) + c2\ r2\ (gbest^{t-1}_i - X^{t-1}_i)$
17: $X^t_i = X^{t-1}_i + V^t_i$
18: calculate f^t_i
19: if $f^t_i > p^{t-1}_{besti}$
20: $p^t_{besti} \leftarrow f^t_i$
21: end if
22: end for
23: calculate n the position of greater f^t_i
24: If p^t_n best > $p^{t-1}_{gbest\ i}$
25: $P^t_{gbest\ i} = p^t_{besti}$
26: end if
27: end for
28: End

The fitness function should be maximized and its value made approximate or equal to the sum of packets in all queues. Therefore, all the variables in the equation are maximizing the fitness value if the particle dimension values are increasing the number of forwarded packets (FP) and decreasing the number dropped packets (DP). The variable DP is added to equation (1) by dividing 1 by DP. The value of 1/DP is increased as the number of dropped packets is decreased. In the code, to avoid division by zero when DP equals 0, we consider 1 is the minimum value for DP.

The swarm's dimensions values are generated as random values ranging from pmin to pmax. The legal swarm is the swarm with weight-sum value equals or less than pmax. Otherwise, the swarm is penalized by subtracting 1 from the fitness value.

Velocity Computation

The velocity is initialized as an array of random numbers from [−200 to 200]. The velocity is arranged as an array of size (sz). Then, the velocity is updated in each iteration

depending on Equation (2) based on the calculation of pbest, the historically best fitness of a specific swarm, and gbest, the best fitness in the whole previous iterations.

$$V_i^t = w\, V_i^{t-1} + c1 \times r1 \times \left(pbest_i^{t-1} - X_i^{t-1}\right) + c2 \times r2 \times \left(gbest_i^{t-1} - X_i^{t-1}\right) \quad (2)$$

Swarm Update

Updating a swarm is straightforward in the psoWRR algorithm. The update occurs according to Equation (3).

$$X_i^t = X_i^{t-1} + V_i^t \quad (3)$$

where X_i^t is the dimention i of a particle at time t and V_i^t velocity of a particle at time t.

Updating the dimensions of all the population of particles happens once at each iteration until the optimal fitness value is reached or the stopping condition becomes valid.

Parameters of the psoWRR Algorithm

The parameters used in the proposed psoWRR are as follows: number of iterations, swarm size, and w. These parameters are used to control the behavior of the psoWRR algorithm.

The number of iterations controls the total number of iterations that the proposed algorithm will run before considering that the optimization is finished. The swarm size represents the number of swarms in each iteration, and the w parameter is used to control the convergence of the optimized weight of the multilevel queue. All these parameters are set according to experiments to find the best value for each of them.

4. Experimental Results and Analysis

This section presents the results obtained from applying the psoWRR algorithm, a discussion of the results, and a comparison of the results with the conventional round-robin algorithm. The results of applying psoWRR were obtained from a framework that was implemented to simulate the scheduling and forwarding of packets. The framework was implemented using C++ language and contained the following functions: generating packets of different QoS requirements, multilevel queue implementation, and a classification function to enqueue packets to the corresponding queue, the PSO algorithm to find the optimal weight for each of the queues, forwarding packets, and finally reporting result. This framework was implemented to ensure that the proposed algorithm psoWRR improved with the optimized weights to schedule and forward packets by reducing the starvation and packet loss problems.

PSO algorithm efficiency depends on various parameters, therefore the proposed psoWRR algorithm has undergone many experiments to adjust the value of these parameters to accelerate convergence of optimal weights and avoid local minimums. The results of these experiments are shown in Figures 2–6. The size of the population is one of the parameters in the PSO algorithm that needs to be adjusted. The high population size contributes to the convergence of the optimal result. However, a high population size consumes memory space, which is considered a drawback for algorithms. Therefore, the value of population size must be selected carefully. Using the particle swarm algorithm requires processing a population of candidate solutions at each iteration. Keeping the population size as small as possible is important to preserve the memory space in the router. Therefore, we need to consider the minimum population size that allows a minimum convergence time. Minimum convergence time is gained by the reach to the best fitness value within minimized number of iterations.

In psoWRR, the size of the population was tested with values ranging from 3 to 10, and the results are shown in Figure 2.

Figure 2 shows the relationship between the number of packets and fitness value in the following four population size values: 3, 5, 7, and 10. The 7 and 10 population sizes gave the highest fitness values, and both may be considered good numbers for the population size because both values are not large enough to challenge the memory space, and they provide a good fitness value for various packet densities.

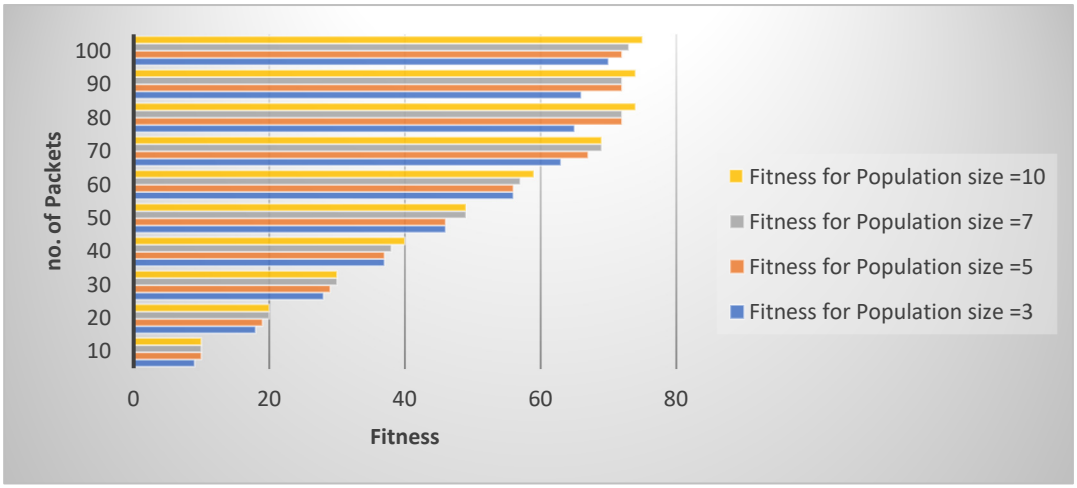

Figure 2. Relation between the number of packets within 10 s. and fitness value for different population sizes.

Additionally, the relationship between the number of packets and fitness value has been studied considering the time available to schedule the various numbers of packets, ranging from 10 packets to 100 packets, which must be scheduled within a limited time range of 10 s to 1 min. As shown in Figure 3, when there is a low number of packets in all queues, the fitness value of psoWRR is high and equals to the number of packets approximately. For instance, when the number of packets equals 10, 20, and 30, fitness value is 10, 20, 30, respectively. However, when the number of the packets is high in one or all the queues, the fitness value is retreated even for high available time, e.g., when number of packets 100, the fitness value is less than 85 for variant available time ranging from 10 to 60 s. The reason behind this problem is the limited size of the queues and some of the packets are lost because the queue is full. In this experiment, the population size was 10, the number of iterations was 10, and the maximum queue size was 60.

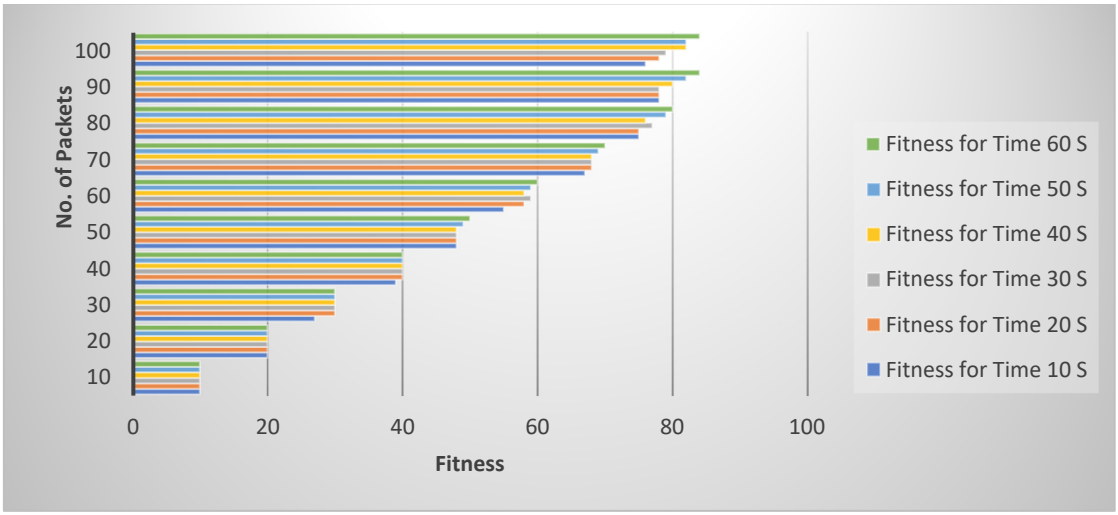

Figure 3. Relation between population size and fitness value for time ranges from 10 s to 1 min.

The number of iterations in the psoWRR algorithm is also an important parameter that must be optimized. A high number of iterations does not ensure the optimal swarm convergence, as it may lead to overfitting and consumes time and memory space. Therefore, we need to find the best and smallest number of iterations that optimize the weights' values of the multilevel queuing system in routers. The relationship between the number of

packets and fitness values is studied according to the following number of iterations: 3, 5, 10, 15, and 20. Both 10 and 15 iterations gave good values for fitness within various number of packets, i.e., the number of forwarded packets is approximating the whole number of packets. In these experiments, although higher number of iterations may provide better results without suffering from overfitting, we have considered 20 for highest number of iterations is to limit execution time and keep it as small as possible. Figure 4 shows the result of experiments for the number of iterations parameter adjustment. For a small number of packets, even 3 and 5 iterations result in optimal fitness values. For the number of packets ranging from 40 to 100, both 10, 15, and 20 iterations gave the best fitness values. We consider 10 and 15 to be the best number of iterations as they gave the best fitness values in a smaller number of iterations. Although 20 iterations gave the highest values for some experiments, we have considered 10 and 15 as the best because the difference of fitness values among the three numbers of iterations are very convergent.

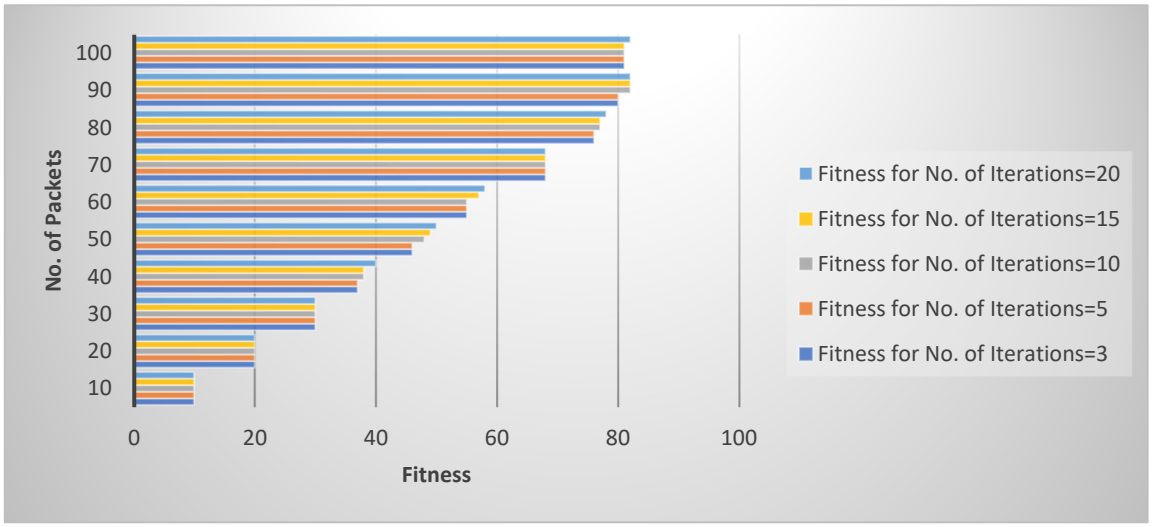

Figure 4. Relation between number of packets and fitness for iterations ranging from 3 to 20.

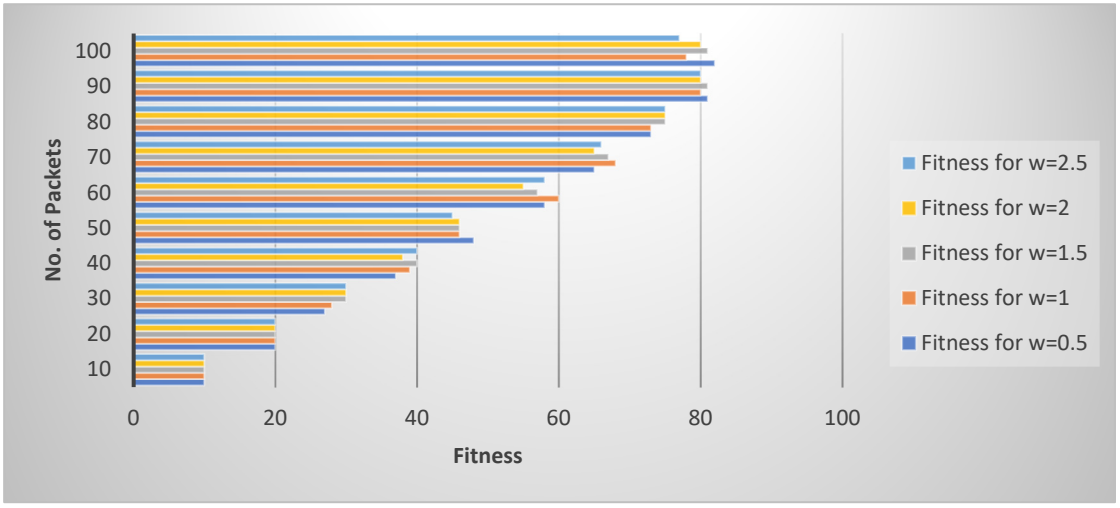

Figure 5. Relation between number of packets and fitness for w parameters.

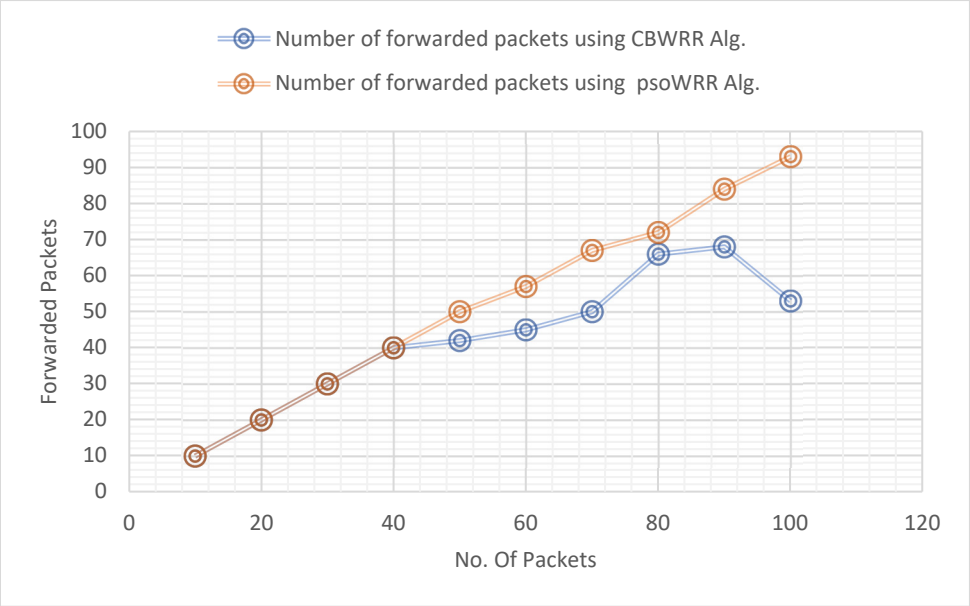

Figure 6. A comparison between CBWRR algorithm [13] and psoWRR alg. For the relation between the numbers of packets and forwarded packets.

One of the important parameters in the PSO algorithm that should be adjusted is the w value. The optimal w value controls the speed of convergence to prevent local minima. Therefore, adjusting this parameter is crucial to find the optimal swarm that gives the global best fitness value. Figure 5 shows the relation between various packet densities and fitness values when the value of w changes in each experiment with the following values: 0.5, 1, 1.5, 2, and 2.5. The results of previous experiments suggest that 1.5 is the best w value for the psoWRR algorithm. The fitness value of psoWRR gave the best value for various numbers of packets and in the majority of experiment results, fitness value is maximized when w = 1.5. In addition, w = 1.5 is an intermediate value as it is not small enough to slow down the convergence or large enough to make convergence premature.

psoWRR Algorithm Efficiency Measurement

The global best (gbest) fitness value resultant after all the determined iterations of psoWRR is compared with the result of scheduling the same packets using the CBWRR algorithm.

The same generated packets in the framework are also scheduled using the conventional round-robin algorithm, and the result of the round-robin algorithm is computed using Equation (4).

$$RR - result = FP + \frac{1}{DP} \tag{4}$$

Here, FP is the sum of forwarded packets, and DP is the sum of the dropped packets when scheduling using the CBWRR algorithm.

The time for the test is 10 s, and these 10 s are used to schedule packets, starting from 10 packets and increasing to 100 packets. Each experiment was repeated 100 times, and the results were reported for both CBWRR and psoWRR algorithms. The weights in CBWRR algorithm were set as [2000, 3000, 5000] in milliseconds as an example [13].

The result of the comparison shows that keeping static weights CBWRR algorithm increases the number of dropped packets due to the variance in traffic intensity. Therefore, optimizing the weights of the queues using psoWRR algorithm reduces the dropped packets and increases the forwarded packets which resultant in packet loss ratio reduction. The majority of dropped packets are the packets from the lower priority queue. Therefore, reducing the number of dropped packets will result in reducing packet starvation.

The comparison between psoWRR and CBWRR algorithms is shown in Figure 6; the number of dropped packets is reduced in the psoWRR. Although the difference is very small when the number of packets is low, in higher queues intensities, the difference is higher.

The proposed psoWRR is compared with references [21,23] according to the packet loss ratio, and the result are shown in Table 2. In psoWRR algorithm, packet loss ratio is calculated using Equation (5) for 100 times repeated experiments to the algorithm.

$$\text{psoWRR Loss Ratio} = \frac{\text{no. of dropped packets}}{\text{no. of Packets}} \times 100 \quad (5)$$

while the packet loss ratio of FDWFQ [21] and LLQ [23] are calculated according to results shown in these works.

Table 2. A comparison between the related work and proposed psoWRR for the packet loss ratio.

Work	Packet Loss Ratio
FDWFQ [21]	35%
LLQ [23]	7% (only for voice packets)
psoWRR	7% (for all types of packets)

5. Conclusions and Suggestions for Future Work

In this work, we implemented a PSO-based network traffic scheduling algorithm. The conventional round-robin algorithm uses fixed size weights for each queue in the multilevel queue system within routers. Considering the variable traffic density, we suggest making the weights dynamic rather than static. Using dynamic weights in multilevel queue scheduling reduces the following two common problems: packet loss and starvation. The proposed psoWRR algorithm is implemented within a framework and simulates scheduling packets in a router. The proposed algorithm and framework were tested using a number of experiments, and the comparison of the results of psoWRR, round robin, and the related works [21,23] shows that the proposed algorithm is better to schedule a larger number of packets within a specific time. Additionally, the proposed algorithm reduces the problem of packet loss that results from the starvation of low-priority packets. Moreover, the work in [23] reduces the packet loss of voice packets using an LLQ. However, other types of packets will suffer more from packet loss due to using LLQ.

It is also possible to create templates of weights to replace applying psoWER algorithm if a pattern of traffic intensity is continuing for a period of time. This approach will reduce the time to apply psoWRR and accelerate forwarding tasks.

In this work, only the latency time is considered as criteria for the QoS. In future work, other criteria, such as delay and jitter, should be considered. Furthermore, other optimization algorithms should be investigated, especially for the required processing time and memory space.

Author Contributions: Conceptualization R.F.G. and L.A.-J.; methodology, software, R.F.G.; validation, R.F.G. and L.A.-J.; formal analysis, investigation, resources, and data curation, R.F.G.; writing—original draft preparation, writing—review and editing, and visualization, R.F.G. and L.A.-J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

References

1. Akpovi, A.O.; Adebayo, A.O.; Osisanwo, F.Y. Introduction to software defined networks (SDN). *Int. J. Appl. Inf. Syst.* **2016**, *11*, 10–14. [CrossRef]
2. Almadani, B.; Beg, A.; Mahmoud, A. DSF: A distributed SDN control plane framework for the east/west interface. *IEEE Access* **2021**, *9*, 26735–26754. [CrossRef]
3. Cisco. Software-Defined Networking. Available online: https://www.cisco.com/c/en/us/solutions/software-defined-networking/overview.html?dtid=osscdc000283 (accessed on 24 September 2022).
4. Rana, D.S.; Dhondiyal, S.A.; Chamoli, S.K. Software defined networking (SDN) challenges, issues and solutions. *Int. J. Comput. Sci. Eng.* **2019**, *7*, 884–889. [CrossRef]
5. Wu, S.; Yang, L.; Guo, J.; Chen, Q.; Liu, X.; Fan, C. Intelligent quality of service routing in software-defined satellite networking. *IEEE Access* **2019**, *7*, 155281–155298. [CrossRef]
6. Latah, M.; Toker, L. Artificial intelligence enabled software-defined networking: A comprehensive overview. *IET Netw.* **2019**, *8*, 79–99. [CrossRef]
7. Belgaum, M.R.; Ali, F.; Alansari, Z.; Musa, S.; Alam, M.M.; Mazliham, M.S. Artificial intelligence based reliable load balancing framework in software-defined networks. *Comput. Mater. Contin.* **2021**, *70*, 251–266. [CrossRef]
8. Bagaa, M.; Taleb, T.; Bernabe, J.B.; Skarmeta, A. A machine learning security framework for IoT systems. *IEEE Access* **2020**, *8*, 114066–114077. [CrossRef]
9. Ijaz, A.; Namal, S.; Ylianttila, M.; Gurtov, A. Towards software defined cognitive networking. In Proceedings of the 2015 7th International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 27–29 July 2015; pp. 1–5.
10. Alhaidari, F.; Almotiri, S.H.; Al Ghamdi, M.A.; Adnan Khan, M.; Rehman, A.; Abbas, S.; Masood Khan, K.; Rahman, A. Intelligent software-defined network for cognitive routing optimization using deep extreme learning machine approach. *Comput. Mater. Contin.* **2021**, *67*, 1269–1285. [CrossRef]
11. Nowak, M.P.; Pecka, P. Routing algorithms simulation for self-aware SDN. *Electronics* **2022**, *11*, 104. [CrossRef]
12. Lu, S.; Wu, M.; Zhao, M. Particle swarm optimization and artificial bee colony algorithm for clustering and mobile based software-defined wireless sensor networks. *Wirel. Netw.* **2022**, *28*, 1671–1688. [CrossRef]
13. Odom, W. *CCNA 200–301 Official Cert Guide*; Pearson Education, Cisco Press: Indianapolis, IN, USA, 2020; Volume 2, pp. 549–595.
14. Raheja, S.; Dadhich, R.; Rajpal, S. Designing of vague logic based multilevel feedback queue scheduler. *Egypt. Inform. J.* **2016**, *17*, 125–137. [CrossRef]
15. Mohammed, M.A.; Abdulmajid, M.; Mustafa, B.A.; Ghani, R.F. Queueing theory study of round robin versus priority dynamic quantum time round robin scheduling algorithms. In Proceedings of the 2015 4th International Conference on Software Engineering and Computer Systems, ICSECS 2015: Virtuous Software Solutions for Big Data, Kuantan, Malaysia, 19–21 August 2015.
16. Su, Y.; Wang, S.; Cheng, Q.; Qiu, Y. Learning-based buffer starvation modelling for packets prefetching strategies of video streaming services. *J. Phys. Conf. Ser.* **2021**, *1827*, 1–7. [CrossRef]
17. Khan, S.A.; Engelbrecht, A.P. A fuzzy particle swarm optimization algorithm for computer communication network topology design. *Appl. Intell.* **2012**, *36*, 161–177. [CrossRef]
18. Wang, D.; Tan, D.; Liu, L. Particle swarm optimization algorithm: An overview. *Soft Comput.* **2018**, *22*, 387–408. [CrossRef]
19. Freitas, D.; Lopes, L.G.; Morgado-Dias, F. Particle swarm optimisation: A historical review up to the current developments. *Entropy* **2020**, *22*, 362. [CrossRef]
20. Wang, Y.; Wang, Q.; Zhang, H.; An, K.; Ye, X.; Sun, Y. Improved particle swarm optimization algorithm for optimization of power communication network. *Int. J. Grid Distrib. Comput.* **2016**, *9*, 225–235. [CrossRef]
21. Rezaee, A.; Rahmani, A.M.; Adabi, S.; Adabi, S. A fuzzy algorithm for adaptive multilevel queue management with QoS feedback. In Proceedings of the 2011 International Conference on High Performance Computing and Simulation, HPCS 2011, Istanbul, Turkey, 4–8 July 2011; pp. 121–127.
22. Han, Z.; Zhang, Q.; Shi, H.; Zhang, J. An improved compact genetic algorithm for scheduling problems in a flexible flow shop with a multi-queue buffer. *Processes* **2019**, *7*, 302. [CrossRef]
23. Ahmed, S.; Ali, M.; Baz, A.; Alhakami, H.; Akbar, B.; Khan, I.A.; Campus, A.; Ahmed, P.A.; Junaid, M. A Design of packet scheduling algorithm to enhance QoS in high-speed downlink packet access (HSDPA) core network. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 596–602. [CrossRef]
24. Rekha, S.; Kalaiselvi, C. Multi Level Queue Scheduling With Particle Swarm Optimization (Mlqs-Pso) Of Vms in Queueing Heterogeneous Cloud Computing Systems. *Int. J. Recent Technol. Eng. IJRTE* **2020**, *8*, 664–672. [CrossRef]
25. Khan, M.A.; Kanwal, A.; Abbas, S.; Khan, F.; Whangbo, T. Intelligent model for predicting the quality of services violation. *Comput. Mater. Contin.* **2022**, *71*, 3607–3619. [CrossRef]
26. Cisco. White Paper: Diffserv-the Scalable End-to-End Quality of Service Model. Available online: https://www.cisco.com/en/US/technologies/tk543/tk766/technologies_white_paper09186a00800a3e2f.pdf (accessed on 26 September 2022).
27. Aureli, D.; Cianfrani, A.; Diamanti, A.; Manuel Sanchez Vilchez, J.; Secci, S. Going beyond DiffServ in IP Traffic Classification. In Proceedings of the NOMS 2020–2020 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 20–24 April 2020; pp. 1–6.
28. Barik, R.; Welzl, M.; Elmokashfi, A.; Dreibholz, T.; Islam, S.; Gjessing, S. On the utility of unregulated IP DiffServ Code Point (DSCP) usage by end systems. *Perform. Eval.* **2019**, *135*, 102036. [CrossRef]

29. Joung, J.; Kwon, J.; Ryoo, J.D.; Cheung, T. Asynchronous deterministic network based on the DiffServ architecture. *IEEE Access* **2022**, *10*, 15068–15083. [CrossRef]
30. Asingwire, B.K.; Ngenzi, A.; Sibomana, L.; Kabiri, C. Performance analysis of IoT-based healthcare heterogeneous delay-sensitive multi-server priority queuing system. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 666–673. [CrossRef]
31. Zeeshan, M.; Ali, A.; Naveed, A.; Liu, A.X.; Wang, A.; Qureshi, H.K. Modeling packet loss probability and busy time in multi-hop wireless networks. *EURASIP J. Wirel. Commun. Netw.* **2016**, *2016*, 1–16. [CrossRef]
32. Subramaniam, S.K.; Panessai, I.Y.; Ramlee, R.A.; Sujatha, R.; Rajagopal, N. Tcp Performance and throughput fairness optimization in a multi-hop pipeline network. *Int. J. Recent Technol. Eng.* **2019**, *8*, 499–505.
33. Zhigalov, K.; Skorikova, N.A.; Daudov, I.M. Interaction of models and methods of providing QoS in networks. *J. Phys. Conf. Ser.* **2020**, *1582*, 012095. [CrossRef]
34. Al-Haddad, R.; Velazquez, E.S.; Fatima, A.; Winckles, A. A novel traffic shaping algorithm for SDN-sliced networks using a new WFQ technique. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 1–9. [CrossRef]
35. Fedor, B.; Straub, J. A Particle Swarm Optimization Backtracking Technique Inspired by Science-Fiction Time Travel. *AI* **2022**, *3*, 390–415. [CrossRef]
36. Gad, A.G. Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review. *Arch. Comput. Methods Eng.* **2022**, *29*, 2531–2561. [CrossRef]