



## Article

# Directionally-Enhanced Binary Multi-Objective Particle Swarm Optimisation for Load Balancing in Software Defined Networks

Mustafa Hasan Albowarab , Nurul Azma Zakaria \*  and Zaheera Zainal Abidin

Fakulti Teknologi Maklumat Dan Komunikasi, Universiti Teknikal Malaysia Melaka (UTeM), Hang Tuah Jaya, Durian Tunggal, Melaka 76100, Malaysia; p031710041@student.utem.edu.my (M.H.A.); zaheera@utem.edu.my (Z.Z.A.)

\* Correspondence: azma@utem.edu.my

**Abstract:** Various aspects of task execution load balancing of Internet of Things (IoT) networks can be optimised using intelligent algorithms provided by software-defined networking (SDN). These load balancing aspects include makespan, energy consumption, and execution cost. While past studies have evaluated load balancing from one or two aspects, none has explored the possibility of simultaneously optimising all aspects, namely, reliability, energy, cost, and execution time. For the purposes of load balancing, implementing multi-objective optimisation (MOO) based on meta-heuristic searching algorithms requires assurances that the solution space will be thoroughly explored. Optimising load balancing provides not only decision makers with optimised solutions but a rich set of candidate solutions to choose from. Therefore, the purposes of this study were (1) to propose a joint mathematical formulation to solve load balancing challenges in cloud computing and (2) to propose two multi-objective particle swarm optimisation (MP) models; distance angle multi-objective particle swarm optimization (DAMP) and angle multi-objective particle swarm optimization (AMP). Unlike existing models that only use crowding distance as a criterion for solution selection, our MP models probabilistically combine both crowding distance and crowding angle. More specifically, we only selected solutions that had more than a 0.5 probability of higher crowding distance and higher angular distribution. In addition, binary variants of the approaches were generated based on transfer function, and they were denoted by binary DAMP (BDAMP) and binary AMP (BAMP). After using MOO mathematical functions to compare our models, BDAMP and BAMP, with state of the standard models, BMP, BDMP and BPSO, they were tested using the proposed load balancing model. Both tests proved that our DAMP and AMP models were far superior to the state of the art standard models, MP, crowding distance multi-objective particle swarm optimisation (DMP), and PSO. Therefore, this study enables the incorporation of meta-heuristic in the management layer of cloud networks.

**Keywords:** SDN; load balancing; multi objective particle swarm optimisation; crowding distance; angle searching; execution time; energy consumption; less execution cost



**Citation:** Albowarab, M.H.; Zakaria, N.A.; Zainal Abidin, Z. Directionally-Enhanced Binary Multi-Objective Particle Swarm Optimisation for Load Balancing in Software Defined Networks. *Sensors* **2021**, *21*, 3356. <https://doi.org/10.3390/s21103356>

Academic Editor: Antonio Guerrieri

Received: 5 February 2021

Accepted: 1 April 2021

Published: 12 May 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Recent years have seen a rapid boom in the development of many new technologies such as Internet of Things (IoT) and cloud systems. The emergence of cloud computing and data storage centres has led researchers to focus on optimising functionality and service. The implementation of these optimisations became easier and flexible with the development of software defined networking (SDN). However, the centralized control architecture of SDN generates concerns about reliability, scalability, fault tolerance, and interoperability [1]. Hence, assuring efficient management of SDN based computation is essential for the successfulness of the system. Computing, a common type of service type, can be defined as coordinating the execution of big processes or tasks on a network to meet a set of objectives or goals such as maximum reliability, minimum execution time,

and minimum rental cost. Simultaneously, meeting all these is not possible due to implicit conflicts. Hence, the incorporation of multi-objective optimisation (MOO) is needed to define an optimisation point from a set of non-dominated points. Meta-heuristic based MOO optimisation is a suitable algorithm candidate that can be adapted for this purpose. Meta-heuristic-based SDN load balancing has been suggested as a new and emerging research topic [2].

Meta-heuristic searching algorithms are a family of algorithms that follow the same concept of searching within numerous or infinite sets of candidate solutions to find the best non-dominated solutions, the optimal solutions, to a problem with respect to one or multi-objective functions. Most meta-heuristic searching algorithms are associated with certain metaphors such as black holes [3], simulated annealing [4], bee colonies [5], and genetics [6]. They only differ in the nature of their search and their ability to avoid local minima while completing the search in a minimum amount of time. However, meta-heuristic searching was originally developed as a single-objective optimisation. Considering the multi-objective nature of many real-world problems, researchers have endeavoured to convert many single-objective optimisations to MOO such as the multi-objective non-dominated sorting genetic algorithm II (NSGA-II) [7] and NSGA-III [8]. For particle swarm optimisation (PSO), multi-objective particle swarm optimisation (MOPSO) [9], many other variants [10], and [11] were developed. Various tools and frameworks were also provided to compare MOO algorithms and develop their features.

Analysing multi-objective performance requires relying on non-dominated sorting to evaluate solutions and travel change within the search space of a meta-heuristic particle swarm. However, non-dominated sorting is not the only means in evaluating the solution in the swarm. Other aspects of performance, such as the diversity of the solutions and their spread in the decision space, warrant identification. Hence, other criteria are used by researchers such as crowding distance in NSGA-II [12] and directional distribution of solutions in angle quantization [13]. However, an integrated criterion capable of combining both angle and direction has yet to be used effectively in a MOPSO variant model.

Computational load balancing of tasks, or tasks allocation, on a set of computing nodes is a multi-objective combinatorial optimisation problem (MOCOP) where the goal is to minimise the execution time and the cost of renting the necessary nodes. However, the problem is regarded as a non-deterministic polynomial time NP-hard optimization problem which motivates researchers to use meta-heuristic searching to solve it. Furthermore, the problem is a discrete searching problem. Hence, solving this type of optimisation problem requires searching in discrete spaces with optimised multi-objective functions while being mindful of the execution time and the node renting cost.

There are two categories in the context of load balancing using SDN technology—computational load balancing and traffic load balancing. The objective of this study was to present new joint mathematical formulation for load balancing problems from the perspectives of computational load balancing, namely, execution time, energy consumption, and cost. The study also aimed to develop a novel MP variant model with enhanced exploration, crowding distance, and angle quantization. Two variant models, continuous and discrete, were developed and compared with existing MP and crowding-distance-based multi-objective particle swarm optimisation (C-MOPSO).

This study presents a set of contributions. We summarize them as follows.

(1) It proposes a joint mathematical formula that addresses three load balancing problems: time, energy, and cost.

(2) It proposes a MOO-based PSO algorithm which not only takes into account the directions of the solutions in the space but also provides a novel solution selection algorithm by combining crowding distance and angle, distance angle multi-objective particle swarm optimisation (DAMP), or using the angle criterion at one time, angle multi-objective particle swarm optimisation (AMP).

(3) It evaluates the developed MOO-based PSO algorithm using mathematical functions and it compares with benchmarking algorithm from MOO-PSO literature. In addition,

binary variants of the approaches were generated based on transfer function and they are denoted by B-(Method Name), e.g., BAMP for AMP.

The literature review is presented in Section 2, and the methodology is detailed in Section 3, while the evaluation and the results are explained in Section 4. The conclusion and the future considerations are expressed in Section 5.

## 2. Literature Review

The problem of load balancing in SDN networks has become an active research topic in the recent years. In the work of [14], the architecture of SDN layers considers load balancer as one block of SDN application tier. In the survey of [15], the authors provided a taxonomy of load balancing in SDN with discussing the various objectives such as response time, resources optimization throughout, and bottlenecks. Their taxonomy has classified load balancing in control plane and data plane. The former was divided into hierarchal and virtualization controller, while the latter was divided into server and link.

There are many research articles exploring the use of meta-heuristic searching in SDN-based network applications, for instance, modified genetic searching for SDN placement in networks [16] and chaotic salp swarm algorithm (CSSA) optimisation to obtain the optimal number of SDN in networks [17], while [18] proposed a resource selection MOO genetic algorithm using SDN network. There are two categories in the context of load balancing using SDN technology—computational load balancing and traffic load balancing. In the work of [19], an approach for enabling real-time traffic matrix for the traffic measurement system in SDN was proposed. Their design includes fixed and elastic schemas in order to achieve overhead reduction without compromising on accuracy. Hence, it falls under the category of multi-objective SDN network traffic measurement. In the work of [20], a flow-aware elephant flow detection applied to SDN was proposed in order to enable sharing the elephant flow classification tasks between the controller and the switches, which is a type of traffic load balancing.

Many studies have used meta-heuristics for traffic load balancing, such as a study by [21], where genetic algorithm was integrated with ant colony optimisation for traffic load balancing, and a study by [22] that proposed genetic optimisation for traffic loading using SDN. In this study, the researchers used a load balancing algorithm to identify the shortest path, which requires the least number of operations, by looking for the lowest capacity among the switches. This was determined by the load balancer using information provided by the controller. Three pieces of information were used for this objective: the path cost, the switch capacity, and the operational cost. The algorithm was also designed to have a mutation operation that used a path and a link to create a new path [22]. While that study only merely used a single optimisation, other studies have adopted MOO algorithms for similar purposes [16]. Computational load balancing aims to optimise task execution from various perspectives such as execution, reliability, and cost. In a study by [23], a framework for load balancing using two meta-heuristic optimisation methods focused on makespan and cost metrics. However, their framework failed to evaluate exploratory searches. Another work by [24] used MOO for SDN-based load balancing where quality of service (QoS) was considered a constraint, while energy saving and load balancing were optimised using MOPSO. This method also failed to evaluate the exploration or the diversity of the solutions. Load balancing has been applied in advanced simulations such as molecular dynamics in a study by [25], where the optimisation was based on heterogeneous supercomputers, which made the optimisation more difficult or a non-deterministic polynomial time NP-hard problem and combined genetic optimisation and PSO.

Numerous recent studies explore developing effective MOPSO with a focus on how to best supply searching via a combination of methods to enhance and obtain more optimal solutions. In a study by [26], four strategies—multi-population, dynamic clustering, solution life, and probability lottery—were used in conjunction with their MOPSO variant model. The study concluded that their MOPSO variant model was superior since it included more

than one strategy while searching. However, it failed to carefully evaluate other strategies such as set coverage, hyper-volume, and delta measure, thereby rendering their algorithm inadequate. Another aspect that is currently under consideration is the computation cost of an algorithm when multi-populations are added.

In another study [27] using MOPSO, the moving strategy of particles toward local and global was embedded in the cross-over operation, which cannot be considered a real improvement in the search itself. The MOPSO variant models developed by the studies mentioned in this literature review used various concepts to ensure the discovery of an adequate set of non-dominated solutions. In a study by [28], MOPSO was used with crowding distance to perform clustering. Considering that multi-objective optimisation is evaluated using wide set of indicators [29], various approaches were developed based on the concept of using one of the indicators of diversity to select solutions from one iteration to another. In another study, R2 indicator contribution was used to select particles instead of crowding distance value, which is capable of achieving higher diversity in the search [30]. R2 indicator contribution value was used also by another MOO-based PSO method study in order to scalarise the solutions in the archive of non-dominated solutions [31]. Another study used unary epsilon indicator and Pareto dominance [32] in addition to direction-based reference points similar to a study by [33].

A study by [34] used angle-based searching in MOO-based PSO, which considered selecting solutions from the low density angle region and deleting the extra particles from the high density angle region. While the method was based on adaptive angle division, distance-based crowding was not included in the search, which affected one aspect of the diversity of the discovered solutions. Other researchers have developed MOPSO that incorporated crowding distance [35]. The study modified the velocity formula by including the sharing-learning factor. The sharing factor was added as a third term in the velocity equation to move the particle not only to the direction of personal and global best particles but also to the average of all other particles in the swarm, which added more diversity to the swarm. The approach also added Gaussian mutation to the particles to achieve higher exploration. Furthermore, the study proposed an update of best global using greedy strategy with respect to each particle's changing position. This method, however, ignored the mobility direction in the particles, which is regarded as another factor in the diversity of exploration. A study by [36] evaluated an aspect that is usually ignored, leaders evaluation in traditional PSO. They went on to propose a new concept that a good leader takes feedback from his/her followers and modifies their decisions accordingly. The study went on to identify various cases of follower-based improvements in the swarm according to the change of the fitness values. Based on that, the velocity of the leader which expressed the changes of the movement speed and direction toward the leaders was changed. However, this method was only applied in single-objective optimisation. Another MOPSO variant model, which incorporated a new concept to achieve diversity and exploration, was proposed by [37]. The researchers divided the space into sets of hyper-boxes and tracked the number of solutions in each hyper-box, finally considering only the solutions at the boundary of each hyper-box. Some methods have developed new criteria for selecting leaders from the repository of non-dominated solutions. For example, a study by [38] developed an improved MOO variant model and provided an algorithm for selecting leaders from sets of non-dominated solutions using a geometrical approach. The approach selected points that had the least distance from the line fitted model for the set of non-dominated solutions. However, the problem with this approach was its invalid assumption of the straight-line approximation of the set of non-dominated solutions occurring in most cases.

Other studies have incorporated clustering in solution selection from one iteration to another, where the set of non-dominated solutions were decomposed to clusters and solutions belonging to different clusters were selected to achieve diversity in the solutions. For instance, [39] used Euclidean distance for clustering. This can be criticised for the fact that an implicit sphere assumptions of Pareto parts geometric model was made, which is not

valid in many types of optimisation surfaces. Many other methods have opted to convert MOO to mono-objective optimisations via decomposition and use it along with dominance to select solutions. This was done in a study by [40] where penalty-based boundary intersection (PBI) was used with dominance to create a hybrid strategy. In our opinion, researchers can fall in the non-convexity trap by applying mono-objective mapping. In the same vein, a study by [41] estimated solution domination using cosine transformation and reference vector association and also used simplified leader-oriented mobility equation to counter the slow convergence and simplify the calculation. The approach presented elite velocity-based selections and a twofold leader definition. However, the cosine distance and the reference vector association lacked accurate estimation of solution diversity in the search space. Studies that have developed MOO-based PSO by exploiting existing information and communication theory concepts, namely entropy and its usefulness in probing the convergence of algorithms based on the entropy behaviour, also warrant mention. This method was first proposed by [42] where a simulation was used to prove the association between the change of entropy in the particles of the Pareto front and the convergence of the algorithm. This method was further developed by [43] where, using particle entropy, the particles were mapped on a parallel-cell coordinate system and a feedback information system, and the difference of entropy was used to change the parameters of the algorithm. Although entropy can be a useful metric indicator of the diversity of the solutions, it still lacks actual geometric and directional description of the particles in the space.

While numerous studies have evaluated load balancing using meta-heuristic, none have evaluated it from the perspective of non-dominated solutions. Furthermore, the exploration of the solution space has not received adequate attention since it is critical in providing the decision maker with sets of choices. Although MOPSO was among the evaluated methods, the improvements done to it ignored searching direction, which plays an important role in the diversity of the offered solutions. A summary of the models and the objectives listed in this literature review are presented in Table 1. None of these studies explored incorporating all five objectives in one model. Therefore, we present the findings of our MP-based load balancing model in subsequent sections.

**Table 1.** Summary of various existing models and objectives for load balancing.

Authors	Meta-Heuristic	Application	Network Load	Reliability	Energy	Cost	Execution Time
[16]	Discrete Particle swarm optimization Algorithm	Virtual Network Embedding Algorithm for SDN	Yes	Yes	No	No	No
[17]	Chaotic Slap Optimization	Distributed Multi-Controller Deployment	No	Yes	No	No	Yes
[18]	The Reference Vector Based Algorithm	SDN Based Resource Selection	No	No	Yes	Yes	No
[21]	Genetic-Ant Colony Optimization	Traffic Load Balancing.	Yes	No	Yes	No	No
[22]	Genetic Optimization	Load Balancing	Yes	No	Yes	No	No
[23]	Bat Algorithm	SDN Based Load Balancing	Yes	No	No	No	No
[24]	Multi-Objective Particle Swarm Optimization Algorithm	SDN Based Load Balancing	Yes	No	Yes	No	No
Ours	DAMP/AMP	SDN Based Load Balancing	Yes	Yes	Yes	Yes	Yes

SDN: software-defined networking; PSO: particle swarm optimisation.

### 3. Methodology

The symbols used throughout this study are explained in Table 2. The methodology starts with presenting the network model in Section 3.1. Next, the task model is provided in Section 3.2. Afterwards, the energy model, the time execution metric, and the renting cost metric are provided in Sections 3.3–3.5, respectively. Next, the optimization objective functions are given in Section 3.6. Next, we present the transfer function model for dealing with the binary space in Section 3.7. The crowding distance is presented in Section 3.8, and the developed DAMP algorithm is provided in Section 3.9. Next, a big O notation is given in Section 3.10. Lastly, a separated section is dedicated for the evaluation analysis.

**Table 2.** Explanation of symbols used throughout this study.

Symbol	Meaning
$N$	Set of nodes of network
$R$	Set of connections or edges between nodes
$e_{ij}$	Connection between node $i$ and node $j$
$N_j$	Node of index $j$
$d_{ij}$	Distance between node $i$ and node $j$
$v_j$	Speed of node $j$
$e_j$	Average energy consumption
$e_{init}$	Initial energy of node
$P_0$	Maximum computational load that can be handled by one node
$L_0$	Maximum communication load that can be handled by one node
$P_i$	Computational load of task $i$
$L_i$	Communication load of task $i$
$K$	The number of nodes required to execute the task
$M$	Set of tasks
$E_M$	Set of directions of tasks dependency
$E_{comp}$	Computation energy
$E_{Tx}(k, d)$	Energy consumption for transmitting $k$ bits for distance $d$
$E_{Rx}(k)$	Energy consumption for receiving $k$ bits
$E_{elec}$	Constant to run the transmitter or receiver circuit
$\epsilon_{amp}$	coefficient related to the transmitter amplifier
$d$	Distance between transmitter and receiver
$w$	inertia
$V_{max}$	Maximum velocity of particle
$V_{min}$	Minimum velocity of particle
$C1$	Constant of local target or leader
$C2$	Constant of global target or leader
$RS$	remaining solutions to be selected from the swarm

#### 3.1. Network Model

The network was represented by an undirected graph  $G(N, R)$ , where the networks were  $N = \{N_j : j = 1, 2, \dots, n\}$   $R = \{e_{ij}, i, j = 1, 2, \dots, n \text{ and } i \neq j\}$ . The edge between two nodes  $(i, j)$  had a weight that represents the distance between the two nodes ( $d_{ij}$ ). When the  $i$  node was not connected to the  $j$  node, the distance between the two was infinity. When unconnected, the nodes were represented as  $N_5$  and  $N_6$ .

Each node was described by variables to determine its computational and power specifications. In order to describe node  $n_j$ , we used the tuple  $(v_j, e_j, e_{init})$  where  $v_j$  denoted computation power of the node, which was measured as instruction per second (IPS),  $e_j$  denoted average energy consumption as measured by Joules per second ( $J/sec$ ), and  $e_{init}$  denoted the initial energy. Each node was also described by two constant variables,  $P_0$  and  $L_0$ .  $P_0$  represented the maximum computational load, while  $L_0$  represented the maximum communication load.

### 3.2. Task Model

The task model  $G(M, E_M)$  was provided by directed acyclic graph (DAG) where we assumed that we had  $m$  tasks  $M = \{M_i : i = 1, 2, \dots, m\}$ , and each task was described by computation or,  $P = \{P_i : i = 1, 2, \dots, m\}$ , which indicated the number of instructions (NI) and the communication or,  $L = \{L_i : i = 1, 2, \dots, m\}$ , measured in bytes. Since the nodes that executed the tasks had  $P_0$  and  $L_0$ , Equation (1) was used to determine the number of nodes required to execute any  $i$  task as  $K$ .

$$K = \max\left(\left\lceil \frac{P_i}{P_0} \right\rceil, \left\lceil \frac{L_i}{L_0} \right\rceil, K_0\right) \quad (1)$$

where  $K_0$  denotes the minimum number of nodes required to execute one task.

### 3.3. Energy Model

The energy consumption model was a combination of two parts; the first part,  $E_{comp}$ , denoted energy consumption based on the execution of the instructions of the  $P_i$  task and was determined using Equation (2).

$$E_{comp} = \frac{P_i}{v_j} \times e_j \quad (2)$$

The second part,  $E_{comm}$ , was a combination of two other parts and was expressed in Equation (3).

$$E_{comm} = E_{Tx}(k, d) + E_{Rx}(k) \quad (3)$$

It assumed that the radio emitted  $E_{elec} \left[ \frac{nJ}{bit} \right]$  to power the transmitter or the receiver circuit and was expressed in Equation (4).

$$E_{Tx}(k, d) = E_{Tx-elec}(k) + E_{Tx-amp}(k, d) = E_{elec} \times k + \epsilon_{amp} \times k \times d^2 \quad (4)$$

In order to receive, the energy was expressed in Equation (5) as follows:

$$E_{Rx}(k) = E_{Rx-elec}(k) = E_{elec} \times k \quad (5)$$

where:

$E_{Tx}(k, d)$  was the energy consumption when transmitting  $k$  bits over a distance  $d$ ,

$E_{Rx}(k)$  was the energy consumption when receiving  $k$  bits,

$k$  was the number of transmitted bits and derived from  $L$  in the task model,

$d$  was the distance between the two nodes and derived from the network model,

$E_{elec} = 50 \left[ \frac{nJ}{bit} \right]$  was the constant required to power the transmitter or the receiver circuit, and

$\epsilon_{amp}$  was the coefficient related to the transmitter amplifier and equalled  $100 \left[ \frac{pJ}{bit} / m^2 \right]$

$$E_{ij} = E_{comm} + E_{comp} = E_{Tx}(k, d) + E_{Rx}(k) + E_{comp} \quad (6)$$

Using  $L_i$  which was communication load of task  $i$  and substituting Equations (2), (4) and (5) in (6) provided (7)

$$E_{ij} = E_{elec} \times L_i + \epsilon_{amp} \times L_i \times d^2 + E_{elec} \times L_i + \frac{P_i}{v_j} \times e_j \quad (7)$$

### 3.4. Time Execution Metric

Equation (8) expresses how long each node ( $n_i$ ), at a velocity of  $v_i$ , spent executing task  $j$  two times. Its computation and communication are given as follows:

$$t_{ij} = t_{com}^{ij} + t_{comp}^{ij} + t_{queue}^{ij} = \frac{L_j}{B} + \frac{p_j}{v_i} + t_{queue}^{ij} \quad (8)$$

Equation (9) was used to determine the makespan

$$T = \sum_{i,j} t_{ij} \quad (9)$$

### 3.5. Renting Cost Metric

By assuming that each node in the network had a renting rate of  $r_i$ , and assuming that the  $n_i$  node needed to operate for  $t_{ij}$  in order to execute task  $j$ , then the total renting cost of the node was Equation (10)

$$RC_{ij} = r_i t_{ij} \quad (10)$$

However, to minimize the renting cost of all nodes to execute all tasks, we used the Equation (11)

$$RC = \sum_{i,j} RC_{ij} \quad (11)$$

### 3.6. Optimisation Objective Functions

The solution was optimised by assigning specific tasks to specific nodes as defined by the matrix  $X = \{x_{ij}\} \in \{0, 1\}^{n \times m}$ . Therefore, the problem was a binary optimisation problem. The objective function was described by the five following Equations (12)–(16)

$$f_1 = \sum_{i,j} E_{ij} \quad (12)$$

$$f_2 = \sum_{i=1}^n \frac{(E_i - \bar{E})^2}{n-1} \quad (13)$$

$$f_3 = T \quad (14)$$

$$f_4 = RC \quad (15)$$

s.t

$$k = \max\left(\frac{P_i}{P_0}, \frac{L_i}{L_0}, k_0\right) \quad (16)$$

and the connectivity limitation of the dependent tasks.

### 3.7. Transfer Function Model

Transfer function was used to convert particle swarm searching to binary to solve the described problem. Assuming that the  $i$  particle had a velocity of  $v_{i,d}^t$  for the dimension  $d$  and the iteration  $t$ , the corresponding particle bit changed its value with a probability of  $TF(v_{i,d}^t) = \frac{1}{1 + \exp(-v_{i,d}^t)}$ . We used this probability to generate the random numbers  $r \in [0, 1]$  and compared it with  $TF(v_{i,d}^t)$ . If the value was lower than  $TF(v_{i,d}^t)$ , then it changed its bit value, otherwise it did not change. After converting the approaches of binary space, we added the letter B to indicate the binary space in the method name.

### 3.8. Crowding Distance

The concept of crowd distance was first proposed by [12] in NSGA-II. It measured the density of the solutions in the space with respect to the objectives. The purpose of using



this concept was to add a solutions selection criterion when they were non-dominated. Basically, solutions located in less crowded areas or that had bigger crowd distances were favoured. Incorporating this criterion in PSO provided higher exploration in the solution space. The pseudo-code for determining crowd distance is provided in Algorithm 1.

---

**Algorithm 1.** The algorithm for calculating crowd distance

---

**Input**

Pareto front // set of non-dominated solutions

**Output**

CD // crowd distance

**Start**

N = size of Pareto front

Sorted Pareto Front = sort(Pareto front)

initiate CD of size N-2 with zeros

for each solution in the Sorted Pareto Front

    find the distance from the previous objective

    find the distance from the next objective

        calculate the distance of the subject solution CD(i) as the summation

        of both the distance from the previous objective and the next objective

    endfor

**End**

---

### 3.9. Developed DAMP Model

The distance angle multi-objective particle swarm optimisation (DAMP) model developed by this study is explained in this section. A directionally aware MOO differs from traditional MOO by exploring power that aims at probabilistic spreading of the solutions in the space according to their crowding distance and directions. Contrary to a study by [13], which determined combined crowding distance with direction in the search and allocated more priority to direction, DAMP probability performs the search while allocating equal weight or selection probability to both direction and distance. The pseudo-code for DAMP is provided in Algorithm 2.

---

**Algorithm 2.** The algorithm for distance angle multi-objective particle swarm optimisation (DAMP).

---

**Inputs**

f1,f2, ... fm //set of objectives

gmax //maximum number of generations

sizeOfSwarm // size of solutions of swarm

Vmax //maximum velocity

Vmin //minimum velocity

personal best //interial, coefficient of moving toward best  
and coefficient of moving toward best global

angleRes //angle resolution

**Output**

PF //pareto front

gmax //maximum number of iterations

**Start**

Initialize swarm

Evaluate(swarm,f1,f2, ... fm)

g = 0

While g < gmax

    newSwarm=[]

    leaders =Select(swarm)

    For each particle until sizeOfSwarm

        newParticle=Update Position (particle,leaders,w,c1,c2,Vmin,Vmax)

        particle=Mutation(particle)

        add particle to newSwarm

---

---

```

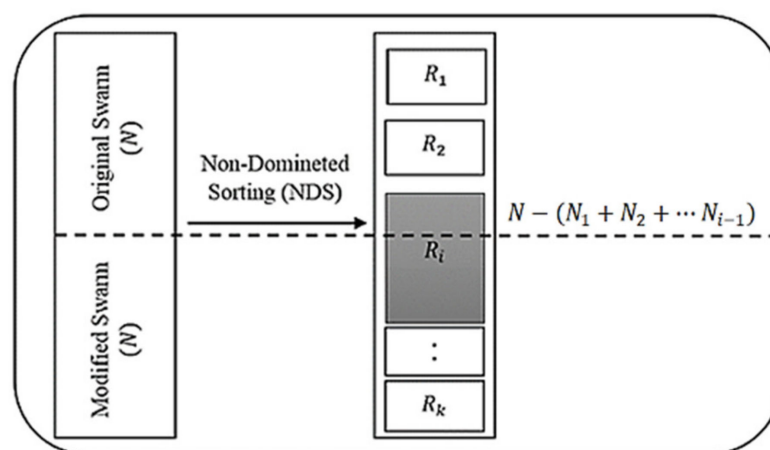
EndFor
Repository=Combine(swarm,newSwarm)
Swarm=Select(Repository,angleRes)
Evaluate(swarm,f1,f2,... fm)
g++
EndWhile
PF=ParetoFront(swarm)
End

```

---

To ensure the progress of the search, the algorithm selected the best solution out of a combined pool of solutions from both the original swarm and the swarm after mobility and mutation. The selection was performed via non-dominated sorting and sorted using non-domination criterion  $\prec$ . The solutions were then ranked accordingly, where  $x, y$  belonged to the same rank, followed by  $x \prec y$  and  $y \prec x$ .

Assuming that the solutions were sorted within  $k$  ranks, as shown in Figure 1, solutions ranked in  $R_1$  were the most optimal solutions and dominated over the subsequent solutions. Solutions in rank  $R_2$  were the second most optimal and, while they dominated over other ranks, they were dominated by solutions in rank  $R_1$ . Additionally, if  $\forall x, y \in R_i$ , then  $x \prec y$  and  $y \prec x$ . The objective was to select  $N$  solutions from the sorted solutions.



**Figure 1.** Conceptual diagram for selecting best  $N$  solution from  $2N$  pool.

As shown in the pseudocode in Algorithm 3, the selection algorithm began by selecting solutions from the most optimal ranks  $R_1, R_2, \dots, R_i$ , where the following equations applied:

$$R_1 + R_2 + \dots + R_{i-1} < N, \quad (17)$$

$$R_1 + R_2 + \dots + R_i > N \quad (18)$$

$N - (N_1 + N_2 + \dots + N_{i-1})$  was then selected from  $N_k$  solutions and was consistent with the exploration. The remaining solutions, up to  $N$ , were selected in a way that was consistent with exploration. This method considered two criteria—the angle distribution and the crowding distance distribution. It started by sorting the solutions from the highest to the lowest crowding distance and the highest to the lowest angle range. Angle range rank was defined as the number of selected solutions within an angular sector in the solution space. The first and the second sets of sorted solutions were assigned to sorted solutions distance and sorted solutions angle, respectively. It then underwent an iterative process from 1 until  $N - (N_1 + N_2 + \dots + N_{i-1})$  and generated a random number ( $r$ ) in each iteration. If the generated number was between 0 and 0.5, a solution was selected from sorted solutions distance. Otherwise, it was selected from sorted solutions angle. This ensured a balance between both angle and crowding distance explorations. The pseudo-code is provided in Algorithm 4. For DAMP, the solutions selection probability combined two criteria,

crowding distance and angle range rank. To demonstrate the concept, we assumed that we had six non-dominated particles, as shown in Figure 2a–c, which shows the corresponding angular range ranks and the crowding distance arrays, respectively. Using the probabilistic calculation, a probability of  $0.5/2$  was assigned to solutions three and four and then to solutions one and two. An overall probability of  $1/4$  was assigned to all four solutions.

---

**Algorithm 3.** The algorithm for selecting N solutions out of 2N pool.

---

**Input**  
 original swarm  
 modified swarm  
 N // sizeOfSwarm  
**Output**  
 selectedSolutions  
**Start**  
 poolSolutions=combine(original swarm,modified swarm )  
 SortedSolutions=nonDominatedSorting(poolSolutions)  
 for each rank k  
 lk=length(rank)  
 cumulativeLength=0;  
 if(cumulativeLength<N)  
 add rank k to selectedSolutions  
 cumulativeLength=cumulativeLength+lk  
 else  
 RemainingSolution=select(poolSolutions,N-cumulativeLength)  
 add to selectedSolutions  
 end  
**End**

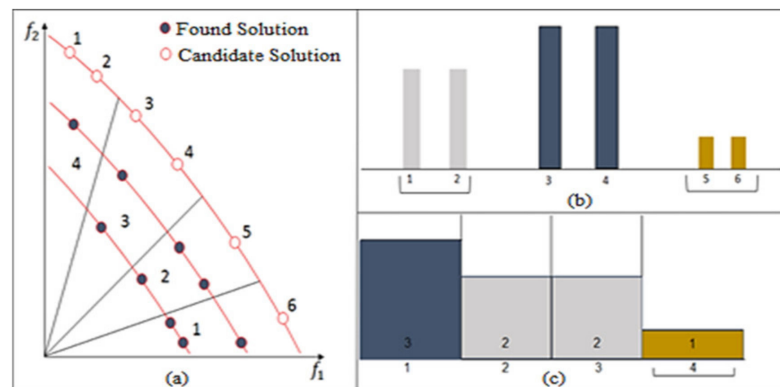
---

**Algorithm 4.** The algorithm for selecting RS solutions out of a pool of non-dominated solutions using crowding distance and angle range rank.

---

**Input**  
 poolSolutions // repository  
 RS //remaining to reach the size of swarm  
 foundSolutions  
 selectedSolutions  
**Output**  
 selectedSolutions  
**Start**  
 center = generateCrowdCenter(foundSolutions)  
 sortedSolutionsDistance=sortingDistance(poolSolutions,center)  
 anglesRangesRank=generateAngleRangeRank(foundSolutions)  
 sortedSolutionsAngles=sortingAngle(foundSolutions)  
 for i = 1 until RS  
 r = generateRandom(0,1)  
 if (r < 0.5)  
 add sortedSolutionsDistance(1) to selectedSolutions  
 delete sortedSolutionsDistance(1)  
 else  
 add sortedSolutionsAngle(1) to selectedSolutions  
 delete sortedSolutionsAngle(1)  
 end  
 end  
**End**

---



**Figure 2.** Illustration of candidate solutions selection using DAMP according to the following criteria: (a) objective space, (b) crowding distance curve, and (c) angle range rank curve.

### 3.10. Big O Notation

We present the complexity analysis of four variants of MP. Assuming that we had for our meta-heuristic searching  $N$  particle,  $M$  iterations, the particle length  $d$ , the number of objectives  $m$ , then the complexity of the searching was given as

$$O(M\{N[O(\text{mobilityEquation}) + O(\text{Evaluation})] + O(\text{sorting})\})$$

$O(\text{Evaluation})$  was related to the tested mathematical function

$$O(\text{mobilityEquation}) = d$$

$$O(\text{sorting}) = mN^2$$

$$O(M\{N[O(\text{mobilityEquation}) + O(\text{Evaluation})] + O(\text{sorting})\}) = O(MNO(\text{Evaluation}) + MNd + MmN^2)$$

We observed that the only difference between DAMP, DMP, AMP, and MP was the sorting part where the algorithm had to sort the solutions based on their angle as well as their distance. The sorting part for the three algorithms was the same, which was  $O(mN^2)$ , because the big O notation of summation to two functions was the maximum big O notation of them.

$$f_1 = O(g_1) \text{ and } f_2 = O(g_2) \Rightarrow f_1 + f_2 = O(\max(g_1, g_2)) \quad (19)$$

### 3.11. Evaluation

The MOO performance measures used to evaluate our proposed method and a comparison between our method and the benchmarking MOO mathematical functions are provided in this sub-section.

#### 3.11.1. C-Metric Measure

C-metric, or set coverage, compared the two Pareto fronts of two approaches in terms of domination. If we had both approach A and approach B, the Pareto front generated from approach A was labelled  $P_A$ , and the Pareto front generated from approach B was labelled  $P_B$ . C-metric  $C(A, B) = C_B$  indicated the number of solutions from B that were dominated by solutions in A. The lower the  $C_B$  value was, the better the performance was. Therefore, the objective was to develop an approach with the lowest  $C_B$  value. The formula for this measure is expressed in Equation (20):

$$C(P_A, P_B) = \frac{|\{y \in P_B | \exists x \in P_A : x \succ y\}|}{|P_B|} \quad (20)$$

### 3.11.2. Hyper-Volume Measure

This measure was a simultaneous indicator of diversity and domination. It was defined as the correlation between the hypercube, the diagonal distance between the solutions in the Pareto front, and the worst set in terms of domination. Therefore, the higher the hyper-volume volume contributed to a better quality of the solutions. The formula for this measure is expressed:

$$HV = volume\left(\bigcup_{x \in P_S} HyperCube(x)\right) \quad (21)$$

### 3.11.3. Delta Measure

This measure was a simultaneous indicator of the uniformity of the Pareto fronts' distributions and spread. Therefore, it was a measure of diversity. It was denoted by  $\Delta$  and needed to be minimal.

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}} \quad (22)$$

where:

$d_f$  and  $d_l$  were the Euclidean distances between the extreme solutions and the boundary solution,

$d_i$  was distances where  $i = 1, 2, \dots, N - 1$ ,

$\bar{d}$  was the average of all the consecutive distances  $d_i$  for  $i = 1, 2, \dots, N - 1$ .

The pseudocode for calculating this measure is provided in Algorithm 5, [12].

---

**Algorithm 5.** The algorithm for calculating the delta measure

---

**Input**

d //Pareto front

dT //True Pareto Front

Output:

$\Delta$  //Delta Measure

**Start**

Sort the Pareto set

Calculate the Euclidean distance between consecutive solution and assign them to matrix M

Calculate the average of matrix M

Fit the curve of the true Pareto front and calculate the distance between the two extreme solutions

calculate the distance between the two extreme solutions

Apply equation and find  $\Delta$

**End**

---

### 3.11.4. Generational Distance (GD)

This measure was an indicator of the optimality of the solutions in terms of their closeness to the true Pareto front solutions. It measured the average distances between the Pareto front solutions and the true Pareto front solutions. Therefore, the lower the GD value was, the more optimal a solution there was [44].

$$GD(P_S, P_T) = \frac{\left(\sum_{i=1}^{|P_S|} d_i^2\right)^{\frac{1}{2}}}{|P_S|} \quad (23)$$

where:

$|P_S|$  was the number of solutions in the Pareto set,

$P_T$  was the true Pareto front,

$d_i$  was the Euclidean distance between the solutions in  $P_S$  and the nearest solutions in  $P_T$ .

### 3.11.5. Number of Non-Dominated Solutions

This measure was an indicator of MOO algorithm performance in terms of the number of non-dominated solutions (NDS) in the Pareto front. Therefore, the higher the NDS values were, the higher the performance was [45].

$$NDS(N) = |P_S| \quad (24)$$

### 3.11.6. Mathematical Benchmarking Functions

Schaffer (SCH), Fonseca-Fleming (FON), Poloni (POL), Kursawe (KUR), and Zitzler-Deb-Thiele (ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6) functions were used [12]. The mathematical formulas of these functions with optimal solutions and the type of the functions are provided in Table 3.

**Table 3.** Benchmark multi-objective optimisation (MOO) functions.

Problem	N	Variables Bounds	Objectives functions	Optimal Solutions	Comments
FON	3	$[-4, 4]$	$f_1(x) = 1 - \exp\left(-\sum_{i=1}^3 \left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_2(x) = 1 - \exp\left(-\sum_{i=1}^3 \left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$	$x_1 = x_2 = x_3$	Non-convex
KUR	3	$[-5, 5]$	$f_1(x) = \sum_{i=1}^{n-1} \left(-10 \exp\left(-0.2 \sqrt{x_i^2 + x_{i+1}^2}\right)\right)$ $f_2(x) = \sum_{i=1}^n ( x_i ^{0.8} + 5 \sin x_i^3)$	Refer to [12]	Non-convex
POL	2	$[-\pi, \pi]$	$f_1(x) = [1 + (A_1 - B_1)^2 + (A_2 - B_2)^2]$ $f_2(x) = [(x_1 + 3)^2 + (x_2 + 1)^2]$ $A_1 = 0.5 \sin 1 - 2 \cos 1 + \sin 2 - 1.5 \cos 2$ $A_2 = 1.5 \sin 1 - \cos 1 + 2 \sin 2 - 5.0 \cos 2$ $B_1 = 0.5 \sin x_1 - 2 \cos x_1 + \sin x_2 - 1.5 \cos x_2$ $B_2 = 1.5 \sin x_1 - \cos x_1 + 2 \sin x_2 - 5.0 \cos x_2$	Refer to [12]	Non-convex, Disconnected
SCH	1	$[-10^3, 10^3]$	$f_1(x) = x^2$ $f_2(x) = (x - 2)^2$	$x \in [0, 2]$	Convex
ZDT1	30	$[0, 1]$	$f_1(x) = x_1$ $f_2(x) = g(x) \left[1 - \sqrt{\frac{x_1}{g(x)}}\right]$ $g(x) = 1 + 9 \left(\frac{\sum_{i=2}^n x_i}{n-1}\right)$	$x_1 \in [0, 1]$ $x_1 \in [0, 1]$ $x_1 \in [0, 1]$	Convex
ZDT2	30	$[0, 1]$	$f_1(x) = x_1$ $f_2(x) = g(x) \left[1 - (x_1 / g(x))^2\right]$ $g(x) = 1 + 9 \left(\frac{\sum_{i=2}^n x_i}{n-1}\right)$	$x_1 \in [0, 1]$ $x_i = 0$ $i = 2, 3, \dots, n$	Non-convex
ZDT3	30	$[0, 1]$	$f_1(x) = x_1$ $f_2(x) = g(x) \left[1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)} \sin(10\pi x_1)\right]$ $g(x) = 1 + 9 \left(\frac{\sum_{i=2}^n x_i}{n-1}\right)$	$x_1 \in [0, 1]$ $x_i = 0$ $i = 2, 3, \dots, n$	Convex, Disconnected

Table 3. Cont.

Problem	N	Variables Bounds	Objectives functions	Optimal Solutions	Comments
ZDT4	10	$x_1 \in [0, 1]$ $x_i \in [-5, 5], i = 2, \dots, n$	$f_1(x) = x_1$ $f_2(x) = g(x) \left[ 1 - \sqrt{\frac{x_1}{g(x)}} \right]$ $g(x) = 1 + 10(n-1) + \sum_{i=1}^n [x_i^2 - 10 \cos(4\pi x_i)]$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, 3, \dots, n$	Non-convex
ZDT6	10	[0, 1]	$f_1(x) = 1 - \exp(-4x_1) \sin^6(6\pi x_1)$ $f_2(x) = g(x) \left[ 1 - \left( \frac{f_1(x)}{g(x)} \right)^2 \right]$ $g(x) = 1 + 9 \left[ \left( \sum_{i=2}^n x_i \right) / (n-1) \right]^{0.25}$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, 3, \dots, n$	Convex, Non-uniformly Spaced

SCH: Schaffer; FON: Fonseca-Fleming; POL: Poloni; KUR: Kursawe; ZDT: Zitzler-Deb-Thiele.

#### 4. Evaluation and Results

DAMP, DMP, and AMP were evaluated using MATLAB 2019b and other benchmark MOPSOs [9], as shown in Table 4. We used the same common parameters of the proposed methods and the benchmarks, and we used the same objective for comparison. Furthermore, each experiment was repeated for 10 runs, changing the seed of the random number generator. The parameters that were selected were based on tuning processes for  $c_1$  and  $c_2$  that represent the coefficients of the effect of personal best and global best respectively. They were selected to be 1/3 and 2/3, respectively. We also set the parameters of  $w$ ,  $V_{max}$ , and  $V_{min}$  to be 0.5, 0.1, and 0.001, respectively. These parameters were related to the original equation of particles mobility of PSO that are presented in Equations (25)–(28).

$$V_{id}(t+1) = \omega V_{id}(t) + C_1 r_{1d} (P_i - X_{id}) + C_2 r_{2d} (P_{gd} - X_{id}), \quad (25)$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (26)$$

$$\text{If } V_{id} > V_{max}, \text{ then } V_{id} = V_{max} \quad (27)$$

$$\text{If } V_{id} < V_{min}, \text{ then } V_{id} = V_{min} \quad (28)$$

Table 4. Parameters used in the evaluations and their values.

Parameter Name	Value
numberOfParticles	50
numberOfIterations	100
$c_1$	1/3
$c_2$	2/3
nRep	100
$w$	0.5
$V_{max}$	0.1
$V_{min}$	0.001

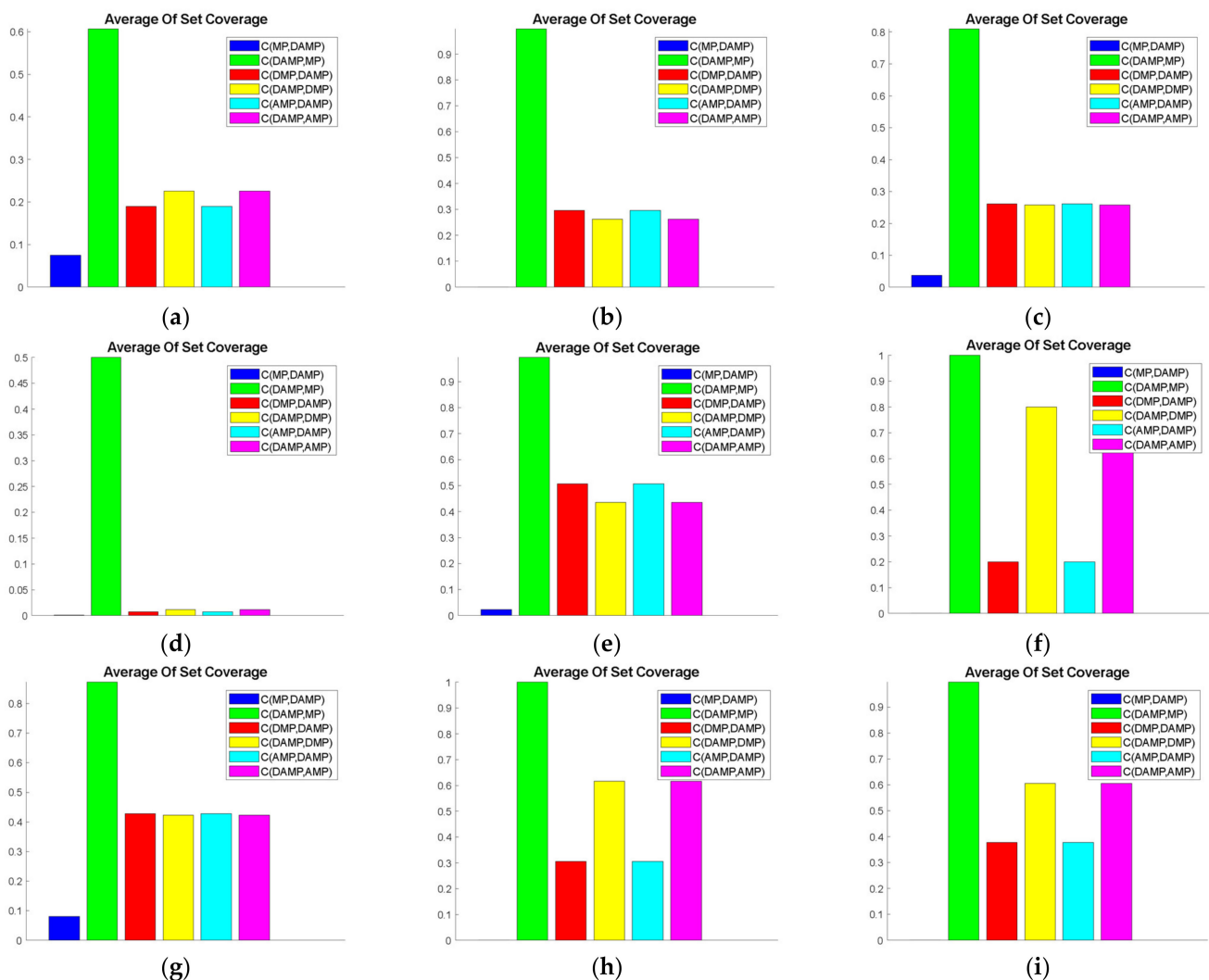
##### 4.1. Evaluation By Mathematical Functions

This section evaluates MOO with the MOO mathematical functions mentioned in the sub-section above. The results are presented in the following Section 4.1.1 Set Coverage Analysis, Section 4.1.2 Hyper-volume, Section 4.1.3 Number of Non-Dominated Solutions (NDS), Section 4.1.4 Delta Measure, Section 4.1.5 Generational Distance, and Section 4.1.6 Statistical Evaluation.

#### 4.1.1. Set Coverage Analysis

The role of set coverage was to judge the superiority of the developed DAMP model over industry benchmarks, such as MP and DMP, and intermediate variant models, such as AMP. Therefore, we attempted to compare  $C(DAMP, x)$  values.

With  $C(x, DAMP)$ ,  $x$  could represent any one of the three compared models. Figure 3 shows the obvious superiority of the DAMP model over the C-metric values of all the other mathematical functions. However, a difference existed in domination performance between the models. For instance, the DAMP model had the most domination over the MP model compared to the other two models. We also observed similar domination performance between AMP and DMP models. We also noticed that DAMP had higher  $C(DAMP, x)$  values when evaluated using ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, and FON. On the other hand, we found that the  $C(DAMP, x)$  values were nearly identical when evaluated using POL, and that the  $C(x, DAMP)$  values were slightly lower when evaluated using KUR when  $x = AMP, DMP$ .



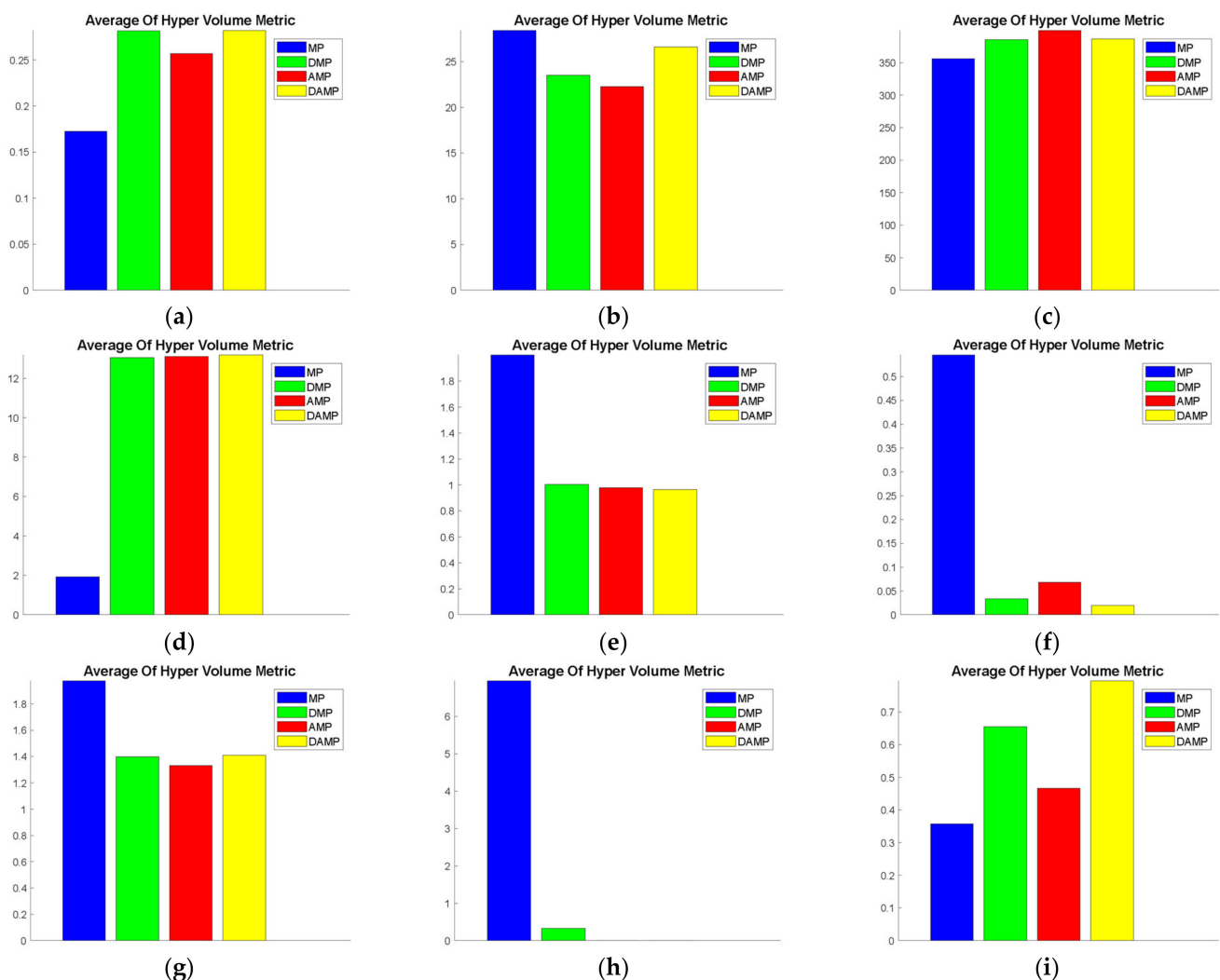
**Figure 3.** Average set coverage values of the DAMP model compared to MP, AMP, and DMP models: (a) FON; (b) KUR; (c) POL; (d) SCH; (e) ZDT1; (f) ZDT2; (g) ZDT3; (h) ZDT4; (i) ZDT6.

Therefore, the overall performance of the DAMP model was clearly superior in comparison to all other models in terms of domination.



#### 4.1.2. Hyper-Volume

This function measure judged the diversity of the developed solutions. While it was regarded as a secondary measure, after a set coverage, where domination was more important than diversity, judging diversity helped identify the overall performance of the algorithm. As shown in Figure 4, we observed that the MP model provided higher hyper-volume values than the other models when evaluated using ZDT1, ZDT3, ZDT4, and KUR. Furthermore, Table 5 shows the statistical significance of hyper-volume values for ZDT1, ZDT4, ZDT6, SCH, and FON based on *t*-test values. Cross-references of these visualisation results with statistical results are shown in Table 5 where the superiority of the DAMP model in terms of hyper-volume value is statistically shown for FON, SCH, and ZDT6. Conversely, the hyper-volume value of the MP model was superior to the DAMP when evaluated using ZDT1 and ZDT4. Nevertheless, for all these functions, the MP model had relatively low domination, as seen in the set coverage analysis sub-section. We also found that the DAMP model had higher hyper-volume values than the MP model when evaluated using SCH. We observed that the hyper-volume performance, when evaluated using KUR, was similar to the MP model and superior to the AMP and the DMP models. These findings support the quality of diversity of the DAMP model solutions apart from the observed domination in the previous sub-section.



**Figure 4.** Average hyper-volume values of the DAMP model compared to MP, AMP, and DMP models: (a) FON; (b) KUR; (c) POL; (d) SCH; (e) ZDT1; (f) ZDT2; (g) ZDT3; (h) ZDT4; (i) ZDT6.

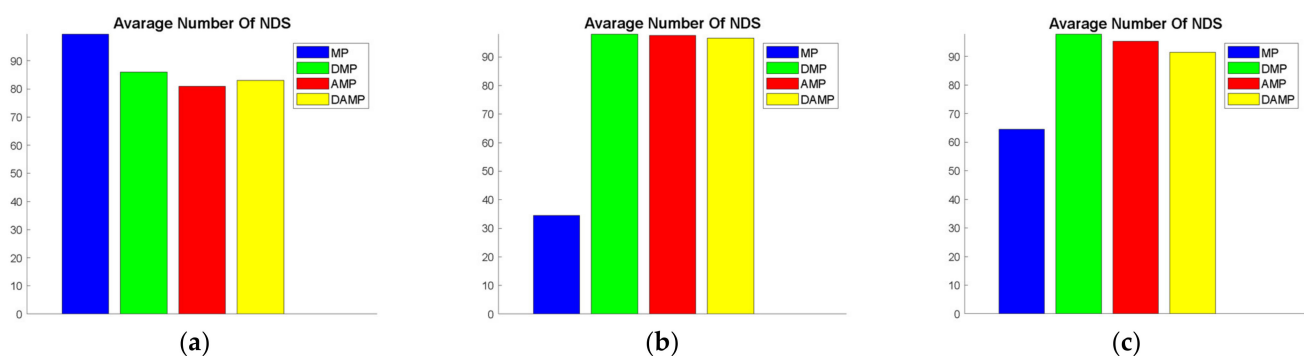
**Table 5.** T-values of the DAMP model compared to MP, AMP, and DMP models.

Function Test	T-Test	Measure			
		Delta	Hyper volume	NDS	GD
FON	DAMP/AMP	0.315850996	0.307418865	0.58492454	0.622265942
	DAMP/DMP	0.719826015	0.995765743	0.49363686	0.652045003
	DAMP/MP	$5.0242 \times 10^{-5}$	0.000207451	$7.39 \times 10^{-5}$	0.049132833
KU	DAMP/AMP	0.316571234	0.136064963	0.54369604	0.573461561
	DAMP/DMP	0.012042727	0.269537335	0.51557762	0.396190518
	DAMP/MP	0.90710705	0.80889372	$7.7633 \times 10^{-7}$	0.002574095
POL	DAMP/AMP	0.164387112	0.209533066	0.07679627	0.302064115
	DAMP/DMP	0.351181697	0.871518999	$2.89 \times 10^{-2}$	0.393267196
	DAMP/MP	0.351181697	0.871518999	0.0206565	0.593382272
SCH	DAMP/AMP	0.304664635	0.383026813	0.60095345	0.533349004
	DAMP/DMP	0.480290477	0.06290748	0.351256638	0.085563604
	DAMP/MP	0.001485655	$1.4044 \times 10^{-5}$	0.05283945	0.142239574
ZDT1	DAMP/AMP	0.142123082	0.904186015	0.05720644	0.636188124
	DAMP/DMP	0.001343439	0.740246281	0.26176837	0.209288404
	DAMP/MP	0.016179938	0.000552411	0.26176837	0.209288404
ZDT2	DAMP/AMP	0.096184093	0.123852474	0.19186711	0.575465083
	DAMP/DMP	0.350353147	0.52239308	0.0815855	0.1455833
	DAMP/MP	0.041056651	0.52239308	0.616373095	0.691184369
ZDT3	DAMP/AMP	0.728433661	0.751269882	0.14275619	0.872306061
	DAMP/DMP	0.000428487	0.948368027	0.106679999	0.206603271
	DAMP/MP	0.592841876	0.948368027	$1.8854 \times 10^{-6}$	$3.31535 \times 10^{-5}$
ZDT4	DAMP/AMP	0.939868234	0.906971038	0.92851591	0.178865555
	DAMP/DMP	0.013632197	0.341959448	0.81800295	0.140962795
	DAMP/MP	0.013632197	0.000838889	0.04702054	$8.29888 \times 10^{-7}$
ZDT6	DAMP/AMP	0.534051523	0.06054342	0.09434971	0.058047202
	DAMP/DMP	0.224179011	0.749439813	0.051886	0.099316485
	DAMP/MP	0.224179011	0.005584877	0.00128414	$1.34771 \times 10^{-5}$

GD: generational distance.

#### 4.1.3. Number of Non-Dominated Solutions (NDS)

This measure indicated the number of non-dominated solutions which was an indicator of the choices provided to the decision maker after optimisation. As seen in Figure 5, the NDS of the DAMP model was comparable with the AMP and the DMP models when evaluated using SCH, ZDT1, ZDT3, ZDT4, KUR, and POL. Moreover, the NDS of all the other models was superior and statistically significant to the MP model when it was evaluated using ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, KUR, and POL, as shown in Table 5.

**Figure 5.** Cont.

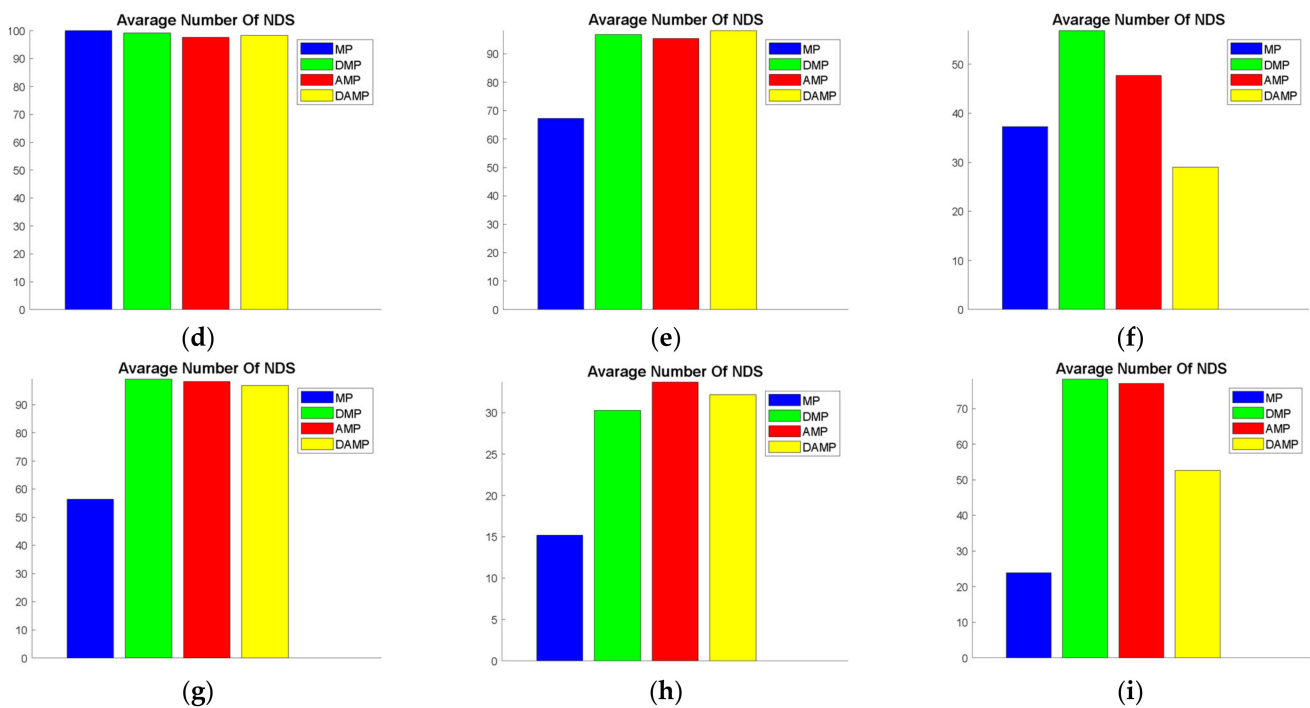


Figure 5. Average non-dominated solutions (NDS) values of the DAMP model compared to MP, AMP, and DMP models: (a) FON; (b) KUR; (c) POL; (d) SCH; (e) ZDT1; (f) ZDT2; (g) ZDT3; (h) ZDT4; (i) ZDT6.

#### 4.1.4. Delta Measure

This measure was regarded as a measure of diversity. An observation of the values revealed that the DAMP model had an average diversity compared to AMP and DMP models when it was evaluated using almost all of the functions, whereas the MP model had higher diversity when evaluated using KUR and less diversity when evaluated using FON. This measure was, again, a secondary measure after the domination of solutions. Cross-referencing of the average delta values is shown in Figure 6, and the statistical findings in Table 5 confirmed that the DAMP model was statistically superior to the MP model when evaluated using FON, whereas the MP model was not superior, with a statistical significance only for KUR.

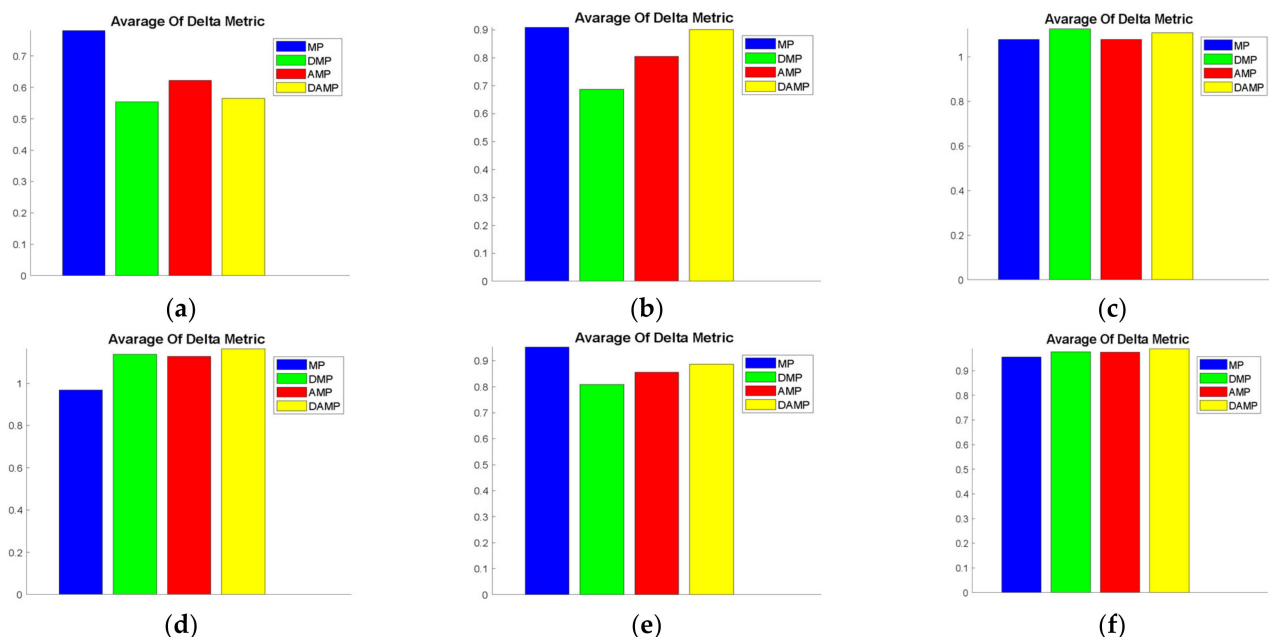
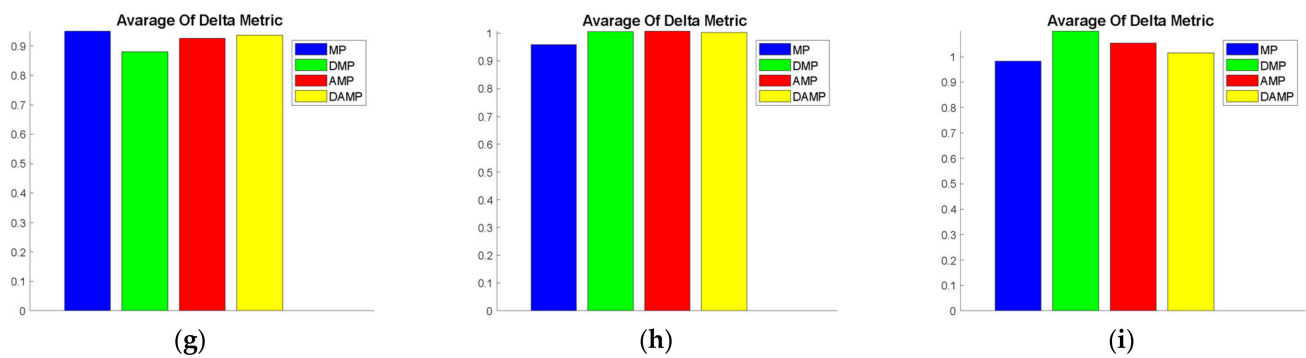


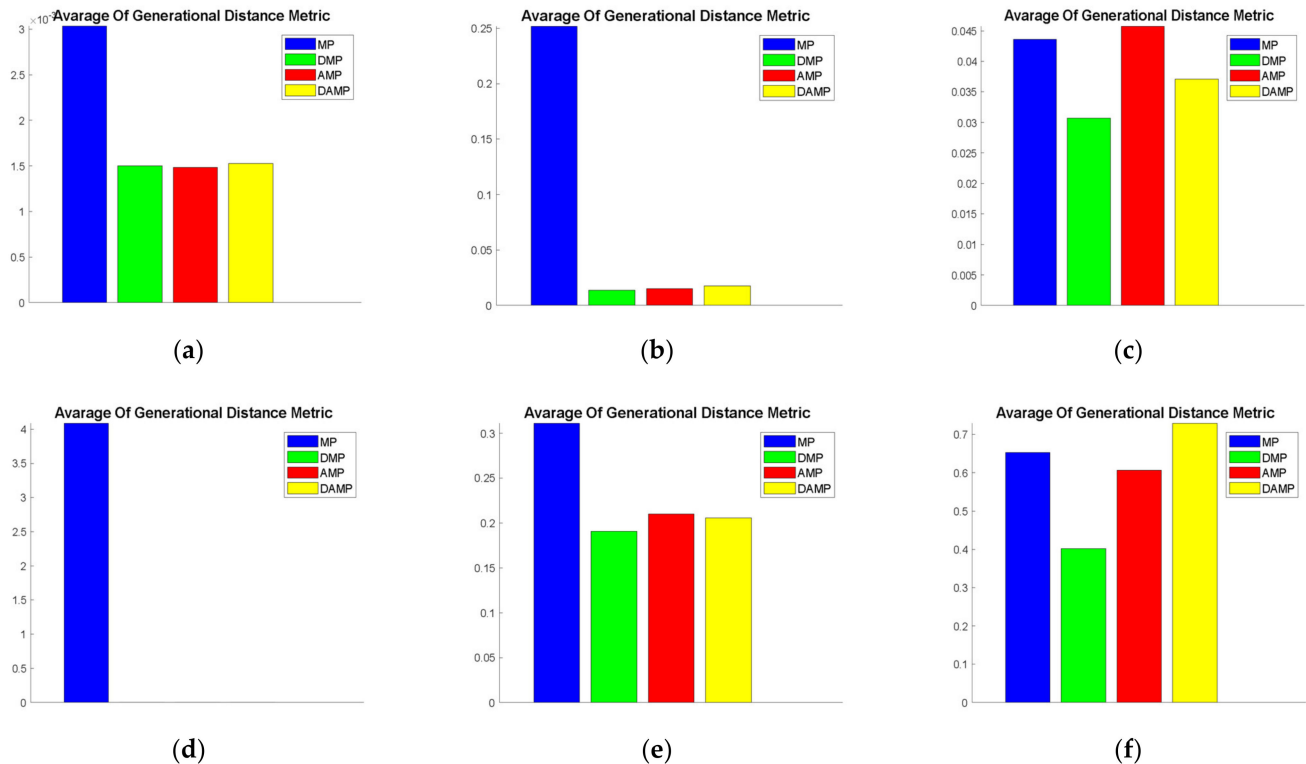
Figure 6. Cont.



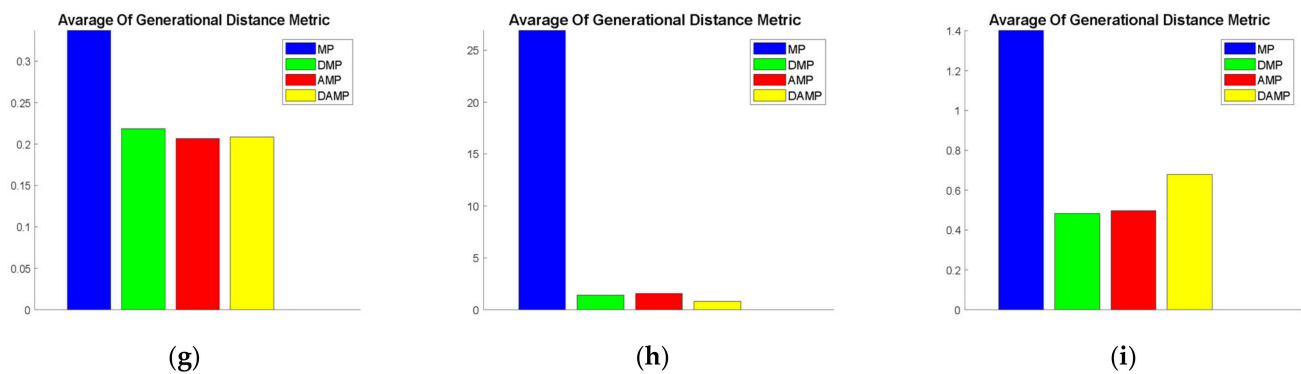
**Figure 6.** Average delta measure values of the DAMP model compared to MP, AMP, and DMP models: (a) FON; (b) KUR; (c) POL; (d) SCH; (e) ZDT1; (f) ZDT2; (g) ZDT3; (h) ZDT4; (i) ZDT6.

#### 4.1.5. Generational Distance

This measure was an indicator of the closeness between the discovered solutions and the true Pareto solutions. It did not provide accurate description of domination as set coverage. However, it did indicate the distance between the discovered solutions and the true Pareto front. In Figure 7, we could see that, for all mathematical functions, DAMP, AMP, and DMP models had similar performance values and were better than the MP model in terms of distance to the true Pareto front. Cross-referencing of these visualisation results with statistical test values are shown in Table 5, and it shows that the DAMP model was statistically significant over the MP model when evaluated using ZDT3, ZDT4, ZDT6, FON, and KUR. Bear in mind that the results of this measure do not reflect domination performance and should, therefore, be read in conjunction with the set coverage results.



**Figure 7.** Cont.



**Figure 7.** Average MOO function values for generational distance of the DAMP model compared to MP, AMP, and DMP models: (a) FON; (b) KUR; (c) POL; (d) SCH; (e) ZDT1; (f) ZDT2; (g) ZDT3; (h) ZDT4; (i) ZDT6.

#### 4.1.6. Statistical Evaluation

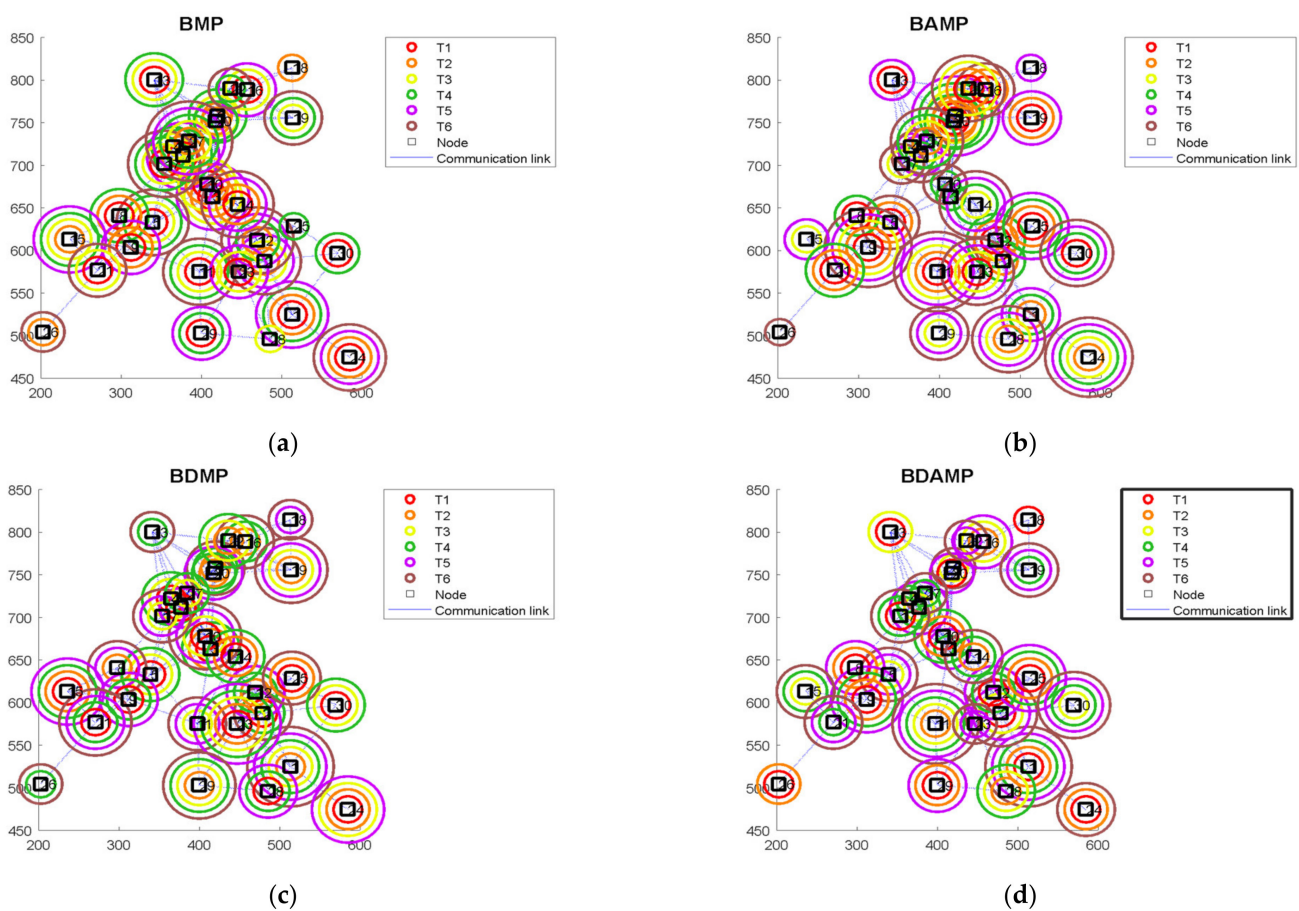
For evaluation purposes, each measure was generated via 10 experiments using random seed for our DAMP model, the AMP model, and the two industry standard DMP and MP models. As seen in Table 5, our DAMP model dominated almost all MOO functions in comparison to the DMP and the MP models. Furthermore, statistical significance also proved the superiority of our DAMP model over the AMP model for some measures, namely, NDS and GD, when evaluated using ZDTs.

#### 4.2. Evaluation Based on Load Balancing Model

This section provides the evaluation of the load balancing in terms of the MOO evaluation metrics. The evaluation was based on number of tasks equal to six and number of nodes in the network equal to 30 nodes. The network is depicted in Figure 8 with the assigned tasks to each of the nodes. This shows that each of the algorithm assigned different tasks to the different nodes where there were different solutions with different performance metrics of the same problem. We also observed that the links of the networks were established to produce connected graphs, which enabled the data exchange while executing the tasks. The parameters and the setting of the evaluation are presented in Table 6.

**Table 6.** Parameters and setting of evaluation.

Parameter Name	Value
number of nodes	6
number of tasks	30
transmission range	100
speed	30 until 100 IPS
power consumption	4 until 10 mW
initial energy	2
Eelc	$50 \times 10^{-6}$ [mj/b]
epsilonAmp	$10 \times 10^{-9}$ mJ/b/m <sup>2</sup>
$P_0$	40 MIPS
$L_0$	50 Byte



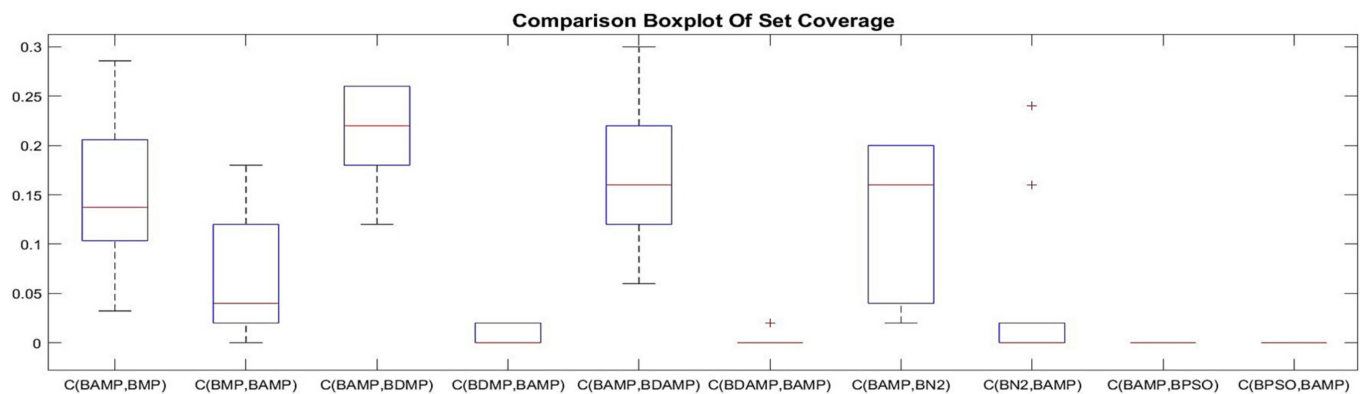
**Figure 8.** The network used to evaluate the task allocation model: (a) binary MP (BMP); (b) BAMP; (c) BDMP; (d) BDAMP.

We added a comparison with two additional algorithms, namely, binary non-dominated sorting genetic algorithm (BN2) and binary particle swarm optimisation (BPSO). We used the same number of solutions and iterations, 50 and 100, respectively. As for objective comparison with the benchmarks, we show the evaluation results in the following Section 4.2.1. Set Coverage, Section 4.2.2. Hyper-Volume, Section 4.2.3 Number of Non-Dominated Solutions, Section 4.2.4 Relative Generational Distance, and Section 4.2.5. Statistical Evaluation.

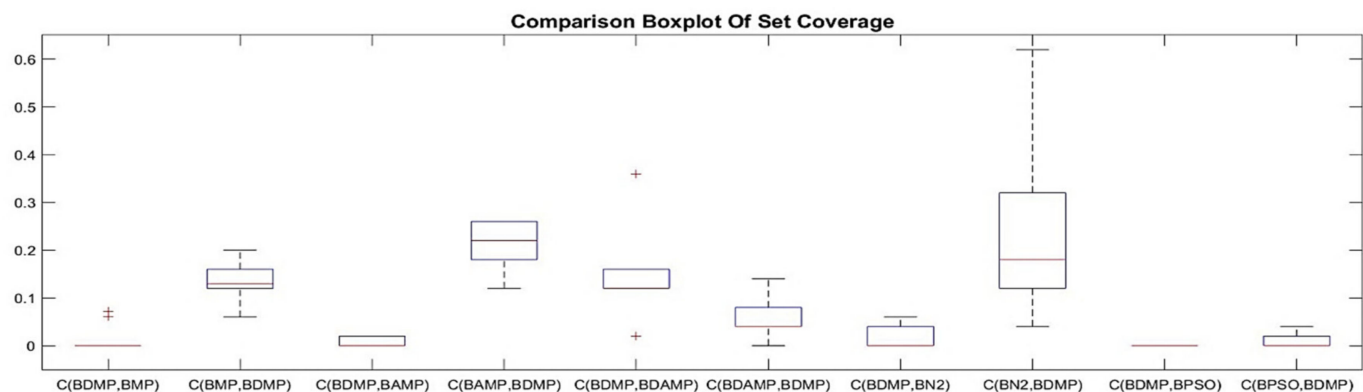
#### 4.2.1. Set Coverage

For objective evaluation, each of the methods was executed 10 times, and its generated set coverage results were compared with the other methods. The results are provided as boxplots depicted in Figures 9–11. As it is observed in Figure 9, BAMP achieved higher set coverage for most approaches, which provided the superiority of using the angle as criterion of searching compared with the other approaches. Another observation was the use of the angle, which provided a wide range of possibilities according to the seed compared with not using the angle or the distance, as it has been shown in  $C(\text{BAMP}, \text{BMP})$  when compared with  $C(\text{BMP}, \text{BAMP})$  where the latter was narrower than the former. In addition, we observed that BAMP dominated BN2 with higher percentage than the domination of BN2 over BAMP. Contrary to the angle, the usage of distance as criterion for exploration had less influence on the domination, as it is shown in Figure 10, where the majority of approaches accomplished more dominance than standalone usage of distance represented by BDMP. Another observation was that the usage of angle and distance was better than using the distance solely. This is observed in Figure 11 because the value of  $C(\text{BDAMP}, \text{BDMP})$  was higher than the value of  $C(\text{BDMP}, \text{BDAMP})$ . However, BPSO generated a non-dominated solution compared with the other benchmark. This solution

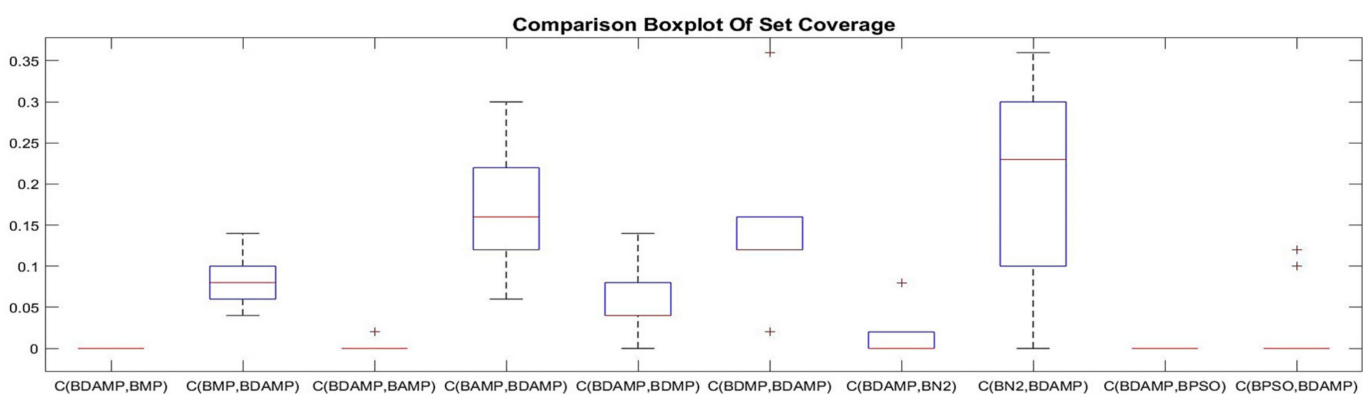
dominated from a single objective over other objectives, namely, the objective of energy distribution, while it was dominated with respect to other objectives.



**Figure 9.** Box plot of the set coverage between the approaches of BAMP with respect to BMP, BDMP, BDAMP, BN2, and BPSO.



**Figure 10.** Boxplot of the set coverage between the approaches of BDMP with respect to BMP, BAMP, BDAMP, BN2, and BPSO.



**Figure 11.** Box plot of the set coverage between the approaches of BDAMP with respect to BMP, BAMP, BDMP, BN2, and BPSO.

#### 4.2.2. Hyper-Volume

In order to evaluate the exploration in the objective space, we generated the hyper-volume. The results of the hyper-volume as boxplots are given in Figure 12. Observing the Figure 12, we noticed that each of the experiments provided different values of hyper-volume. In some experiments, BAMP indicated higher value of hyper-volume (HV), while in others, there was superiority of BDMP. This revealed that the performance of exploration

was highly sensitive to the initial seed. However, BN2 outperformed other approaches of swarm family in the hyper-volume.

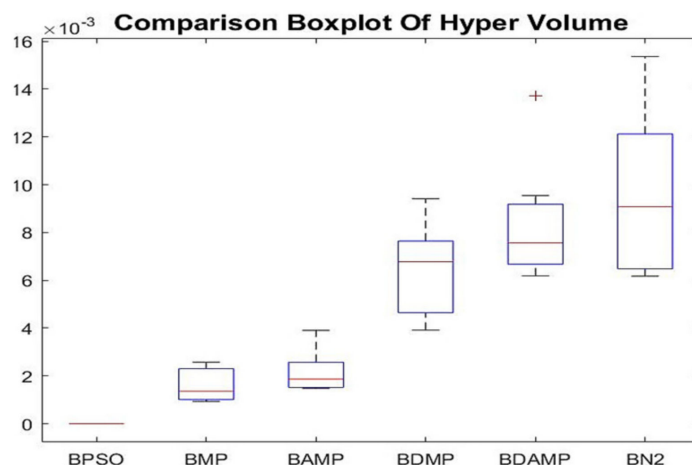


Figure 12. Boxplot of hyper volume results of BAMP and BDAMP compared with benchmarks.

#### 4.2.3. Number of Non-Dominated Solutions

The other metric that was used to evaluate the multi-objective optimization approaches from the perspective of load balancing was the number of non-dominated solutions, which is given in Figure 13. We observed that the approaches had almost the same NDS except for BMP due to the high non-domination reached by searching within each algorithm.

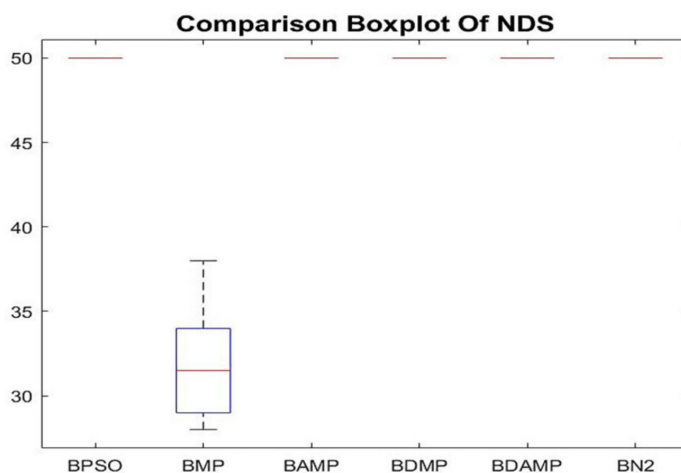
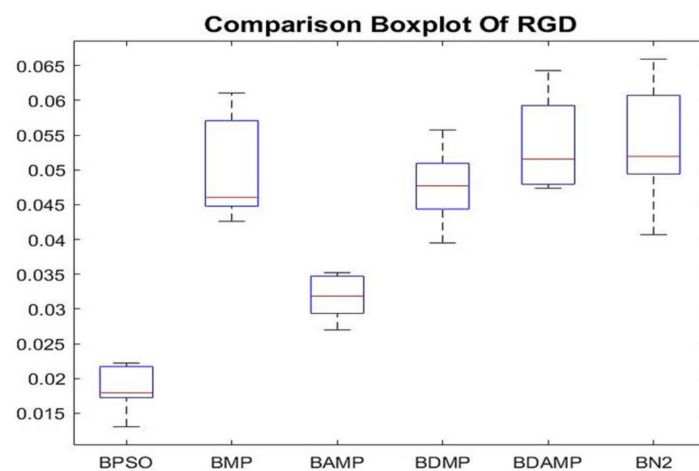


Figure 13. Boxplot of NDS results of BAMP and BDAMP compared with the benchmarks.

#### 4.2.4. Relative Generational Distance

The last metric was the relative generational distance, which needed to be minimized. Figure 14 shows that BAMP and BPSO were the best in terms of minimizing this metric compared with the other benchmark. However, it is important to distinguish between lower Euclidean distance and more domination, which is not always associated.

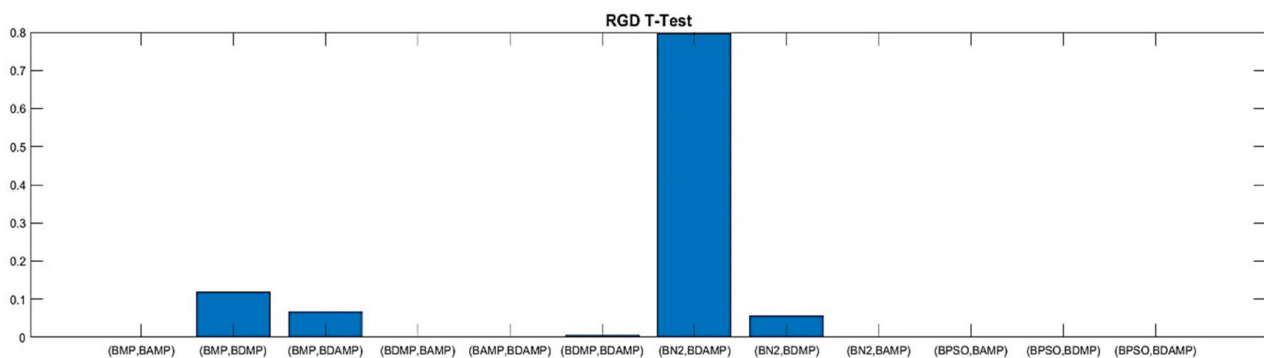




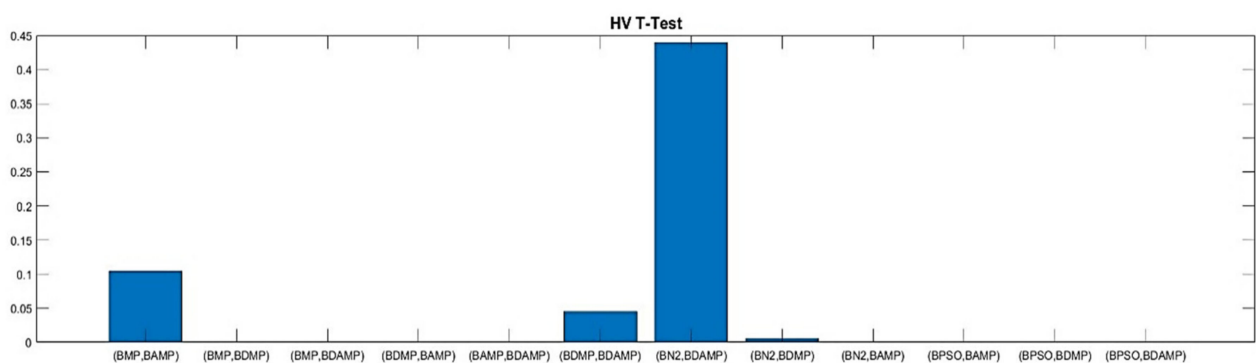
**Figure 14.** Boxplot of relative generational distance (RGD) results of BAMP and BDAMP compared with the benchmarks.

#### 4.2.5. Statistical Evaluation

For thorough evaluation, we conducted a *t*-test to verify the superiority from a statistical perspective. The *t*-test evaluation was conducted based on three metrics, namely, RGD, HV, and NDS. The *t*-test was based on a series of 10 runs, changing the random seed. The evaluation used a confidence level of 0.05 for rejecting the null hypothesis and accepting the statistical significance of the difference between the approaches. As we observe in the Figures 15 and 16, respectively, RGD and HV were the two metrics showing most statistical differences between the approaches, while for NDS, statistical difference was only observed for the comparison with BPSO because it was a single objective optimization and was weak for providing the number of non-dominated solutions, as shown in Figure 17.



**Figure 15.** RGD T-test values between our approach and the benchmarks.



**Figure 16.** HV T-test values between our approach and the benchmarks.

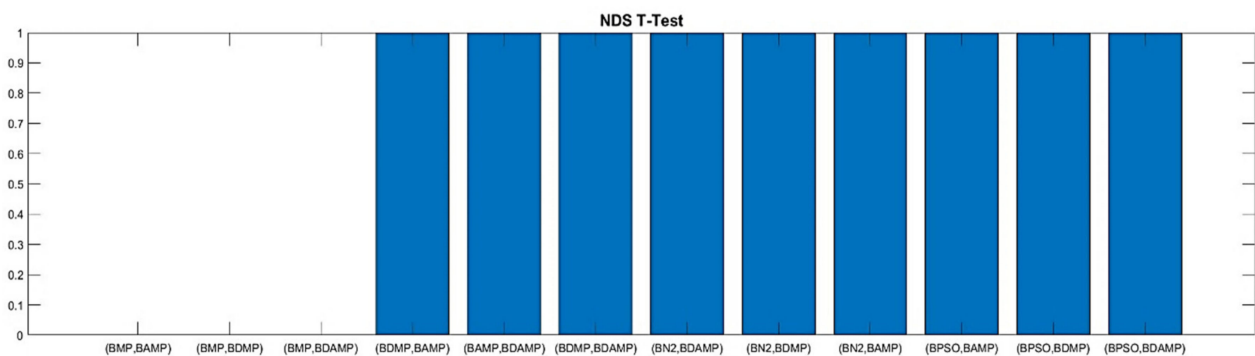


Figure 17. NDS T-test values between our approach and the benchmarks.

## 5. Conclusions and Future Considerations

The study successfully optimised load balancing in software defined networking (SDN) using multi-objective optimisation (MOO) based on particle swarm optimisation (PSO). The industry-benchmark-MP model was expanded to include two additional search criteria, crowding distance and crowding angle. The former provided AMP, the latter provided DMP algorithm, and joining them provided DAMP. In addition, Sigmoid transfer function was then incorporated to convert them to binary, which provided BMP, BAMP, BDMP, and BADMP. The evaluation was decomposed into two phases; the first one was conducted based on benchmarking mathematical functions while the second one was conducted on a developed load balancing SDN model with four objectives: energy (E), energy distribution (D), makespan (T), and renting cost (R). It was found from the evaluation that both AMP and DAMP were superior over DMP and MP in terms of the optimization of the benchmarking mathematical functions. Both BAMP and BDAMP were also superior over BMP and BDMP in terms of the load balancing metrics. Hence, the hypothesis of the superiority of directionality or angle in the optimization was confirmed. Furthermore, it was concluded that using conversion to binary space did not affect the performance of the optimization.

As potential applications for our method, we give edge networks where the users have some applications that need tasks to be executed in real time or with less latency, which requires renting some local nodes for this purpose instead of sending them to the cloud. Other potential applications are for the developed multi-objective optimization, which can be used for various combinatory problems, such as surgeries planning in hospitals [46] and job-shop planning [47].

Several limitations of the approach that can be addressed are as follows. Firstly, it uses fixed angle resolution for dividing the solution space. This might lead to non-stable performance based on the value that is given to the angle. Future studies should explore extending load balancing by adding other objectives, such as node reliability, and incorporating more search criteria in the optimisation algorithm. Another future work is to enable adaptive angle decomposition of the solution space. Secondly, it uses probabilistic selection of non-dominated solutions based on angle or distance in an equal way. Another future work is to make the selection based on adaptive probability of selecting non-dominated solutions. Thirdly, it uses global learning based on moving the particle toward its global best following the conventional mobility equation of PSO. The global learning might lead to premature convergence; a better approach is to use comprehensive learning [48]. Fourthly, we will extend the model to handle dynamical aspects such as running tasks twice on the same node and the cache effect.

**Author Contributions:** Conceptualization, M.H.A. and N.A.Z.; methodology, M.H.A.; validation, M.H.A. and N.A.Z.; formal analysis, M.H.A.; investigation, Z.Z.A.; writing—original draft preparation, M.H.A.; writing—review and editing, M.H.A., N.A.Z., and Z.Z.A.; supervision, N.A.Z., Z.Z.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** The APC was funded by MINISTRY OF HIGHER EDUCATION MALAYSIA, grant number FRGS/2018/FTMK-CACT/F00392.

**Acknowledgments:** The authors would like to thank the Ministry of Higher Education Malaysia and Universiti Teknikal Malaysia Melaka, in providing financial support for this research through the Fundamental Research Grant Scheme (FRGS) with reference number (FRGS/2018/FTMK-CACT/F00392). The authors would also like to thank the UTeM Zamalah Scheme for sponsoring the scholarship.

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

## References

1. Abdelaziz, A.; Fong, A.T.; Gani, A.; Garba, U.; Khan, S.; Akhuzada, A.; Talebian, H.; Choo, K.-K.R. Distributed controller clustering in software defined networks. *PLoS ONE* **2017**, *12*, e0174715. [[CrossRef](#)] [[PubMed](#)]
2. Akbar Neghabi, A.; Jafari Navimipour, N.; Hosseinzadeh, M.; Rezaee, A. Nature-inspired meta-heuristic algorithms for solving the load balancing problem in the software-defined network. *Int. J. Commun. Syst.* **2019**, *32*, e3875. [[CrossRef](#)]
3. Hatamlou, A. Black hole: A new heuristic optimization approach for data clustering. *Inf. Sci.* **2013**, *222*, 175–184. [[CrossRef](#)]
4. Karagul, K.; Sahin, Y.; Aydemir, E.; Oral, A. A simulated annealing algorithm based solution method for a green vehicle routing problem with fuel consumption. In *Lean and Green Supply Chain Management*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 161–187.
5. Hancer, E.; Xue, B.; Zhang, M.; Karaboga, D.; Akay, B. Pareto front feature selection based on artificial bee colony optimization. *Inf. Sci.* **2018**, *422*, 462–479. [[CrossRef](#)]
6. Wang, T.-H.; Wu, H.-C.; Meng, J.-H.; Yan, W.-M. Optimization of a double-layered microchannel heat sink with semi-porous-ribs by multi-objective genetic algorithm. *Int. J. Heat Mass Transf.* **2020**, *149*, 119217. [[CrossRef](#)]
7. Deb, K.; Agrawal, S.; Pratap, A.; Meyarivan, T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *International Conference on Parallel Problem Solving from Nature*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 849–858.
8. Mkaouer, W.; Kessentini, M.; Shaout, A.; Kolighe, P.; Bechikh, S.; Deb, K.; Ouni, A. Many-objective software modularization using NSGA-III. *ACM Trans. Softw. Eng. Methodol.* **2015**, *24*, 1–45. [[CrossRef](#)]
9. Coello, C.A.C.; Lechuga, M.S. MOPSO: A proposal for multiple objective particle swarm optimization. In Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), Honolulu, HI, USA, 12–17 May 2002; Volume 2, pp. 1051–1056.
10. Coello, C.A.C.; Pulido, G.T.; Lechuga, M.S. Handling multiple objectives with particle swarm optimization. *IEEE Trans. Evol. Comput.* **2004**, *8*, 256–279. [[CrossRef](#)]
11. Sierra, M.R.; Coello, C.A.C. Improving PSO-based multi-objective optimization using crowding, mutation and  $\epsilon$ -dominance. In *International Conference on Evolutionary Multi-Criterion Optimization*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 505–519.
12. 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](#)]
13. Metiaf, A.; Wu, Q.; Aljeroudi, Y. Searching With Direction Awareness: Multi-Objective Genetic Algorithm Based on Angle Quantization and Crowding Distance MOGA-AQCD. *IEEE Access* **2019**, *7*, 10196–10207. [[CrossRef](#)]
14. Khan, S.; Gani, A.; Wahab, A.W.A.; Abdelaziz, A.; Ko, K.; Khan, M.K.; Guizani, M. Software-defined network forensics: Motivation, potential locations, requirements, and challenges. *IEEE Netw.* **2016**, *30*, 6–13. [[CrossRef](#)]
15. Hamdan, M.; Hassan, E.; Abdelaziz, A.; Elhigazi, A.; Mohammed, B.; Khan, S.; Vasilakos, A.V.; Marsono, M.N. A comprehensive survey of load balancing techniques in software-defined network. *J. Netw. Comput. Appl.* **2020**, *174*, 102856. [[CrossRef](#)]
16. Chai, R.; Xie, D.; Luo, L.; Chen, Q.; Member, S. Multi-Objective Optimization-Based Virtual Network Embedding Algorithm for Software-Defined Networking. *IEEE Trans. Netw. Serv. Manag.* **2019**. [[CrossRef](#)]
17. Ateya, A.A.; Muthanna, A.; Vybornova, A.; Algarni, A.D.; Abuarqoub, A.; Koucheryavy, Y.; Koucheryavy, A. Chaotic salp swarm algorithm for SDN multi-controller networks. *Eng. Sci. Technol. Int. J.* **2019**, *22*, 1001–1012. [[CrossRef](#)]
18. Bao, N.; Zuo, J.; Zhu, H.; Bao, X. Multi-objective Optimization for SDN Based Resource Selection. In Proceedings of the 18th International Conference on Communication Technology (ICCT), Chongqing, China, 8–11 October 2018; pp. 811–816.
19. Tahaei, H.; Salleh, R.; Khan, S.; Izard, R.; Choo, K.-K.R.; Anuar, N.B. A multi-objective software defined network traffic measurement. *Measurement* **2017**, *95*, 317–327. [[CrossRef](#)]
20. Hamdan, M.; Mohammed, B.; Humayun, U.; Abdelaziz, A.; Khan, S.; Ali, M.A.; Imran, M.; Marsono, M.N. Flow-Aware Elephant Flow Detection for Software-Defined Networks. *IEEE Access* **2020**, *8*, 72585–72597. [[CrossRef](#)]
21. Xue, H.; Kim, K.T.; Youn, H.Y. Dynamic Load Balancing of Software-Defined Networking Based on Genetic-Ant Colony Optimization. *Sensors* **2019**, *19*, 311. [[CrossRef](#)]
22. Jamali, S.; Badirzadeh, A.; Siapoush, M.S. On the use of the genetic programming for balanced load distribution in software-defined networks. *Digit. Commun. Netw.* **2019**, *5*, 288–296. [[CrossRef](#)]
23. Kaur, A.; Kaur, B.; Singh, D. Meta-heuristic based framework for workflow load balancing in cloud environment. *Int. J. Inf. Technol.* **2019**, *11*, 119–125. [[CrossRef](#)]

24. Zhu, R.; Wang, H.; Gao, Y.; Yi, S.; Zhu, F. Energy saving and load balancing for SDN based on multi-objective particle swarm optimization. In *International Conference on Algorithms and Architectures for Parallel Processing*; Springer: Cham, Switzerland, 2015; pp. 176–189.
25. Li, D.; Li, K.; Liang, J.; Ouyang, A. A hybrid particle swarm optimization algorithm for load balancing of MDS on heterogeneous computing systems. *Neurocomputing* **2019**, *330*, 380–393. [[CrossRef](#)]
26. Yu, H.; Wang, Y.; Xiao, S. Multi-objective particle swarm optimization based on cooperative hybrid strategy. *Appl. Intell.* **2020**, *50*, 256–269. [[CrossRef](#)]
27. Rahimi, S.; Abdollahpouri, A.; Moradi, P. A multi-objective particle swarm optimization algorithm for community detection in complex networks. *Swarm Evol. Comput.* **2018**, *39*, 297–309. [[CrossRef](#)]
28. Rashed, A.B.; Hamdan, H.; Sharef, N.M.; Sulaiman, M.N.; Yaakob, R.; Abubakar, M. Multi-objective clustering algorithm using particle swarm optimization with crowding distance (MCP SO-CD). *Int. J. Adv. Intell. Informat.* **2020**, *6*, 72–81. [[CrossRef](#)]
29. Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.M.; Da Fonseca, V.G. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans. Evol. Comput.* **2003**, *7*, 117–132. [[CrossRef](#)]
30. Wei, L.-X.; Li, X.; Fan, R.; Sun, H.; Hu, Z.-Y. A hybrid multiobjective particle swarm optimization algorithm based on R2 indicator. *IEEE Access* **2018**, *6*, 14710–14721. [[CrossRef](#)]
31. Li, X.; Li, X.-L.; Wang, K.; Li, Y. A Multi-Objective Particle Swarm Optimization Algorithm Based on Enhanced Selection. *IEEE Access* **2019**, *7*, 168091–168103. [[CrossRef](#)]
32. Luo, J.; Huang, X.; Yang, Y.; Li, X.; Wang, Z.; Feng, J. A many-objective particle swarm optimizer based on indicator and direction vectors for many-objective optimization. *Inf. Sci.* **2020**, *514*, 166–202. [[CrossRef](#)]
33. Jiang, S.; Yang, S. A strength Pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization. *IEEE Trans. Evol. Comput.* **2017**, *21*, 329–346. [[CrossRef](#)]
34. Feng, Q.; Li, Q.; Chen, P.; Wang, H.; Xue, Z.; Yin, L.; Ge, C. Multiobjective Particle Swarm Optimization Algorithm Based on Adaptive Angle Division. *IEEE Access* **2019**, *7*, 87916–87930. [[CrossRef](#)]
35. Peng, G.; Fang, Y.-W.; Peng, W.-S.; Chai, D.; Xu, Y. Multi-objective particle optimization algorithm based on sharing–learning and dynamic crowding distance. *Optik* **2016**, *127*, 5013–5020. [[CrossRef](#)]
36. Mishra, K.K.; Bisht, H.; Singh, T.; Chang, V. A direction aware particle swarm optimization with sensitive swarm leader. *Big Data Res.* **2018**, *14*, 57–67. [[CrossRef](#)]
37. Mokarram, V.; Banan, M.R. A new PSO-based algorithm for multi-objective optimization with continuous and discrete design variables. *Struct. Multidiscip. Optim.* **2018**, *57*, 509–533. [[CrossRef](#)]
38. Fan, Z.; Wang, T.; Cheng, Z.; Li, G.; Gu, F. An Improved Multiobjective Particle Swarm Optimization Algorithm Using Minimum Distance of Point to Line. *Shock Vib.* **2017**, *2017*, 16. [[CrossRef](#)]
39. Zhang, W.; Li, G.; Zhang, W.; Liang, J.; Yen, G.G. A cluster based PSO with leader updating mechanism and ring-topology for multimodal multi-objective optimization. *Swarm Evol. Comput.* **2019**, *50*, 100569. [[CrossRef](#)]
40. Sedarous, S.; El-Gokhy, S.M.; Sallam, E. Multi-swarm multi-objective optimization based on a hybrid strategy. *Alexandria Eng. J.* **2018**, *57*, 1619–1629. [[CrossRef](#)]
41. Pan, A.; Wang, L.; Guo, W.; Wu, Q. A diversity enhanced multiobjective particle swarm optimization. *Inf. Sci. (Ny)* **2018**, *436*, 441–465. [[CrossRef](#)]
42. Pires, E.J.S.; Machado, J.A.T.; de Moura Oliveira, P.B. Entropy diversity in multi-objective particle swarm optimization. *Entropy* **2013**, *15*, 5475–5491. [[CrossRef](#)]
43. Sun, Y.; Gao, Y.; Shi, X. Chaotic multi-objective particle swarm optimization algorithm incorporating clone immunity. *Mathematics* **2019**, *7*, 146. [[CrossRef](#)]
44. Niu, P.; Ma, Y.; Li, M.; Yan, S.; Li, G. A kind of parameters self-adjusting extreme learning machine. *Neural Process. Lett.* **2016**, *44*, 813–830. [[CrossRef](#)]
45. Zitzler, E.; Thiele, L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Trans. Evol. Comput.* **1999**, *3*, 257–271. [[CrossRef](#)]
46. Marques, I.; Captivo, M.E.; Barros, N. Optimizing the master surgery schedule in a private hospital. *Oper. Res. Heal. Care* **2019**, *20*, 11–24. [[CrossRef](#)]
47. Türkyılmaz, A.; Şenvar, Ö.; Ünal, İ. A research survey: Heuristic approaches for solving multi objective flexible job shop problems. *J. Intell. Manuf.* **2020**, *31*, 1949–1983. [[CrossRef](#)]
48. Liang, J.J.; Suganthan, P.N. Dynamic multi-swarm particle swarm optimizer with a novel constraint-handling mechanism. In *Proceedings of the 2006 IEEE International Conference on Evolutionary Computation*, Vancouver, BC, Canada, 16–21 July 2006; pp. 9–16.