A Performance Evaluating Simulation for PSO Algorithm by Applying Traceroute Feature

Hsing-Chung Chen [1,2,*], Agung Mulyo Widodo [1,3], Bambang Irawan [1,3], Cahya Damarjati [1,4] and Aristophane Nshimiyimana[1]
[1]Dept. of Computer Science and Information Engineering, Asia University, Taiwan, cdma2000@asia.edu.tw, shin8409@ms6.hinet.net, aristophanenshimiyimana@gmail.com
[2]Dept. of Medical Research, China Medical University Hospital, China Medical University, Taiwan.
[3]Department Of Computer Science, Esa Unggul University, West Jakarta, Indonesia, agung.mulyo@esaunggul.ac.id, bambang.irawan@esaunggul.ac.id
[4]Dept. of Information Technology, Universitas Muhammadiyah Yogyakarta, Yogyakarta, Indonesia, cahya.damarjati@umy.ac.id
*Corresponding e-mail addresses: shin8409@ms6.hinet.net, cdma2000@asia.edu.tw

Abstract. The Particle Swarm Optimization (PSO) algorithm for the routing protocol implementing in this paper. This research has been carrying out a series of experiments to observe its performance. We assume a network model with 5 nodes and 6 paths, which is a modified topology in our simulation. The performance of the algorithm evaluated by using the traceroute feature in MikroTik, where the data packet will choose its path. In this experiment, the PSO algorithm compared with Dijkstra algorithm. Finally, the results show that the performance of PSO algorithm is better than Dijkstra algorithm in comparison to both packet throughputs obtained and packet delay.

1 Introduction

The routing work is the process of choosing the best path to reach the destination network or the process of sending packet data from the source to the destination, not on the same network. The routing table which contains all routing information, e.g. routing path [11, 13, 14, 17, 18]. The routing work related to the calculation of costs in forwarding packages in the corresponding network [11, 13, 14, 17, 18]. The metric in a routing table could be represented as the distance, such as the amount of bandwidth used, memory consumption and processor usage, or the time required to forward the delivered packets. Open Shortest Path First (OSPF) routing protocol is a routing protocol that is commonly used in networks. The OSPF router tracks the status of all the different network connections [12] between itself and other networks that the data is trying to send. This makes it a link-state routing protocol. Routers maintain routing table for successful packet delivery from the source node to another destination. This routing protocol is using the Dijkstra Algorithm to determine the lowest cost path. However, this algorithm has a drawback that is caused by a bottleneck so that the packets delivered are accumulated at one node, even though another paths with a high cost have a low density.

The PSO algorithm is a computational algorithm that adopts the behavior of living things that move in groups. The routing protocol which is using PSO will optimize the route path selection problem by moving the particles or potential solutions of the route path using functions for the position and speed of the particles. This paper compares the routing protocol using the PSO algorithm with Dijkstra algorithm.

These research results are shown that the performance comparison [15, 16] of both two algorithms to establish the best path and the intricacy efficient even at the scale of network expansion. PSO algorithm is the best algorithm for to find optimization solution for packet delay and throughput in this experiment with comparing Dijkstra algorithm, it showed the performance PSO algorithm implemented in routing network is better than Dijkstra algorithm.

This paper organized as follows. In Section 1, OSPF routing protocol and the motivation for this research are introduced. Section 2 presents the related works of this research. In this section 3, our research approach explains how to implement the PSO algorithm dealing with the routing protocol work. Section 4 explains the results of the experiment and then carries out the discussions. Finally, we conclude this research in Section 5.

2 Related works

This research is conducted using PSO algorithm for applying tracerouter feature in Mikrotik to evaluate packet delay and throughput. The following related works are recently done by the previously researchers as explained in this section.

Some swarm intelligence algorithms are the PSO algorithm inspired by a flock of birds and the AntNet algorithm inspired by colony of ants. The AntNet algorithm framework has the ability to select techniques that can be used to optimize the solution's search of a variety of different problems, which is based on intuition and the rules through an empirical approach. The AntNet algorithm could be used as an adaptive best-effort routing in the IP network to solve the combinatorial optimization. It has been carried out the research which modifies this algorithm to initialize the routing table and select a node hop to send a packet when one of the links broken. The AntNet algorithm produces throughput and packet delay preferable than OSPF routing protocols [1, 8]. Unlike the AntNet algorithm, the PSO algorithm has three main components including: the particle, the cognitive component and the social component, and the particle speed. The particle represents a solution [9]. A study of the applying of the PSO algorithm to optimize routing protocol has been carried out. This study tries to combine with genetic algorithms to solve invalid looping or backward path problems, and to find the valid path and close to optimal with the fewest looping. Particles choose the explicit location based on *pbest* from the set of *gbest* particles [2, 3]. In another study, the use of Adaptive Mutation Genetic Algorithms (AMG) was used to optimize routing on a network. This study is compared with the Ant Colony Optimization (ACO) and PSO Algorithm and is gotten the conclusion that the AMG has smaller looping than the other algorithms with weights determined. [3, 7]. Several algorithms implemented in the routing protocol and compare all the algorithms. In the routing protocol that relies on algorithms as the best path search, sometimes always greedy, all considered good even though in the middle of the route a bottleneck. From the drawing back, a new algorithm formed to search for the best path [4, 7, 10]. The routing algorithm classified as an adoption routing algorithm and a non-adoption routing algorithm. The adoption routing algorithm is an algorithm network paths could change the way they route by changes in network topology and traffic. It has a dynamic routing table where sends data over the network. Distance vector routing algorithm, state link routing algorithm, distributed routing algorithm are under the category of the adoption routing algorithm. A non-adaptive routing algorithm is an algorithm used to follow a static routing table for data that enables transmission over a network. This algorithm does not adjust to traffic flow and network topology. For the shortest routing path, the flooding algorithm included in the category of the non-adaptive routing algorithm. The result is that for the adoption routing algorithm, it could easily find the best routing path in network traffic because it could adapt compared to the non-adoption routing algorithm [5]. A hierarchical routing algorithm based on fuzzy mathematics (HRAFM) used to analyze a comprehensive network robustness detection in Wireless Sensor Network (WSN). The traditional robustness detection model assumes that all nodes have the same weight, so it is not possible to get accurate results. Thus, the fuzzy mathematical theory introduced for the detection of WSN robustness. The results show that HCRAFM achieves load balance between WSN nodes, extends the life cycle of each node, and extends the service life of the network [6]. The OSPF routing protocol and Routing Information Protocol version 2 (RIPv2), the Dijkstra algorithm used to select the shortest path between two graphical vertices that represent a network topology. The RIPv2 is a distance-vector protocol that uses a count hop to measure. Paths that have a lower hop count selected to skip the data packet. Based on the research that carried out, the OSPF routing protocol could provide the choice of the best path (best path) to deliver data packets [7].

3 Research Approach

In this research, the performance measured by comparing the PSO algorithm with the Dijkstra algorithm in the routing protocol. The packet delay, and throughput used to measure the performance. In this simulation, we use five routers and six paths in the combination of general topology as shown in Fig. 1 and Fig. 2.

The components had been used for implementing PSO algorithm in the following Table 1 below.

Table 1. The definition of components.

Components	Operational definitions
IP address	An address that will be placed on one router and is unique
Queue	Bandwidth
Destination	A router loopback IP address to be addressed
PSO	A routing selection process technique

The components that affect routing optimization include two following factors.

1) Queue on a link, each node will transmit packets, and connected using a link. In our research, we used a wire for connection. Each link will be regulated by the width of the bandwidth that is passed, the goal is to obey the situation of a path to load many packets.

2) The destination on a node is the last place where the packet was sent.

This algorithm is implemented into the topology in figure 1 it will look like below :

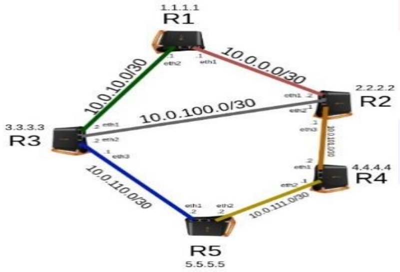


Fig. 1. Topology with IP allocation.

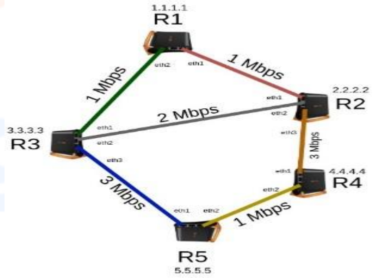


Fig. 2. Topology with bandwidth label.

The MikroTik router is used to easily implement the PSO algorithm, which supports the scripting language. In addition, the requirements are required below.

1) Five MikroTik routers with RB-931-2nd series;
2) UTP Cat5e, 1.5 meters as much as 6 pieces, to connect the routers;
3) Six Port PDU;
4) WinBox ver. 3.19.

From Fig. 1 we define the IP address of each router in Table 2. The related script* was made by us, which had been uploaded to each router.

*Remark:*the script shows the link below.**https://drive.google.com/drive/u/3/folders/1CNpxXhXMm4-8ek8zjqZogAL8sHNcG4qP.*

For the experiment, Table 3 presents the bandwith value from one routher to other routers that is shown in Fig. 2.

Table 2. IP aaddres allocation.

Router	Interface	Destination	Ip address
R_1	Loopback	R_1	1.1.1.1/32
	Ether1	R_2	10.0.0.1/30
	Ether2	R_3	10.0.10.1/30
R_2	Loopback	R_2	2.2.2.2/32
	Ether1	R_1	10.0.0.2/30
	Ether2	R_3	10.0.100.1/30
	Ether3	R_4	10.0.101.1/30
R_3	Loopback	R_3	3.3.3.3/32
	Ether1	R_1	10.0.10.2/30
	Ether2	R_2	10.0.100.2/30
	Ether3	R_5	10.0.110.1/30
R_4	Loopback	R_4	4.4.4.4/32
	Ether1	R_2	10.0.101.2/30
	Ether2	R_5	10.0.111.1/30
R_5	Loopback	R_5	5.5.5.5/32
	Ether1	R_4	10.0.0.111.2/30
	Ether2	R_3	10.0.110.2/30

Table 3. Bandwidth (Mbps) between any two routers.

Nodes	R_1	R_2	R_3	R_4	R_5
R_1	-	1	1	-	-
R_2	1	-	2	3	-
R_3	1	2	-	-	3
R_4	-	3	-	-	1
R_5	-	-	3	1	-

3.1 Implementation of PSO Algorithm.

For better understanding of the remaining contents in this paper, we define the notations that are listed in Table 4.

The three equations that are applied in the implementation of PSO algorithm could be listed as follows.

Velocity update equation: $v_i^{t+1} = c_1 * v_i^t \oplus c_2((pBest_i^t + \frac{1}{2}(gBest_g^t - pBest_i^t) - x_i^t)$ (1)

Particle update equation: $x_i^{t+1} = x_i^t + v_i^{t+1}$ (2)

Fitness equation: $\frac{1}{f(x)} = \frac{1}{\sum_{i=1}^{N} x_i(t)}$ (3)

Table 4. Notations.

Notation	Definition
Particles	A Particle represents as set of routing path that passing through every router once.
$pBest_t$	It represents the particle that has the highest fitness valus in *t* time. In routing representation, fitness calculation is adjusted to routing implementation.
gBest	It represents the best newest particle that resulted from comparing all $pBest_t$.
v	Velocity of particle in PSO Algorithm.
c	Acceleration coefficient, assume: $c_1 = 0.5$ and $c_2 = 1$.
x	The bandwith.
f(x)	The total amount of bandwidth to complete all path in particular particle.
t	Time.
Fitness	A function to determine $pBest_t$. The formula is defined in equation (3).

Next, PSO algorithm is shown below.

Routing Protocol by Using PSO Algorithm.

```
Begin
    Input: t=0
    initialize the x_i(t=0) and v_i(t=0)
    initialize the pBest_i(t=0) = x_i(t=0)
    initialize the gBest_{g=1}(t=0)
    output: x_i(t)  gBest_{g=i}(t=n)  for each individual
    i ∈ N  : calculate fitness(i) = 1 / (∑_{i=1}^{N} x_i(t=0))
      do
            t=t+1
            update the velocity v_i(t) by using
            equation (1);
            update the particles x_i(t) by using
            equation (2);
            for each individual i ∈ N :calculate
            fitness(i) = 1/f(x) = 1 / (∑_{i=1}^{N} x_i(t))
            update the pBest_i(t) and gBest_{g=i}(t)
      while  (not a stop condition)
End.
```

Definition 1. A transposition operation is a method to exchange two values with certain dimensions based on the index order of the particle position. In addition, Transposition (a, b) means the position of element a, is swapped with the position of element b.

3.2 Representation of Particles

The PSO algorithm used to find the best solution for the best routing path from the source to the destination. In this case, the solution represented as particles with certain dimensions mentioned below from step one until step six.

Step 1**.** To initialize the Position of particles.

Initialization carried out to generate a new set of random solutions of a string of dimensions of particles and placed in a buffer called the population. In this stage *the population size* (*popSize*) must be determined. This value represents the number of individuals or particles contained in the population. The experiment is using 5 nodes, then it is assumed 4 *population sizes*, of *pop Size* = 4. We define a *fitness function* $= \frac{1}{f(x)}$, where $f(x) = \textit{the total}$

of bandwidth to complete one path. Initial particles, at the 0*th* iteration (*t* = 0) are generated randomly in the form of an integer number stating the node number and the combination is unique, as shown in Table 5.

Table 5. Position of particle initialization, (*x*).

$x_i(t=0)$	***Particle***	$f(x)$	*fitness function* $1/f(x)$
$x_1(0)$	$[R_1R_2R_4R_5R_3]$	$1+3+1+3+1=9$	$1{,}111\times10^{-1}$
$x_2(0)$	$[R_1R_3R_2R_4R_5]$	$1+2+3+1+0=7$	$1{,}423\times10^{-1}$
$x_3(0)$	$[R_1R_3R_5R_4R_2]$	$1+3+1+3+1=9$	$1{,}111\times10^{-1}$
$x_4(0)$	$[R_2R_3R_1R_2R_4]$	$3+1+3+1+1=9$	$1{,}111\times10^{-1}$

Step 2**.** Initialization of the initial particle velocity.

At the *0th* iteration (at time, *t* = 0), we assume that the initial velocity values of all particles are at v_i^t is 0.

Step 3. Initialization of the pBest and gBest.

When it is still the *0*th iteration (*t* = 0), $pBest_0$, it could be obtained from the highest fitness values in this *t*. Thus, $1{,}423\times10^{-1}$ $pBest_0$ for particle $[R_1R_3R_2R_4R_5]$.

The *gBest*, is sought by selecting the newest $pBest_i$. Thus, in this step, $gBest = pBest_0 = 1{,}423\times10^{-1}$.

Step 4. Velocity updating phase.

Enter the first iteration, $(t=t+1=0+1=1)$.

The equation (1) is used to update the velocity. If, we use c1 = 0.5 and c2 = 1 then to obtain the result of velocity update calculated as follows:

$$v_i^{t+1} = c_1 \times v_i^t \otimes c_2\left(\left(pBest_i^t + \frac{1}{2}\left(gBest_g^t - pBest_i^t\right)\right) - x_i^t\right).$$

Assume that: $v_1(1) = v_1(0+1)$,

$$v_i^{0+1} = c_1 \times v_i^0 \otimes c_2\left(\left(pBest_i^0 + \frac{1}{2}\left(gBest_g^0 - pBest_1^0\right)\right) - x_i^0\right),$$

$$v_1^1 = 0.5\times0\oplus1\left(\left([R_1R_2R_4R_5R_3] + \frac{1}{2}\left([R_1R_3R_2R_4R_5] - [R_1R_2R_4R_5R_3]\right)\right) - [R_1R_2R_4R_5R_3]\right)$$

Thus, in this case for the equation mentioned above, the problem would be solved according to Definition 1 by using the transposition operation for every different routing path with $[R_1R_3R_2R_4R_5]$ as reference.

i. $[R_1R_2R_4R_5R_3] \rightarrow [R_1R_4R_2R_5R_3]$ by using transposition operation (2,3) because R_2 is in third position in $[R_1R_2R_4R_5R_3]$ swapped with R3 in second position.

ii. $[R_1R_4R_2R_5R_3] \rightarrow [R_1R_4R_2R_3R_5]$ by using the transposition operation (4,5) because R_3 is in fifth position in $[R_1R_4R_2R_5R_3]$ swapped with R_5 in fourth position.

iii. $[R_1R_4R_2R_3R_5] \rightarrow [R_1R_3R_2R_4R_5]$ by using the transposition operation (2,4) because R_4 is in second position in $[R_1R_4R_2R_3R_5]$ swapped with R_3 in fourth position.

Next, the equation above become the following equation,

$$v_1^1 = 0.5 \times 0 \oplus 1\left(\left([R_1R_2R_4R_5R_3] + \frac{1}{2}\left([R_1R_3R_2R_4R_5] - [R_1R_2R_4R_5R_3]\right)\right) - [R_1R_2R_4R_5R_3]\right)$$

by using the transposition operation ((2,3), (4,5), (2,4)), and the equation could be written below.

$$v_1^1 = 0.5 \times 0 \oplus 1\left(\left([R_1R_2R_4R_5R_3] + \frac{1}{2}\left((2,3),(4,5),(2,4)\right)\right) - [R_1R_2R_4R_5R_3]\right) \text{ I}$$

$\|v\|$ is number of transpositions listed. So, how to multiply the velocity with constant could be explained below:

If, $v = \left((2,3),(4,5),(2,4)\right)$ then, $\| v \| = 3$ and form the equation above, $c = \frac{1}{2} = 0.5$ where c is a constant $\in R$

It is known, $\| v' \| = 2 \Rightarrow \| v' \| = [0.5 * 3]$, *i.e.* the operation of transposition become two. So, $v' = \left((2,3),(4,5)\right)$ and using addition position with velocity and subtraction position with position can be obtained below:

i. $v_1^1 = \left(\left([R_1R_2R_4R_5R_3] + \left((2,3),(4,5)\right)\right) - [R_1R_2R_4R_5R_3]\right)$;

ii. $v_1^1 = \left(\left([R_1R_2R_4R_5R_3] + \left((4,5)\right)\right) - [R_1R_2R_4R_5R_3]\right)$;

iii. $v_1^1 = \left([R_1R_4R_2R_3R_5] - [R_1R_2R_4R_5R_3]\right)$ by using transposition (2,3);

iv. $v_1^1 = \left([R_1R_4R_2R_3R_5] - [R_1R_4R_2R_5R_3]\right)$ by using transposition (4,5);

v. $v_1^1 = \left([R_1R_4R_2R_3R_5] - [R_1R_4R_2R_3R_5]\right)$ and $v_1^1 = \left((2,3),(4,5)\right)$.

Table 6. $v_i(1)$ value for second iteration.

$v_i(t=1)$	Initial Velocity
$v_1(1)$	$((2,3),(4,5))$
$v_2(1)$	0
$v_3(1)$	$((2,4),(3,5))$
$v_4(1)$	$((2,4),(3,5),(4,5))$

Step 5. Position updating phase.

The equation (2) is used to update the position of particle.

$$x_i^{t+1} = x_i^t + v_i^{t+1}.$$

It is assumed to calculate $x_1(1)$.

$$x_i^0 = x_i^0 + v_i^0;$$
$$\Rightarrow x_1^1 = [R_1 R_2 R_4 R_5 R_3] + ((2,3),(4,5));$$
$$\Rightarrow x_1^1 = [R_1 R_2 R_4 R_5 R_3] + ((4,5));$$
$$\Rightarrow x_1^1 = [R_1 R_4 R_2 R_3 R_5].$$

From Table 3, $f(x_1^1)$ could be obtained as follow:

$$f(x_1^1) = 0+3+2+3+0 = 8.$$

For the fitness value, the calculation is as follow:

$$\frac{1}{f(x_1^1)} = 1.25\times10^{-1}.$$

For the similar calculation result of all $x_i(1)$, it is shown in Table 7.

Step 6. The *pBest* and *gBest* updating phase.

In this step, we compare *pBest* in the previous iteration with the highest fitness in current iteration. The higher value of it would be the current *pBest*. For $t=0$, $pBest_0$ is $1,423\times10^{-1}$ as stated in Table 5. The highest fitness value for $t=1$ is 1.43×10^{-1} as stated in Table 7. Thus, $pBest_1$ is 1.43×10^{-1} because the both *t* have same values .

The newest *pBest* would be the *gBest*. Then, if the next iteration is continued (t = t + 1), the PSO algorithm steps will be repeated continuously until, *n* iteration has been reached convergence. Table 8 shows the list of $pBest_i$ for current iteration.

Table 7. The position of, $x_i\left(1\right)$.

$x_1\left(t=1\right)$	Particle	$f\left(x\right)$	Fitness
$x_1\left(1\right)$	$\left[R_1R_4R_2R_3R_5\right]$	$0+3+2+3+0=8$	1.25×10^{-1}
$x_2\left(1\right)$	$\left[R_1R_3R_2R_4R_5\right]$	$1+2+3+1+0=7$	1.43×10^{-1}
$x_3\left(1\right)$	$\left[R_1R_4R_2R_3R_5\right]$	$0+3+2+3+0=8$	1.25×10^{-1}
$x_4\left(1\right)$	$\left[R_1R_4R_2R_3R_5\right]$	$0+3+2+3+0=8$	1.25×10^{-1}

Table 8. The *pBest* value for second iteration.

$pBest_i$	Particle	$f\left(x\right)$	Fitness
$pBest_0$	$\left[R_1R_2R_4R_5R_3\right]$	$0+3+2+3+0=8$	1.25×10^{-1}
$pBest_1$	$\left[R_1R_3R_2R_4R_5\right]$	$1+2+3+1+0=7$	1.43×10^{-1}

4 Results and Discussions

The testing result is carried out by comparing between PSO Algorithm and the Dijkstra algorithm usage with the same weight of the packet and the same path.

4.1. Testing of PSO and Dijsktra Algorithm Performance

The testing for PSO and Dijsktra algorithm performance is using *traceroute feature*, the goal is to know the packet that passes through the node and the path to reach the destination. the IP loopback on the R_5-PSO router is 5.5.5.5. Using in MikroTik, it looks like in Fig. 3. The Dijkstra will look like in Fig. 4.

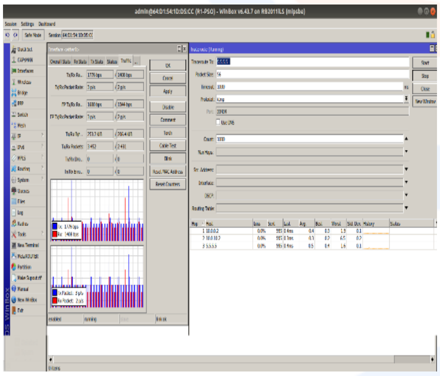


Fig. 3. Traceroute Using PSO Algorithm.

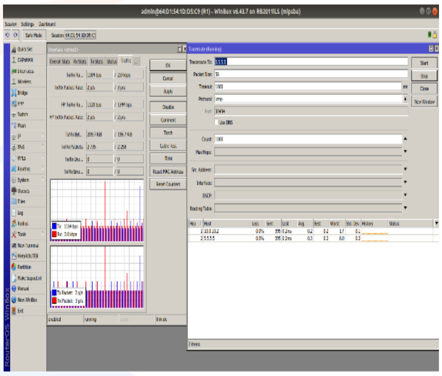
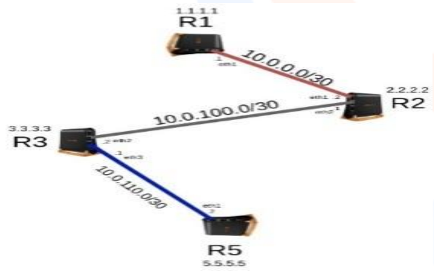


Fig.4. Traceroute Using Dijkstra Algorithm.

The results (host) then the path or route selected by the PSO algorithm is from R_1 > R_2 > R_3 > R_5. While the route is chosen by Dijkstra algorithm is different to PSO, Dijkstra chooses from R_1> R_3> R_5, if it is makes an image then as shown in Fig. 5 and Fig. 6.



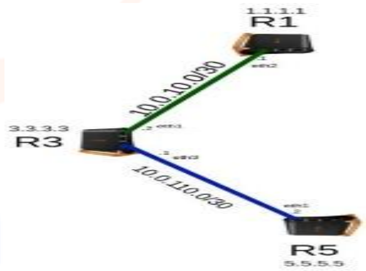
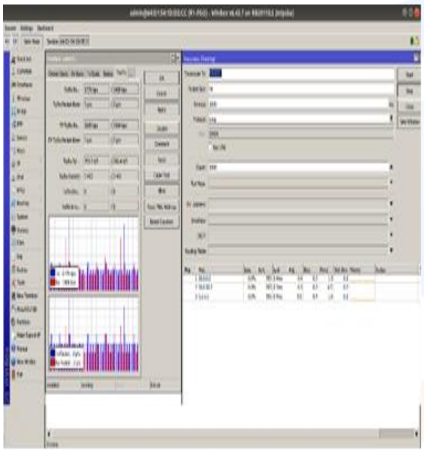


Fig. 5. Route Selection with PSO Algorithm.

Fig. 6. Route Selection with Dijkstra Algorithm.

The testing *traceroute* with the load of 56 bytes using the PSO algorithm as protocol and 995 count sent (packet sending trial) is shown in Fig.7. The Performance of Dijsktra with load of 56 bytes and the 995 count is shown in Fig.8.



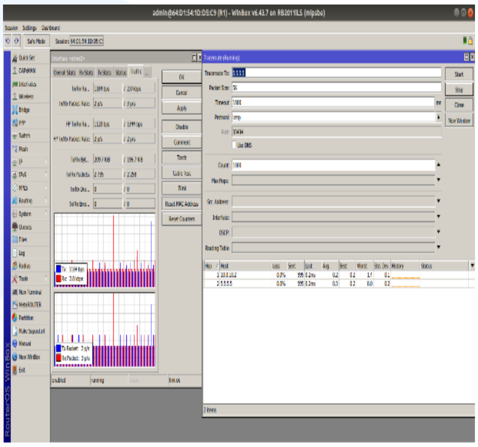


Fig. 7. Testing with a load on the PSO Algorithm.

Fig. 8. Testing with a load on the Dijkstra Algorithm.

Table 9 shows the comparasion of throughput performance between the PSO and Dijsktra algorithms with load of 56 bytes and the 995 count.

Table 9. Comparison of Performance between the PSO algorithm and the Dijkstra algorithm.

	PSO Algorithm		***Dijkstra* Algorithm**	
	T_x	R_x	T_x	R_x
Rate	1776 bps	1408 bps	1184 bps	2.0 kbps
Packet Rate	3 p/s	2 p/s	2 p/s	3 p/s
FP *Rate*	1680 bps	1344 bps	1120 bps	1344 bps
FP *Packet Rate*	3 p/s	2 p/s	2 p/s	2 p/s
Bytes	253.2 KB	206.4 KB	209.7 KB	196.7 KB
Packets	3452	2431	2735	2258
Packet Delay	1.2 ms		0.5 ms	
Route Selection	3 Hops		2 Hops	

The packet data generated by the PSO algorithm could send a packet of T_x (transmitter) 3452 / R_x (receiver) 2431 with a number of bytes of T_x 253.2KB / R_x 206.4KB, while the Dijkstra algorithm is smaller, that is from 995 count could send packets as much as T_x (transmitter) 2735 / R_x (receiver) 2258 with many bytes of T_x 209.7KB / R_x 196.7KiB, it could be concluded that the PSO algorithm is greater throughput than the Dijkstra algorithm.

Table 10 shows the comparasion of packet delay performance between the PSO and Dijsktra algorithms with load of 56 bytes and the 995 count.

Table 10. Packet Delay between PSO algorithm.

	Hop	Host	Loss	Sent	Last	Avg	Best	Worst	Std. Dev.
PSO	1	10.0.0.2	0.0%	995	0.4	0.4	0.3	1.9	0.1
	2	10.0.10.2	0.0%	995	0.3	0.3	0.2	6.5	0.2
	3	5.5.5.5	0.0%	995	0.4	0.5	0.4	1.6	0.1
Dijkstra	1	10.0.0.2	0.0%	995	0.2	0.2	0.2	1.4	0.1
	2	5.5.5.5	0.0%	995	0.2	0.3	0.8	8.0	0.2

From Table 10 shows the Djikstra algorithm has a shorter route than the PSO algorithm. The PSO algorithm passes 3 hops to reach the destination, while the Dijkstra algorithm only needs 2 hops to reach the destination. The Dijkstra algorithm chooses the path via R3, while the PSO algorithm has more hops because the nature of Dijkstra algorithm is Short Path First. While the PSO algorithm has a longer route, the packet delay time required is 1.2 ms, greater than the packet delay time required by the Djikstra algorithm, 0.5 ms, because PSO need more time for encapsulation and fragmenting packet. However, the weakness of the Dijkstra algorithm is that when a data packet occurs at *t*-time, traffic density becomes a bottleneck. This is a factor causing the package to arrive at the destination pending. It also includes packages returned to the sender and through other routes.

Next, total number of the operations is calculated below.

$$T(n) = n + n + 1 + n\log n + \log n + 16 = 2n + 17 + n\log n + \log n;$$

$$T(n) = 2n + 17 + (n+1)\log n.$$

Otherwise, the total number of operations is performed and obtained below via *t* times until the termination condition.

$$T(n) = 2n + 17 + (n+1)\log n.t = O(n.t\log n) = O(n\log n).$$

Thus, the PSO algorithm could be easily implemented because of its complexity is linear, and even smaller than the complexity of the Dijkstra algorithm, $O(n^2)$.

5 Conclusion

In this section, it concluded that the PSO algorithm enables to be implemented into the routing protocol, because it has linear complexity, $O(n.t\log n)$, and smaller than the complexity of the Dijkstra algorithm, $O\left(n^2\right)$. The PSO algorithm throughput is much greater than the Dijkstra algorithm. By using iteration with n equal 995, the PSO algorithm could send packets within T_x (transmitter) 3452 / R_x (receiver) 2431 and with bytes of T_x 253.2 KB / R_x 206.4 KB, while the Dijkstra algorithm is smaller, by using iteration with n equal 995, it could send packets within T_x (transmitter) 2735 / R_x (receiver) 2258 packets and with bytes as T_x 209.7 KB / R_x 196.7 KB. The packet density sent by the PSO algorithm is denser compared to Dijkstra algorithm. The PSO algorithm takes longer to send packets to destination, from the n iteration equal to 995 *avg* calculated by 1.2 ms while the Dijkstra algorithm is faster by 0.5 ms. This is because the Djikstra algorithm chooses a shorter route than the PSO algorithm. The PSO algorithm passes three hops to reach the destination, while the Dijkstra algorithm only needs two hops to reach the destination. Many factors in the delay of a packet to its destination, include packet return back or through other routes. However, the encapsulation and the fragmentation of the Dijkstra algorithm make the smaller packet so it becomes faster to the destination. It hoped there will be another research, about the process of the fragmentation and the encapsulation in the PSO algorithm.

Acknowledgments. This work was supported by Asia University, Taiwan, and China Medical University Hospital, China Medical University, Taiwan, under Grant ASIA-108-CMUH-05 and ASIA-107-CMUH-05. This work was also supported by Asia University, Taiwan, UMY, Indonesian, under Grant 107-ASIA-UMY-02. This study is also supported by the Ministry of Science and Technology (MOST), Taiwan, Republic of China, under the grants of MOST 108-2221-E-324-013 and MOST 107-2221-E-468-015.

References

1. Baran B. and Sosa R.: AntNet: Routing Algorithm for Data Networks based on Mobile Agents. Argentine Symposium on Artificial Intelligence, Vol. 5. No. 12. (2001) 75-84
2. Sriramoju A. B. and Corporation M.: Particle Swarm Optimization Algorithm for Routing Network, Vol. 3, No. 2 (2017) 339-345
3. Lestandy M., Pramono S. H., and Aswin M.: Optimasi Routing pada Metropolitan Mesh Network Menggunakan Adaptive Mutation Genetic Algorithm. J. Nas. Tek. Elektro dan Teknol. Inf., Vol. 6, No. 4 (2018) 430–435
4. Shandilya S.: Understanding Network Routing Problem and Study of Routing Algorithms and Heuristics through Implementation. Global Journal of Computer Science and Technology, Vol. 17, No. 5-E (2017)
5. Sumitha J.: Routing Algorithms in Networks. Research Journal of Recent Sciences, Vol. 3. (2014) 1–3
6. Xiang Y., Chen M., Zhuang X., and Li X.: Routing algorithm of wireless sensor network and robustness analysis based on fuzzy mathematics. Int. J. Online Eng. Vol. 13. No. 12. (2017) 85–103
7. Musril H. A.: Penerapan Open Shortest Path First (OSPF) Untuk Menentukan Jalur Terbaik Dalam Jaringan. J. Elektro dan Telekomunikasi. Terapan. Vol. 4.No. 1. (2017) 421
8. Deng, Wu, et al.: A novel intelligent diagnosis method using optimal LS-SVM with improved PSO algorithm. Soft Comput 23, (2019) 2445–2462
9. Dhanachandra, N., and Chanu Y. J.: An image segmentation approach based on fuzzy c-means and dynamic particle swarm optimization algorithm. Multimedia Tools and Applications (2020) 1-20

10. Prithi, S., and S. Sumathi.: LD2FA-PSO: A novel Learning Dynamic Deterministic Finite Automata with PSO algorithm for secured energy efficient routing in Wireless Sensor Network. Ad Hoc Networks 97 (2020) 102024
11. Chen H.-C.: TCABRP: A Trust-Based Cooperation Authentication Bit-Map Routing Protocol Against Insider Security Threats in Wireless Ad Hoc Networks, IEEE Systems Journal, Vol. 11, No. 02 (2017) 449 – 459
12. Kung T.-L. and Chen H.-C.: Topological dynamics of comparison-based fault identification in ad hoc networks, Pervasive and Mobile Computing, Vol. 41 (2017) 69-82
13. Chen H.-C. and Su H.-K.: A Cooperative Trust Bit-Map Routing Protocol Using the GA Algorithm for Reducing the Damages from the InTs in WANETs, Journal of Internet Services and Information Security, Vol. 4, No. 4 (2014) 52 – 70
14. Chen H.-C. *et al.*: A Routing Algorithm Based on Event-Oriented Applications for Digital Home Wireless Heterogeneous Networks, International Journal of Engineering and Industries, Vol. 2, No. 3 (2011) 96 ~ 103
15. Weng C.-E., Chen H.-C.: The performance evaluation of IEEE 802.11 DCF using Markov chain model for wireless LANs, Computer Standards & Interfaces, Vol. 44 (2016) 144 – 149
16. Weng C.-E., Wen J.-H., Chen H.-C. and Yang L.: The Performance Analysis of Direct/Cooperative Transmission to Support QoS in WLANs, Computer Science and Information Systems, Vol. 11, No. 3 (2014) 1043 – 1156
17. Weng C.-E., Sharma V., Chen H.-C. and Mao C.-H.: PEER: Proximity-Based Energy-Efficient Routing Algorithm for Wireless Sensor Networks, Journal of Internet Services and Information Security, Vol.6, No.1 (2016) 47–56
18. Huang Y.-F., Wang J.-W., Jenq J., Chen H.-C. and Hsu C.-H.: Performance on Clustering Routing for Naturally Deployed Wireless Sensor Networks, Communications in Computer and Information Science, Vol 797, Springer, Singapore (2018) 1–9