Improving AODV Route Recovery Mechanisms with Connectivity and Particle Swarm Optimization

Nadilma C. V. N. Pereira, Carmelo J. A. Bastos-Filho, and Renato M. de Moraes

***Abstract*—This paper proposes a new function for the Ad Hoc On-Demand Distance Vector (AODV) algorithm using the connectivity concept to decide between source and local repair when a link break occurs. Besides, we used a Computational Intelligence algorithm, called Particle Swarm Optimization (PSO), which enables to find candidate solutions in all considered search space in order to reduce average data packet delivery delay sent over the network. The performance of the original AODV protocol, according to its Request for Comments (RFC), is compared with this new implemented approach, called here as AODV-PSO. Three metrics were used to evaluate the performance of the algorithm: throughput (as data packet delivery fraction), routing overhead and average data packet delivery delay. We observed that the AODV-PSO outperformed the original AODV protocol in the scenarios studied in this paper.**

***Index Terms*—Ad hoc networks, AODV, PSO.**

I. Introduction

GREAT advances in communication devices and wireless applications have been observed in the last years. It has increased mainly due to the benefits supported by these technologies when compared to wired networks, such as installation facilities and portability. These devices are often used by people who need connectivity and flexibility.

Among the wireless communication systems, ad hoc networks have been pointed as a great alternative for the future, since they can provide the same services of a conventional wireless network, without the requirement of a base station for establishing communication [1]. In ad hoc networks, the goal is to send information from a source to a destination node in a peer-to-peer connection through intermediate mobile nodes, *i.e.*, across multiple hops. Accordingly, the network nodes also act as routers where the management and control are distributed among the mobile nodes. Therefore, the traditional wired network routing algorithms cannot be effectively applied to ad hoc networks, since they present some issues that are very peculiar and different from wired networks.

Many protocols have been proposed to solve the multihop routing problem, each one based on different assumptions and features. Some of them use a broadcast mechanism to discover routes through the network [2], [3].

This work was developed in part at Polytechnic School of Pernambuco - UPE. This work was supported in part by Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE), by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Brazil.

N. C. V. N. Pereira is with Informatics Center, Federal University of Pernambuco, Recife, Brazil. Email: ncvnp@cin.ufpe.br

C. J. A. Bastos-Filho is with Polytechnic School of Pernambuco, University of Pernambuco, Recife, Brazil. Email: carmelofilho@ieee.org

R. M. de Moraes is with Department of Electrical Engineering, University of Brasília, Distrito Federal, Brazil. Email: renatomdm@unb.br

One of the leading protocols for routing in ad hoc wireless networks is the AODV algorithm [3], [4], [5]. This protocol is reactive, meaning that it does not require the nodes to maintain routes to their destinations that are not in active communication. AODV routing also provides two different mechanisms to recover routes when a link break in active routes occurs due to changes in network topology. In the first case, the route repair can be initiated locally at the node located immediately upstream of the broken link. In the second case, the route repair can be made by the source node after receiving a notification of the route failure [5].

In the AODV route recovery mechanisms, for networks with limited diameters, error messages can be propagated back to the source node relatively quickly, and a repairing action is performed. However, for networks with larger diameter and longer paths, the error message may have to propagate for more hops to reach the source which may increase the time to repair. Hence, this mechanism can become ineffective for more stressful scenarios. One can observe that, in the current AODV algorithm implementation, the protocol chooses to do either a source repair or a local repair based only on the number of hops involved across the path [6].

In this way, we propose a new AODV based approach for route failure recovery by adding the connectivity concept at the candidate nodes in the decision process to perform the route repair. Connectivity is defined here as the number of neighboring nodes present in the transmission range of the candidate node chosen to make the route repair (either the source node or the predecessor node to the failure).

In addition, we consider to use a Computational Intelligence technique called Particle Swarm Optimization (PSO) [7], to generate the appropriate composition of weights for all considered parameters in AODV route recovery, in order to improve the choice in local repair decision aiming to reduce data packet delivery delay.

After that, to validate our proposal, we developed similar simulation scenarios used by related work that describe the characteristics of the AODV protocol [4], [8], [9], [10], [11], [12], [13], employing the Network Simulation-2 (NS-2) [14]. Finally, the new approach is evaluated based on simulation results for three network metrics: data packet delivery fraction, routing overhead and data packet delivery delay.

This paper is organized as follows. Section II briefly reviews the AODV routing protocol focusing on its routing maintenance and describes the related work. Section III presents the proposed modifications to the AODV protocol based on the connectivity and the PSO technique. Section IV describes the simulation scenario and results. Finally, Section V brings the conclusion and suggestion for future work.

II. Ad Hoc On-Demand Distance Vector Routing

The AODV routing protocol enables dynamic, self-starting, multi-hop routing between participating mobile nodes wishing to establish and maintain an ad hoc network. The AODV algorithm allows mobile nodes to quickly obtain routes for new destinations, and it does not require nodes to maintain routes to destinations that are not in active communication. Also, AODV routing permits mobile nodes to respond to link breakages and changes in network topology in a timely manner [5]. The main objective of this protocol is to quickly and dynamically adapt itself to changes of conditions on the network links, for example, due to mobility of nodes.

The AODV protocol works as a pure on-demand route acquisition system. When a source node desires to send a message to some destination node and does not already have a valid route to that destination, it initiates a path discovery process to locate the other node. It broadcasts a route request (RREQ) control packet to its neighbors, which then forward this request to their neighbors, and so on. It occurs until either the destination or an intermediate node with a "fresh enough" route to the destination is located.

The AODV protocol utilizes destination sequence numbers to ensure that all routes contain the most recent route information. Each node maintains its own sequence number. During the process of forwarding the RREQ, intermediate nodes record in their routing tables the address of the neighbor from which the first copy of the broadcast packet is received, thereby establishing a reverse path. Once the RREQ reaches the destination or an intermediate node with a fresh enough route, the destination or the intermediate node responds by unicasting a route reply (RREP) control packet back to the neighbor from which it first received the RREQ.

A. Route Recovery

An active route is defined as a route which has recently been used to transmit data packets. If a link break occurs while the route is active, the node upstream of the break propagates a route error (RERR) message to the source node to inform it that the destination is now unreachable [5]. After receiving the RERR, if the source node still desires the route, it can reinitiate the route discovery process.

Alternatively, the algorithm may initiate a local repair mechanism when a link failure happens on an active route and the first node upstream of this break (the predecessor) chooses to repair the link locally if the destination is not too far away. In such case, the node increments the sequence number for the destination and then broadcasts a RREQ for the destination [5]. Thus, local repair attempts will often be invisible to the source node. The node that initiates the repair waits the discovery period to receive RREP in response to the RREQ. During local repair, data packets should be buffered. If, at the end of the discovery period, the repairing node has not received a RREP (or other control message creating or updating the route) for that destination, the node propagates a RERR [5]. When it happens, long delays and huge losses of packets due to exhaustion of the queues will occur. However, if the repairing node receives a RREP, it ensures lower overhead and delay.

In the current implementation of the AODV protocol, it chooses to do either source repair or local repair depending on the number of hops involved on the path. Notice that the choice holds following the condition depicted in Algorithm 1, where *packetForward* is the amount of hops from the source node until the upstream node (*i.e.*, the predecessor) before the failure and *predecessorHopCount* is the quantity of hops from the predecessor node to destination which information is stored in the routing table of the predecessor node. If *packetForward* has more hops than *predecessorHopCount*, the algorithm chooses to make local repair, otherwise the upstream node sends a RERR to the source. Based on this characteristic of the AODV routing protocol, several studies proposed to locally recover from the disconnected route [11], [12], [13], [15], [16], [17], [18], [19], such that the repairing action can still be on time, as described in the next subsection.

Algorithm 1: Route recovery decision.

```
1 if ($packetForward \geq predecessorHopCount$) then
2     localRepair();
3 end
4 else
5     sourceRepair();
6 end
```

B. Related Work

In [15], Pan *et al.* suggested, as an enhancement to AODV, an approach that leads to two routing protocols, called AODV - Local Repair TTL (AODV-LRT) and AODV - Local Repair Quota (AODV-LRQ), which aimed to efficiently repair the link errors. These approaches include a decrease breadth or depth of the extent to which the repair mechanism applies. Decreasing the breadth of the repair mechanisms means that one can limit the maximum number of hops, *i.e.*, time-to-live (TTL), that RREQ packets have to pass, assuming that the size of the network topology and transmission range of every node are known. For example, if the size of the network topology is $1500m \times 600m$ and the transmission range for each node is 250 m, the longest path will be 6 hops. In such scenario, TTL will not exceed $\frac{\textit{longest known path}}{2}$ hops. Decreasing the depth of the repair mechanisms means to limit the number of times a node is allowed to forward the route repair request, *i.e.*, each node carries out the route repair *k* times in some period *T* of time.

In [11], Youn *et al.* proposed a new local repair scheme using a promiscuous mode, which is mainly composed of two parts: adaptive promiscuous mode and quick local repair scheme. Adaptive promiscuous mode repeats the switching processes between promiscuous mode and non-promiscuous mode. The proposed scheme adopts promiscuous mode such that each node keeps monitoring the overheard packets from which the routing information about the route path in adjacent nodes can be obtained. This action can cause excessive energy consumption and reduce network efficiency.

Other proposed studies aim to change the AODV protocol to make it more efficient in relation to performance metrics such as throughput or overhead utilizing the information from the neighbors to prevent breaks or provide spare routes [12], [13], [16], [17], [18], always monitoring restored routes after occurrences of link breaks due to the dynamic of the network. Another approach is to gradually migrate to a clustering route protocol to deal with dense networks [19].

Although the previous studies have proposed new schemes for route recovery of AODV algorithm in case of broken links, to the best of our knowledge, none of them attempted to use the concept of node connectivity, beyond hop counters, employing weight in each argument considered to obtain the repair decision. Furthermore, any of these previous approaches did not consider the use of Computational Intelligence techniques to address route failure recovery.

III. New Approach to Improve The AODV Route Recovery Mechanisms with Connectivity and PSO

In the approach currently implemented on the AODV protocol, as aforementioned, the source repair and the local repair are used, but the protocol chooses either one depending on the number of hops involved on the path to destination according to the original draft of the protocol specified in its RFC [5]. However, the amount of neighbors around the candidate nodes (source and predecessor) to repair the failure may indicate the possibility of finding another good route to destination.

A. Connectivity

Following this rationale, we introduce the connectivity concept defined as the number of neighbors in the transmission range of a node, which can be obtained from the AODV exchange of HELLO messages. Accordingly, our proposal to improve the performance of the AODV also considers the connectivity information for the source and predecessor nodes on the decision of the recovery. This is accomplished by associating weights to the number of hops on the path to destination and to the connectivity, according to Algorithm 2. In this algorithm, $source$ and $local$ are functions of the amount of hops on the path to destination and their respective connectivities. The weights *A*, *B*, *C* and *D* have direct influence on the choice between one and other repair action and are real numbers defined over the interval $[-1, 1]$. The PSO technique was employed to optimize the weight values in Algorithm 2.

Algorithm 2: Route recovery using the connectivity concept. A, B, C and D are obtained from the PSO technique.

```
1 $source = (A \times packetForward) + (B \times sourceConnectivity)$;
2 $local = (C \times predecessorHopCount) + (D \times predecessorConnectivity)$;
3 if $(source \geq local)$ then
4     localRepair();
5 end
6 else
7     sourceRepair();
8 end
```

B. Particle Swarm Optimization for AODV Routing

Particle Swarm Optimization (PSO) is a Computational Intelligence technique proposed by Kennedy and Eberhart [7] in 1995, which implements a metaphor of social behavior for the interaction between individuals (particles) of a group (swarm). From this initial objective, the concept evolved to a simple and efficient optimization algorithm. More specifically, the ability of groups of some species (like simple agents) work as a whole for locating desirable positions in a given area. This seeking behavior was associated with that of an optimization search for solutions to non-linear equations in a real-valued search space[1] (or hyperspace) [20]. In PSO, each single particle acts as an agent in the search space. The set of particles (the group of agents) is named swarm.

1) PSO Technique: In the most common implementations of PSO, particles move through the search space using a combination of an attraction to the best solution that they individually have found, and an attraction to the best solution that any particle in their neighborhood has found during the search process so far. In PSO, a neighborhood is defined for each individual particle as the subset of particles which it is able to communicate with [20].

Particle swarm optimization is initialized with a population of random solutions for which is also assigned a randomized velocity. The potential solutions, called particles, are then "flown" through the hyperspace.

Each particle keeps the coordinates of the position in the hyperspace which is associated with the best solution it has achieved so far $\overrightarrow{p}_i$. The best solution is determined according to the fitness evaluation at the points considered along the trajectories of the particles. The value of the fitness is also stored. This value is often called P_{best}. Another "best" value is also tracked: the "global" version of the particle swarm optimizer G_{best} which keeps track of the overall best value, and its location, obtained thus far by any particle in the population $\overrightarrow{p}_g$.

The particle swarm optimization concept consists of, at each time step, changing the velocity (accelerating) each particle toward its $\overrightarrow{p}_i$ and $\overrightarrow{p}_g$ (global version). Acceleration is weighted by a random term with separate random numbers being generated for acceleration toward $\overrightarrow{p}_i$ and $\overrightarrow{p}_g$. The previous velocity of the particle is also considered for changing the velocity of the particle.

The particles move throughout the search space by a fairly simple set of update equations. The algorithm updates the entire swarm at each time step by updating the velocity and position of each particle in every dimension according to [21]

$$\overrightarrow{v}_i(t+1) = \omega\overrightarrow{v}_i(t) + c_1\epsilon_1[\overrightarrow{p}_i(t) - \overrightarrow{x}_i(t)] + c_2\epsilon_2[\overrightarrow{p}_g(t) - \overrightarrow{x}_i(t)], \tag{1}$$

where for each particle there are three vectors: a position vector $\overrightarrow{x}_i$, a velocity vector $\overrightarrow{v}_i$ and the vector which stores its best found position $\overrightarrow{p}_i$, in which the subscript *i* represents the index of the particle. These values represent the individual knowledge of the particle. Besides these three vectors, the PSO algorithm should keep another array that stores the

[1]In our study, since four parameters (A, B, C and D) need be optimized, our search space has four dimensions.

best position found by any particle in the neighborhood, representing the group's experience $\overrightarrow{p}_g$.

The velocity vector $\overrightarrow{v}_i$ directly influences the direction that the particle will take on each iteration, allowing the particles to reach different regions of the search space.

The knowledge acquired by the particle in accordance with their own experience is represented in Equation (1). $c_1\epsilon_1[\overrightarrow{p}_i(t) - \overrightarrow{x}_i(t)]$ is called the cognitive component. This term represents how much, from its current position $\overrightarrow{x}_i$, the particle is far away from its best position ever found $\overrightarrow{p}_i(t)$.

The experience of an entire neighborhood is represented by the social component, which appears in Equation (1) as $c_2\epsilon_2[\overrightarrow{p}_g(t) - \overrightarrow{x}_i(t)]$. This component corrects the velocity of particle i based on the best position found within its neighborhood $\overrightarrow{p}_g(t)$. For a neighborhood with the size of the entire swarm, this term represents the correction of particle velocity for the whole group. In this case, the particle will be aware of the best overall position within the group, and thus, adjust its position relative to the whole swarm.

The terms c_1 and c_2 are acceleration constants and assume real values [20]. These values are used to balance the influence of the cognitive component and social component, defining the search behavior of the swarm. The terms ϵ_1 and ϵ_2 are generated separately from an uniform distribution between 0 and 1 [21].

The inertia weight parameter ω was used to strike a better balance between global exploration and local exploitation by adjusting the influence of the previous particle velocities on the optimization process, described in Equation (1).

The change of velocity modifies the position of particles making them to move through the hyperspace over successive iterations. Upon updating their information, the swarm particles reorganize their experiences. Depending on the problem that the PSO will handle, either minimizing or maximizing a function, each particle will adjust its objective function (fitness) to converge towards a possible solution.

2) PSO Algorithm: PSO initially provides a set of random values for the weights and velocities and a network scenario is executed using them. Equation (1) is utilized to update the velocities of each particle in the swarm. At the end of the execution, data packet delivery delay measured is taken as the PSO fitness and this process is repeated until the best values for the weights is reached according to the stopping criteria. Thus, the evaluated objective function (fitness) of each particle is the average data packet delivery delay for each network scenario simulated, so that this measure could be minimized over the course of iterations. This process is described in Algorithm 3.

The stopping criteria employed was the number of iterations. Therefore, regarding the computational complexity of the PSO technique, considering that the algorithm is running with *P* particles and it evaluates each particle at each iteration, the execution time is roughly PNT, where *N* is the number of iterations and *T* is the time to evaluate a scenario with the predefined set of parameters (*i.e.*, A, B, C and D).

3) PSO Configuration: For the simulations performed in this work, we used the PSO with global topology, that can make search for appropriate parameters to the AODV decision function. The global topology converges fast and can be used for this problem since the search space is not highly multimodal[2]. In this kind of topology the particles have knowledge about the position of all the other particles in the swarm, being able to know which one has the best position [20]. [3]

Algorithm 3: PSO algorithm for AODV routing.

```
1  startSwarm();
   // Set particle dimensions randomly from
      considered search space
   // Initialize particle positions and
      velocity as in Equation (1)
2  repeat
3      for eachParticle do
4          for eachDimension do
5              updateVelocityPosition();
               // Update particle velocity and
                  position as in Equation (1)
6          end
7          calculateFitness;
           // For each particle, calculate its
              fitness value from the data packet
              delivery delay measured
8      end
9      Uptade leader;
       // If the fitness value is better than
          the previous $P_{best}$, set the current
          fitness value as the new $P_{best}$
10     if currentFitness $<$ bestIndividualFitness then
11         bestIndividualFitness = currentfitness;
12     end
13     if currentFitness $<$ bestNeighborhoodFitness then
14         bestNeighborhoodFitness = currentfitness;
15     end
       // Best particle is selected as $G_{best}$
16 until stoppingCriteria;
```

A population of 20 particles was utilized. The algorithm terminated when the number of 200 iterations of the PSO was reached. For ω we used values decreasing linearly from 0.9 to 0.4 during the execution of the algorithm. According to [21], 20 particles, 200 iterations and the linearly decreasing values in the above interval is appropriate to low dimension search space like the one utilized here.

IV. Network Simulation Results and Analysis

Aiming to produce results relevant to this study, three performance measures were chosen for our analysis: throughput, overhead and average packet delivery delay. The measures of performance are explained below:

- *Data packet delivery fraction (or throughput)*: The ratio of the data packets delivered to destinations to those generated by the constant bit rate (CBR) sources.
- *Routing overhead:* The total number of control routing packets transmitted during the simulation.
- *Average data packet delivery delay*: This includes all possible delays caused by buffering during route discovery latency, queuing at interfaces, retransmission delays at the MAC layer, propagation and transfer times.

[2]A multimodal problem is characterized by many local maxima which prevents the algorithm to quickly reach a solution [20].

[3]Note that PSO topology and network topology are distinct definitions, as well as particles and nodes are different things.

A. Network Simulation Environment

For the network simulations performed in this work, we used the NS-2, version 2.33 [14]. The simulation environment, analogous to related works, has area of $1500m \times 300m$, containing 50 mobile nodes and simulation duration of 900 seconds. The physical (PHY) and medium access (MAC) layers have the following characteristics: IEEE 802.11 at bit rate of 2 Mbits/s and transmission range of 250 m. The radio propagation model was the Two-Ray Ground [14]. All data and routing packets sent by the network layer are queued at the interface queue until the MAC layer can transmit them. Each node has a queue for packets awaiting transmission by the network interface that holds up to 50 packets.

The mobility model utilized was the Random Waypoint model [22] using seven different pause times: 0, 30, 60, 120, 300, 600, and 900 seconds in order to simulate different degrees of movement. The maximum speed (V_{max}) is 20 m/s. The traffic pattern was peer-to-peer CBR with the following parameters: 30 traffic sources, packet rate of 4 packets/s with packet size of 512 bytes. We opted for User Datagram Protocol (UDP) traffic because it is not connection oriented, *i.e.*, it does not have mechanisms to adapt to network load. Our intention is to measure the network layer effect over the metrics. Similar simulation scenarios were used by other authors that describe the characteristics of the AODV protocol [4], [8], [9], [11], [12], [13].

For minimizing the PSO objective function, here considered as the average data packet delivery delay, only a network simulation scenario was called as a sample for each utilized pause time. Accordingly, the PSO was run in 7 different times, obtaining a sample for each different pause time parameter setting.

For each iteration of PSO with different weights for the parameters for each considered particle, the same network simulation scenario was run 20 times for a pause time defined, because a total of 20 particles were employed. Upon executing 200 iterations in PSO, 4000 calls were made to the network simulation scenario for the delay analysis. However, as we considered seven different pause times, the PSO worked separately for each, resulting in a total of 28000 calls to the network simulation scenario. Accordingly, the optimal values of the weights A, B, C and D found by the PSO technique for each pause time utilized are shown in Table I.

B. Performance Results

Each produced curve point in the results shown below represents an average of ten runs in the NS-2 simulator with identical traffic load, but with different randomly generated mobility scenarios. The confidence level on the plotted intervals is 95%. The weights found by the PSO technique, according to Table I, were employed for the parameters considered in the decision function implemented in the modified AODV protocol.

Figures 1, 2 and 3 show the average data packet delivery delay, the packet delivery fraction and the routing overhead, respectively, as a function of the pause time. In the figures, AODV refers to the original AODV protocol, while AODV-PSO refers to the modified AODV protocol using the weights found by the PSO technique.

TABLE I
OPTIMUM PARAMETERS VALUES FOUND BY THE PSO TECHNIQUE FOR EACH PAUSE TIME EMPLOYED IN THE STUDIED SCENARIO.

Pause Time (s)	A	B	C	D
0	0.368487	-0.3521118	-1	-0.704438
30	0.794896	-0.129794	-0.0223986	-0.593785
60	-0.0259248	0.459122	-0.372466	-0.5
120	0.222188	-0.0126874	0.114947	-0.90658
300	0.389383	-0.214353	-0.589134	-0.5
600	0.0502506	-0.246525	-0.426209	-0.367086
900	0.219695	-0.546503	-1	-0.5

Figure 1 illustrates that, for average data packet delivery delay, the AODV-PSO achieved a better performance when compared to the original AODV. One can note that for lower pause times, *i.e.*, higher mobility (the pause times between 0 and 300 seconds), the AODV-PSO presents an improvement of almost 30%. At 600 seconds it shows an improvement of almost 10%. At 900 seconds no improvement was verified since in such case the network remained static during the entire simulation period. These results outperform those obtained in [11], [12] and [13], for similar scenarios. [4]

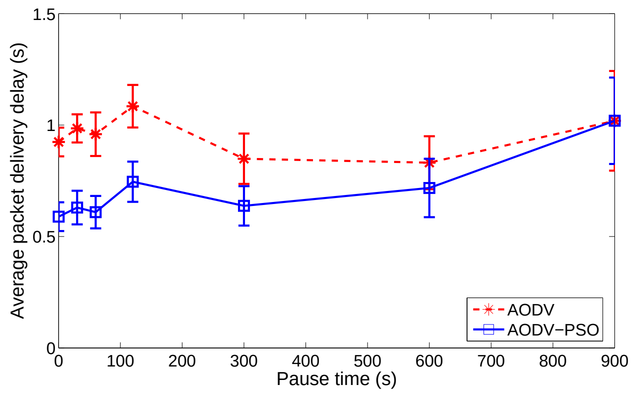

Fig. 1. Average data packet delivery delay *versus* Pause time.

Although the PSO has been used only to minimize the average delay, the other performance measures were positively influenced by the weights matches as shown by the following results.

Figure 2 shows that the AODV-PSO throughput outperforms the original AODV implementation.This can be explained by the fact that, considering the PSO function implemented, better decisions in relation to nodes capable of doing repairs are happening. That is, good routes are being established for the data to quickly flow to their destination, thus increasing the throughput.

In Figure 3, it is observed that the number of control packets sent by AODV-PSO is less than the amount of packets sent by the original AODV protocol. This is a result of the new repair mechanism chosen, since it is successful in the first attempts to fix routes; hence, lowering traffic control packets across the network.

All results show performance improvement due to the use of connectivity and PSO algorithm for the route recovery decision

[4]In [15], [16], [17], [18] and [19] the data delivery delay was not evaluated.

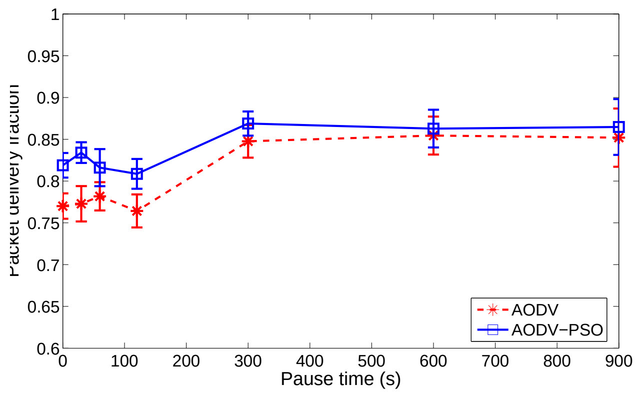

Fig. 2. Data packet delivery fraction *versus* Pause time.

in the AODV protocol for all investigated metrics. The better results were obtained for the scenarios with higher mobility, *i.e.* lower pause times, indicating that the AODV-PSO is an interesting option for mobile ad hoc networks.

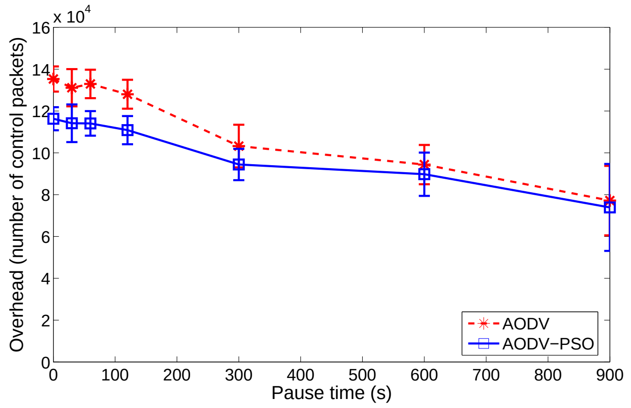

Fig. 3. Routing overhead *versus* Pause time.

V. Conclusion and Future Work

This paper proposed a new decision function to improve the performance behavior of the route recovery mechanism of the AODV protocol in ad hoc networks. We also proposed to use Particle Swarm Optimization in order to find the best set of parameters in the decision. More specifically, the AODV protocol was modified to make a decision to recover from route failure either at the source node or at the link failure predecessor node implementing the connectivity concept and using weights for each argument considered in the decision function. The values for each weight was found through PSO basic form. We observed that the PSO showed satisfactory behavior improving the performance of AODV for all metrics on the investigated scenarios. These results indicate that depending on the network topology, parameters and user application, it can be advantageous for AODV routing to use computational intelligence algorithm.

For future work, a possible extension of the present study is to analyze the application of a multi-objective PSO approach to improve several aspects simultaneously in order to maximize throughput and minimize overhead and delay.

References

[1] C. S. R. Murthy and B. S. Manoj, *Ad Hoc Wireless Networks: Architectures and Protocols*. Prentice Hall, 2006.

[2] D. B. Johnson, J. Broch and D. A. Maltz, “DSR: the dynamic source routing protocol for multihop wireless ad hoc networks,” *Ad Hoc Networking*, pp. 139-172, Addison-Wesley Longman Publishing, 2001.

[3] C. E. Perkins and E. M. Belding-Royer, “Ad hoc On-Demand Distance Vector Routing,” in *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, USA, Feb. 1999.

[4] C. E. Perkins, E. M. Belding-Royer, S. R. Das and M. K. Marina, “Performance comparison of two on-demand routing protocols for ad hoc networks wireless networks,” *IEEE Personal Communication*, vol. 8, no.1, pp. 16-28, 2001.

[5] C. E. Perkins, E. M. Belding-Royer and S. R. Das, “Ad Hoc On-Demand Distance Vector Routing.” *Request for Comments*: 3561, July 2003.

[6] N. C. V. N. Pereira and R. M. de Moraes, “A Comparative Analysis of AODV Route Recovery Mechanisms in Wireless Ad Hoc Networks,” in *Proc. of IEEE Latin-American Conference on Communications (LatinCom)*, Medellin, Colombia, Sept. 2009.

[7] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proc. of IEEE International Conference on Neural Networks*, Perth, Australia, Dec. 1995.

[8] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols,” in *Proc. of ACM/IEEE International Conference on Mobile Computing and Networking*, Dallas, TX, USA, Oct. 1998.

[9] E. M. Belding-Royer and C. K. Toh, “A review of current routing protocols for ad hoc mobile wireless networks.” *IEEE Personal Communications*, vol. 6, no. 2, pp. 46-55, 1999.

[10] K. Kim and H. Seo, “The effects of local repair schemes in AODV - Based Ad Hoc Networks,” *IEICE Transactions on Communications*, vol. E87-B, no.9, pp. 2456-2466, 2004.

[11] J. Youn, J. Lee, D. Sung and C. Kang, “Quick local repair scheme using adaptive promiscuous mode in mobile ad hoc networks,” *Journal of Networks*, vol.1, no.1, pp. 1-11, 2006.

[12] J. Liu and C. Richard Lin, “RBR: refinement-based route maintenance protocol in wireless ad hoc networks,” *Computer Communications*, vol. 28, no. 8, pp. 908-920, 2005.

[13] M. K. Marina and S. R. Das, “On-demand multipath distance vector routing in ad hoc networks,” in *Proc. International Conference on Network Procotols (ICNP)*, Riverside, CA, USA, Nov. 2001.

[14] http://www.isi.edu/nsnam/ns/doc/, access in July 2012.

[15] M. Pan, S. Chuang and S. Wang, “Local repair mechanisms for on-demand routing in mobile ad hoc networks,” in *Proc. of Pacific Rim International Symposium on Dependable Computing*, Changsha, China, Dec. 2005.

[16] H. Liu and D. Raychaudhuri, “Label switched multi-path forwarding in wireless ad-hoc networks,” in *Proc. of IEEE International Conference on Pervasive Computing and Communications Workshops*, Seattle, WA, USA, Mar. 2005.

[17] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi, “A Framework for Reliable Routing in Mobile Ad Hoc Networks,” in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, San Francisco, CA, USA, Mar. 2003.

[18] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, “Highly-resilient, energy-efficient multipath routing in wireless sensor networks,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 1, no. 2, pp. 11-25, Oct. 2001.

[19] Z. Kai, W. Neng and L. Ai-fang, “A new AODV based clustering routing protocol,” in *Proc. of Wireless Communications, Networking and Mobile Computing (WCNM)*, Wuhan, China, Sept. 2005.

[20] D. Bratton and J. Kennedy, “Defining a Standard for Particle Swarm Optimization,” in *Proc. of IEEE Swarm Intelligence Symposium*, Honolulu, HI, USA , April 2007.

[21] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in *Proc. of IEEE International Conference on Evolutionary Computation*, Anchorage, AK, USA, 1998.

[22] T. Camp, J. Boleng, and V. Davies, “A Survey of mobility models for ad hoc network research,” *Wireless Communication and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483-502, 2002.

[23] M. Reyes-sierra and C. A. C. Coello, “Multi-Objective particle swarm optimizers: A survey of the state-of-the-art,” *International Journal of Computational Intelligence Research*, v. 2, no. 3, pp. 287-308, 2006.

Nadilma C. V. N. Pereira Currently doing Ph.D. in Computer Science at Federal University of Pernambuco (UFPE). She received the M.Sc. degrees in Computer Engineering from University of Pernambuco (UPE) in 2010. Graduated in Computer Science from the Catholic University of Pernambuco (UNICAP) in 2007. She has experience in Computer Science, with an emphasis on networks Computers and Computational Intelligence, acting on the following topics: ad hoc networks, routing and Computational Intelligence.

Carmelo J. A. Bastos-Filho Was born in Recife, Brazil, in 1978. He received the B.Sc. in electronics engineering and the M.Sc. and Ph.D. degrees in electrical engineering from Federal University of Pernambuco (UFPE) in 2000, 2003, and 2005, respectively. In 2006, he received the best Brazilian thesis award in electrical engineering. His interests are related to optical networks, swarm intelligence, evolutionary computation, multiobjective optimization, and biomedical applications. He is currently an Associate Professor at the Polytechnic School of the University of Pernambuco. He is the head of the research division of the Polytechnic School of Pernambuco and coordinates the masters course on systems engineering. He is an IEEE senior member and a Research fellow of the National Research Council of Brazil (CNPq).

Renato M. de Moraes Received the Bachelor degree in Electrical Engineering from Federal University of Pernambuco, Brazil in 1996, and the Master degree from State University of Campinas (UNICAMP), São Paulo, Brazil in 1998. He obtained his Ph.D. degree in EE from University of California, Santa Cruz, USA in 2005. Currently, Renato is an Assistant Professor at the Department of Electrical Engineering at the University of Brasília, Brazil. Renato has served as TPC member for IWCMC 2007 and 2008, IEEE WCNC 2010, 2011 and 2012, IEEE ICCC 2012, IEEE PIMRC 2012, publication chair for IEEE SPAWC 2011 and Technical Program Committee Chair for SBrT 2012. He has published more than 50 publications. His areas of interest include ad hoc networks, information theory and communication networks.