2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

An Ant Colony Optimization Algorithm for Multi-objective Clustering in Mobile Ad Hoc Networks

Chung-Wei Wu, Tsung-Che Chiang, and Li-Chen Fu

***Abstract*—Due to the proliferation of smart mobile devices and the developments in wireless communication, mobile ad hoc networks (MANETs) are gaining more and more attention in recent years. Routing in MANETs is a challenge, especially when the network contains a large number of nodes. The clustering technique is a popular method to organize the nodes in MANETs. It divides the network into several clusters and assigns a cluster head to each cluster for intra- and inter-cluster communication. Clustering is NP-hard and needs to consider multiple objectives. In this paper we propose a Pareto-based ant colony optimization (ACO) algorithm to deal with this multiobjective optimization problem. A new encoding scheme is proposed to reduce the size of search space, and a new decoding scheme is proposed to generate high-quality solutions effectively. Experimental results show that our approach is better than several benchmark approaches.**

I. Introduction

With the rise of the fourth generation (4G) communication standards and the growing usage of mobile computing devices such as Personal Digital Assistants (PDA) and smart phones, MANETs have become more and more popular in recent years. MANET is a self-organizing and self-configuring multi-hop wireless network, which comprises of a set of mobile hosts that can move around freely and cooperate to pass data between each other without fixed infrastructures. It has the advantages of low cost, plug-and-play convenience, and flexibility. It facilitates wireless communication in places where fixed infrastructures are hard to be set up, for example in battle fields, the steep mountain areas, and temporary conferences constructed by mobile devices.

An MANET is often constructed by hundreds of mobile devices in the real world, and it encounters the scalability problem in the traditional flat form [1]. In the flat form each node needs to record the information (node ID, location, etc.) of all other nodes in the network, and it is hard to maintain all the information when the number of nodes increases. Clustering leads to a hierarchical form. Nodes inside a cluster only need to keep the information of the cluster head. This helps to deal with the scalability problem.

Chung-Wei Wu is with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. (e-mail: wsrts@hotmail.com).

Tsung-Che Chiang is with the Department of Computer Science and Information Engineering, National Taiwan Normal University, Taipei, Taiwan, R.O.C. (e-mail: tcchiang@ieee.org).

Li-Chen Fu is with the Department of Computer Science and Information Engineering and the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. (e-mail:lichen@ntu.edu.tw).

A cluster structure facilitates the spatial reuse of resources to increase the system capacity [2]. With the non-overlapping multi-cluster structure, two clusters may deploy the same frequency or code set if they are not neighboring clusters. When passing data, the cluster heads will form a relatively stable virtual backbone for communication [3]. Fig. 1 is an illustration of the virtual backbone. Data route on the virtual backbone, and only cluster heads participate in the routing process. The cluster structure makes the network topology easier to maintain [4]. When a node changes its associated cluster, only the nodes in the affected clusters need to update the information. Hence, we can reduce the cost of transmitting messages for information update.

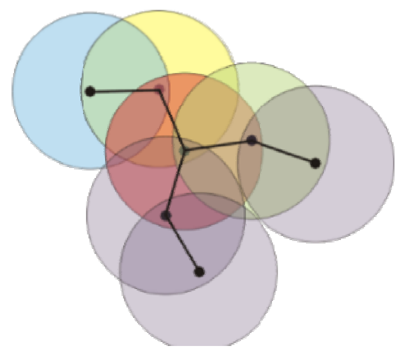

Fig. 1. Virtual backbone: black points represent the cluster heads.

Clustering is an NP-hard [5] and multiobjective optimization problem. Most past works only focused on a single objective in solving the clustering problem. The single-objective algorithms cannot deal with multiple objectives easily since improving one objective usually degrading other objectives. Some works tried to optimize multiple objectives by aggregating them into a single value through weighted sum or some scalarizing functions. This kind of algorithm places burden on users to determine the scalarizing function and the weights of objectives.

In this paper we propose a Pareto-based ACO algorithm to solve the multiobjective clustering problem. We propose an encoding scheme to reduce the size of search space and a decoding scheme to generate high-quality solutions effectively. The rest of this paper is organized as follows: Section II will give the problem formulation. Section III will review related literature. The proposed algorithm will be detailed in Section IV. Section V will present experiments and results, and finally conclusions will be made in Section VI.

II. Problem Formulation

In this section, we will describe the problem and define the objectives. To solve the clustering problem of MANET, we want to separate the mobile nodes into several clusters and assign a cluster head for each cluster. We consider three objectives in this study.

First, the number of clusters/cluster heads (one cluster has only one cluster head) is to be minimized to make the virtual backbone compact and to reduce the number of hops during communication. Equation (1) represents the first objective, where S denotes the set of cluster heads.

$$f_1: \quad \text{Minimize } |S| \tag{1}$$

Second, we want to balance the number of nodes covered by each cluster head. When a cluster contains too many nodes, the cluster head may use up its power fast due to frequent communication. Besides, there is a high probability of collision in a large-size cluster when the nodes inside want to transmit messages. In (2), n_i is the number of nodes within the cluster i and μ is the ideal number of nodes in each cluster. The value of μ is defined by ($N/|S| - 1$), where N is the number of nodes in whole network.

$$f_2: \quad \text{Minimize } \sum_i (n_i - \mu)^2 \tag{2}$$

The third objective is to minimize the total power consumption. Power consumption is proportional to the distance between two communicating nodes. We define the power consumption D_v of a cluster head v in (3), where v' is a cluster member of v. We want to minimize the total power consumption of the network, as defined in (4).

$$D_v = \sum_{v' \in cluster(v)} Distance(v, v') \tag{3}$$

$$f_3: \quad \text{Minimize } D_t = \sum_{v \in S} D_v \tag{4}$$

A multiobjective minimization problem is formulated as (5), where Ω denotes the solution space, x denotes a solution, $f_i(x)$ denotes the i^{th} objective function, and M denotes the number of objectives.

$$\text{Minimize } F(x) = \{f_1(x), f_2(x), \dots f_M(x)\} \; x \in \Omega, \tag{5}$$

We say that solution *A dominates* solution *B* if and only if *A* is not worse than *B* for all objectives and is better than *B* for at least one objective. A solution is *Pareto optimal* if it is not dominated by any other solution. In other words, we cannot improve any objective of a Pareto optimal solution without degrading any other objective. The set of Pareto optimal solutions forms the *Pareto optimal set*, and the set of objective vectors of the Pareto optimal set is the *Pareto front*. In this study we will propose a Pareto-based ACO algorithm to solve the 3-objective clustering problem. Our goal is to find the Pareto optimal set. Users do not need to determine the scalarizing functions and objective weights. They can check the trade-off between objectives and select the favorite solution among the Pareto optimal set.

III. Literature Review

Many clustering algorithms have been proposed in the literature. We classify them into three categories according to the number of concerned objectives and how they dealt with the objectives.

The first category of work focused on a single objective, for example, minimization of total power consumption. An and Papavassiliou [6] proposed a method based on node's mobility. They assigned nodes with lower mobility to be cluster heads. Ho and Ewe [7] did load-balanced clustering based on the ACO algorithm, and Cheng et al. [8] resorted to the genetic algorithm (GA). Bagci and Yazici [9] proposed an energy-aware unequal clustering approach to minimize energy consumption.Watfa et al. [10] proposed a battery-aware reliable clustering algorithm, which assigned the nodes with the maximal battery power as cluster heads to avoid depleting power too fast. Lo et al. [11] proposed a multi-head clustering algorithm. They found a slave cluster head to share the load of master head to avoid the power of the master cluster head being used up too fast. These works performed well in the single-objective problem, but they are not suitable for the multiobjective scenario.

The second category of algorithm dealt with the multiobjective clustering problem by a scalarizing function. Chatterjee et al. [12] proposed a weighted clustering algorithm (WCA), whose main idea is to sum up all objectives into a single value and always choose the node with the highest value as a cluster head. Turgut et al. [13] proposed a GA to optimize the WCA algorithm, and later Turgut et al. [14] used simulate annealing to optimize WCA. Shahzad et al. [15] proposed a comprehensive learning particle swarm optimization to choose cluster heads. Chakraborty et al. [16] proposed a clustering algorithm based on differential evolution. Thirumurugan and Raj [17] proposed an efficient weighted mechanism for ad hoc networks. They used Manhattan distance instead of Euclidean distance to the evaluation step. Zhao et al. [18] added a new objective called communication load of node and intended to find nodes with low communication load as cluster heads. Although this category of algorithm took multiple objectives into consideration, they just combined the objectives into a single value through weighted sum or some scalarizing functions and then optimized the aggregated value. Summing the objective values with very different ranges makes the meaning of the sum unclear, and setting objective weights is also a difficult job for users.

The third category of work is based on Pareto dominance and gets more attention in recent years. Cheng et al. [19] proposed a stability-based clustering algorithm. It maintained the relative stable neighbors (the nodes staying within the transmission range for a certain time duration) of each node and used the well-known algorithm SPEAII [20] to deal with multiple objectives. Wang and Hung [21] proposed the reliable node clustering algorithm based on scatter search. Ali et al. [22] proposed an energy efficient clustering method based on the particle swarm optimization algorithm.

IV. PROPOSED ALGORITHM

We propose a multiobjective ACO (MOACO) to address the clustering problem in MANETs. The key innovation is the proposal of the encoding and decoding schemes. Compared with the commonly-used permutation encoding, our scheme can reduce the size of search space, avoid redundant solutions, and generate solutions that the permutation encoding scheme cannot generate.

A. Encoding

The encoding scheme transforms the solution space into the search space. The permutation encoding scheme was usually used in past studies, where each integer in the permutation sequence represents the ID of a node. This encoding way results in $O(N!)$ search space size, where N is the number of nodes in the network.

We propose a bit-string encoding scheme. A solution is encoded as a bit string of length N, in which 1 means a node is chosen to be a cluster head and 0 means the node is not. The encoding scheme decreases the search space size from $O(N!)$ to $O(2^N)$. In our MOACO, we associate pheromone with the nodes in the network. The amount of pheromone τ_v specifies the probability of a node v to be selected as a cluster head. In other words, the v^{th} bit in the bit string is set by 1 in probability τ_v when we generate a solution. Pheromone update will be detailed in sub-section IV-E.

B. Decoding

The decoding scheme transforms a solution from the search space back to the solution space. The common decoding scheme for the permutation encoding scheme simply chooses the nodes as cluster heads one by one in the order they appear in the permutation. Each time a node is chosen as a cluster head, all nodes within its transmission range are made as its cluster members. One drawback of this decoding scheme is when some nodes are within the transmission range of the chosen cluster head, they will lose the opportunity to be chosen as cluster heads even though that may be a better solution. Fig. 3 is an example. When we choose node 3 as a cluster head, node 4 will lose the opportunity to be chosen as a cluster head. This forces node 5 and node 6 to be chosen as cluster heads. We are unable to find the solution in Fig. 4, which needs only two cluster heads. Another drawback is that different permutations may generate identical solutions. In Fig. 3, permutations [3 1 2 4 5 6] and [3 2 5 4 6 1] will be decoded to the same solution because we choose node 3 as a cluster head, delete its members {1, 2, 4}, and finally choose node 5 then node 6.

In this paper we propose a decoding scheme to improve the quality of solutions. Our decoding scheme takes only the nodes which are not tagged as 1 in the bit string as cluster members when a cluster head is chosen. The bit string [0 0 1 1 0 0] will be decoded as the solution in Fig. 4.

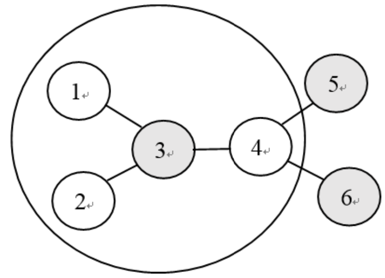

Fig. 3. Node 4 will lose the opportunity to be chosen as cluster head in the traditional decoding way.

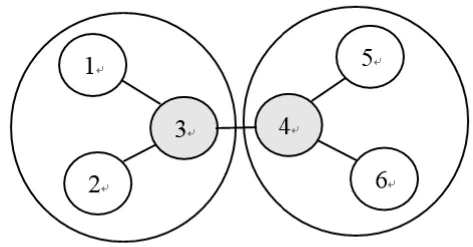

Fig. 4. A solution with fewer cluster heads and more balanced cluster sizes than the solution in Fig. 3.

C. Repair function

In our encoding scheme a bit string [0 0 0 0 0 0] will be decoded to an infeasible solution (where no node serves as a cluster head). We propose a repair algorithm to make every bit string correspond to a feasible solution (where all nodes in the network are either cluster heads or are covered by a cluster head). Algorithm I shows the pseudo code.

The first phase of the repair algorithm keeps choosing the uncovered node with the lowest weight value defined in (6) as a cluster head until all nodes are covered. The weight of a node v is the weighted sum of the normalized power consumption and the normalized load deviation. D_v is the power consumption of cluster head v; D^{max} is the maximal value of power consumption among all nodes; $load_v$ represents the number of members of cluster v; th is median of load among all nodes, and L^{max} is the maximal value of load among all nodes.

$$weight_v = \alpha \cdot \frac{D_v}{D^{max}} + \beta \cdot \frac{(load_v - th)}{L^{max}} \tag{6}$$

The second phase finds and removes the redundant cluster heads. We say a cluster head is *redundant* if it and all its members can be covered by another cluster head. Fig. 5 is an example, where node 4 and its member (node 5) can be covered by node 2. In this case node 4 is redundant. We keep deleting the redundant cluster heads with the highest weight value in (6) until there is no redundant head. The repair algorithm makes our solutions not only feasible but also better quality.

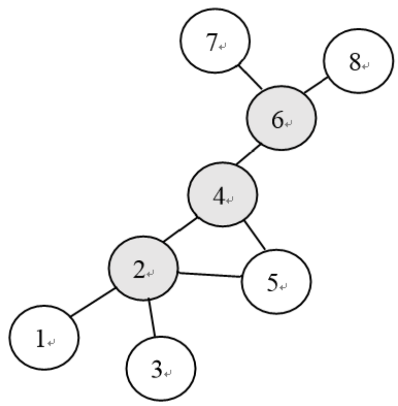

Fig. 5, Colored nodes stand for cluster heads. Node 4 and its member (node 5) can be covered by node 2. Thus, node 4 is considered as redundant in our repair algorithm.

ALGORITHM I. REPAIR ALGORITHM

/ make the solution valid */*
Decode the bit string. Let S be the set of cluster heads.
While (S can't cover the whole network)
  Select the uncovered node v_b with min $\{weight_v\}$ into S
  Mark the nodes within the transmission range of v_b as covered
End while

*/*remove redundant cluster heads */*
Identify the set of redundant cluster heads R.
While (R is not empty)
  Delete the node v_w with max $\{weight_v\}$ from R and S
  Re-judge all the nodes in R after deleting node v_w
    / When a node is deleted, some nodes in R may not be redundant anymore. Fig. 6 is an example. */*
  Remove the non-redundant nodes from set R
End while

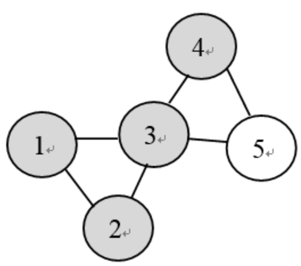

Fig. 6. Nodes 1, 2, 3, and 4 are cluster heads. Each of them is *redundant* according to the definition in Section IV-C.

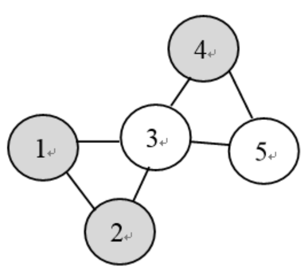

Fig. 7. After node 3 is removed from the set of cluster heads, node 4 is not redundant anymore. (If node 4 is deleted, node 5 cannot be covered by any cluster head.)

D. Fitness evaluation

After generating solutions, we need to identify which solutions are good and which are bad. We do non-dominated sorting [23] to rank solutions. Fitness of a solution x is defined by (7) and (8), where P denotes the set of solutions.

$$fitness(x) = R_x / \sum_{y \in P} R_y \tag{7}$$

$$R_x = \max_{y \in P} \{rank(y)\} - rank(x) \tag{8}$$

E. Pheromone update

Pheromone update is the most important part in the ACO algorithm. In the nature, a shorter path accumulates more pheromone to attract more ants to follow; in the optimization process, components in a higher-quality solution should also accumulate more pheromone so that these components will appear more frequently in generating new solutions. We associate pheromone with nodes in the network. The amount of pheromone represents the probability of a node to be chosen as a cluster head. Let $\tau_v(t)$ denote the amount of pheromone of node v at generation t. The amount of pheromone is updated by (9).

$$\tau_v(t) = \tau_v(t-1) \cdot \rho + \Delta\tau_v, \tag{9}$$

where ρ is the evaporation factor and $\Delta\tau_v$ is the pheromone deposited on node v.

$$\Delta\tau_v = \frac{\sum_{x \in one_v} fitness(x)}{|one_v|} - \frac{\sum_{x \in zero_v} fitness(x)}{|zero_v|} \tag{10}$$

In (10), one_v is the set of solutions in which node v is a cluster head, and $zero_v$ is the set of solutions in which node v is not. More specifically, the change of the amount of pheromone on a node v is the difference between the average fitness of the solutions which choose v as a cluster head and the average fitness of the solutions which do not. If choosing v as a cluster head leads to better solutions, the pheromone on v will increase.

Algorithm II gives the main steps of the proposed MOACO algorithm.

ALGORITHM II. PROPOSED MOACO ALGORITHM

$g = 0$
While ($g < G$)
  Step1: $g = g + 1$. Generate population P_g based on pheromone information.
  Step2: Use Algorithm I to make each solution feasible.
  Step3: Do non-dominated sorting to rank solutions.
  Step4: Evaluate each solution using fitness function in (7).
  Step5: Update pheromone.
End while
Output the non-dominated solutions in P_g.

V. EXPERIMENTS AND RESULTS

A. Experiment simulation

We simulated a MANET with N nodes in an $M \times M$ grid to evaluate the performance of algorithms. We generated six scenarios, where N was in {100, 200, 300} and M was in {100, 200}. Transmission range of a node was set by $M/10$. Three benchmark algorithms were implemented. The WCA-based algorithm greedily chooses the node with the minimal value of the weight defined in (6) as the cluster head. We set α and β by 0.7 and 0.3, respectively; th was set by $M/20$. The GA used $f(x) = 0.3 \cdot f'_2(x) + 0.7 \cdot f'_3(x)$ as the fitness function, where $f_i'(x)$ is the normalized value of $f_i(x)$. The WSACO is the same as our proposed MOACO with the exception that WSACO also took the above-mentioned linear weighted sum as the fitness function. Table I shows the parameter setting of our MOACO in the experiment.

TABLE I. PARAMETER SETTING OF MOACO

Parameter	Values
evaporation factor ρ	0.95
initial pheromone $\tau_v(1)$	0.2
α	0.5
β	0.5
th	$M/20$
number of ants	100
number of generations	100

B. Results analysis

For each scenario, we run each of the four tested algorithms five times and recorded the average objective values. In our MOACO we chose the best solution from the non-dominated solutions in terms of the same weighted sum function $(0.3 \cdot f'_2(x) + 0.7 \cdot f'_3(x))$ for fairness. Tables II and III show the results of scenarios of 100×100 and 200×200 grids, respectively. The best objective value among the compared algorithms is marked in bold.

Our MOACO performs well in f_1 (number of clusters) and f_2 (load balance) in most cases. WSACO sets a high weight on f_3 and thus outperforms MOACO in this objective. We ran WSACO with four different sets of objective weights and compared these two algorithms in Table IV. Performance of MOACO dominates that of WSACO in all four cases. The results show that performance of WSACO is very sensitive to the weight setting. In contrast, MOACO provides good performance without the need of weights. Besides, MOACO can provide a set of solutions for users to choose. Fig. 8 gives an illustration of the distribution of non-dominated solutions on the objective space.

VI. CONCLUSION

In this paper we proposed an ACO algorithm to address the multiobjective clustering problem in MANETs. We proposed bit-string encoding and decoding schemes to reduce search space size, avoid duplicate solutions, and generate high-quality solutions effectively. We considered three objectives simultaneously, ranked solutions based on Pareto dominance, and deposited pheromone based on ranks. Performance of the proposed MOACO was compared with three algorithms which evaluated solutions by weighted sum. Experimental results showed that our MOACO can find a set of high-quality solutions without the need of objective weights. Our algorithm can solve 300-node problems within one second and is applicable in dynamic environments.

In our future work, we will investigate the impact of parameter values on the algorithm performance. We will also elaborate the fitness function to deal with multiple objectives. The proposed encoding and decoding schemes are not specific to ACO, and we will examine their effects in other metaheuristics such as particle swarm optimization.

TABLE II. PERFORMANCE OF FOUR TESTED ALGORITHMS IN 100×100 GRID WITH 100, 200, AND 300 NODES ($w_2 = 0.3,\ w_3 = 0.7$)

N	Algorithm	f_1	f_2	f_3
100	WCA	43.2	145.2	415.2
	GA	**36.4**	120	399.4
	WSACO	40.4	49.8	**368**
	MOACO	37.6	**46.4**	387.6
200	WCA	53.6	138.4	1012.4
	GA	47.2	111.2	**976.8**
	WSACO	49.8	97.6	**976.8**
	MOACO	**45**	**91.2**	989.4
300	WCA	57.2	174.2	1685.2
	GA	49.4	151.2	1590.2
	WSACO	52.4	157.8	**1583.8**
	MOACO	**48.4**	**143.8**	1594.8

TABLE III. PERFORMANCE OF FOUR TESTED ALGORITHMS IN 200×200 GRID WITH 100, 200, AND 300 NODES ($w_2 = 0.3,\ w_3 = 0.7$)

N	Algorithm	f_1	f_2	f_3
100	WCA	46.2	145.2	803.6
	GA	37.6	126.4	759.2
	WSACO	42.8	50.4	**728.2**
	MOACO	**35.2**	**44.4**	775.8
200	WCA	58.4	245.2	1956.2
	GA	49	216.2	1876
	WSACO	53.8	105.4	**1851.4**
	MOACO	**46.8**	**101**	1887.8
300	WCA	62.2	332.2	3196.4
	GA	52.4	316.2	3071.6
	WSACO	58	159.2	**3065.2**
	MOACO	**50.4**	**158**	3111.2

TABLE IV. PERFORMANCE OF WSACO AND MOACO IN 100×100 GRID WITH 100 NODES USING DIFFERENT OBJECTIVE WEIGHTS

w_2	w_3	Algorithm	f_1	f_2	f_3
0.4	0.6	WSACO	38.2	43.4	404.2
		MOACO	**36.6**	**41**	**398.6**
0.5	0.5	WSACO	37.2	42.8	412.4
		MOACO	**36.2**	**38.8**	**402.6**
0.6	0.4	WSACO	37.8	42	408.6
		MOACO	**36**	**40**	**407**
0.7	0.3	WSACO	37.4	42.6	426.8
		MOACO	**36.6**	**41**	**403**

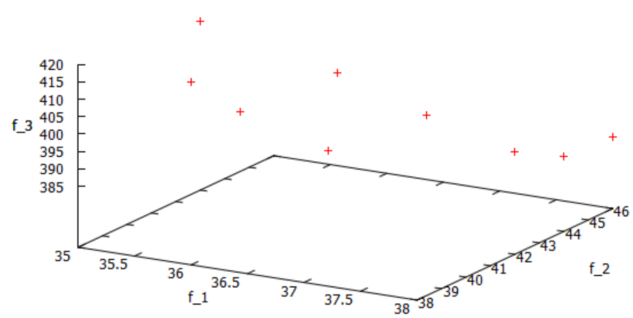

Fig. 8. Distribution of non-dominated solutions on the objective space.

ACKNOWLEDGMENT

The authors are grateful to the anonymous referees for their insightful comments. This research was sponsored by the Ministry of Science and Technology, Taiwan, under grant 102-2627-E-002-001.

REFERENCES

[1] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," IEEE Communications Surveys and Tutorials, vol. 7, pp. 32-48, 2005.

[2] T.-C. Hou and T.-J. Tsai, "A access-based clustering protocol for multihop wireless ad hoc networks," IEEE Journal on Selected Areas in Communications, vol. 19, pp. 1201-1210, 2001.

[3] U. C. Kozat, G. Kondylis, B. Ryu, and M. K. Marina, "Virtual dynamic backbone for mobile ad hoc networks," in IEEE International Conference on Communications., 2001, vol.1, pp. 250-255.

[4] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T.-W. Chen, "Scalable routing strategies for ad hoc wireless networks," IEEE Journal on Selected Areas in Communications, vol. 17, pp. 1369-1379, 1999.

[5] S. Basagni, I. Chlamtac, F. Andras, and E. Jonsson, "A generalized clustering algorithm for peer-to-peer networks," in Workshop on Algorithmic Aspects of Communication, 1997.

[6] B. An and S. Papavassiliou, "A mobility-based clustering approach to support mobility management and multicast routing in mobile ad-hoc wireless networks," Int. J. Netw. Manag., vol. 11, pp. 387-395, 2001.

[7] C. K. Ho and H. T. Ewe, "A hybrid ant colony optimization approach (hACO) for constructing load-balanced clusters," in IEEE Congress on Evolutionary Computation, vol. 3, pp. 2010-2017, 2005.

[8] H. Cheng, S. Yang, and J. Cao, "Dynamic genetic algorithms for the dynamic load balanced clustering problem in mobile ad hoc networks," Expert Systems with Applications, vol. 40, pp. 1381-1392, 2013.

[9] H. Bagci and A. Yazici, "An energy aware fuzzy approach to unequal clustering in wireless sensor networks," Applied Soft Computing, vol. 13, pp. 1741-1749, 2013.

[10] M. K. Watfa, O. Mirza, and J. Kawtharani, "BARC: A Battery Aware Reliable Clustering algorithm for sensor networks," Journal of Network and Computer Applications, vol. 32, pp. 1183-1193, 2009.

[11] S.-C. Lo, Y.-J. Lin, and J.-S. Gao, "A Multi-Head Clustering Algorithm in Vehicular Ad Hoc Networks," International Journal of Computer Theory & Engineering, vol. 5, pp. 242-247, 2013.

[12] M. Chatterjee, S. K. Das, and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks," Cluster Computing, vol. 5, pp. 193-204, 2002.

[13] D. Turgut, S. K. Das, R. Elmasri, and B. Turgut, "Optimizing clustering algorithm in mobile ad hoc networks using genetic algorithmic approach," in IEEE Global Telecommunications Conference, vol.1, pp. 62-66, 2002.

[14] D. Turgut, B. Turgut, R. Elmasri, and T. V. Le, "Optimizing clustering algorithm in mobile ad hoc networks using simulated annealing," in IEEE Wireless Communications and Networking, vol.3, pp. 1492-1497 2003.

[15] W. Shahzad, F. Khan, and A. Siddiqui, "Clustering in Mobile Ad Hoc Networks Using Comprehensive Learning Particle Swarm Optimization (CLPSO)," in Communication and Networking. vol. 56, 2009, pp. 342-349.

[16] U. K. Chakraborty, S. K. Das, and U. T. E. Abbott, "Clustering in mobile ad hoc networks with differential evolution," in IEEE Congress on Evolutionary Computation 2011, pp. 2223-2228.

[17] S. Thirumurugan and E. G. D. P. Raj, "W-PAC: an efficient weighted partitioning around cluster head mechanism for ad hoc network," presented at the Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology, Coimbatore UNK, India, 2012.

[18] X. Zhao, W. N. N. Hung, Y. Yang, and X. Song, "Optimizing communication in mobile ad hoc network clustering," Computers in Industry, vol. 64, pp. 849-853, 2013.

[19] H. Cheng, J. Cao, X. Wang, and S. K. Das, "Stability-based multi-objective clustering in mobile ad hoc networks," presented at the Proceedings of the 3rd international conference on Quality of service in heterogeneous wired/wireless networks, Waterloo, Ontario, Canada, 2006.

[20] E. Zitzler, M. Laumanns, L. Thiele, E. Zitzler, E. Zitzler, L. Thiele, et al., "SPEA2: Improving the strength Pareto evolutionary algorithm," ed: Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK), 2001.

[21] T. Wang and W. N. N. Hung, "Reliable Node Clustering for Mobile Ad Hoc Networks," Journal of Applied Mathematics, vol. 2013, Article ID 285967, 8 pages, 2013.

[22] H. Ali, W. Shahzad, and F. A. Khan, "Energy-efficient clustering in mobile ad-hoc networks using multi-objective particle swarm optimization," Applied Soft Computing, vol. 12, pp. 1913-1928, 2012.

[23] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002.