

A New Method for Multi-objective TDMA Scheduling in Wireless Sensor Networks Using Pareto-Based PSO and Fuzzy Comprehensive Judgement

Tao Wang^{1,2}, Zhiming Wu², and Jianlin Mao²

¹ School of Electrical and Computer Engineering
Georgia Institute of Technology, USA
twang31@mail.gatech.edu

² Department of Automation
Shanghai Jiao Tong University, 200240, Shanghai, China
{reno_wang0823,ziminwu,jlmao}@sjtu.edu.cn

Abstract. In wireless sensor networks with many-to-one transmission mode, a multi-objective TDMA (Time Division Multiple Access) scheduling model is presented, which concerns about the packet delay and the energy consumed on node state transition. To realize the scheme, a mapping between the problem and evolutionary algorithm is reasonably set up. A multi-objective particle swarm optimization based on Pareto optimality (PAPSO) is then proposed to solve such multi-objective optimization problem and find a better tradeoff between time delay and energy consumption. The simulation results validate the effectivity of PAPSO algorithm and also show that PAPSO outperforms other techniques in the literature.

1 Introduction

Large-scale networks of wireless sensors are becoming a hot topic of research due to their potential usage in defense, pervasive commercial and scientific applications. In such networks, sensors are units with sensing, processing, and wireless networking capability. They can automatically collect information and report the results to an access point. However, as the sensors are usually battery-powered, saving energy becomes an essential problem in sensor networks.

The medium access control (MAC) method is a major consumer of sensor energy [2]. Different medium access methods result in different time and energy efficiencies. Among those proposed MAC protocols, including contention based access and contention free access, TDMA is a suitable access method for wireless sensor networks. First, TDMA can save energy by eliminating collisions, avoiding idle listening, or entering inactive states until being allocated time slots. Secondly, as a collision-free access method, TDMA can bound the delays of packets and guarantee reliable communication.

However, allocating time slots to each sensor to finish a couple of data collection tasks is an NP-complete problem [3]. Additionally the energy constraint makes the problem more difficult to solve. Therefore, in sensor networks, the main challenge of TDMA is how to allocate time slots to each node to minimize the energy consumption and time delay.

As to the aspect of TDMA scheduling against time performance, some references have studied how to minimize packet delay [4], how to improve fairness [5], how to maximize parallel operation [6], and how to shorten the total slot cost to finish a set of transmission tasks [7]. By setting up a convex optimization model, Cui et al. [4] adopted relaxation methods to solve the problem, and got a pareto optimal energy-delay curves, where the power consumption on switching was neglected. Sridharan et al. [4] developed a linear programming formulation and presented a distributed solution, which outperformed than random MAC in terms of fairness and delay etc. To minimize the overall transaction time, which was modeled as a graph partitioning problem, Gandham et al. [5] proposed a distributed edge-coloring algorithm. In order to save time for data collection, Ergen et al. [6] proposed three algorithms based on coloring method in graph theory. However, these two references did not consider the energy saving problem in sensor networks.

In this paper a practical hierarchical solution approach is proposed to solve the multi-objective TDMA scheduling problem. Given the strong search ability in combinatorial optimization, particle swarm optimization (PSO) is introduced. And to reach multi-objective optimality, Pareto optimization serves as evaluation criterion of candidate solutions during the evolutionary process of PSO, by means of which we can get Pareto optimal solutions to TDMA scheduling problem. In this sense, the whole framework of the method can deal with several constraints flexibly and reach a multi-objective optimal slot-allocation scheme.

The remainder of the paper is organized as follows: the problem statement is introduced in section 2; the part of optimization algorithm, including coding method and evaluation system of PSO solutions, is expatiated in section 3; and the computational results are given in section 4; and section 5 gives some final conclusions.

2 Problem Statement

2.1 Network and Scheduling Model

From a viewpoint of network, a sensor network can be represented by an undirected graph $G = (V, E)$, where V represents the set of all sensors in the network and $E \subset V \times V$ represents the set of communication links between a pair of nodes. There is one access point (AP) in V . All traffic generated at sensors are destined for AP, composing a routing tree. Such a network is called many-to-one sensor network.

The distance $d(i, j)$ between nodes i and j is defined as the minimum number of edges to go from one to the other. From this definition, the topology of sensor network can be described by an $N \times N$ symmetric connectivity matrix C , which

is defined as $C_{ij} = 1$, if $d(i, j) = 1$; else $C_{ij} = 0$. In addition, the conflict relationship in the network can be described by an interference matrix $I_{N \times N}$, where if $d(i, j) \leq 2$, $I_{ij} = 1$; else $I_{ij} = 0$. As a result, node i and j can transmit data at the same time if the communication distance $d(i, j)$ is larger than 2.

In TDMA scheduling problem, time is splitted into equal intervals called time slots. Each time slot is designed to accommodate a single packet to be transmitted and received between pairs of nodes in the network. And once the routes are established, the allocation of time slots directly influences the performance of transmission in network. Moreover, TDMA also ensures collision-free communication when several transmission tasks run simultaneously. So what we need to do is to find a schedule of time slots to reach our requirements of network transmission performances, such as energy consumption and time delay.

In many-to-one sensor networks, as the sensor data flows toward AP from respective source nodes, the TDMA scheduling problem can be formulated as follows. There are a set of sensor data packets, forwarding to AP, over the routing trees, which are established using GPSR [7]. Each data-collection process that a packet flows to AP from its source node is called a task. On the established routes, each task consists of a sequence of transmission actions called subtasks, where one subtask needs one slot occupation. The aim of the problem is to determine a slot-allocation sequence of subtasks so that collision would not happen, and some optimization criteria could be satisfied.

2.2 Description of Optimization Objectives

In this paper, two optimization objectives are considered : the average energy consumption and the average time delay of data packets of sensor nodes.

To save energy, a common idea is just to switch off the radio when it is neither transmitting nor receiving. However, frequently turning on/off the radio also consumes large amounts of power, especially when the packet is small. Hence, we take into account this part of energy, which is ignored in many TDMA researches. According to Ref. [3], the formula of average consumed energy of sensor nodes is as follows:

$$AvgEngy = \frac{1}{N} \sum_{i=1}^N [P_i^{tx} \cdot (t_i^{tx} + t_i^{s-tx}) + P_i^{rx} \cdot (t_i^{rx} + t_i^{s-rx})]$$

where N denotes the number of nodes in the network, P_i^{tx} (P_i^{rx}) is the power consumption of transmitter (receiver) at node i . t_i^{tx} (t_i^{rx}) is the total work time of the transmitter (receiver) at node i . t_i^{s-tx} (t_i^{s-rx}) is the total transition time consumed between the sleep and active states.

As a performance index, the average time delay of data packets should be as small as possible, which can help those sensor nodes to increase their sampling rate to the maximum possible level. In other words, the smaller the average time delay of data packets is, the more data those nodes can collect in every unit time and more efficiently they can communicate with each other.

3 PSO-Based Energy-Delay Pareto Optimization

3.1 Standard PSO

Particle swarm optimization (PSO) is a new swarm intelligence technique proposed by Eberhart and Kennedy [8], inspired by social behavior of bird flocking or fish schooling.

The main idea of PSO is as follows. There is a population of random solutions. Each potential solution, called particle, flies through the problem space by following the current optimal particle. Flying in the search space, each particle has a velocity which is dynamically adjusted according to the experiences of its own and its colleagues. This makes the swarm have an intelligent ability of flying towards the optimal position.

The global model equations of PSO are:

$$V_{id}(t+1) = W \cdot V_{id}(t) + C_1 \cdot rand_1() \cdot (p_{id}(t) - X_{id}(t)) + C_2 \cdot rand_2() \cdot (p_{gd}(t) - X_{id}(t)) \quad (1)$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t) \quad (2)$$

$$|V_{id}| \leq V_{max} \quad (3)$$

where V_{id} and X_{id} are respectively the velocity and position of particle. p_{id} and p_{gd} respectively represent the best position of i_{th} particle and the swarm. W called inertia weight, is a user-specified parameter. A large inertia weight pressures towards global exploration while a smaller inertia weight pressures towards fine-tuning the current search area. Proper selection of the inertia weight and acceleration coefficients can provide a balance between the global and the local search. C_1 and C_2 are acceleration factors, which are usually set to 2. $rand_1$ and $rand_2$ are random numbers between (0,1). And during the recursive process, the selections of p_{id} and p_{gd} depend on the evaluation system, which will be expatiated later.

According to our TDMA problem, some parameters in PSO are defined as follows. There are N tasks and each task i includes M_i hops. The dimension of particles is set to M , the number of the total subtasks, i.e., $\sum_{i=0}^{N-1} M_i$. The border of the searching space is defined by X_{max} , which is set to $N-1$. Parameter V_{max} determines the maximum change one particle can take during one iteration, which is set as $N-1$. Under this setting, V_{id} is a value in the range $[-(N-1), N-1]$, thus the positions of the flying particles are under control.

3.2 Pareto-Based Evaluation System of PSO Solutions

In PSO, there are concepts of individual and population. During the process of iterations of the algorithm, the flying direction of particle derives from its own optimal position p_{id} and global optimal position p_{gd} of population. Consequently, the selection of p_{id} and p_{gd} directly influences the searching performance of PSO. Thus, the evaluation system of candidate solutions is critical to the whole optimization algorithm.

However, there exists confliction between time delay and energy consumption in TDMA scheduling, i.e., pursuing the optimization of power consumption on switching inextricably damages the performance of time delay index. It is mainly because to lessen state transitions and thus save energy, the good working continuity of nodes is required, which means that after collecting data from its child nodes, the sensor node better wait to transmit data packets to its own parent node instead of switching off. As a result, the performance of time delay would be damaged, and vice versa. Consequently, as to such multi-objective optimization problem in which the objectives cannot be optimized simultaneously, the concept of Pareto optimality was introduced into the evaluation system.

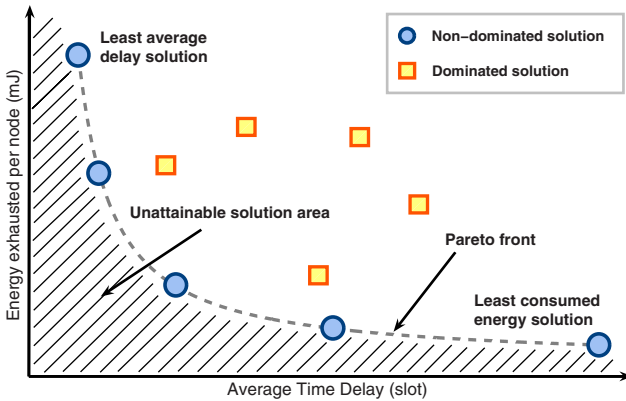


Fig. 1. Pareto frontier of candidate scheduling solutions

3.3 Pareto Optimality

In general, the multi-objective optimization problem is described as follows:

$$\begin{aligned} \min f(x) &= (f_1(x), f_2(x), \dots, f_k(x)) \\ \text{s.t. } g_i(x) &\leq 0, \quad i = 1, 2, \dots, m \end{aligned}$$

Where $x \in R^n$ is the decision vector belonging to the feasible region S , which is described as follows: $S = \{x \in R^n | g_i(x) \leq 0, i = 1, 2, \dots, m\}$

A decision vector $x_1 \in S$ is said to *dominate* a decision vector $x_2 \in S$ (denoted $x_1 \prec x_2$) iff:

- The decision vector x_1 is not worse than x_2 in all objectives, or $f_i(x_1) \leq f_i(x_2) \quad \forall i = 1, 2, \dots, q$.
- The decision vector x_1 is strictly better than x_2 in at least one objective, or $f_i(x_1) < f_i(x_2)$ for at least one $i = 1, 2, \dots, q$.

If any of above conditions is not met, then x_1 does not dominate x_2 . All the solutions that do not dominate each other compose nondominated solution set. If the

comparative space of picked solutions is the global space, then the nondominated solution set is called *Pareto-optimal* set.

In this paper, the objective functions of our TDMA scheduling algorithm are:

$$f_1 = AvgEngy, f_2 = AvgDelay \tag{4}$$

which are used in the Pareto dominance comparison as Fig.1.

Crowding-Measure-Based Maintenance of Pareto Archive. Since Pareto optimal set is often infinite, Pareto archive is used to offer available optimal solutions representing the Pareto frontier. In order to make the nondominated solutions in the archive uniformly distribute on the Pareto frontier, here crowding-measure is introduced into the maintenance of archive.

$$d_i = (d_i^1 + d_i^2)/2 \tag{5}$$

where d_i^1 and d_i^2 are the minimum two Euclidean distances between individual i and other members of archive.

The crowding measure d_i reflects the distribution of other individuals around i . The smaller d_i is, the more the number of individuals surrounding i is. Consequently during the evolutionary process, when the Pareto archive is full, we filter the member with the minimum crowding measure. By means of the maintenance, the retained members of archive would evenly distribute on the Pareto frontier.

Evaluation of Global Optimal Solution. The selection of global best position p_{gd} is crucial to the whole PSO algorithm. In multi-objective problem, the population generates several nondominated solutions, all of which can be p_{gd} . Then how to assign suitable p_{gd} for every single particle is one important task. Here, combined with the maintenance of Pareto archive, we select the global best position p_{gd} as follows:

To every newly-generated nondominated solution x_i :

- If x_i dominates some members of archive, then x_i takes the place of all the dominated ones, and let x_i be the new p_{gd} of those particles whose previous p_{gd} are the replaced;
- Else if the current archive is full, let x_i replace the least-crowding-measure solution x_l as well as its position as a p_{gd} .
- Else if the archive is not full, directly insert x_i first, and use a newly-defined variable $np(x_i)$, the number of particles whose p_{gd} is x_i , to make every member of Pareto archive at least be p_{gd} of some particles, thus ensuring the diversity of solutions:
 - a For all the solutions x_k in the archive, define $s = \min\{np(x_k)\}, k = 1, 2, \dots, M$ (M is the swarm size),
 if $s \geq g$, then $s = g$ and $np(x_i) = 0$; where $g = (0.025 \sim 0.05)M$, an index to prevent a solution from being p_{gd} of too many particles;
 - b Define $F = \{x_k | np(x_k) > s\}, u = |F|, v = 1$; “||” means the size of set.

- c Find x_k in F , closest to x_i , and let x_i be p_{gd} of one of the particles whose previous p_{gd} is x_k . $F = F \setminus \{x_k\}$, $np(x_i) = np(x_i) + 1$, $v = v + 1$;
- d If $np(x_i) < s$ and $v < u$, go to step **c**; If $np(x_i) < s$ and $v = u$, go to step **b**; If $np(x_i) = s$, then add x_i to the archive.

After the process of PSO flying iterations, the Pareto solution archive represents the Pareto-optimal solutions of the multi-objective TDMA scheduling.

Evaluation of Local Optimal Solution. As to the selection of local optimal solution p_{id} in every iteration, we take the method proposed by Coello Coello [10]:

every time the particle j gets the new position x_j and the current local optimal position p_{jd} after the j iterations, compare the dominance between p_{jd} and x_j . If $x_j \succ p_{jd}$, then the updated $p_{jd} = x_j$; else if $x_j \prec p_{jd}$, then p_{jd} does not change; else if x_j and p_{jd} do not dominate each other, then randomly pick one between them two.

3.4 PAPSO Optimization

The procedure of PAPSO optimization is as follows:

1. Initialize parameters of PSO, including max_generation_PSO, W , C_1 , C_2 , M and M' , and make Pareto archive empty;
2. Initialize Pareto solution archive:
As to all the initial solutions, calculate their fitness f_1, f_2, \dots, f_i one by one. Then according to Pareto optimality, judge their dominance with each other, add all non-inferior solutions to P' , and form an initial Pareto solution archive.
3. While (max_generation_PSO is not reached)
 { generation=generation+1;
 generate next generation of swarm by eq.(1)~(3);
 evaluate the new swarm according to the evaluation system, and update the local optimal position P_{id} of individual particle and the global optimal position P_{gd} of swarm (i.e. Pareto solution archive) }
4. When the evolutionary process is over, output the result of Pareto optimization.

3.5 Encoding and Decoding Scheme

To apply our evolutionary algorithm on TDMA time slot scheduling, encoding is a key step. The aim is to express a solution as a sequence code, which is an individual in population-based algorithms.

According to the description of the scheduling problem in section 2.1, a data collection task can be divided into several hops called subtasks in sequence. Hence, a set of data collection tasks can be viewed as a combination of all the subtasks. Therefore, we first denote a subtask as (TaskID,HopNo.), where the TaskID points

out which task the subtask belongs to, and the HopNo. gives the sequence number of this subtask, showing its position in the whole task TaskID. For example, if task 1 includes 2 hops, then the two corresponding subtasks can be denoted as (1,0) and (1,1). Observing this representation method, we take the TaskIDs out of a subtask sequence to form an individual.

To expatiate the above encoding approach, an example shown in Fig. 2 is given. There are three tasks, i.e., transmitting a packet from node 0 to AP, another from node 1 to AP and the third one from 4 to AP. According to their routes, each task contains 4 subtasks, i.e., (0,0)(0,1)(0,2)(0,3), (1,0)(1,1)(1,2)(1,3), and (4,0)(4,1)(4,2) respectively. There is a random combination of above subtasks, such as,

$$(0, 0) (1, 0) (0, 1) (4, 0) (0, 2) (4, 1) (4, 2) (0, 3) (1, 1) (1, 2) (1, 3)$$

then taking the TaskIDs out, the above sequence can be encoded as follow:
01040440111

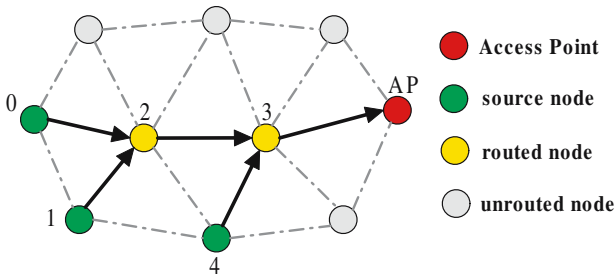


Fig. 2. Example of Network

To summarize the above process, the encoding rule is concluded as follows. There are N tasks and each task i includes M_i hops. An individual is a sequence composed of all the task ID numbers from 0 to $N - 1$, where each task number i appears M_i times. Obviously, the length of the individual is $\sum_{i=0}^{N-1} M_i$. According to this rule, an individual can be generated at random.

The decoding method of an individual is denoted as follows. First, an individual is transformed to a sequence of subtasks. Then, from the first subtask to the last one, we assign slots in sequence at the same time, trying to assign those subtasks to one slot without rousing collisions. Take an individual from the above example network, i.e., 00001111444, the corresponding sequence of subtasks is (0,0)(0,1)(0,2)(0,3)(1,0)(1,1)(1,2)(1,3)(4,0)(4,1)(4,2). Then the slot allocation scheme is shown in Fig.3.

From the decoding scheme, it can be seen that an individual can finally be decoded to a slot scheduling, which allows parallel operations. The mapping between an individual and a TDMA scheduling solution is now completely established.

Slot No.	1	2	3	4	5	6	7	8	9
Subtask Sequence	(0,0)	(0,1)	(0,2)	(0,3) (1,0)	(1,1)	(1,2)	(1,3) (4,0)	(4,1)	(4,2)
Execution Node	0	2	3	AP 1	2	3	AP 4	3	AP

Fig. 3. Slot allocation of the example

4 Simulation Results

In the simulation, to validate the viability of our proposed method, four networks with different numbers of nodes were deployed, randomly generating the network topology. The access point is located in the center of the area. The capacity of the channel is set to 500kbps and the packet lengths are 1kbits. Some parameters in energy aspect are as follows: similar to Ref. [2], the transition time between the sleep and active states is assumed to 470s. The power consumed in transmission and reception of a packet is set to 81 and 180mW, respectively. The power consumed in idle state is set as same as the reception state. In addition, we assume that the clock drift can be ignored by lengthening the slot time slightly.

To illustrate the effectiveness and performance of our PAPSO optimization algorithm, two other algorithms are used as comparisons: Max Degree First coloring algorithm (MDFCA, which is a common 2-distance coloring algorithm), and Node Based Scheduling Algorithm (NBSA) proposed by Ergen and Varaiya [6]. In the initialization of PAPSO, the swarm size M was set as 50 and the maximal generation was 200. And the task amount was that every sensor node generated a data packet and sent it to the access point. For the sake of fair comparison, all these algorithms run in energy-efficient mode, i.e. a node with no packet to send keeps its radio off during its allocated time slots.

Table 1 to 3 respectively illustrates the results of average time delay, average energy consumption and total time cost of three scheduling algorithms in the four networks. Among the three algorithms, MDFCA performs the worst on all the three indice. It is mainly because in MDFCA, many slots are allocated to those nodes which do not have the transmission task, thus wasting a lot of slots and decreasing the time performance of transmission. Moreover, in MDFCA, the node could only work on one slot in each coloring period, which makes the node have to switch its state in every single transmission. Consequently, graph coloring is not suitable to the data collecting sensor networks.

Although NBSA is still an algorithm based on coloring idea, its optimization performance for sensor networks is much better. This is because NBSA excludes such empty slots assigned to those nodes without packets, which can help the algorithm to save time and energy. However, both of the two algorithms lack the ability to adjust energy consumption of network.

The seven PAPSO solutions were continuous 7 members of the Pareto archive, which size is set as 7 to make data table succinct. According to Pareto optimality, the results of NBSA and PAPSO dominated the ones of MDFCA obviously

Table 1. Performance of average delay (slot)

No. of Nodes	25	49	121	169
MDFCA	16.67	44.06	136.13	198
NBSA	12.21	28.58	76.68	108.25
PAPSO(1)	9.08	24.38	70.32	100.75
PAPSO(2)	9.11	24.57	71.18	101.32
PAPSO(3)	9.53	24.77	71.92	101.92
PAPSO(4)	10.28	25.52	73.15	103.25
PAPSO(5)	11.82	26.65	74.82	105.38
PAPSO(6)	13.50	27.95	76.80	107.98
PAPSO(7)	14.58	29.35	78.82	110.91

Table 2. Performance of average energy consumption (mJ)

No. of Nodes	25	49	121	169
MDFCA	0.419	1.211	4.656	7.353
NBSA	0.378	1.057	4.186	6.624
PAPSO(1)	0.394	1.069	4.194	6.565
PAPSO(2)	0.387	1.049	4.138	6.520
PAPSO(3)	0.374	1.035	4.075	6.483
PAPSO(4)	0.365	1.023	4.030	6.450
PAPSO(5)	0.361	1.015	3.990	6.432
PAPSO(6)	0.357	1.009	3.982	6.426
PAPSO(7)	0.355	1.005	3.975	6.425

as table 1 and 2 showed. To illustrate the advantage of PAPSO over NBSA, we picked data in the 121-node network and compared the two algorithms as Fig. 4 showed. The solution of NBSA was dominated by solution (2)(3)(4)(5) of PAPSO. Furthermore, since during the maintenance of archive, we fixed two extreme cases as the border of archive and solutions are evenly distributed, so we can easily find a tradeoff by picking the middle one, i.e. PAPSO(4). , the performance of total time cost was not sacrificed. From table 3, we can see that the performance of total time cost of PAPSO was comparative to NBSA. Even when a tradeoff is met, the performance of total time cost was not sacrificed.

In a whole, the optimization effect of PAPSO outperforms the other two algorithms. It is mainly because: 1. Effective idea of solving the scheduling problem. As to the scheduling of TDMA time slots, we assigned time slots to subtasks sequentially, which avoids generating idle slots and thus ensures the good time performance. And time slots of subtasks can be flexibly adjusted, which leads to good working continuity of sensor nodes and decreases the energy consumption. 2. Good search algorithm. As an efficient evolutionary algorithm, PSO not only owns good search ability in the solution space but also is good at solving multi-objective optimization problem. Combined with Pareto optimality, PSO can search the solution space more thoroughly and find out a better tradeoff between energy consumption and time delay.

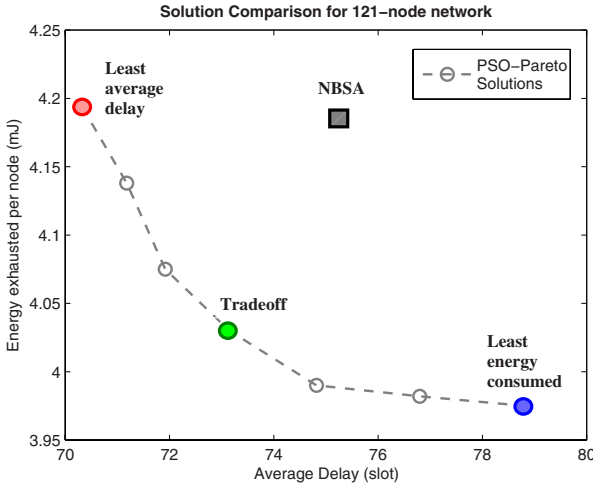


Fig. 4. Comparison of results of NBSA and PAPS0 in 121-node network

Table 3. Performance of total time cost(slot)

No. of Nodes	25	49	121	169
MDFCA	46	123	397	602
NBSA	30	62	154	212
PAPS0(1)	28	59	149	205
PAPS0(2)	31	61	155	207
PAPS0(3)	29	59	151	210
PAPS0(4)	31	62	153	209
PAPS0(5)	31	61	153	213
PAPS0(6)	30	60	155	209
PAPS0(7)	32	61	156	215

5 Conclusions

In this paper, according to the characteristics of many-to-one data transmission in wireless sensor networks, we combined Pareto optimality with PSO and proposed an effective PAPS0 optimization to solve the TDMA scheduling problem. And the optimization has the following advantages: 1. Under the framework of evolutionary search algorithm, it is easy to deal with multi-objective optimization problem and set up the model of multi-objective optimization; 2. We can make the best of the problem-solving ability of evolutionary search algorithm over NP problem; 3. The introduction of Pareto optimality makes the evaluation system of multi-objective PSO solutions more reasonable and effective, thus offering more choices of network performance to decision-makers.

In wireless sensor networks, multi-objective optimization problem widely exists and there is usually confliction among such objectives. Consequently, our

proposed multi-objective optimization algorithm can serve as a good idea to such kind of problem.

References

1. Jolly, G., Younis, M.: An Energy-Efficient, Scalable and Collision-Free MAC layer Protocol for Wireless Sensor Networks. *Wireless Communications and Mobile Computing* (2005)
2. Ergen, S.C., Varaiya, P.: TDMA: Scheduling Algorithms for Sensor Networks, Technical Report, Department of Electrical Engineering and Computer Sciences U.C. Berkeley (July 2005)
3. Cui, S., et al.: Energy-Delay Tradeoffs for Data Collection in TDMA-based Sensor Networks. In: The 40th annual IEEE International Conference on Communications. Seoul, Korea (May 16-20, 2005)
4. Sridharan, A., Krishnamachari, B.: Max-Min Fair Collision-free Scheduling for Wireless Sensor Networks. In: Workshop on Multihop Wireless Networks (MWN 2004) (April 2004)
5. Gandham, S., Dawande, M., Prakash, R.: Link scheduling in sensor networks: Distributed edge coloring revisited. In: INFOCOM, pp. 2492–2501 (2005)
6. Ergen, S.C., Varaiya, P.: TDMA: scheduling algorithms for sensor networks, Technical Report, Department of Electrical Engineering and Computer Sciences University of California, Berkeley (July 2005)
7. Karp, B., Kung, H.T.: GPSR: Greedy perimeter stateless routing for wireless networks. In: Proc. ACM/IEEE MobiCom (August 2000)
8. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, pp. 39–43 (1995)
9. Deb, K.: Evolutionary Algorithms for Multi-Criterion Optimization in Engineering Design [A]. In: Proceedings of Evolutionary Algorithms in Engineering and Computer Science (EUROGEN-99), pp. 135–161 (1999)
10. Coello Coello, C.A., Salazar Lechuga, M.: MOPSO: A Proposal for Multi Objective Particle Swarm Optimization. In: Congr. on Evolutionary Computation, Piscataway, New Jersey. vol. 2, pp. 1051–1056 (2002)