A HYBRID GENETIC ALGORITHM FOR TASK SCHEDULING IN INTERNOT OF THINGS

Junqing Li, Yuting Wang, Tao Sun

School of Computer Science, Liaocheng University
Liaocheng, Shandong, China
lijunqing.cn@gmail.com

Abstract

This paper addresses the task scheduling problem in Internet of Things (IoT) by using a hybrid genetic algorithm. The task scheduling considered the communication cost in task transportation. The scheduling sequence relation among tasks is also considered. The proposed algorithm combined genetic algorithm (GA) with several local search approaches. Experimental results show the efficiency and effectiveness of the proposed algorithm.

Keywords - Internet of Things; genetic algorithm; task scheduling; local search

1 INTRODUCTION

The Internet of Things (IoT) refers to uniquely identifiable objects (things) and their virtual representations in an Internet-like structure. The term IoT was first used by Kevin Ashton in 1999 [1]. Radio-frequency identification (RFID) is often seen as a prerequisite for the Internet of Things. If all objects of daily life were equipped with radio tags, they could be identified and inventoried by computers [2, 3]. In IoT, there are lots of computers, physical devices, and mobile devices. To complete a complex task, the task may be decomposed into a lot of simple tasks, and be assigned to lots of devices in IoT, which can improve the whole performance of the system. However, by this way, the communication devices in the IoT should make a balance each other, thus to increase the system performance while balance all the resources. Many researchers have verified that the task scheduling problem is NP-hard [4, 5]. In this paper, we propose a hybrid genetic algorithm (GA) for solving the task scheduling in IoT system. Experimental results show the efficiency of the proposed algorithm.

2 PROBLEM DEFINITION

In this study, we consider the task scheduling problem with n tasks to be processed on m devices. In the task scheduling problem, there are sequence constraints in many tasks. That is, some tasks must be processed after their predecessor tasks. Let t_i denotes the i^{th} task, $G(t_i)$ denotes the group of task which have the sequence relation with the task t_i. In order to schedule tasks on each device, we must consider the sequence relation constraints in each task group. Meanwhile, communication costs will occur if some tasks select different RFID devices with their pre-processed tasks. Therefore, in the scheduling problem, we should balance the total communication cost and the maximum completion time. In this study, the main task of the problem is to assign each task on each RFID device to minimize the completion time of the last task.

Suppose eleven tasks are to be processed on eleven RFID devices in a given operating sequence. The processing time on each device is deterministic and given before scheduling. For example, in Fig. 1, the string "T1-10" below device 1 means that the task T1 is assigned to be processed on device D1, which consumes 10 seconds to complete the task. The following task T2 is allocated on device D2 with 3 seconds. The two points named "T0" and "T*" are the two dummy points, which consumes zero processing times, and is the start and end points of the system. Fig. 1 also tells the scheduling sequence constraints of the given problem. For example, in the figure, the task T1 should be processed before the starting time of the task T2, that is, the task T2 cannot be started before the completion of T1. The number up the line between T1 and T2 denotes the communication cost of transporting task T1 and T2 from device D1 to D2. Therefore, in this example problem, the starting time of T2 should have 2 seconds late than the completion time of T1. The late time between each task on different device is the communication cost and should be considered in the proposed algorithm.

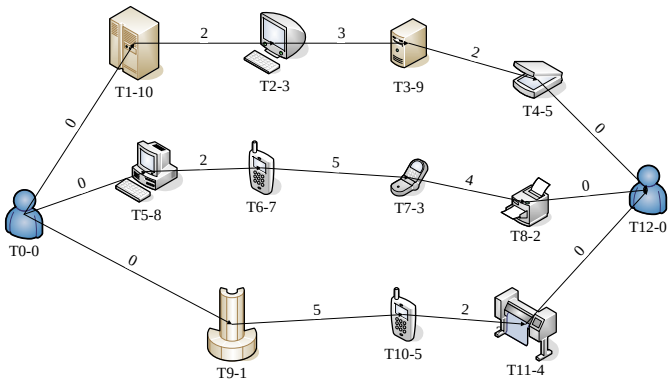

Fig. 1. Example of a task scheduling problem with communication cost

Next, in Fig. 2, we give a realistic task scheduling example with parallel RFID devices for each task. In Fig. 2, we can see there are four tasks and three RFID devices. In scheduling each task, the three RFID devices can be selected. Each device, with different processing time, can be selected for operating the given task, thus generate processing cost. The successor task, if a task has, will repeat the above process to select a suitable device. However, if the successor task chooses a different RFID device with its predecessor, then a given number of communication cost will occur. For example, in Fig. 2, the proposed algorithm selects D2 for processing task T1, while selects D1 for task T2. Then 4 seconds transportation cost will occurs between the completion time of T1 and the starting time of T2. Therefore, we should add the transportation cost to the processing time of T2 on D1.

In addition, when assign two tasks on one RFID device, for example, assign T1 and T3 on device D1, then T1 and T3 should be scheduled on D1. That is, we should determine the starting time of T1 and T2 on D1 and guarantee the starting time of the two tasks should not overlap with each other. In this example, the algorithm should consider two issues: (1) to select suitable RFID devices for each task; (2) to schedule each task on every RFID device. The processing time table, communication costs and processing sequence are given in Table I and Table II, respectively.

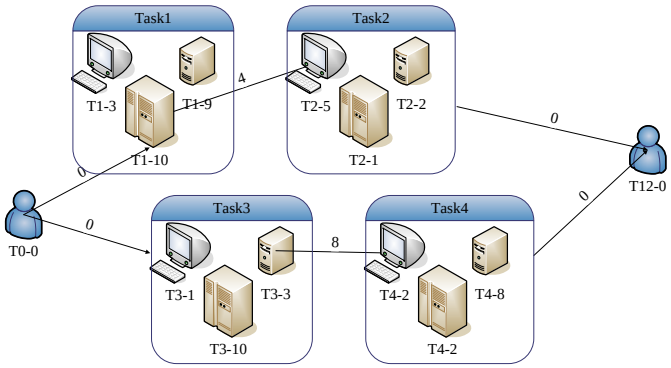

Fig. 2. Example of a task scheduling problem with parallel devices

The processing time table, communication costs and processing sequence are given in Table I and Table II, respectively. Table I shows the processing time for each task on each RFID device. For example, the task T1 consumes 3 seconds on D1, 10 seconds on D2, and 9 seconds on D3. Table II illustrates the processing sequence and communication costs. For example, tasks T1 and T2 are a group of tasks which have certain processing sequence. That is, the task T1 must be first processed, then the task T2. If T2 has been processed on a RFID device different with the device which has processed T1, then 4 seconds will consumed for the communication cost. In Table II, {T1, T2}, {T3, T4} are two groups. In each group, tasks have given processing sequence, while in different group, tasks have none sequence constraints.

Table 1. Processing time table

	D1	D2	D3
T1	3	10	9
T2	5	1	2
T3	1	10	3
T4	2	2	8

Table 2. Communication Cost and processing sequence

T1	T2	T3	T4
0	4	0	8

3 THE PROPOSED ALGORITHM

In this study, we propose a hybrid algorithm which combines GA and several local search approaches for solving the task scheduling problem in IoT system.

3.1 The classical GA

The genetic algorithm (GA) is proposed by Holland in 1975, which is a search heuristic that mimics the process of natural evolution [6-9]. This heuristic is routinely used to generate useful solutions to optimization and search problems. The main features of GA are its operators by mimicking the nature evolution, such as inheritance, mutation, selection, and crossover.

In the classical, each solution is represented by a population of strings (called chromosomes). After generating a random initial population, the solutions in the initial population will perform an evolution progress to converge to an optimal space. In each generation, each solution in the current population is evaluated by a certain fitness function. Each solution will be stochastically selected from the current population to form a new population with the same size. The above steps repeated until the stop conditions satisfy.

3.2 Solution Representation

In the proposed algorithm, each solution is represented by two vectors, i.e, the scheduling vector and the RFID device assignment vector. The first vector tells the task scheduling sequence while the second vector displays the RFID device for processing the task. For the problem given in section 2, a solution may be represented with {T1,T2,T3,T4}{1,2,3,2}. This solution tells the following information: (1) the sequencing order is <T1,T2,T3,T4>; (2) the assigned RFID device for processing each task is also given. That is, <T1, D1>, <T2, D2>, <T3, D3>, <T4, D2>.

3.3 Initial population

To produce the initial population with high level of quality, several initial approaches are used for the scheduling vector in the hybrid algorithm [10,11]:

- Random rule

In random rule, first, generate $n \times m$ integer value one by one; then, shuffle all integer values in a random order.

- High Workload First rule (HWF).

In HWF, the first task to be scheduled is the task with the highest workload. That is, the task with the most remained workload will be selected first. If there are more than one task with the most workload, randomly select one as the next task to be scheduled.

- Least Workload First rule (LWF)

In LWF, the first task to be scheduled is the task with the least workload. That is, the task with the least remained workload will be selected first. If there is more than one task with the least workload, randomly select one as the next task to be scheduled.

The initial population is constructed as follows. First, generate one solution by using HWF rule, one solution by LWF rule. Then, generate P_{size}-2 solutions by using the random rule.

To balance the resource workload in the initial population, several initial approaches are used for the RFID device vector in the hybrid algorithm.

- Random rule

In random rule, randomly select a RFID device for each task from the candidate device set.

- Lowest Processing First (LPF).

In LPF, the device with the minimum processing time will be selected for processing the corresponding task.

3.4 Crossover operator

The crossover operator in the basic GA is to learn information from the other chromosomes, thus to improve the quality of the whole population. In this study, we chose one-point crossover operator as the process to learn information in scheduling vector among solutions.

Given two parent chromosomes p_1, p_2, the two new-generated chromosomes are denoted as c_1 and c_2, respectively. The detailed steps of the one-point crossover operator are given as follows.

*Step*1. Randomly generate a number r ranges in [1, $n \times m$].
*Step*2. Copy the genes between [1, r] from p_1 to the corresponding locations at c_1.
*Step*3. Copy the genes between [1, r] from p_2 to the corresponding locations at c_2.
*Step*4. Copy the genes between [r, $n \times m$] from p_1 to the corresponding locations at c_2.
*Step*5. Copy the genes between [r, $n \times m$] from p_2 to the corresponding locations at c_1.

3.5 Mutation operator

The mutation operator in the basic GA is to increase the population diversity. In this study, we include several mutation operators for scheduling vector as follows.

(1) Insert mutation operator

First, generate two random integer values, r_1 and r_2, which are both range in [1, $n \times m$]. Suppose that r_1 is less than r_2. Then, insert the genes at location r_2 before r_1.

(2) Swap mutation operator

First, generate two random integer values, r_1 and r_2, which are both range in [1, $n \times$]. Suppose that r_1 is less than r_2. Then, swap the two genes at location r_2 and r_1, respectively.

In the proposed algorithm, the mutation operator for RFID device assignment vector is using the random approach. That is, for a randomly select task, randomly select a different RFID device with the current operating device.

3.6 Framework

The whole framework of the proposed algorithm is given as follows.

*Step*1. Generate P_{size} solutions as the initial population.

*Step*2. Evaluate each solution in the population. If the stop criterion satisfies, then stop the algorithm; otherwise perform steps 3 to 5 $n \times m$ times.

*Step*3. Randomly select two solutions in the current population. Then, perform crossover operator on the two solutions with probability p_c. Replace the two solutions with the two new-generated solutions.

*Step*4. Perform mutation operator on the two solutions with probability p_m.

*Step*5.Evaluate the new generated solutions and record the best solution found so far.

4 EXPERIMENTAL RESULTS

This section describes the computational experiments to evaluate the performance of the proposed algorithm. The current instantiation was implemented in C++ on a Pentium IV 1.8GHz with 512M memory.

4.1 Experimental parameters

The experimental parameters for the proposed algorithms are given as follows.

- P_{size}. The initial population size P_{size} is set 10;
- h. The neighboring solution number for each local search is set $n \times m$.
- P_c. The probability of crossover is set to 0.3;
- P_m. The probability of mutation is set to 0.7.

4.2 Experimental results

We select a 29-tasks-7-devices instance to test the performance of the proposed algorithm. Table III tells the processing time for the randomly generated benchmark. Fig. 3 gives the best solution by the proposed algorithm. In the Gantt chart, the rectangle filled with green color represents the communication cost for the task transportation. The other rectangles illustrate the scheduling task on each RFID device. From the experimental results and the Gantt charts, we can see that the proposed algorithm is efficient for solving the task scheduling in IoT systems. The experimental results verify the efficiency and effectiveness of the proposed algorithm.

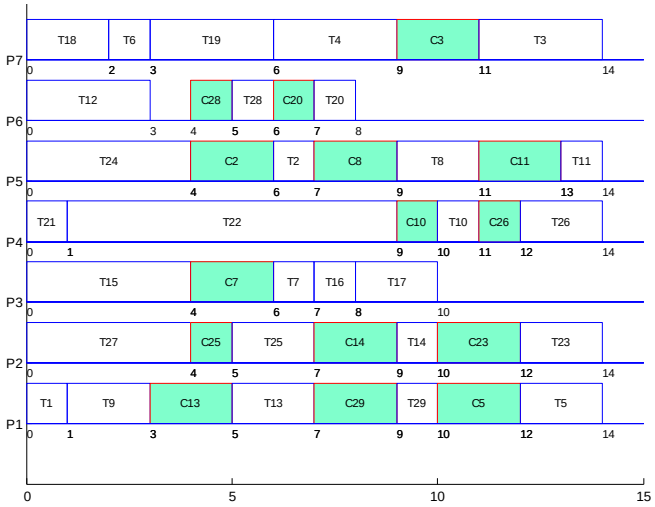

Fig. 3. Gantt chart for the best solution

Table 3 Processing time table

	P1	P2	P3	P4	P5	P6	P7
T1	1	4	6	9	3	5	2
T2	8	9	5	4	1	1	3
T3	4	8	10	4	11	4	3
T4	6	9	8	6	5	10	3
T5	2	10	4	5	9	8	4
T6	15	4	8	4	8	7	1
T7	9	6	1	10	7	1	6
T8	11	2	7	5	2	3	14
T9	2	8	5	8	9	4	3
T10	5	3	8	1	9	3	6
T11	1	2	6	4	1	7	2
T12	7	1	8	5	4	3	9
T13	2	4	5	10	6	4	9
T14	5	1	7	1	6	6	2
T15	8	7	4	56	9	8	4
T16	5	14	1	9	6	5	8
T17	3	5	2	5	4	5	7
T18	5	6	3	6	5	15	2
T19	6	5	4	9	5	4	3
T20	9	8	2	8	6	1	7
T21	6	1	4	1	10	4	3
T22	11	13	9	8	9	10	8
T23	4	2	7	8	3	10	7
T24	12	5	4	5	4	5	5
T25	4	2	15	99	4	7	3
T26	9	5	11	2	5	4	2
T27	9	4	13	10	7	6	8
T28	4	3	25	3	8	1	2
T29	1	2	6	11	13	3	5

5 CONCLUSIONS

In this study, we proposed a hybrid algorithm which combines GA and other local search approaches for solving the task scheduling problem in IoT systems. The communication cost and the task sequence constraints are considered synchronized in the proposed algorithm. Experimental results and comparison with well-known algorithms show the efficiency of the proposed algorithm. The future work is to introduce other meta-heuristic algorithms for solving the task scheduling problems in the IoT system.

6 ACKNOWLEDGMENT

This research is partially supported by National Science Foundation of China under Grant 61104179, and Science Research and Development of Provincial Department of Public Education of Shandong under Grant (J09LG29, J11LG02 and J10LG25).

7 REFERENCES

[1] Kevin Ashton: That 'Internet of Things' Thing. In: RFID Journal, 22 July 2009.

[2] P. Magrassi, A. Panarella, N. Deighton, G. Johnson, Computers to Acquire Control of the Physical World, Gartner research report T-14-0301, 28 September 2001.

[3] Commission of the European Communities (2009-06-18). "Internet of Things — An action plan for Europe" (PDF). COM(2009) 278 final.

[4] Jung YH., Kim HS., Choe Y. (2009) Ant colony optimization based packet scheduler for peer-to-peer video streaming. IEEE Commun Lett 13(6):441-443.

[5] Seriven I., Lewis A., Mostaghim S. (2009) Dynamic search initialization strategies for multi-objective optimization in peer-to-peer networks. Proc. IEEE Congress on Evol, Comput. (CEC), Trondheim, Norway, May 2009, pp. 1515-1522.

[6] Wang L. Shop Scheduling with Genetic Algorithms. TsingHua Press, 2003.5.

[7] Eiben, A. E. et al (1994). "Genetic algorithms with multi-parent recombination". PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: 78–87.

[8] Sakawa M, Mori T (1999) An efficient genetic algorithm for job shop scheduling problems with fuzzy processing time and fuzzy due date. Comput Ind Eng 36(2):325-341.

[9] Song XY, Zhu YL, Yin CW, Li FM (2006) Study on the combination of genetic algorithms and ant colony algorithms for solving fuzzy job shop scheduling. In Proceedings of the IMACS multi-conferences on computational Engineering in Systems Application. Beijing, pp 1904-1909.

[10] Li JQ, Pan QK, Liang YC (2010) An effective hybrid tabu search algorithm for multi-objective flexible job shop scheduling problems. Computers and Industrial Engineering, 59(4):647-662.

[11] Li JQ, Pan QK, Suganthan, PN, Chua TJ (2011) A hybrid tabu search algorithm with an efficient neighborhood structure for the flexible job shop scheduling problem. Int J Adv Manuf Technol 52(5-8):683-697.