International Journal of Applied Optimization Studies
Year 2018 Volume 1 Issue 1 pp. 11-24
www.ijaos.com

A Gravitational Emulation Local Search Algorithm for Task Scheduling in Multi-Agent System

Ali Asghar Rahmani Hosseinabadi[1,*] , Erfan Babaee Tirkolaee [2]

[1]*1Young Researchers and Elite Club, Ayatollah Amoli Branch, Islamic Azad University, Amol, Iran*
[2]*Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Itan*

Abstract

One of the challenges in designing Multi-Agent Systems (MAS) on agents is breaking a job into several tasks and scheduling them among agents so that execution time is reduced and energy consumption become optimized as well load balancing is considered as a major factor on performance. On the other hands, one of the decisive factors in task scheduling and load balancing among agents is how to deploy tasks on agents. In this paper, a novel method is proposed based on a Gravitational Emulation Local search (GELS) algorithm for task scheduling among agents and load balancing. The performance of the proposed algorithm is evaluated in comparison with different sized test problems. Finally, simulation results show that proposed algorithm can solve the problem perfectly.

Original Article:

Received 2018-06-03
Revised 2018-07-20
Accept 2018-07-30

Keywords:
Gravitational Emulation Local search;
Multi-agent Systems;
Task Scheduling;
Prioritization.

1. Introduction

The increased complexity of distributed systems has led to the entry of new challenges to the design of interactions and management of decentralized components in industrial and research field. One way to overcome these new challenges is to provide more dynamic, self- aware, consistent and optimal systems, like Multi-Agent Systems (MAS) [1]. A MAS consists of several agents which are defined as software (hardware) entities deployed in a specific environment and can react automatically to environmental changes [2]. An agent has three basic characteristics: Reactivity, pro-activeness and Social ability. Reactivity refers to the ability of an agent to react to changes in its external environment. Pro-activeness indicates the objective behavior of the agent and social ability means that based on Agent Communication language, agents can negotiate instead of sending data and then interact with another agent [2]. Therefore agents are independent and autonomous entities which can gain information from the environment and collaborate on solving complex problems. Agents are used in many applications, such as real-time requirements including communication systems, database search and retrieval, control systems, and sensor fusion

*Corresponding author email address: A.R.Hosseinabadi@iaubeh.ac.ir

[2]. A MAS is a collection of entities that interact with each other to solve a problem or to reach a goal beyond the ability of one agent [3].

Figure 1 shows an outline of a MAS and interactions between agents. As we know, in real-world there are complex problems that only one agent cannot solve it, so in such a situation having a team of agents with the ability to communicate and sharing information can be useful. It means that, instead of having a centralized approach in each agent, solution goals and tasks must be distributed among the team. Such solutions can better control the system and can be dynamically controlled in different situations. Each agent is able to perform a unique task and delegates the rest tasks to other agents. This way, the complexity level of the entire system is reduced to a controllable level [1].

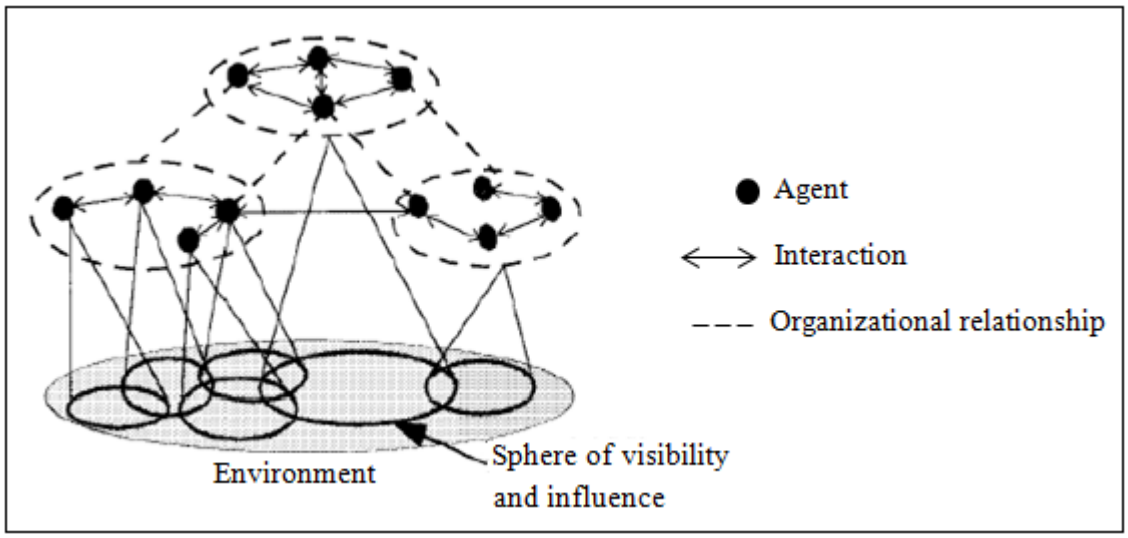

Fig. 1. An overview of multi-agent systems [1].

MAS have some advantages over other traditional and central systems. Firstly, they are suitable for large and dynamic distributed systems, and also have good flexibility and scalability, because by increasing the size of the problem, it is possible to add more agents to the system [3]. A MAS also works in a non-concurrent and parallel manner and is faster and more efficient than centralized systems [3]. In addition, MAS is an open system, and new tasks are created based on changes in the external environment and agents are used to solve them [4]. According to the unique characteristics of the agents (mobility, autonomy and load imbalance, etc.), if tasks are assigned without coordination, some agents become overloaded very quickly, and this situation leads to a bottleneck situation for the system [5]. Task scheduling problem is one of the main problems in MAS and suitable scheduling techniques are needed to perform tasks with high efficiency. The aim of scheduling of these systems is to reduce execution time and also the overhead of agents [4, 5]. In recent years, different scheduling methods have been proposed for MAS that can be divided into the following categories: task scheduling based on auction protocol [6, 7]; (2) forming coalition methods [8, 9]; (3) social network based methods [10]; (4) decision theory based methods [11]; (5) cooperative and non-cooperative participation based methods [12].

Optimization algorithms are categorized into two groups of exact algorithms and approximate algorithms [13]. Optimization algorithms are a kind of intelligent algorithms used to find optimal solutions. These algorithms have strategies for tackling from local optimality and are applicable to a wide range of problems [13].

In recent years, many optimization algorithms are developed for solving various problems. Some of them are as follows: Genetic Algorithm (GA) [14], Ant Colony Optimization (ACO) [15], Bee Colony Optimization (BCO) [16] Particle Swarm Optimization (PSO) [17], Tabu Search (TS) [18] Simulated Annealing (SA) [19] and so on.

Recently, a new algorithm named Gravitational Emulation Local Search (GELS) Algorithm is proposed by Barry Webster in 2004 which is inspired by Newton's law for optimization [20]. This algorithm is based on randomization concepts along with two parameters of velocity and force which uses random numbers of existing local search algorithms in order to avoid local optimum. The idea of this algorithm is based on the principle of gravitational force in nature that causes objects to be attracted to each other so that the heavier object has a greater gravitational force and applies it to other objects and attracts objects with less weight [20].

So far we have noticed no report on using GELS algorithm in MAS. In this paper, a new optimization algorithm named GELS is proposed to solve task scheduling in MAS. The purpose of this algorithm is load balancing in the system and having the maximum number of scheduled tasks.

The rest of the paper is organized as follow: Section 2 review some related works. Section 3 explains GELS algorithm. The proposed algorithm is introduced in Section 4. Simulation results and conclusion are presented in Sections 5 and 6 respectively.

2. Literature Review

By increasing global connectivity, In order to process the information in different parts of the world, various distributed systems, including multi-agent systems, have been implemented. In such environments, due to the difference in the volumes of processing data in different nodes, the processing of some agents may be completed faster than others, so the speed of the system depends on the longest processing time. So we need scheduling mechanisms to speed up this process [5]. In recent years many methods have been proposed, some of them are explained below.

Given the processing status of each agent, CPU power and the amount of calculation of each user, and with the aim of reducing the system return time, Ranatunga et al. [21] provided a fair and effective algorithm for scheduling in which an agent with the smallest computational request has higher priority.

Adhau et al. [22] considered the time of sending resources and cost of execution and control to propose a new algorithm named DMAS/RIS to allocate and send shared resourcesto large and complex multi-purpose systems.

Hsie and Lin [23] proposed an algorithm for workflow scheduling for multi-agent systems by combining multi-agent system, contract net protocol and workflow models specified by Petri nets, in which original workflow scheduling problem is divided into several sub-problems and are solved by agents.

Zhenqiang et al. [24] solved task scheduling problem with bottleneck resources by considering imbalance of resource capacity in Production Scheduling System and combining dynamic and autonomous agents. In this method, Bottleneck Resource Agent (BRA) is used for finding bottleneck resources and Non-Bottleneck Resources Agent (NBRA) is used for finding non-bottleneck resources and finally BAD is used to

coordinate the relationship between the arrival time and delivery time of tasks in BRA and NBRA.

Multi-agent scheduling model [25] is provided with the aim of improving the system flexibility and reducing the return time of dynamic job shop scheduling. But the combination of this method with optimization algorithms can lead to better scheduling. Yumin and Shufen [26] proposed a new efficient and hybrid algorithm for Job-Shop Scheduling using GA and a MAS with the aim of increasing performance and efficiency and reducing the time of reaching global optimum.

The aim of the agile multi-agent scheduling system [27] is to increase fault tolerance in uncertainties and increase system efficiency and reduce the makespan. This method uses negotiation strategies to solve the problem.

LI and DU [28] combined MAS with GA and proposed a new method for Job Shop Scheduling Problem (JSCP) which allows agents to choose optimum jobs dynamically. This method utilizes a hybrid genetic algorithm to re-schedule jobs in order to reach global optimum.

Nie et al. [29] proposed an agent-based dynamic scheduling model for FMS which performs scheduling and control of the system through agent collaboration and without interrupting the operation of the system and without user intervention. The proposed method is capable of responding to dynamic changes including machine failure or out-of-date tasks.

Shah et al. [30] proposed agent-based scheduling model for scheduling in grid computing with the aim of increasing robustness, efficiency, performance, scalability, and heterogeneity. APDRR scheduling algorithm proposed by Shah et al. [31] is a combination of round robin and priority scheduling algorithms to simultaneously meet user goals and increase system efficiency. Chronos MAS [32] is designed to set up user sessions; this system assigns a smart regulator agent to each user. Users can schedule time, location, the day of negotiation, based on user interests and preferences. In order to provide better performance and better allocation, agents can learn from the user as well as interact with other agents. For further studies on the application of MAS in solving various problems, refer to [33-37].

3. The GELS algorithm

Many researchers have been working on the optimization problems and the applications of the computers in order to optimize them [44-48]. In 1997, GLS algorithm was proposed by Voudouris and Tesang [38] to search and solve complex problems for the first time. In 2004, Webster [20] called this method GELS and was used as a powerful local search algorithm for solving complex problems. The main idea of GELS is based on gravitational force, which causes to attract objects with each other such a way that heavy object has the higher gravitational force and attract low weigh objects. The attraction force between two objects depends on the distance between them [39-44].

In GELS algorithm, possible solutions in search space are divided into several categories according to their fitness's. Each of these categories named a dimension of the solution and there is initial velocity for them. Eq. (1) computes the gravitational force between Current Solution (CU) and Candidate Solution (CA). This force (F) is added to velocity vector in the path of current motion. If velocity exceeds the maximum value (threshold), maximum

velocity becomes the current velocity, and if the velocity becomes negative due to this force, the velocity is considered zero [43, 44].

$$F = \frac{G(Cu - CA)}{R^2} \tag{1}$$

4. Proposed algorithm

In the proposed method, GELS Algorithm is used as a strategy for solving task scheduling problem in MAS. The purpose of this algorithm is load balancing in the system and having the maximum number of scheduled tasks. In the following, GELS_MAS is presented to get a suitable solution for the problem in a reasonable time.

To provide an appropriate scheduling to perform *N* tasks by *M* agents, it is needed to define some models for determining the status of agents. Therefore in this section, we define the models of the proposed algorithm.

4.1. Agent load model

For optimal task scheduling, we must first determine agent's load according to agent's distance from the service center and determine the cost of establishing this connection. So agent's load is calculated in Eq. (2).

$$W_i = D_i \times C_i \qquad \forall i \in I \tag{2}$$

Where W_i is the load of agent *i*, *i*={1,2,…,*n*}, D_i indicates the distance of agent *i* from the service center and C_i is the cost of establishing this connection.

4.2. Agent credit model

In order to load balancing and assigning appropriate tasks to each agent in the process of task scheduling in MAS, a credit model is needed. The system assigns a credit value to each agent which is based on agent's load and decides which input task should be assigned to which agent according to this credit value. An agent with higher credit value (an agent with lower load) will have a greater chance of accepting input task as it is shown in Eq. (3).

$$Cr_i = Max(W_i) - W_i \tag{3}$$

Where Cr_i is the credit of agent *i*, Max(W_i) is the maximum load that can be assigned to the agent and W_i is the existing load on the agent. In this equation, agent's credit decreases by increasing agent's load and vice versa.

4.3. Gravitational mass of agent

The gravitational mass of agents has a direct relationship with agent's capacity and the distance between them. Agents' capacities and their distances to the center are used to determine the gravitational mass between agents and service center and the average capacities of two agents and the distance between them are used to determine the gravitational mass between two agents (Eq. (4)).

$$M_{A,B} = \frac{D_{A,B} \times C_{A,B}}{V_{A,B}} \quad (4)$$

Where, $M_{A,B}$ is the gravitational mass between two agents, $D_{A,B}$ is the distance between two agents, $C_{A,B}$ is agents' loads, and $V_{A,B} = 10$ is the velocity between agents.

The initial velocity of each dimension of the solution is selected randomly in the interval [1, 100] and in the proposed algorithm initial velocity is set to 10.

4.4. Defining solution dimensions

In the proposed method, each solution dimension can be considered equal to one task; in fact, the number of solution dimensions is equal to the number of input tasks of the problem. The neighbor of the CU is the task with the least distance from the center, the least execution time and the highest velocity for the task of the CU of each agent and that task has not been done so far.

4.5. Neighborhood definition

In the GELS algorithm, unlike other algorithms, neighbors are not searched randomly, instead, each CU has different neighbors, each of which is based on a particular change which is named the direction for moving towards the neighboring solution and all the neighbors obtained this way are only based on this neighbor.

In the proposed method, the neighbor solution is obtained this way, the task with the least distance and time and more velocity than CU is selected as a neighbor and CA for each agent.

4.6. Fitness function

The purpose of the proposed algorithm is load balancing in the system and having the maximum number of scheduled tasks. So a better solution is the one that holds the system in a balanced state and also schedules more tasks. Eq. (5) shows the fitness function.

$$Fitness = \sum_{i=1}^{M} T_i \quad (5)$$

Where M is the number of agents and T_i indicates input task.

4.7. Gravitational force calculation

After choosing the best candidate solution, the gravitational force between CU and candidate solution is calculated and added to the velocity dimension of the corresponding solution. This force is the difference between the fitness function of CA and the fitness function of the current solution. Therefore the gravitational force between two solutions is calculated in Eq. (6).

$$F = G \times \frac{Fitness(candidate\_ch_i) - Fitness(current\_ch_i)}{R^2} \quad (6)$$

4.8. Solution Method

In the mentioned problem, several tasks enter to the system. These tasks must be scheduled, so that maximum numbers of tasks are performing and load on agents is balanced. In this system, each task can be done by only one agent, but agents may perform more than one tasks or nothing. Using this method we can have an appropriate allocation of tasks in the system. In this method GELS algorithm is used. Initial or CU is obtained by sequentially assigning tasks to the agents. Figure 2 shows a sample CU in which the length of the CU is equal the number of agents and numbers inside cells indicates the number of input tasks. For example task 1 is on agent 1, task 2 is on agent 2 … and task 6 is on agent 6.

CU:	T_1	T_2	T_3	T_4	T_5	T_6

Fig.2. A sample initial solution

Each generated solution is called a solution dimension which is obtained in next rounds by changing CU based on gravitational force. The initial velocity of each solution dimension is chosen randomly from [1, 100], which is set to 10 in the proposed algorithm. The gravitational force between any generated solution having better fitness function and its parent solution is calculated and added to the velocity dimension of that solution.

In order to generate next solution that is CA act based on the gravitational mass of agents, so that the task given to an agent may be greater than the maximum capacity of that agent, so according to the gravitational mass matrix, we must assign this task to an agent with less gravitational mass. If better solution is obtained is considered as CA and gravitational force between as CA and as CU is calculated and added to the velocity dimension. Therefore, gravitational mass between agents is updated. This trend repeats until the desired solution with the best fitness is obtained.

5. Simulation results

In this section, the simulation results obtained by the proposed algorithm and its comparison with GA (according to [45]) are presented. The algorithm is coded in C# programming language. It has been run on the system with the configuration of CPU 2.2 GHz and RAM 4 GB. The length of tasks scheduling on agents is one of the comparing parameters for solving the Task Scheduling Problem in Multi-Agent System in line with the evaluation of the proposed algorithm. The proposed algorithm is run for different sized (small, medium and large) problems including multiple agents and multiple tasks (tasks are assigned to the agents randomly). The input information is shown in Table 1.

Table 1. Input information for test problems.

Problem	#Agent	#Task
Small sized	2	40
	3	40
	4	30
	5	40
Medium sized	20	50
	20	100
	20	150
	20	200
	20	250
Large sized	10	100
	10	150
	10	200
	10	250
	10	500

Table 2 presents the comparison results of the proposed algorithm with GA. As it is clear, our proposed algorithm could generate better solutions rather than GA and it has a significant preference. The length of tasks scheduling is equal to the reduction of the required time for performing tasks.

Table 2. The obtained results of the proposed algorithm in comparison with GA**.**

Problem	#Agent	#Task	GA [45]	GELS_MAS
			Best solution (makespan)	**Best solution (makespan)**
Small sized	2	40	2.67	1.35
	3	40	8.74	5.17
	4	30	6.89	3.54
	5	40	7.23	4.07
Medium sized	20	50	3.45	1.20
	20	100	5.71	2.27
	20	150	7.11	3.50
	20	200	8.95	4.36
	20	250	10.69	6.40
Large sized	10	100	5.24	2.00
	10	150	9.36	5.11
	10	200	11.83	6.75
	10	250	15.31	8.83
	10	500	3.53	1.19

Figures 1-3 depicts the output results of the proposed algorithms for three problems. As it is shown in figures, the algorithm could generate acceptable results.

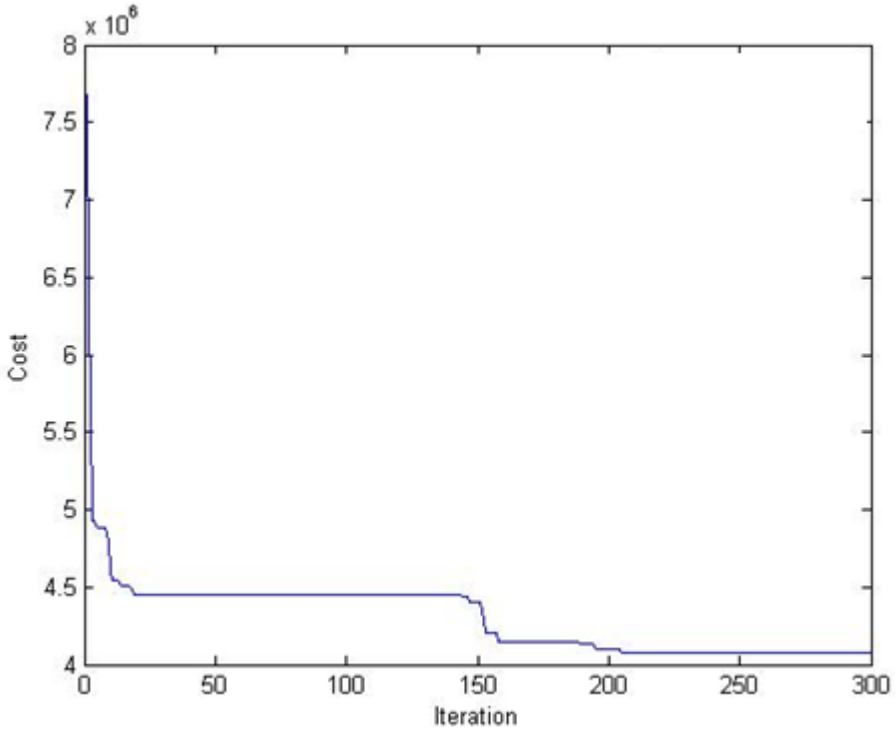

Fig. 1. The obtained results for 5 agents and 40 tasks.

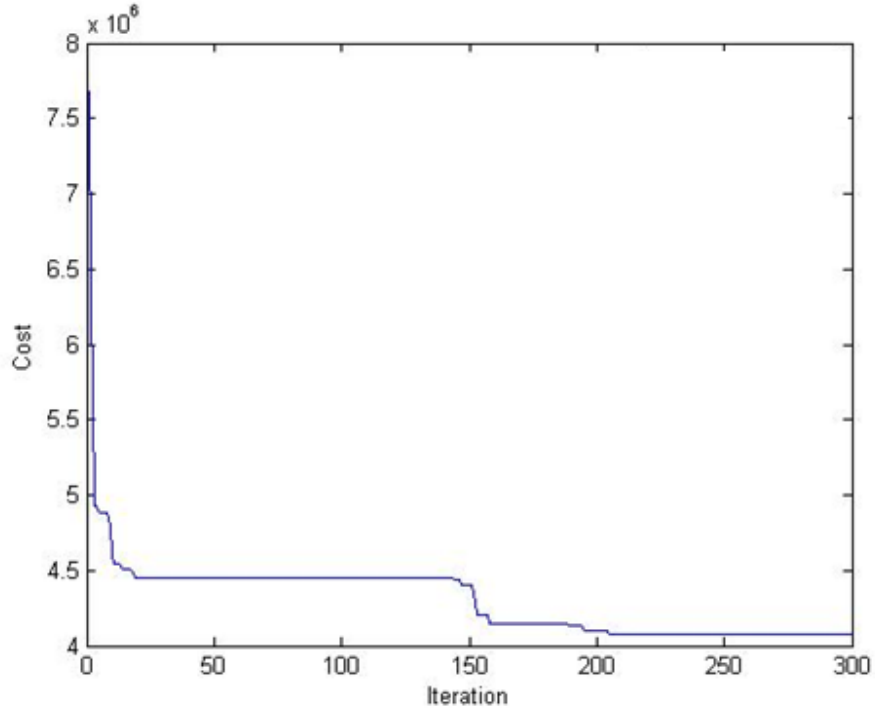

Fig. 2. The obtained results for 20 agents and 200 tasks.

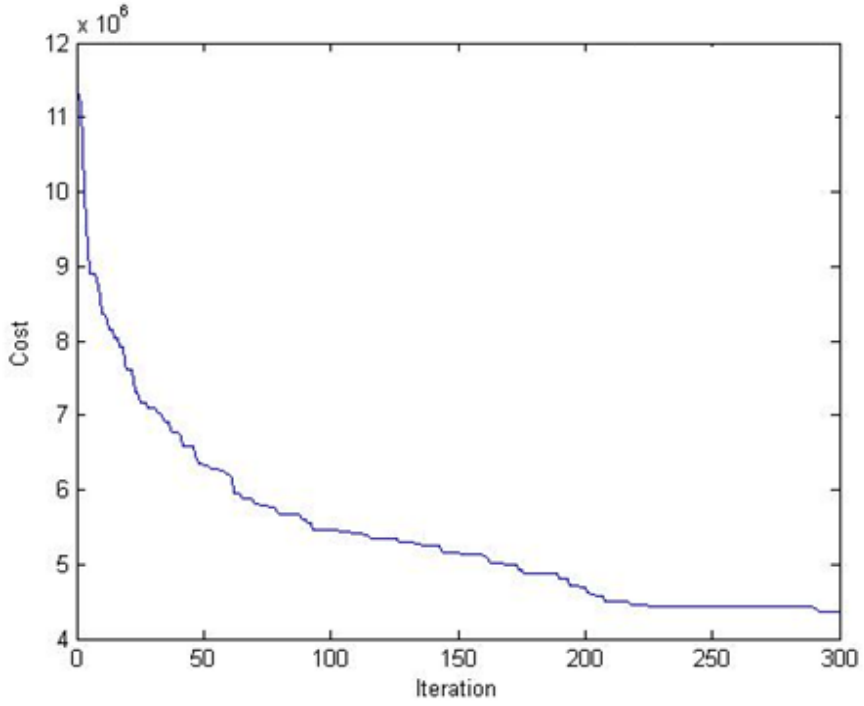

Fig. 3. The obtained results for 10 agents and 250 tasks.

6. Conclusion

Nowadays, distributed computing model has been transformed to a developing concept. Beside, agent scheduling is one of the main component of any practical agent system. In this paper, a new Gravitational Emulation Local Search Algorithm based algorithm named GELS_MAS is proposed for task scheduling problem in MAS with the objective of minimizing its overall completion time or scheduling length. The main advantages of this algorithm are reducing execution time and increasing the efficiency of the system. The performance of the proposed algorithm is compared with Genetic Algorithm which has been presented in the literature previously. The obtained results showed that proposed algorithm has a high efficiency in solving task scheduling problem in Multi-agent System and can obtain better results. As an applicable suggestion for future studies, another efficient algorithm can be developed in order to generate much better solutions that may be achievable by hybridizing the algorithm with some other algorithm such as GA.

References

[1] Morris, A. (2008). *SAMSON: Strong multi-agent simulation of wireless sensor networks*. Ph.D., thesis, University of Edinburgh, 1-87.

[2] Yang, R., & Wang, L. (2013). Development of multi-agent system for building energy and comfort management based on occupant behaviors. *Energy and Buildings*, 56, 1-7.

[3] Baig, Z. A. (2012). Multi-agent systems for protecting critical infrastructures: A survey. *Journal of Network and Computer Applications*, 35(3), 1151-1161.

[4] Li, B., Zhang, X., & Wu, J. (2009, December). A Constrained Optimal Task Scheduling Problem in Multi-Agent System. International Conference on Computational Intelligence and Software Engineering, CiSE 2009, 1-4.

[5] Kim, Y. H., Han, S., Lyu, C. H., & Youn, H. Y. (2009, August). An efficient dynamic load balancing scheme for multi-agent system reflecting agent workload. International Conference on Computational Science and Engineering, 216-223.

[6] Shehory, O., & Kraus, S. (1998). Methods for task allocation via agent coalition formation. *Artificial intelligence*, 101(1), 165-200.

[7] Choi, H. L., Brunet, L., & How, J. P. (2009). Consensus-based decentralized auctions for robust task allocation. *IEEE transactions on robotics*, 25(4), 912-926.

[8] Tošić, P. T., & Agha, G. A. (2004, December). Maximal clique based distributed coalition formation for task allocation in large-scale multi-agent systems. In International Workshop on Massively Multiagent Systems, Springer, Berlin, Heidelberg, 104-120.

[9] Aknine, S., & Shehory, O. (2006, December). A feasible and practical coalition formation mechanism leveraging compromise and task relationships. In Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology, IEEE Computer Society, 436-439.

[10] De Weerdt, M., Zhang, Y., & Klos, T. (2007, May). Distributed task allocation in social networks. In Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems, ACM, 76.

[11] Abdallah, S., & Lesser, V. (2005, July). Modeling task allocation using a decision theoretic model. In Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, ACM, 719-726.

[12] Manisterski, E., David, E., Kraus, S., & Jennings, N. R. (2006, May). Forming efficient agent groups for completing complex tasks. In Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, ACM, 834-841.

[13] Talbi, E. G. (2009). *Metaheuristics: from design to implementation*. John Wiley & Sons, 74.

[14] Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press, 1-223.

[15] Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical computer science*, 344(2-3), 243-278.

[16] Yonezawa, Y., & Kikuchi, T. (1996, October). Ecological algorithm for optimal ordering used by collective honey bee behavior. Proceedings of the Seventh International Symposium in Micro Machine and Human Science, 249-256.

[17] Bai, Q. (2010). Analysis of particle swarm optimization algorithm. *Computer and information science*, 3(1), 180.

[18] Glover, F., & Laguna, M. (1998). Tabu search. *In Handbook of combinatorial optimization*, Springer, Boston, MA, 2093-2229.

[19] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671-680.

[20] Webster, B. L. (2004). *Solving Combinatorial Optimization Problems Using a New Algorithm Based on Gravitational Attraction*. Ph.D., thesis, Melbourne, Florida Institute of Technology, 1-250.

[21] Ranatunga, V., Hosokawa, A., Kinoshita, K., Yamai, N., & Murakami, K. (2007, January). A Fair and Effective Scheduling Algorithm for Multi-Agent Systems. 2nd International Conference on Communication Systems Software and Middleware, COMSWARE 2007, 1-8.

[22] Adhau, S., Mittal, M. L., & Mittal, A. (2013). A multi-agent system for decentralized multi-project scheduling with resource transfers. *International journal of production economics*, 146(2), 646-661.

[23] Hsieh, F. S., & Lin, J. B. (2013, May). A simulation environment for scheduling workflows in multi-agent systems. International Conference on Advanced Robotics and Intelligent Systems (ARIS), 116-121.

[24] Zhenqiang, B., Weiye, W., Peng, W., & Pan, Q. (2012). Research on production scheduling system with bottleneck based on multi-agent. Physics Procedia, 24, 1903-1909.

[25] Li, Q. S., & Du, L. M. (2009, July). Model design of job shop scheduling based on Multi-agent system. International Conference on Services Science, Management and Engineering, SSME'09. IITA, 233-236.

[26] Yumin, D., & Shufen, X. (2007, July). Hybrid Application On Job-Shop Scheduling by Genetic Algorithm and MAS Spring-Net. Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD 2007, 2, 579-584.

[27] Wang, Z., & Liu, Y. (2006, October). A multi-agent agile scheduling system for job-shop problem. Sixth International Conference on Intelligent Systems Design and Applications, ISDA'06, 2, 679-683.

[28] Li, Q., & Du, L. (2009, October). Research on hybrid-genetic algorithm for mas based job-shop dynamic scheduling. Second International Conference on Intelligent Computation Technology and Automation, ICICTA'09, 1, 404-407.

[29] Nie, L., Bai, Y., Wang, X., Liu, K., & Cai, C. (2012, May). An agent-based dynamic scheduling approach for flexible manufacturing systems. 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD), 59-63.

[30] Shah, S. N. M., Haron, N., Zakaria, M. N. B., & Mahmood, A. K. B. (2012). Agent-based Robust Grid Scheduling Framework for High Performance Computing. *AASRI Procedia*, 1, 554-560.

[31] Shah, S. N. M., Zakaria, M. N. B., Haron, N., Mahmood, A. K. B., & Naono, K. (2012). Design and evaluation of agent based prioritized dynamic round robin scheduling algorithm on computational grids. *AASRI Procedia*, 1, 531-543.

[32] Zunino, A., & Campo, M. (2009). Chronos: A multi-agent system for distributed automatic meeting scheduling. *Expert Systems with Applications*, 36(3), 7011-7018.

[33] Barenji, A. V., Barenji, R. V., & Hashemipour, M. (2016). Flexible testing platform for employment of RFID-enabled multi-agent system on flexible assembly line. *Advances in Engineering Software*, 91, 1-11.

[34] Ma, Q., & Miao, G. (2015). Output consensus for heterogeneous multi-agent systems with linear dynamics. *Applied mathematics and computation*, 271, 548-555.

[35] Saboori, I., & Khorasani, K. (2015). Actuator fault accommodation strategy for a team of multi-agent systems subject to switching topology. *Automatica*, 62, 200-207.

[36] Huang, N., Duan, Z., & Chen, G. R. (2016). Some necessary and sufficient conditions for consensus of second-order multi-agent systems with sampled position data. *Automatica*, 63, 148-155.

[37] Sun, Y., Wang, Y., & Zhao, D. (2015). Flocking of multi-agent systems with multiplicative and independent measurement noises. *Physica A: Statistical Mechanics and its Applications*, 440, 81-89.

[38] Voudouris, C., & Tsang, E. P. (2003). *Guided local search*. In Handbook of metaheuristics (pp. 185-218). Springer, Boston, MA.

[39] Hosseinabadi, A. A., Kardgar, M., Shojafar, M., Shamshirband, S., & Abraham, A. (2014, November). GELS-GA: hybrid metaheuristic algorithm for solving multiple travelling salesman problem. 14th International Conference on Intelligent Systems Design and Applications (ISDA), 76-81.

[40] Rostami, A. S., Mohanna, F., Keshavarz, H., & Hosseinabadi, A. A. R. (2015). Solving multiple traveling salesman problem using the gravitational emulation local search algorithm. *Applied Mathematics*, 9(2), 1-11.

[41] Farahabadi, A. B., & Hosseinabadi, A. R. (2013). Present a new hybrid algorithm scheduling flexible manufacturing system consideration cost maintenance. *International Journal of Scientific & Engineering Research*, 4(9), 1870-1875.

[42] Hosseinabadi, A. A. R., Farahabadi, A. B., Rostami, M. H. S., & Lateran, A. F. (2013). Presentation of a new and beneficial method through problem solving timing of open shop by random algorithm gravitational emulation local search. *International Journal of Computer Science Issues* (*IJCSI*), 10(1), 745.

[43] Hosseinabadi, A. A. R., Siar, H., Shamshirband, S., Shojafar, M., & Nasir, M. H. N. M. (2015). Using the gravitational emulation local search algorithm to solve the multi-objective flexible dynamic job shop scheduling problem in Small and Medium Enterprises. *Annals of Operations Research*, 229(1), 451-474.

[44] Hosseinabadi, A. A. R., Rostami, N. S. H., Kardgar, M., Mirkamali, S., & Abraham, A. (2017). A new efficient approach for solving the capacitated vehicle routing problem using the gravitational emulation local search algorithm. *Applied Mathematical Modelling*, 49, 663-679.

[45] Meng, A., Ye, L., Roy, D., & Padilla, P. (2007). Genetic algorithm based multi-agent system applied to test generation. *Computers & Education*, 49(4), 1205-1223.

[46] Tirkolaee, E. B., Goli, A., Bakhsi, M., & Mahdavi, I. (2017). A robust multi-trip vehicle routing problem of perishable products with intermediate depots and time windows. *Numerical Algebra, Control & Optimization*, 7(4), 417-433.

[47] Tirkolaee, E. B., Alinaghian, M., Hosseinabadi, A. A. R., Sasi, M. B., & Sangaiah, A. K. (2018). An improved ant colony optimization for the multi-trip Capacitated Arc Routing Problem. *Computers & Electrical Engineering*.

[48] Mirmohammadi, S. H., Babaee Tirkolaee, E., Goli, A., & Dehnavi-Arani, S. (2017). The periodic green vehicle routing problem with considering of time-dependent urban traffic and time windows. *Iran University of Science & Technology*, 7(1), 143-156.