A MULTI-AGENT SYSTEM FOR GENERATION OF UNIVERSITY TIME SCHEDULES

Abderrahim Siam[1], Souidi Mohammed El Habib[1] and Samir Safir[2]

[1]ICOSI Lab, Abbes Laghrour University, Khenchela, Algeria
[2]Department of Mathematics and Computer Engineering,
Abbes Laghrour University, Khenchela, Algeria

ABSTRACT

This paper presents an approach based on a multi agent system vision for the resolution of the problem of university timetable. This problem is very important. Indeed a bad schedule affects negatively affects the level of student acquisition. This problem is an arduous problem whose handmade achievement is a drastic task that can mobilize several people several days of work without any guarantee on the quality of the found solutions. The proposed approach involves agents to find timetables that constitute a compromise of a multitude of points of view. Each agent works on behalf of a teaching actor. After a first phase where all the possible timetables are generated, a second phase is executed in the form of several iterations. In each cycle, each agent eliminates the least preferred schedule among possible schedules. Thus, it remains timetables that represent compromises between all agents and therefore the teaching actors.

KEYWORDS

Multi Agent Systems, University Timetable, &Resolution

1. INTRODUCTION

The problem of the time schedules which consists in general on assigning a set of tasks and / or activities to a set of resources based on specific time periods is an instance of the problems of scheduling tasks. It is an NP-complete problem. In other words, none of the existing algorithms is able to solving all the instances of the problem on its own in a polynomial time. This problem is omnipresent in all practical aspects of modern society and in many organizations such as hospitals, transport companies, protection and emergency services and universities.

Several researchers have confronted the problem of time schedules according to several points of view and with different approaches based on different paradigms of resolution. In this paper, we propose a Multi-Agents solution. Multi-agent systems [1] present an excellent paradigm for the analysis, modelling and implementation of systems with a set of conflicting objectives. This is due to the interesting features of multi-agent systems in terms of coupling flexibility and abstraction. By coupling flexibility we refer to the ability to structure the system to be developed as a combination of interacting software entities; the ability to describe the relationships between the software entities constituting the system independently of software entities themselves; and highly advanced direct and indirect communication and interaction mechanisms such as message exchange, coordination, negotiation, ... etc. By abstraction, we refer to the concepts of very high levels of abstraction introduced by this paradigm ranging from the agent to artificial organizations and the manipulation and exchange of knowledge rather than data.

Dhinaharan Nagamalai et al. (Eds) : WIMO, ICAIT, ICDIPV, NC - 2019
pp. 23-32, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.90403

2. THE UNIVERSITY TIME SCHEDULING PROBLEM

The time scheduling problem consists on assigning a set of tasks and / or activities to a set of resources based on specific time periods, subject to a set of constraints. The constraints to consider differ from one problem to another depending on the problem specificity as well as the expected characteristics of the desired timetable. These constraints are to classify into two classes [2]. The first one is hard constraints that must always be vitrified; a schedule or a timetable that does not meet these constraints is unfeasible or unacceptable. The second category includes constraints often called soft constraints and preferences that do not require strict verification; they allow to approach as much as possible of the desired objective. These preference constraints are used to express what must be a good schedule. These constraints are more difficult to formalize than hard constraints and their processing is more difficult. Several approaches dealing with this problem relax the preference constraints and introduce them as an Objective Function whose optimization allows getting closer to the satisfaction of the constraints.

The University timetabling UTT problem is an instance of the most well-known cyclical scheduling problems. It is a question of scheduling the tasks (which have a cyclical character) of a set of teachers by allocating them a set of classrooms and setting them the start and end dates. Building a timetable that meets all the needs of a university is really important; But also it is quite difficult and complicated. With advances in hardware and software technologies, the scientific community continues to work on this issue to develop formal and automated procedures for developing effective and desirable timetables. The University timetabling problem concerns two aspects [3]: courses and exams. Different aspects separate these two categories. For example, we try to group courses, while we prefer to distance the exams from each other as much as possible. Or, a class can be held at a given time in a room, while several exams can be held at the same time in the same room or the same exam can be arranged in several rooms. In this work, we focus on courses timetabling.

3. EXISTING RESOLUTION APPROACHES

In the literature, there are a very large number of approaches for the resolution of the UTT problem. These approaches are to be classified in centralized approaches and distributed or decentralized approaches [2]. Unlike centralized approaches, decentralized approaches provide methods in which sequential execution is decreased.

3.1. Centralized Approaches

Centralized approaches are classified in: approaches based on exact methods; approaches based on constraints satisfaction problems and approaches based on approximate methods.

The exact methods such as linear programming and graph theory try to ensure the completeness of the resolution. Several approaches based on exact methods have been proposed to solve the UTT problem. Among these approaches, we cite as examples the approach proposed in [4] based on graph theory and the approaches proposed in [5] based on the Graph Coloring Problem GCP technique. With such methods, the computational time required usually increases exponentially with the size of the problem. despite the advances made with exact methods, which have helped to solve problems in an optimal way, these methods generally have difficulties with large size problems because the search for an optimal solution may be totally inappropriate in some practical applications because the size of the problem, the dynamics that characterize the work environment, the lack of precision in the data collection, the difficulty of formulating constraints in explicit terms or the presence of conflicting objectives.

Several researchers have chosen to formulate the scheduling problem as a constraint satisfaction problem (C.S.P). Constraint satisfaction problems or CSP were introduced by [6]. CSP are mathematical problems where we look for states or objects satisfying a certain number of constraints or criteria. A constraint satisfaction problem (CSP) consists of a set of variables defined by a corresponding set of possible values and a set of constraints. One solution to the problem is to assign a value from the first set to each variable from the second one so that all constraints are satisfied. The constraints are managed through a propagation system that reduces the variable domains and pruning the search space. The propagation mechanism associated with a backtracking scheme allows possible to fully explore the search space. Several formulations of the UTT problem by CSP have been proposed. We cite as examples the works proposed in [7], [8] [9] and [10]. The intention in the work [9], is to explore two different heuristics to segment the UTT problem into sub-problems in order to solve them effectively. Each sub-problem is solved as a constraint satisfaction problem (CSP). Once the UTT is partitioned and each part solved separately, two different strategies are proposed to integrate the solutions and obtain a complete solution. The work [10] proposes a mathematical constraint satisfaction model which defines the UTT problem with the use of a constructive Approach to obtain solutions in the proposed model.

All these works suffer from the same insufficiencies, such as the useless exploration of certain areas of the research space, the redundancy of local inconsistencies, the non-taking into account of information on the inconsistency of partial instantiations, and especially its exponential complexity which is strongly combinatory.

In the approximate methods, the goal is to find a good solution in a reasonable calculation time without trying to guarantee the optimality of the obtained solution. The approximate methods are based mainly on various heuristics, often specific to a type of problem. Meta heuristics are an important part of the approximate methods. A Meta heuristic is an optimization algorithm aimed at solving difficult optimization problems for which no more efficient classical method is known. Several classifications of Meta heuristics have been proposed; most generally distinguish two categories: Neighborhood-based methods such as: simulated annealing and tabu; Population-based methods: such as genetic algorithms, ant colony algorithms and evolutionary programming and evolution strategies. We quote as examples three works for solving the UTT problem based on approximate methods. In [11], authors chose to adapt genetic algorithms to solve this problem. Their formulation consists in that each chromosome is considered as a schedule of a room and each gene in the chromosome contains information on the various courses that are programmed in this room for specific periods of time. In addition to an evaluation function to measure the degree of violation of hard constraints and for determining the number of compilation errors. The authors presented an experiment to find a better combination where the most optimal solution was found with the following parameters: 200 iterations, Popsize = 4, crossover = 0.75 and mutation between 0.3 and 0.4. In [12], authors used tabu research to explore the research space of this problem by going through three constructive phases, the initialization phase which allows for a rapid construction of an initial schedule of tasks using the greedy heuristic, then the intensification phase whose execution of the tabu search algorithm and finally the diversification phase which is based on a disturbance operator for the reduction of the number of constraint violations. In [13] authors propose a heuristic algorithm for university course timetabling problem. In the proposed solution several timetables are generated because the random or stochastic character of operations and steps of the algorithm. The best solution is chosen among the generated ones through measure index.

It is admitted that, from a very general point of view, no metaheuristic is really better than another. Indeed, a metaheuristic can not claim to be more efficient on all the problems, although certain instances (that is to say the algorithm itself, but also a choice of parameters and a given implementation) could be more adapted than others on certain classes of problems.

In the final analysis, it is sometimes possible that the choice of the representation of the solutions, or more generally methods associated with the metaheuristic, has more influence on the performances than the type of algorithm itself. In practice, however, metaheuristics are more powerful than exhaustive or purely random search methods. It is often necessary to adapt the algorithm to the optimized problem. The choice of the representation of the manipulated solutions is crucial. Then, most metaheuristics have parameters whose adjustment is not necessarily trivial. Finally, obtaining good performance usually involves a step of adapting the various steps of the algorithm.

Neighborhood -based methods and population-based methods are often combined into so-called hybrid methods in order to fully exploit the power of searching for neighborhood and recombination methods of evolutionary algorithms on a population of solutions.

3.2. Distributed Approaches

We can distinguish the existence of a few distributed approaches for solving the UTT problem. Several approaches based on the distributed satisfaction of constraints have been proposed, such as the approach proposed in [14]. On the other hand, several approaches based on multi-agent systems have been proposed. In [15] a multi-agent architecture using Mobile Agent technology is presented. This architecture, is composed of five platforms "Week-day" (presenting the five days of work a week), and two types class of agents: a first class of type mobile agents representing the set of courses to be taught "Agent-course", and a second class of type stationary agents containing an "Agent-Interface" for the initialization of the system, an "Agent-Publisher" for combination and display of results, and five "Agent-Signboards" successively in the five "Week-day" platforms to organize and facilitate the negotiation between the different agents-courses. The approach has been tested by two implementations. One centralized implementation and another distributed. With the following parameters: (100 courses, 100 teachers, 10 classrooms, 5 days of weeks with 9 periods of time per day), it was found that the distributed version is more efficient in terms of time with a difference of 33 , 22 minutes compared to the centralized version. With 45 periods per day while maintaining the other parameters the distributed version is more efficient in terms of time with a gap of 15.8 minutes compared to the centralized version. In [16], authors presented a distributed model based on cooperative agents for the resolution of the university time-use problem. This model is composed of two classes of agents: a first class grouping together a set of agents each of whom is responsible for a single department of the faculty, they are of the "Timetable Agent (TA)" type, a second class containing a single agent "Mediator Agent (MA)" to guide the process of negotiations between the various agents of the system. By passing the startup step, the authors did not impose a stopping parameter. Each algorithm can only finish if it finds a locally satisfactory solution. In the experiments, we can distinguish that the approach gives good results by using the local search heuristic for each TimeTable-Agent.

Several multi agents' systems architectures were proposed for the UTT problem. In addition to the two examples presented, we can mention the works [17],[18],[19], and [20].A survey of various agent systems that have been utilized in solving university course time tabling problem may be found in [21]. Some of the existing proposed systems suffer from lack of intelligence approach, some are not implemented and the majority of existing approaches did not manage to adapt this formalism well to generate a solution that satisfy the totality of the constraints of the problem.

4. MULTI AGENT PROPOSED APPROACH

It is clear that approaches based on exact methods suffer from the exponential dependence of computation time on the size of the problem. Hence their application for generating large UTT is not possible. Although approaches based on approximate methods can produce good solutions, these approaches are materialized in a sequential execution form and do not exploit the possibilities of executions on several threads of activities. As well as, in certain situations, it is necessary to have a solution of good quality (that is to say fairly close to the optimal) in a context of limited resources (computing time and / or memory). In this case, the optimality of the solution is not guaranteed, nor even the difference with the optimal value. Nevertheless, the time required to obtain this solution is much lower than in the case of an exact method.

The proven power of multi agent approaches is not fully exploited to provide Multi agents solutions for the UTT problem. This is evidenced by the existence of few approaches to solving the UTT problem based on Multi agent systems. The approaches that we find in the literature do not focus on the preferences of teachers who are looking for organized schedules given their multiple occupations. It is important to remember that the Multi agent systems approach shows very important characteristics that we quoted in the introduction.

In this paper, we propose a new multi agent approach for UTT generation. The general idea of the approach schematized in Figure 1 is as follows: the resolution is done in two phases. In the first phase, a set of agents called initiators agents generate all possible schedules that meet defined criteria. In the second, a set of agents called controllers intervene. Each agent acts on behalf of a teacher and defends his interests. This phase takes place in several iterations. At each iteration, each controller eliminates the least preferred UTT. Thus, as the system progresses in the execution, the system converges to maintain the UTTs that satisfied the totality of the agents, and thus the teachers that they represent. In this second phase, controllers agents work in parallel under the constraint that an iteration is completed only if each controller has eliminated one and only one solution. In the end we get the solutions that compromise between all Interferers.

The underlying idea is: The UTT problem can be seen as a non-cooperative game. Each teacher i is a player who must choose a timetable. Players must make their choice in a coordinated way to maximize the expectation of rewards: indeed, it depends on the choice of all teachers. More formally, the UTT problem can be represented as a non-cooperative game TT = (P; A; u). The players are the teachers p ϵ P; The possible actions of the teacher i are the possible generated schedules generated (Ai) i ϵ P;The profile strategy of a teacher i represents all the courses that i teach; The utility function u (pi) is the value of an UTT compared to the preferred UTT (proposed) by the teacher i. This function can be defined in different ways.

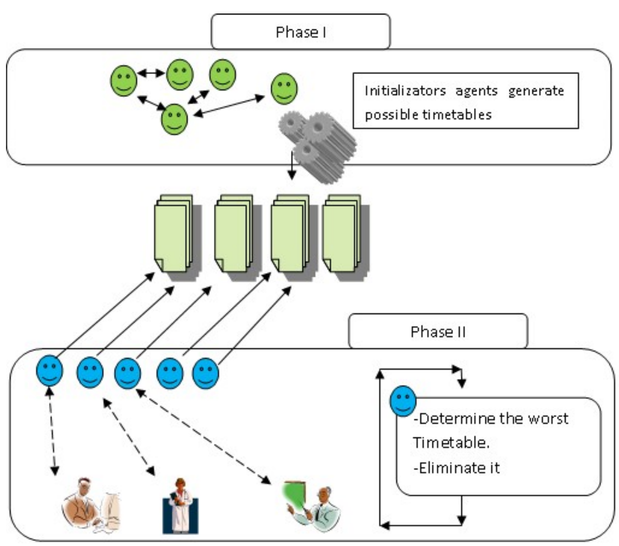

Figure 1. Schematic description of the proposed approach

4.1. First Phase, Generating Possible Schedules

The initialization agents generate all possible timetables, as well as calculating the preference or utility function of each schedule generated with respect to each teacher. This calculation takes place in parallel hence the utility of multi-agent systems.

A timetable tt is represented by the structure: List < Lecture > tt[maxSession; maxDay] where: maxSession and maxDay represent the maximum number of sessions in a day and days in the week respectively; tt[i; j] =set of (List) lectures during the session i in the day j; Lecture is a structure defined as a lecture (i.e. course C is taught by professor P for group G during a given period). The generated timetables are stoked in a hash table of the form:

Dictionary < List < Lecture > [maxSeance; maxJour]; bool > TTs

Where the keys of the hash table are timetables and the value *bool* is a value that is used for marking.

The generation of the timetables may be represented by the algorithm in the figure 2 where the function GeneNewTT generates a timetable from a Lecture and the function CombineTT combines two timetables.

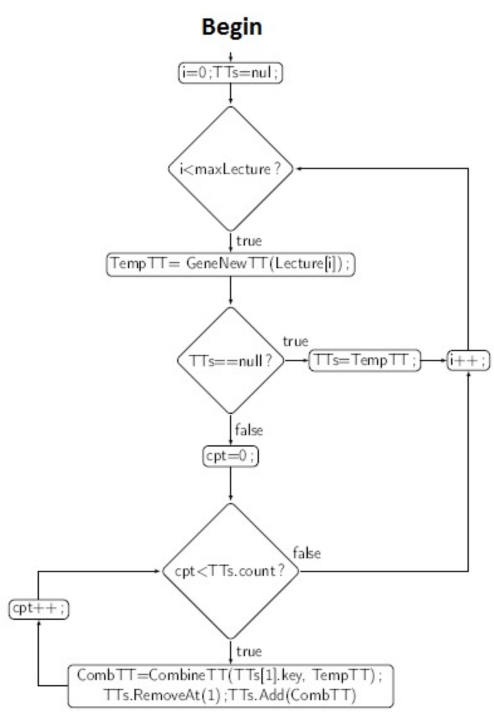

Figure 2. Generation of possible timetables

4.2. Second Phase, Iterative Elimination of Non-Preferred UTTs

In this phase, repetitive processing is performed as iterations. In each iteration, each controller agent proceeds to eliminate the least preferred UTT. The degree of preference of each UTT with respect to each agent is measured by an utility function whose behavior is described in the algorithm illustrated in Figure 3. The communication between the controllers agents is indirect, these agents communicate via sharing a common structure data (blackboard) TTs that contains all the initial UTTs generated by the Initialization Agents. Controllers Agents can read and write to this TTs memory at the same time through a mechanism similar to the .Net Concurrent Dictionary mechanism that represents a thread-safe collection of key / value pairs accessed by multiple threads simultaneously. This indirect communication mechanism is very flexible and more efficient than a direct mechanism. In each iteration, each controller tries to eliminate the worst UTT with respect to its preferences. In each iteration, each agent must eliminate one and only one UTT. The progress of this phase can be described in the form of the algorithms of figure 4. After the last iteration, only the UTTs that have compromises between all the agents of the system remain.

Algorithm 1 preference calculation for the teacher i

procedure PREFERENCE CALCULATION
  Input : TTs TimeTables generated, ω TimeTable proposed by the Teacher i
  Output : P_i Array contain the values of preference function of the Teacher i
  for $m = 0$ to k **do**
    double $f = 0$;
    for each course $c_j \in C$ teached by the Teacher i **do**
      if $TTs[m].key[P_i][H_k][C_j] = \omega[P_i][H_k][C_j]$ **then**
        $f = f + 1$;
    $P_i[m] = f/C.size$;

Figure 3. Preference calculation algorithm

4.3. Discussion

The approach presented in this paper differs from the proposed agent-based approaches in that the results produced can be seen from the angle of emerging phenomena. In reality, there is no direct communication between the agents. Each agent defends his interests by making a choice without being forced to negotiate with other agents in the system. This mode of operation allows a great flexibility and leaves open the choice of the architecture of the agents. It is even possible to use agents of different types and architectures. According to a stop condition representing the number of solutions to hold N. the system ends up preserving the N best solutions constituting compromises between the agents of the system.

To measure its performance, a simulation of this approach is prepared in Visual CSharp .Net where the agents are implemented as threads. The generated UTTs are manipulated by the controller agents through the .Net ConcurrentDictionary mechanism. This mechanism represents a thread-safe collection of key / value pairs accessed by multiple threads simultaneously. The obtained results are encouraging because the system converges in all the experiments conducted to the compromise UTTs. The performance of the approach in terms of computation time is acceptable despite the fact that the environment in which the implementation is carried out is limited. This environment consists in desktop computer with the following characteristics: An Intel (R) Core (TM) 2 Quad processor and a RAM (R.A.M) of 4 GB. The simulation involved two versions. In the first version agents access one after the other sequentially in the second phase. In the second version agents access in parallel and each one eliminates the least preferred solution while ensuring a mutual exclusion mechanism in the case where two or more agents decide to eliminate the same solution. Some results of the simulation are as follows.

In a simple test with 3 teachers, 2 days per week, 3 periods per day; the generation of the possible timetables and the selection of the best solution lasted 0.625 seconds with a satisfaction 100% of all the teachers for the sequential version. For the parallel version with 100% of satisfaction the time is 0,223 seconds.

In another case, with 50 teachers, 3 days per week, 5 periods per day; the generation of the possible timetables and the selection of the best solution lasted 3 hours and 12 minutes seconds with a satisfaction of 65% average for all the teachers for the sequential version. For the parallel version with 65% of satisfaction the time is about 59 minutes. It is important to point out that the simulation is done on a small machine. With a more powerful calculator, the results will be better.

Algorithm 2 eliminate the worst timetable generated by the Teacher i

```
procedure REMOVEWORST
Input : TTs generated TimeTables, P_i Array contain the values of preference function of
the Teacher i
Output : return true if agent i eliminate the worst TimeTable , else false .
    for each (KeyValuePair<int, double> item in P_i.OrderBy(k => k.Value)) do
        if StillOne(TTs) then
            return false ;
        KeyValuePair<List<int>[,], bool> TT = TTs.ElementAt(item.Key) ;
        if TT.Value == false then
            TTs[TT.Key] = true ;
            P_i.Remove(item.Key) ;
            return true ;
        else
            P_i.Remove(item.Key) ;
    return false ;
```

Algorithm 3

```
procedure EQUILIBRIUM
Input : TTs generated TimeTables, AgentTeacherList list of Agents Controlleurs
Output : equilibrated timetable.
    var agentList = new List < Task > ();
    int lenght = TTs.Count ;
    int i = 0;
    bool stop = false ;
    while i < lenght and !stop do
        for each AgentTeacher agent ∈ AgentTeacherList do
            var task = new Task(() ⟹ { if (!agent.RemoveWorst(TTs))stop = true ;}) ;
            task.Start();
            agentList.Add(task);
            i++;
        Task.WaitAll(agentList);
```

Figure 4. Elimination worst and progression algorithms

5. CONCLUSIONS

Papers The UTT problem is a complex process. It is a combinatorial problem very difficult to solve because a solution of the problem is to represent by a set of properties with the aim to find the best combination of these properties. Another cause of complexity of the problem is the size of the problem. Indeed the size of the problem for modest institutions easily reaches the order of 1000 lessons to order per week in a competing manner on three levels: periods, teachers and locals. Also the estimation of the degree of satisfaction of the constraints of preference is often difficult to formulate in addition to the fact that they can be sometimes contradictory.

In this paper we presented a multi agent approach that adds to all the proposed approaches to solve the UTT problem based on different resolution methods. In this approach the parallelism of treatments is pushed forward with the adapted multi agent vision. The approach allows finding good UTTs which present real compromises between different points of view. A Simulation of this approach is carried out in Visual CSharp .Net whose results are encouraging.

Despite encouraging results from our work, many improvements are still possible. We can reduce the execution time of the generation of possible UTTs by using Meta heuristics. Also, the utility Function can be improved by adding other soft constraints.

REFERENCES

[1] Jacques, Ferber (1995) Les systèmes multi-agents Vers une intelligence collective, InterEditions

[2] Houssem, Eddine Nouri & Olfa, Belkahla (2015) Résolution multi-agents du problème d'emploi du temps universitaire, Éditions universitaires européennes Publishers.

[3] Edmund K, Burke & Sanja, Petrovic, (2004) “Timetabling and Rostering”, European Journal of Operational Research, Vol. 153, No. 1, pp1-2.

[4] Timothy, A Redl, (2007) “University timetabling via graph coloring: an alternative approach”, Proceedings of the Thirty-Eighth Southeastern International Conference on Combinatorics, Graph Theory and Computing., Vol. 187, pp174-186.

[5] Edmund K, Burke, Jakub, Mare£ek, Andrew J, Parkes, & Hana Rudová, (2009)) “A supernodal formulation of vertex colouring with applications in course timetabling”, Annals of Operations Research, Vol. 179, No. 1, pp105-130.

[6] Ugo, Montanari, (2009) Networks of constraints : fundamental properties and applications to picture processing, Information Sciences.

[7] Zhang Lixi. & Sim, Kim Lau, (2005) “Constructing university timetable using constraint satisfaction programming approach”, International Conference on Computational Intelligence for Modelling, Control and Automation, pp28-30.

[8] Ho, Sheau Fenn Irene, Safaai-Deris, & Siti, Zaiton-Mohd Hashim, (2009) “Investigating Constraint-Based Reasoning for University Timetabling Problem ”, Proceedings of the International MultiConference of Engineers and Computer Scientists, Vol. 1.

[9] Magaña-Lozano D.J., Conant-Pablos S.E., & Terashima-Marin H, (2013) “Exploring the Solution of Course Timetabling Problems through Heuristic Segmentation ”, Batyrshin I., González Mendoza M. (eds) Advances in Artificial Intelligence. MICAI 2012. Lecture Notes in Computer Science, vol 7629. Springer, Berlin, Heidelberg

[10] Marco, Antonio Cruz-Chávez, Mireya Flores-Pichardo, Alina Martínez-Oropeza, Pedro Moreno-Bernal, & Martín H. Cruz-Rosales, (2016) “Solving a Real Constraint Satisfaction Model for the University Course Timetabling Problem: A Case Study”, Mathematical Problems in Engineering, Vol. 2016.

[11] Aderemi O, Adewumi Babatunde A, Sawyerr & Montaz, Ali, (2009) “A heuristic solution to the university timetabling problem”, Engineering Computations: International Journal for Computer-Aided Engineering and Software, Vol. 26, No. 8, pp972-984.

[12] Zhipeng, Lû & Jin-Kao Hao, (2010) “Adaptive Tabu Search for Course Timetabling”, European Journal of Operational Research, Vol. 200, No. 1, pp235-234.

[13] Mohadese, Kaviani Hadi, Shirouyehzad Seyed, Mojtaba Sajadi & Mohammadreza ,Salehi, (2014) “A heuristic algorithm for the university course timetabling problems by considering measure index: a case study”, International Journal of Services and Operations Management, Vol. 18, No. 1, pp01-20.

[14] Yang, Xiang & Wanling Zhang, (2008) "Distributed University Timetabling with Multiply Sectioned Constraint Networks", Association for the Advancement of Artificial Intelligence, the Twenty-First International FLAIRS Conference.

[15] Yan, Yang Raman, Paranjape Luigi, Benedicenti & Nancy, Reed, (2005) "A Mobile Agent System for University Course Timetabling", Indian International Conference on Artificial Intelligence (IICAI-05).

[16] Joe Henry, Obit Dario, Landa-Silva, Djamila, Ouelhadj, Teong, Khan Vun & Rayner Alfred, (2011) "Designing a Multi-agent Approach System for Distributed Course Timetabling", 11th International Conference on Hybrid Intelligent Systems (HIS). IEEE.

[17] Olfa, Belkahla Driss & Houssem Eddine, Nouri, (2013) "Solving a University Course Timetabling Problem by Negotiation in a Multi-agent System", Journal of E -Technology, Vol. 4, No. 1.

[18] Yan, Yang Raman, Paranjape Luigi, Benedicenti & Nancy, Reed, (2011) "A Multi Agent System for Course Timetabling", Journal of Intelligent Decision Technologies, Vol. 5, No. 2, pp113-131.

[19] Rafal, Tkaczyk Maria, Ganzha & Marcin, Paprzycki, (2016) "AgentPlanner - Agent-based timetabling system", Informatica, Vol. 40, No. 1, pp3-17.

[20] Nicholas, Oluwole Ogini Noah, Oghenefego Ogwara & Obeten, Obi Ekabua, (2018) "Intelligent based Multi-Agent Approach for University Timetable Scheduling System", International Journal of Computer Applications, Vol. 182, No. 1, pp10-21.

[21] Nicholas, Oluwole Ogini Noah, Oghenefego Ogwara & Obeten, Obi Ekabua, (2017) "A Review of Agent Based University Course Time Tabling Systems", International Journal of Engineering Development and Research, Vol. 5, No. 2, pp565-585.

AUTHORS

Dr Abderrahim Siam received his BS degree in Computer Science from University of Batna (Algeria) in 2002, and MS degree in Computer Science from University of Oum El Bouaghi (Algeria) in 2005and his Ph.D in Computer Science from University of Constantine,Algeria. He is working as professor in Department of Mathematics and Computer Science in University of khenchela (Algeria). He is currently the vice rector of University of khenchela (Algeria) and and his research interests include software engineering, fuzzy logics, formal methods, multiagent systems and complex systems.

Dr Souidi Mohammed El Habib received his Master degree in Computer Science from University of Khenchela (Algeria) in 2012 and his Ph.D in Computer Science from Harbin Institute of Technology (China) 2017 . He is working as assistant professor in Department of Mathematics and Computer Science in University of khenchela (Algeria).

Samir Safir : Master student in Computer Science in the University of khenchela (Algeria).