Available online at www.scientiaresearchlibrary.com

Scientia Research Library **ISSN 2348-0424**
USA CODEN: JETRB4

Journal of Engineering And Technology Research, 2019, 7 (4):1-12

(*http://www.scientiaresearchlibrary.com/arhcive.php*)

HEURISTIC AND MULTI-AGENT SOLUTIONS OF A UNIVERSITY TIMETABLING PROBLEM FOR HIGHER EDUCATION INSTITUTIONS

FERNANDO S*. VIRAY JR.
[1]*Pangasinan State University*
Corresponding Author: [1]*skyfher@gmail.com*

ABSTRACT

Higher Education Institutions (HEIs) both locally and abroad are continuously faced with the difficult challenges of preparing university class schedules every semester, thus, making the University Timetabling Problem (UTP) considered as one of the most tedious recurring problems in higher education. The challenges of the UTP coupled with varying institutional policies, constraints, government regulations, and never-ending Higher Educations Orders (CMOs) and other variables needs to be addressed. This research focuses on the design and implementation of a heuristic multi-agent university timetabling solutions by employing local search and optimization algorithms, Tabu Search, Greedy Algorithm, Integer Linear Programming, and Bi-Partite Graph approach of Artificial Intelligence. Simulation results in one of the five campuses with random faculty member projected schedule using actual tertiary level scheduling data have shown that the system yielded feasible and optimum schedule solutions. The study was conducted among 2,000 students and 200 faculty members. The implementation of achievable artificial intelligent and multi-agent-based system in all the campuses has the possibility of gaining tremendous benefits such as less time and conflict-free schedule.

Keywords : local optimization solutions, greedy algorithm, university timetabling problem, multi-agent system, tabu search, and divide-and-conquer

---

INTRODUCTION

Higher education institutions (HEIs) both locally and abroad are continuously faced with the difficult challenges of preparing university class schedules every semester, thus, making the University Timetabling Problem (UTP) commonly known as the University Class Scheduling Problem (UCSP) as one of the most tedious recurring enrollment tasks implemented among schools.[1][2] The challenges of the UTP are coupled with the different institutional policies and parameters.[3] Nonetheless, the necessary variables of the problem are composed of 4; (1) a set of subjects based on a particular course or program; (2) a group of students who belong to a specific program and who are obliged to take subjects as outlined on their respective curriculum; (3) a set of teachers, each with distinct specialization, who will teach each subject belonging to a particular department of knowledge domain; (4) and a set of classrooms, either regular lecture room or specialized facility (such as laboratories, music centers, gymnasiums) where classes will be accommodated on a particular timeslot and day.

The UTP problem aims to find and assign school resources (classrooms and teachers) to a class without conflict to other classes – a class being an event composed of a subject and students to be held in a specific timeslot of the day.[3] Under this definition, elements of the class become the constraints or rules to be strictly observed, followed, and fulfilled to attain conflict-free class schedules. These constraints or rules are always referred to as hard constraints in the area of timetabling. On the other hand, other scheduling preferences that can be ignored to arrive in a conflict-free schedule are called soft constraints.[4] Given the number of teachers, rooms, subjects, days and timeslots, blocks of students, and teacher preferences, entails having several constraints and enormous combinatory possibilities. Thus, the search for a feasible solution is different from finding an optimum solution.[5] Nevertheless, both solutions require a higher level of human logic for the technical solution and practical implementation.[6] Under this context, Artificial Intelligence (AI) algorithms in the form of local search and optimization techniques are commonly utilized to solve the timetabling problem. It is also the reason why UTP is regarded as an NP-complete problem, a type of nondeterministic polynomial time (NP) hard problems characterized by having no known efficient and swift method of solving the problem.[7]

A strategy to solve the UTP is utilizing a “divide and conquer” tactic by dividing the UTP into five sub-problems: teacher assignment, course scheduling, class-teacher timetabling, student scheduling and room assignment sub-problems in which each sub-problem is individually resolved by assigning a specific local search and optimization algorithm and forwarding the result as a partial solution to the other sub-problems. Recent studies also implemented artificial intelligent agents as part of a multi-agent system (MAS) to resolve the UTP by assigning a worker AI agent to each of UTP’s sub-problem and a management agent that will control and manage all working agents.

This paper aims to evaluate the design of a multi-agent university timetabling model by implementing commonly-used local search and optimization algorithms to solve the university timetabling problem. The general constraints of each UTP sub-problem, as well as common tertiary level local state university class scheduling parameters, will be utilized as inputs for a computer simulation that will monitor the speed of finding a conflict-free solution and the total number of soft constraints satisfied for each iteration.

2. Related Works

In the context of higher education institutions, particularly local state universities, the UTP applies to the challenge of searching for a conflict-free class schedules by considering a set of courses/programs, each with a set of subjects, and assigning students, teachers, and rooms given a particular timeslot and day in a week.[8][9] The solution involves local search and optimization approaches as it requires to search and select the best solution among previously found solutions. Recent studies used machine learning (ML) to solve scheduling problems by utilizing artificial intelligent (AI) agents that learn to find an optimum solution among feasible solutions by mimicking humans in performing logic, complex calculations, and relationship cross-checking.[10]

UTP are often denoted as hard or soft constraints-based problems. Hard constraints are rules to be strictly considered and fulfilled; otherwise, no feasible solution can be obtained. On the contrary, soft constraints or preferences are optional factors that can be ignored, but a feasible solution can still be arrived. However, satisfying soft constraints increases the quality of the solution, thus making the feasible solution an optimum one. Due to the varying needs and parameters of each HEI, the difficulty of developing a generic timetabling software model remains high.[11]

Nevertheless, recent studies have shown that by incorporating AI agents to each sub-problem of the UTP and by implementing commonly-used local search and optimization algorithms, a promising model can be achieved.

2.1 Multi-agent Scheduling

An agent is a computational system that is placed in an active setting and has the capability of autonomously rendering intelligent behavior. Its setting may include other agents that form a community of interacting agents, and as a whole, function as a multi-agent system. The solution for a University Timetabling Problem can be obtained by utilizing a "divide and conquer" approach by dividing the problem into five sub-problems and assigning each with a worker AI Agent being controlled and triggered by a manager AI agent. The solution from each AI agent is combined to provide an overall solution.

a. Teacher AI Agent

The teacher AI agent handles assigning and scheduling teachers to specific courses with the following assigned tasks:

- Each course section has a minimum and maximum number of teachers to teach;
- Part-time teachers are required to teach courses that cannot be taught by full-time teachers due to overload or out of specialization scope;
- A maximum and minimum number of teachers can be assigned to a particular course depending on the number of course sections;
- Each teacher should be given ample time for class and lesson preparation so the maximum number of hours per teacher should not exceed or be equal to 8 hours; and
- Full-time teachers are prioritized for faculty loading.

Course AI Agent

The course AI is primarily assigned to resolve the course scheduling which involves the scheduling of subjects on a specific period, year level, and class sections for a particular course program approved by government agencies to be offered by the school. The main goal is to create a conflict-free initial schedule of subjects as specified by each course curriculum per year level regardless of enrolled students with the following constraints:

- Each student strictly follows a specific course curriculum;
- A course curriculum is consisted of a set of subjects to be taken for a particular semester or year;
- Each subject has its corresponding credit units equivalent to several hours of teaching in a week
- Each class day consists at most two half days (i.e., morning and afternoon) with a fixed lunchtime in-between.
- Students are grouped according to their courses and year level which entails the number of subjects the student has finished in following his curriculum;
- Large courses in which numerous students are projected to enroll have to be divided into course sections;

- Sections of a course need not be taught by the same teacher. Hence, sections of a course may have periods and subjects to be taken in common but being taught by different teachers.
- The number of teachers, as well as the set of course sections, is determined in advance based on the approved course offering and previously enrolled students.

c. Class-Teacher AI Agent

The class-teacher AI agent is tasked is to handle the creation of a schedule of teachers and class sections over a set of periods. The basic constraints that must be satisfied are:

- Professors can only be in one class at any given time,
- Students belonging to a section or class can only attend to a teacher's lecture at any given time. This sub-problem also focuses on assigning substitute teacher to a particular class in the absence of the regular assigned teacher and merging of classes having the same subject to take under the same curriculum.[12]

d. Student Scheduling AI Agent

The student scheduling AI agent manages the actual student scheduling which is primarily involved in creating a balanced, almost gap-free or consecutive subject schedules for a specific period within the week based on the course curriculum of the student. A balanced and ideal student schedule relates to an even number of subject hours per day for the student to be taken in a week while gap-free speaks about subjects being contiguously scheduled to maximize the usual 6 to 8 hours whole day class period.[13]

e. Room AI Agent

- The primordial objective of room AI agent is to supervise the room assignment which covers allocating classes to rooms with the following constraints:
- Lecture subjects should be taught inside regular lecture classrooms;
- Laboratory or elective subjects should be taught inside a specialized facility or laboratory room where tools, devices, and equipment aide in learning;
- A particular subject is taught in a classroom by a specific teacher during a specified timeslot; and;
- Each room, regardless of type, has a maximum seating capacity.

f. Central Communication AI Agent

The central communication AI agent is the leader of all worker AI agents. It manages, controls, and triggers each worker agent and server as a communication bridge among other agents. It guarantees that each worker AI agent is working according to their prescribed task at any given time.

2.2 Commonly Used AI-based Timetabling Algorithms

Many works and meta-heuristic algorithms have been proposed to solve the University Timetabling Problem, which is based on local search and optimization techniques. This section provides different analytical timetabling algorithms based on local search optimizations that will be incorporated as part of the design of the multi-agent university timetabling.

a. Heuristics

A heuristic function, or simply heuristic, is a method developed for solving a problem faster when available established approaches are too slow or require a lot of time and computing resources. It is also used in creating an approximate solution when there is no exact solution.[14] Human intuition and learning are still the best forms and sources of heuristics, candidate solutions and options for trade-offs; hence, the availability of a vast number of algorithms and solutions for a particular problem and its related-problems.[15][16] Meanwhile, a meta-heuristic is a high-level function developed to generate or find a heuristic approach that can deliver a satisfactory solution to an optimization problem.[17] It is usually applied in combinatorial optimization to find an optimal solution over a discrete search space.[18]

b. Tabu Search

In Tabu Search, local searches mean taking a candidate solution by checking its immediate solutions (called neighbors) from a solution set that is similar but has very minute details hoping to find an enhanced solution.[19] Although the Tabu Search algorithm is more time consuming as complexity increases with a directly proportional increase in constraints, it is an efficient algorithm in terms of using limited computing resources. It has effectively avoided loops in cycling back to previously visited solutions through memory structure called Tabu List.[20]

c. Greedy Algorithm

Greedy algorithm is a heuristic model that assumes local optima at each phase hoping to find the optimal solution. The utilization of the greedy algorithm to solve the timetabling problem is capable to solve a complicated multiple hard constraint conditions. These conditions are:

- uniform teaching resources, which include teachers, students, and classrooms, with practicable timeslots, and
- a balance for all the curricular loads of students and teachers' satisfaction of preferred lecture schedule.
- The greedy algorithm is indeed useful and the runtime shortens significantly as compared to other techniques used in the timetabling system.

d. Integer Linear Programming (ILP) Algorithm

Linear programming (LP), is a technique to attain the best result in a mathematical model that requires to be characterized through linear relationships. The main goal of this algorithm is to search for a point in the polyhedron's function, where the largest or smallest value exists. If the value being searched for and values to choose from the feasible region needs to be integers, then the problem is referred to as an integer programming (IP) or integer linear programming (ILP) problem. There are various fields of study where linear programming can be utilized, such as vehicle service scheduling, which then can be adapted to university class timetabling by modifying the routes to courses/subjects, vehicles to classrooms, vehicle drivers to teachers, and passengers to students.[21] IP techniques can be used to model university class scheduling which can integrate several academic, operational requirements and constraints and generate a feasible outcome for practical use.

e. Bi-Partite Graph Approach

In a bipartite graph, an element from a set is linked through a line to one or all of the elements of another set, thus, forming bipartite relations. The bipartite graph approach can be used in timetabling problems by converting the list of teachers, classrooms, subjects, students, courses into vertex sets where each element in the list will become an element of a vertex set. Relations will be linked from corresponding vertices, and then vertex value matching, collisions and non-collisions, and possible optimizations can be performed. Results of experimental studies demonstrated that utilizing a bipartite graph approach can solve university timetabling problems.[22]

MATERIALS AND METHOD

This research aimed to develop the design of multi-agent university timetabling to solve the University Timetabling Problem for a tertiary level local state university by considering the characteristics of agents, constraints, and parameters of the five sub-problems of UTP and the characteristics of commonly-used local search and optimization algorithms discussed above.

The researchers used and combined the goal-based and layered architecture models of AI agents to embody each agent's input, methods, goals, thresholds, decisions, and response. The researchers used seven AI agents that will interact and communicate with one another to come up with a conflict-free schedule. The AI agents are composed of two Central Communication AI agents (one each in the frontend and backend of the system), which will be manager agents and 5 worker agents as part of the backend, each corresponds to the sub-problem of UTP. Figure 1 exhibits the organization and communication diagram of the seven AI agents both in the frontend and backend of the multi-agent university timetabling system.

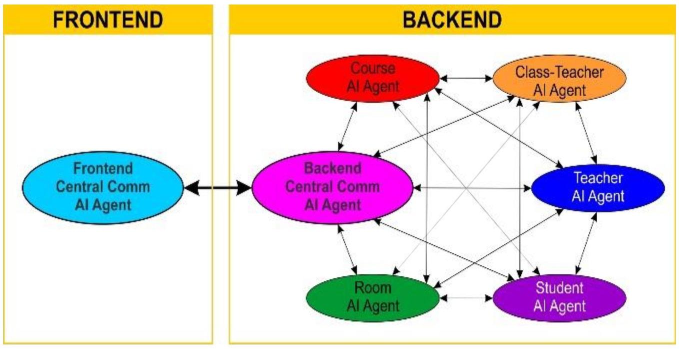

Figure 1: Organization and communication model of the proposed system design

The constraint evaluation targets to assess how the proposed design of the multi-agent university timetabling that will satisfy all hard constraints and some of the soft constraints by considering all required input data of the five sub-problems of the Timetabling problem. These input data were listed and assessed, which forms the hard and soft constraints. The criteria set for a feasible solution is that all hard constraints must not be violated even for a single instance, and class events of all schedules must be conflict-free among the rest of the class events. Conversely, the criteria set for an optimum solution is to have a higher average course curriculum weekly subject load distribution rate among blocks or sections, and a higher teacher preference satisfaction result. Meanwhile, the computational speed evaluation will calculate the total time elapsed until the hybrid algorithm finds a feasible solution.

Sixteen hard constraints were identified and organized about the context of the subject tertiary level local state university. These hard constraints were listed after studying the university's administrative and faculty manuals, government policies for state universities, previous and current enrollment reports of the university, the university's government-approved tertiary level programs, and the four UTP hard constraints.

Table 1 shows the 16 hard constraints identified, their sources, as well as the UTP sub-problem and AI agent in which it is corresponding.

Table 1: Hard Constraints

#	Constraint	Sub-Problem Involve
1	A class must be assigned a specific teacher at any given timeslot and day.	Teacher
2	A class must be assigned one subject only at any given timeslot and day.	Course
3	A class block must attend to only one subject at any given timeslot and day.	Student
4	A class must be assigned to one classroom only at any given timeslot and day.	Room
5	Full-time teachers should be prioritized over part-timers	Teacher
6	Subjects to be offered are based on approved programs only.	Course
7	The teaching load is a maximum of 21 units per semester.	Teacher
8	Teachers with administrative functions will be de-loaded to 12 units per semester.	Teacher
9	The minimum class population is 15.	Class-Teacher
10	The maximum class population is 50.	Class-Teacher
11	Low-class populations of the same subject regardless, of course, can be merged into one.	Class-Teacher
12	Either department/college can use classrooms from other colleges or departments for lecture purposes.	Room
13	A 3-unit lab subject is equivalent to 5 teaching hrs.	Course
14	A 3-unit lab subject is equivalent to 4.25 teaching load	Teacher
15	Basis of class blocks is the enrollment report.	Student
16	Programs to be offered are those approved and certified only by the government.	Course

Listing of hard constraints identified and their respective involved UTP sub-problem.

In terms of soft constraints, the researchers analyzed sample teacher preferences about class scheduling and narrowed down these preferences into six (6) natures of teacher preference. Each preference is given an equivalent score that will be summed up if ever the class events of the solution found will fit into either of these preferences.

Table 2 lists the 6 natures of teacher preferences, which will be used as soft constraints and their corresponding score.

An actual set of teacher preferences, a list of teachers, classrooms, subjects, block sections, and curricula from a chosen college of the subject state university were prepared and served as test data for the simulation.

Table 2: Natures of Teacher Preferences (Soft Constraints)

#	Nature	Short Description	Max Score
1	Subject-based	Preferring a subject to be taught based on specialization	1
2	Timeslot-based	Selecting a timeslot when to teach a subject to perform other officially appointed duties and functions	1
3	Room-based	Opting to teach in a nearby or lower-storey room	1
4	Subject-Timeslot-based	Preferring a to teach a subject based on specialization on a particular timeslot	2
5	Subject-Room-based	Preferring to teach a subject based on specialization on a particular classroom	2
6	Room-Timeslot-based	Preferring to teach a subject a particular timeslot and in a nearby or lower-storey room.	2
7	Subject-Timeslot-Room-based	Preferring to teach a subject based on specialization on a particular timeslot to a nearby or lower-storey classroom	3

Listing of the natures of teaching preferences as soft constraints.

The current enrollment report of the college was also included as part of the test data to simulate the projected number of blocks or class sections. The test data is equivalent to an enrollment scheduling data for a semester of a university college with 5 departments, each with four 4-year degree courses for the morning shift.

RESULTS AND DISCUSSION

Figure 2 exhibits the framework of the proposed design of the multi-agent university timetabling which highlights the organization and communication of the seven AI agents both in the frontend and backend and how the multi-agent scheduling is related to constraints, stakeholders of the university, and the goal of producing a conflict-free schedule.

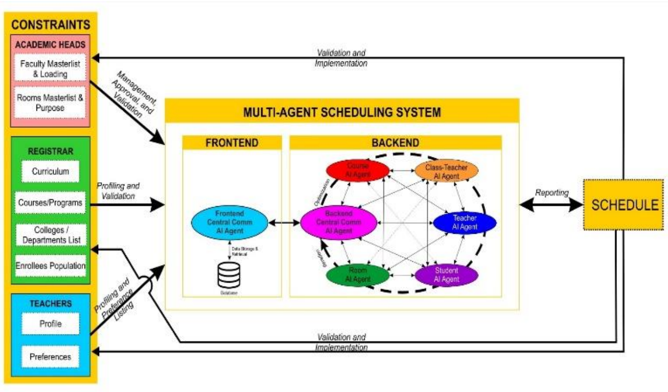

Figure 2: Framework of the proposed multi-agent university timetabling system

The sources of constraints for the UTP come from the different university stakeholders such as the academic heads, who provide a list of faculty, load distribution, room assignment, curriculum, enrollment reports, faculty specialization, and preferred schedule. These data will be encoded in t6he system via the frontend module, where the Frontend Communication Central AI agent resides to manage database access and retrieval and administering schedule requests and outputs from the backend. The backend module of the multi-agent university timetabling is composed of the Backend Central Communication AI agent who handles schedule requests from the frontend and who assigns scheduling tasks to the 5 worker AI agents, which correspond to the 5 sub-problems of UTP. Once a feasible solution is found, the backend AI agent will report it back to the frontend AI Agent, who will then save it to the database for retrieval and reporting purposes.

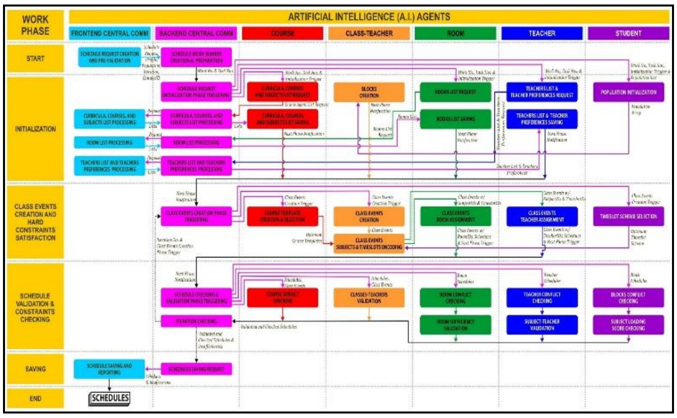

Figure 3: Process Flow of the Proposed Design of the Multi-Agent University Timetabling

Figure 3 is a multi-agent university timetabling process flow was established, which is composed of the different tasks involved in finding a feasible solution to the UTP after carefully evaluating the parameters of each sub-problem of the University Timetabling Problem. It exhibits the step-by-step solution in solving the UTP by dividing the problem into phases and tasks to solve a sub-problem of UTP by using the corresponding local search and optimization algorithm. The figure also shows the steps to be taken by the multi-agent system to iterate and look for an optimum solution after finding the initial feasible solution.

The initialization phase commences once the frontend central communication AI agent receives a schedule creation request from the system end-user and forwards it to the backend central communication AI agent after passing validation. In this phase, backend central communication AI

Agent triggers backend worker AI agents to commence their work after assigning each agent's required input variables. It is also during this phase that the student and class-teacher AI Agents create blocks by observing the minimum and a maximum number of student population per block.

After the initialization phase, the class events creation and hard constraints satisfaction phase will immediately follow. During this phase, the course AI Agent will create and select a course curriculum template for each program offering per year level. The course curriculum template is the distribution of subjects to be taken by each course per year level in a week by observing the number of hours per subject. This task solves the course sub-problem by utilizing heuristics instead of relying on all possible combinations of subjects, timeslots, and days. At this point, the subject loading score is also computed per template, which provides information on how to balance the subjects spread out each day in a week. Once all course templates are created, class events array will be created by the class-teacher AI Agent and will be encoded with the selected timeslot scheme chosen by student AI Agent. A timeslot scheme ensures that schedules do not cluster from one part of the day by using several starting hours of each class block., i.e., 8:00 AM scheme sets class-blocks scheduled to start every 8:00 AM while a 9:00 AM scheme starts schedule of a block at 9:00 AM and ends depending on the total number of subject and subjects hours the class block has. Meanwhile, the integer linear programming is used to assign respective subjects and timeslots for each class event performed by the class-teacher AI agent. After this, Room and Teacher AI Agent will assign the corresponding room and teacher for each class event by implementing a greedy algorithm.

Once all class events are assigned with subject, teacher, room and class blocks/sections, a schedule is created and will be subjected to several battery of tests and checks to validate and ensure that they are conflict-free from one another which occurs during schedule validation and constraints checking phase by each backend worker AI agent utilizing bi-partite graph approach. Each class event is compared to the rest of the class events array and ensure that all hard constraints are followed. If all hard constraints are satisfied with all of the elements of the class events array, it means that the result is a feasible solution.

If a solution is found, the system will compare this from previously found solutions and check whether it is an optimal solution by observing the average subject loading score from each class block and the total number of teacher preferences satisfied. This checking (performed by backend central communication AI Agent) implements tabu search, which saves to memory those right solutions and iterates again until the given iteration count is reached. The solution with the highest average subject loading score and the most teacher preferences satisfied will be considered as the optimum solution and will be forwarded to the frontend central communication AI agent for saving and reporting purposes

The average speed of the proposed design of the multi-agent university timetabling for creating a conflict-free class schedule involving 5 courses, 38 blocks, and 847 class events is 40.22 seconds with an average block schedule loading score of 89.55% and an average teacher preferences satisfaction of 318.20. The data is indicative that the more tries (which entails more time) that are given for the system to generate a conflict-free schedule, the higher the average block schedule loading score and teacher preferences satisfaction it can obtain.

CONCLUSION

The implementation of the proposed design of the multi-agent university timetabling in the multi-constraint inputs and environment of the university timetabling problem provides encouraging results given ample computational resources and time after combining goal-based and layered architecture models of AI agents and utilizing universal AI-based local search and optimization algorithms. With the organized scheduling phases and tasks involved under the proposed design of the multi-agent system, a feasible solution was obtained as it satisfies all the hard constraints. An optimum solution in preparing a conflict-free schedule involving 5 courses, 38 blocks, 1,624 students, and 847 class events was achieved in just 40.22 seconds with an average subject loading score of 89.55% that indicates that the class schedules created are balanced and evenly distributed in a week. The proposed design of the multi-agent university timetabling system also satisfied an average of 318.20 teacher scheduling preferences, which is indicative of a new quality to the feasible solution.

REFERENCE

[1] Domenech, B. & Lusa, A. (**2016**). A MILP model for the teacher assignment problem considering teachers' preferences. *European Journal of Operational Research* Volume 249, Issue 3, 16 March 2016, Pages 1153-1160. Elsevier.

[2] Poole, David; Mackworth, Alan (**2017**). Artificial Intelligence: Foundations of Computational Agents (2nd ed.). Cambridge University Press. ISBN 9781107195394.

[3] Pereira, Valdecy and Costa, Helder Gomes (2016). Linear Integer Model for the Course Timetabling Problem of a Faculty in Rio de Janeiro. Advances in Operations Research Volume **2016**. doi: http://dx.doi.org/10.1155/2016/7597062.

[4] Almeida, Maria Weslane S., Medeiros, João Paulo S. and Oliveira, Patrícia R. (**2015**). Solving the Academic Timetable Problem Thinking on Student Needs. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA). Florida: IEEE. ISBN: 978-1-5090-0287-0.

[5] Babaei, Hamed; Karimpour, Jaber; and Hadidi, Amin (**2015**). A Survey of Approaches for University Course Timetabling Problem.Computers and Industrial Engineering Volume 86. Elsevier.

[6] Christian, Brian and Griffiths, Tom (**2016**). Algorithms to Live By: The Computer Science of Human Decisions. Henry Holt and Company. ISBN: 1627790373.

[7] Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. (**2014**). Introduction to Algorithms (3rd ed.). MIT Press and McGraw-Hill. pp. 966–1021. ISBN 0-262-03293-7

[8] Zhang, D., Guo, S., Zhang, W. & Yan, S. (**2014**). A novel greedy heuristic algorithm for university course timetabling problem. Intelligent Control and Automation (WCICA), 2014 11th World Congress On Intelligent Control and Automation. IEEE. Shenyang, China

[9] Phillips, Antony, Waterer, Hamish, Ehrgott, Matthias, and Ryan, David (**2014**). Integer Programming Methods for Large Scale Practical Classroom Assignment Problems. Lancaster EPrints 2015. Accessed February 12, 2018.

[10] Russell, Stuart J.; Norvig, Peter (**2015**), Artificial Intelligence: A Modern Approach 3rd Edition (Paperback). New Jersey: Prentice Hall, ISBN: 9332543518, 978-9332543515.

[11] Fonseca, George H.G., Santos, Haroldo G., Carrano, Eduardo G., Stidsen, Thomas J.R. (**2017**). Integer programming techniques for educational timetabling. *European Journal of Operational Research* Volume 262, Issue 1, 1 October 2017, Pages 28-39. Elsevier.

[12] Bahel, M. & Thomas A. (**2017**). Innovative Evolutionary Algorithm Approach for Class-Teacher Timetabling Problem. Bhilai Institute of Technology, Durg District, India.

[13] Cheng, E., Kruk, S., & Lipman, M.J. (2013). Flow Formulations for the Student Scheduling Problem. Practice and Theory of Automated Timetabling IV, 4th International Conference, PATAT 2013, Gent, Belgium.

[14] Nilsson, Nils J. (**2014**). Principles of Artificial Intelligence. Morgan Kaufmann Publishing. ISBN: 1483295869, 9781483295862.

[15] Gendreau, M. & Potvin, J.Y. (**2018**). Handbook of Metaheuristics. Springer. ISBN: 3319910868, 9783319910864.

[16] Du, K. & Swamy, M. (**2016**). Search and Optimization by Metaheuristics: Techniques and Algorithms Inspired by Nature. Birkhäuser Publishing. ISBN: 3319411926, 9783319411927.

[17] Castillo, O. & Melin, P. (**2014**). Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics: Theory and Applications. Springer. ISBN: 331910960X, 9783319109602.

[18] Amodeo, L., Talbi, E.G., & Yalaoui, F. (**2017**). Recent Developments in Metaheuristics. Springer. ISBN: 3319582534, 9783319582535.

[19] Rovatsos, Michael, Vouros, George & Julian, Vicente (**2015**). Multi-Agent Systems and Agreement Technologies. 13th European Conference, EUMAS 2015, and Third International Conference, AT 2015, Athens, Greece. Springer. ISBN: 9783319335094.

[20] Bindra, Richa, Modi, Nishank, Shah, Rahil and Puri , Simran (**2014**). University Course Scheduling using Tabu Search Algorithm. Thesis: University of Waterloo. Retrieved 20 March 2018.

[21] Neapolitan, Richard; Jiang, Xia (**2018**). Artificial Intelligence: With an Introduction to Machine Learning. Chapman & Hall/CRC. ISBN 978-1-13850-238-3.

[22] Ganguli, Runa and Roy, Siddhartha (**2017**). A Study on Course Timetable Scheduling using Graph Coloring Approach. *International Journal of Computational and Applied Mathematics* Volume 12, Number 2 (2017), pp. 469-485. Research India Publications. ISSN 1819-4966.