Sequential Constructive Algorithm incorporate with Fuzzy Logic for Solving Real World Course Timetabling Problem

Chapter · January 2020

DOI: 10.1007/978-981-15-0058-9_25

Sequential Constructive Algorithm incorporate with Fuzzy Logic for Solving Real World Course Timetabling Problem

Tan Li June[1], Joe H. Obit[2], Yu-Beng Leau[3], Jetol Bolongkikit[4] and Rayner Alfred[5]

[1] Universiti Malaysia Sabah Labuan International Campus, Malaysia
[2] Knowledge Technology Research Unit, Universiti Malaysia Sabah, Malaysia, Jalan UMS, 88400 Kota Kinabalu, Sabah, Malaysia
lijunetan93@gmail.com, joehenryobit@gmail.com,lybeng@ums.edu.my
jetol@ums.edu.my, ralfred@ums.edu.my

Abstract. Sequential constructive algorithm is one of the popular methods for solving timetabling problems. The concept of the algorithm is to assign event based on their difficulty value by using different sequential heuristic. The most common sequential heuristics are largest enrolment, largest degree and saturation degree. Each sequential heuristic has its own criteria to obtain events' difficulty value. Instead of single sequential heuristic, this paper presents to use fuzzy logic to consider multiple sequential heuristics in order to obtain the difficulty value of the events. The proposed method will be used to generate feasible solution as well as improve the quality of the solution. Another objective of this paper is to tackle a real world course timetabling problem from Universiti Malaysia Sabah Labuan International Campus (UMSLIC). Currently, UMSLIC generates course timetable manually which is very time consuming and ineffective.The experimental results show that the proposed method is able to produce better quality of solution less than one minute. In terms of quality of timetable and efficiency, the proposed method is outperforming UMSLIC's manual method.

Keywords: Sequential Constructive Algorithm, Fuzzy Methodology, Course Timetabling Problem.

1 Introduction

A general definition of course timetabling problem is known as scheduling of a given set of courses to a limited number of timeslots and room under certain criteria and requirements [1]. Every institution has its own set of criteria and requirements and they are often known as a set of constraints in timetabling problem. Basically, constraints can be categorized into two groups: hard and soft constraints. A feasible solution must not involve in hard constraint violation. Although different institutions have different sets of constraints, however there are some hard constraints that commonly used in course timetabling problem. For instance, no student is assigned to attend

R. Alfred et al. (eds.), *Computational Science and Technology*,
Lecture Notes in Electrical Engineering 603,
https://doi.org/10.1007/978-981-15-0058-9_25

more than one lecture concurrently. On the other hand, soft constraints are not necessary to be solved but it is highly desirable when soft constraints violation can be reduced in order to improve the quality of the solution. For instance, a student should not be assigned to only attend a lecture a day. Normally, the feasibility and quality of a timetable is measured by using cost function which indicates the degree of hard and soft constraints violation.

An enormous number of studied have been done with different approaches to solve course timetabling problems since the early of 1960's. For example, methods that are used to generate feasible solution such as sequential constructive algorithm [1], [2] and constraint programming [3], [4]. While metaheuristic algorithms are popular in improving solution such as Great Deluge [5], [6], Simulated Annealing [7], [8], Tabu Search [9], [10] and so forth.

This paper aims to solve course timetabling problems of Universiti Malaysia Sabah Labuan International Campus (UMSLIC). In UMSLIC, the course timetable is generated by timetabling officer manually. The timetabling process is time consuming as it needs to go through several times of amendments in order to produce a feasible timetable. Therefore, this paper proposes to develop an algorithm which integrates sequential constructive algorithm with fuzzy logic [11] to solve UMSLIC course timetabling problems. In this paper, there are two different phases of experiments: construction and improvement phase. The development of algorithm will be further discussed in Section 4.

2 Related Work

2.1 Sequential Constructive Algorithm

Sequential constructive algorithm was first introduced by [12] to solve examination timetabling problems. The idea of the algorithm is to assign those "difficult" events into timetable first. It is very difficult for those difficult events to fit themselves in the timetable when most of the timeslots and rooms are occupied by other events. There are some common sequential heuristics used to generate feasible solution such as Largest Degree (LD), Largest Enrolment (LE), Least Saturation Degree (SD), Largest Colored Degree (LCD) and Weighted Largest Degree (WLD).

However, in [13], the algorithm was unable to produce feasible solution for large instance from benchmark dataset by [14]. Therefore, many researches introduced modified version of sequential constructive algorithms in order to improve the performance of the algorithms.

In [15], a hyper-heuristic framework which employed Tabu search to search for permutations of graph heuristic to solve both examination and course timetabling problems. The framework utilized the Tabu search to store all the possible permutations and select the most suitable heuristic to construct timetable. Any move that is not able to assign course into feasible slot and room will be stored in a Tabu list.

While paper [16] proposed a framework to hybridize sequential heuristics with local search and Tabu search. The framework is composed by three processes which each of them has different objective to achieve. The experimental results showed that the proposed method able to generate feasible solutions for benchmark datasets [14] and ITC 2002 datasets [17]. However, none of them outperformed in terms of efficiency. For instance, the heuristic produced the best quality of timetable would taking longer time than the others.

2.2 Fuzzy Methodology applied in different area of research

In real world situation, most of the knowledge is ambiguous in nature and human reasoning is usually based on fuzzy information. Besides, decision making often involves considering multiple factors simultaneously. The concept of fuzzy logic was introduced is to handle such imprecise information and deal with multiple aspects at the same time. In present, fuzzy logic had been widely applied in variety of real world applications [18], [19] such as washing machines, air conditioners, and even timetabling problem particularly. For instance, fuzzy logic was applied to consider more than one heuristic ordering simultaneously to determine the difficulty value of a course.

In [11], LD, LE and SD were used to incorporate with fuzzy methodology to generate three different fuzzy combinations: Fuzzy LDLE, Fuzzy SDLE and Fuzzy SDLD. The proposed method was tested with examination timetabling benchmark datasets by [20]. The results showed that the proposed approach was able to produce comparable results in literature and overall, Fuzzy SDLE produced the best result among the fuzzy heuristic orderings.

The research on fuzzy heuristic orderings was extended to compare with non-fuzzy heuristic ordering in terms of algorithm performance in [21]. Basically, the non-fuzzy heuristic ordering refers to the single heuristic ordering which are LD, LE and SD. With the same examination benchmark datasets by [20], results showed that scheduling a list of courses based on fuzzy heuristic orderings is way more effective than only based on single heuristic ordering.

3 Fuzzy System and Fuzzy Sets Theory

3.1 Introduction

In 1965, Zadeh introduced fuzzy logic to deal with imprecise information with the idea of fuzzy set [22]. The basic idea of fuzzy logic is to use a fuzzy set to represent a class of events with degrees of membership. The general framework is to translate a fuzzy set into linguistic variable which will be assigned value in the range of zero and one by using membership function $f_A(X)$. A fuzzy set can also be known as an extended version of classical set theory. A membership function quantifies the linguistic variable of the fuzzy set based on its membership degree. This paper illustrates the form of fuzzy set by using room temperature as an example. For instance, the statement of “it is cold if the temperature is below 10°C” can be controversial due to its ambiguous status. In classical set theory, the membership function of the set of temperature is presented as below:

$$f_A(x) = \begin{cases} 1, & if\ x \leq 10°c \\ 0, & if\ x > 10°c \end{cases} \quad (1)$$

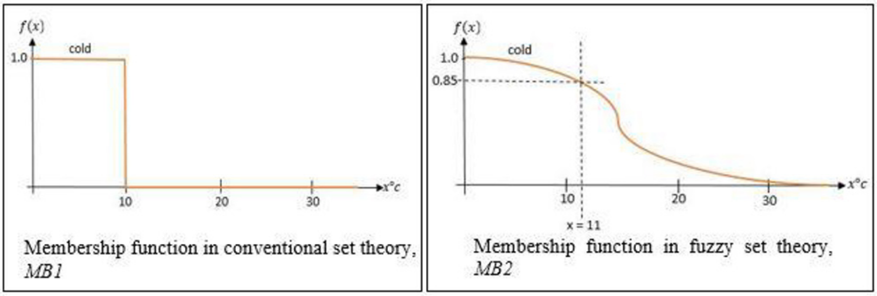

Fig. 1. Different membership function

Fig 1 shows the different membership functions are plotted based on conventional and fuzzy set theory. In *MB1*, all temperature above 10°C do not considered as cold such as 11°C because the degree of membership of 11°C is zero which is not really accurate.

In fuzzy set theory, "temperature" is used as a linguistic variable and assigned with linguistic value which is "cold". Unlike conventional set theory, which is only either "cold" or "not cold", fuzzy control system assigns value to the linguistic variable in the range of zero and one to the room temperature using membership function $f_A(x): X \rightarrow [0,1]$ before evaluating the temperature. In *MB2*, the degree of membership of 11°C is 0.85 which is still considered as cold but with slightly less than 10°C. The result indicates that 11°C is still considered as cold but not as cold as 10°C.

The framework of a fuzzy system consists of four processes: fuzzification, a set of "IF…THEN…" rules, inference engine and defuzzification. Fuzzy set theory employs "IF…THEN…" rule to link input variables to output variables. In a rule, "IF" plays a role as "antecedent" while "THEN" to produce the output variables as "consequent" [11]. A set of rules is generated based on the number of inputs and outputs, and the pre-defined feature of the system.

4 Methodology

4.1 Generate Feasible Solution

This paper considers two sequential heuristics simultaneously with the implementation of fuzzy system to determine the difficulty values of a given set of courses. The three main sequential heuristics that used in this paper are:

1. **Largest Degree (LD)**: Courses are listed in descending order based on the number of courses in conflict. For instance, a course with higher number of courses in conflict will be scheduled into timetable first.

2. **Largest Enrolment (LE)**: Courses are listed in descending order based on the total number of students who take the particular course. For instance, a course with greater number of students will be scheduled into timetable first.
3. **Saturation Degree (SD)**: Courses are listed in ascending order based on the number of available slot for the particular course. For instance, a course with fewer number of available slots will be scheduled into timetable first.

Hence, there will be three different combinations of multiple sequential heuristics which are: fuzzy largest degree and largest enrolment (LDLE), fuzzy saturation degree and largest enrolment (SDLE) and fuzzy saturation degree and largest degree (SDLD). Table 1 summarizes the hard and soft constraints for UMSLIC course timetabling problem:

Table 1. Hard constraints and soft constraints of UMSLIC course timetabling problem.

Hard constraints	Soft constraints
1. No students and lecturers are assigned to attend more than one course simultaneously.	1. Students and lecturers should not attend more two courses consecutively.
2. A course must be assigned into a room that is able to fit all the students of that certain course.	2. Students and lecturers should not attend course at the last time slot of the day.
3. Cannot assign more than one into a room in every timeslot.	3. Students should not only attend a single course in a day.
4. There are certain courses must be assigned into specific timeslot.	4. Extra room space should be reduced.

This section will take largest degree (LD) and largest enrolment (LE) as an example to discuss the process of generating fuzzy multiple sequential heuristics. In this scenario, largest degree (LD) and largest enrolment (LE) will act as linguistic variables with three different linguistic values: small, medium and large. A set of "IF…THEN…" rules will be applied to link the linguistic variable and generate a single output variable which is named as courseweight, refers to the difficulty value of a course. There are total nine rules for fuzzy LDLE used for fuzzy reasoning and only three of them will be used to illustrate the process as below:
Rule 1: IF (LD is small) and (LE is medium) THEN (courseweight is small).
Rule 2: IF (LD is medium) and (LE is medium) THEN (courseweight is medium).
Rule 3: IF (LD is medium) and (LE is large) THEN (courseweight is large).

To determine the linguistic values for LD and LE, first thing is to normalize the input values of LD and LE by using a formula as follow:

$$val' = \frac{val - minValueOfTheList}{maxValueOfTheList - minValueOfTheList} \quad (2)$$

From equation (2), val' is the normalized value while val is each of the values from the list in the range of [minValueOfTheList, maxValueOfTheList]. For example, a set of courses with the LD [4, 90] and LE [1, 176], and assume LD and LE for a course is 33 and 101 respectively. In this scenario, 33 and 101 will be treated as input values and normalized by using equation (2). Therefore, the normalized value of LD and LE are 0.34 and 0.57 respectively.

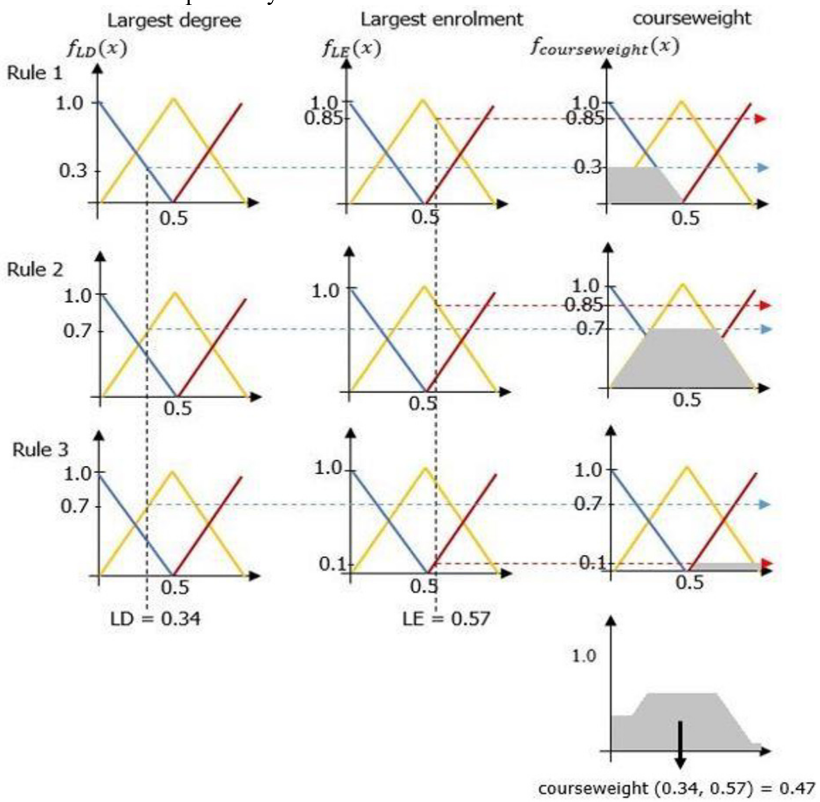

Fig. 2. Illustrate using the Mamdani Inference process to generate courseweight

Fig. 2 shows how to generate an output value, courseweight by using fuzzy rules based system with the three rules stated above.

The output linguistic value from the rules will be transformed into crisp value by using Centre of Gravity (COG) which finds the centroid of a planar figure. From Fig. 2, a value 0.47 is obtained from fuzzy system which indicate the course has 0.47 as difficulty values based on LD and LE criteria.

The scheduling process starts after the list of course is reordered by using fuzzy multiple sequential heuristic and it is named as $unscheduledCourses, UC$. A course with highest $courseweight$ will be selected first and assign into a random feasible timeslot and room. If a course fails to fit into any timeslot and room, it will be moved from the UC to the $failScheduledCourses, FC$. Meanwhile, it is important to note that LD and LE are known as static sequential heuristics which remain the difficulty value of a course unchanged until the end of the process. While SD is a dynamic heuristic ordering which will always calculate and update the difficulty value for the unscheduled courses throughout the whole scheduling process. Therefore, there will be an extra calculation on slot available for fuzzy SDLE and fuzzy SDLD. This step terminates if there is no course left in unscheduled courses. If there is at least one course in the $FC,$ it will trigger $reschedulingProcess$ to disturb the solution. The whole scheduling process will be terminated when all courses are managed to assign into timetable. The number of hard and soft constraints violation will be calculated as penalty cost to evaluate the feasibility and quality of timetable. The construction phase is presented in pseudocode as shown in Fig. 3:

```
Select a fuzzy multiple sequential heuristic and generate a list of unscheduled courses, UC;
while (UC is not empty)
  select a course c with highest courseweight ;
  while (c is not removed from UC)
    select an empty timeslot t and room r at random ;
    if (t and r are feasible for c) then
        assign c into t and r ;
        move c from UC to scheduledCourses, SC ;
    end
    if (no feasible slot and room available for c) then
        move c from UC to failScheduledCourses, FC ;
        break;
    end
  end
    if (fuzzy multiple sequential heuristics consists of SD) then
        recalculate the timeslot and room availability for every course
        reorder the list of course based on the fuzzy multiple sequential heuristic criteria
    end
end
while (FC is not empty)
  reschedulingProcess = false ;
  while (reschedulingProcess is not true)
     select an assigned course c from timeslot t and room r ;
     select an empty timeslot s* and room r* at random ;
     if (t* and r* are feasible for the c) then
        swap c from t and r to timeslot t* and room r* respectively ;
        reschedulingProcess = true ;
     end
  end
  select an unassigned course c, an empty timeslot t and room r at random ;
  if (t and r are feasible for the c) then
    assign c into t and r ;
    move c from FC to SC ;
  end
end
```

Fig. 3. Pseudocode of construction phase

4.2 Improve Solution

The process starts with reordering the set of courses into a list, $RescheduledCourses, RC$ by using the same fuzzy multiple sequential heuristics applied in construction phase. This process is set to run for 10000 iterations as the algorithms do not produce any significant difference after 9000 iterations. At each iteration, a course with highest $courseweight$ will be rescheduled to other timeslot and room which lower the soft constraints violation for the course. After rescheduling, RC will remove the course and the whole timetable will be re-evaluated and employs hill climbing with monte carlo to accept or reject the new solution. In order for algorithm to accept the new solution, one of these criteria must be achieved: 1. If penalty cost of the new solution is lower than current solution or 2. Accept worse solution within certain probability, it is known as Exponential Monte Carlo. If fuzzy SDLE or fuzzy SDLD is implemented in this experiment, the timetable need to be evaluated every iteration to update RC as they are composed of dynamic sequential heuristic. Improvement phase is presented in pseudocode as shown in Fig. 4.

```
Reorder the list of courses as RescheduleCourses, RC by using the fuzzy multiple sequential
heuristic applied in construction phase
S = current solution ;
f(S) = penalty cost of current solution ;
while (iteration < 10000)
  if (RC is not empty) then
    select a course c with highest courseweight from timeslot t and room r ;
    move c to a feasible timeslot t* and room r* which cause the lowest soft constraints
    violation for c and form a new solution S* ;
    β = f(S*) – f(S) ;
    if (f(S*) < f(S)) then
      S ← S*
    end
    else if (f(S*) > f(S)) or (exponential(-β) > rand[0,1]) then
      S ← S*
    end
    else if (f(S*) is zero) then
      break ;
    end
    remove c from RC ;
  end
  else
    recreate RC ;
  end
end
```

Fig. 4. Pseudocode of improvement phase

5 Experimental Results

Java based applications and the fuzzy logic reasoning system generated by [23] are implemented in the research. The dataset semester 1 session 2016/2017 of UMSLIC is used to conduct the experiments. The experiments run 50 times to obtain the average penalty costs. Table 2 presents penalty costs of real world timetable and experimental results.

Table 2. Results comparison between real world application and proposed algorithms

Approach	UMSLIC	Fuzzy LDLE	Fuzzy SDLE	Fuzzy SDLD
Hard constraint	86	0	0	0
Soft constraint	2313	4719	4637.17	5026
Soft constraint (improved)	-	368.17	276.67	274.83

Improvement rate	-	92.14	94.02	94.58

In order to generate a feasible solution, the penalty cost of hard constraint must be zero. In construction phase, the proposed algorithms only focus on generating a feasible solution without considering soft constraints. This explains why the proposed algorithms have higher penalty cost for soft constraints violation. Although UMSLIC timetable has the lowest penalty cost for soft constraints violation among the approaches, but it has penalty cost of 86 for hard constraints violation. This indicates that no matter how low the soft constraints violation is, the timetable is still not feasible and practical. Therefore, students who retake subject and clash with their main subjects, they have to deal with the lecturer personally. Besides, the duration for UMSLIC to produce a timetable can be up to three months. While the proposed algorithm can generate a feasible timetable less than one minute. This indicates that the proposed algorithm performs much more effective than UMSLIC's manual method. Among the fuzzy multiple sequential heuristics, Fuzzy SDLE outperformed in construction phase as it has the lowest average penalty cost for soft constraints violation. While Fuzzy SDLD performed better in improving solutions. It has the lowest average penalty cost for soft constraints violation 274.83 and it is able to improve the quality of feasible solutions up to 94.58%. The overall results show that the proposed algorithm is way more effective than UMSLIC method for solving the course timetabling problem.

6 Conclusion

In conclusion, this paper has proposed the implementation of fuzzy logic to consider more than one sequential heuristic simultaneously. The paper compares the performance of different fuzzy multiple sequential heuristics with UMSLIC real world timetable. The experimental results proved that the proposed methods performed well in both construction and improvement phase for solving real world course timetabling problem. This motivates the research to work further by implementing multi-agent system. By doing so, the three fuzzy multiple sequential heuristics will communicate with each other in order to generate and improve the solution.

References

1. Syariza A. R., Andrej B., Burke E. K., Ozcan E.,: Construction of Examination Timetables Based on Ordering Heuristics. 2009 24th International Symposium on Computer and Information Sciences, pp. 680-685 (2009).
2. S. R. Runa Ganguli.,: A Study on Course Timetable Scheduling using Graph Coloring Approach. International Journal of Computational and Applied Mathematics, vol. 12, pp. 469-485 (2017).
3. Y. J. Kuan, J. H. Obit , A. Rayner,: A Constraint Programming Approach to Solve University Course Timetabling Problem (UCTP). Ameircan Scientific Publishers, pp. 400-407 (2016).

4. June T. L., J. H. Obit, Yu-Beng L., Jetol B.,: Implementation of Constraint Programming and Simulated Annealing for Examination Timetabling Problem, Computational Science and Technology 2018 (ICCST 2018), pp. 175-184 (2018).
5. D. Landa-Silva, J. H. Obit,: Great Deluge with Non-linear Decay Rate for Solving Course Timetabling Problems, IEEE, pp. 11-18 (2008).
6. D. Landa-Silva, J. H. Obit,: Evolutionary Non-linear Great Deluge for University Course Timetabling, Springer-Verlag Berlin Heidelberg, pp. 269-276 (2009).
7. Y. J. Kuan, H. O. Joe and A. Rayner,: Comparison of Simulated Annealing and Great Deluge Algorithms for University Course Timetabling Problems (UCTP), American Scientific Publishers, vol. 4, pp. 400-407 (2015).
8. Abramson D., Krishnamoorthy M., Dang H.,: Simulated Annealing Cooling Schedules for the School Timetabling Problem, Asia Pacific Journal of Operational Research, pp. 1-22 (1999).
9. A. S. Luca Di Gaspero,: Tabu Search Techniques for Examination Timetabling, International Conference on the Practice and Theory of Automated Timetabling, pp. 104-117 (2000).
10. Adeyanju I., Arulogun T., Omidiora E., Omotosho O. i.,: University Examination Timetabling Using Tabu Search, International Journal of Scientific and Engineering Research, vol. 5, no. 10 (2014).
11. Asmuni H., Burke E. K., Garibaldi J. M.,: Fuzzy Multiple Heuritic Ordering for Course Timetabling".
12. S. Broder,: Final Examination Scheduling, Communication of the ACM, pp. 494-498 (1964).
13. Salwani A., Burke E. K., McCollum B.,: An investigation of variable neighborhood search for university course timetabling (2005).
14. Socha K., Knowles J., Sampels M.,: A max-min ant system for the university course timetabling problem., Proceedings of the 3rd International Workshop on Ant Algorithms, ANTS 2002, Springer Lecture Notes in Computer Science, vol. 2463, no. 10, pp. 1-13 (2002).
15. Burke E. K., McCollum B., Meisels A., Petrovic S., Qu R.,:A graph-based hyper-heuristic for educational timetabling problems," European Journal of Operational Research , pp. 177-192 (2007).
16. D. Landa-Silva, J. H. Obit,: Comparing Hybrid Constructive Heuristics for University Course Timetabling, pp. 222-225 (2011).
17. International timetabling competition 2002. Metaheuristic Network, http://www.idsia.ch/Files/ttcomp2002/.
18. S. C. Pappis C.P., "Fuzzy Reasoning," Burke and Kendall, pp. 437-474 (2005).
19. P. Gupta,: Applications of Fuzzy Logic in Daily life, International Journal of Advanced Research in Computer Science, pp. 1795-1800 (2017).
20. Carter M. W., Laporte G., Lee S. Y.,:Examination Timetabling: Algorithmic Strategies and Applications, Journal of the Operational Research Society, pp. 373-383 (1996).
21. Asmuni H., Burke E. K., Garibaldi J. M.,: A Comparison of Fuzzy and Non-Fuzzy Ordering Heuristics for Examination Timetabling, Proceeding of 5th International Conference on Recent Advances in Soft Computing, pp. 288-293 (2004).
22. L. Zadeh,: Fuzzy Sets, Information and Control, pp. 338-353 (1965).
23. J. A.-F. Pablo Cingolani,: jFuzzyLogic: A Robust and Flexible Fuzzy-Logic Inference System Language Implementation, 2012 IEEE International Conference on Fuzzy Systems, pp. 1-8 (2012).

24. Asmuni H., Burke E. K., Garibaldi J. M., McCollum B.,: An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables, Computers & Operations Research, pp. 981-1001 (2009).
25. Y. J. Kuan, H. O. Joe and A. Rayner,: A Constraint Programming Approach to Solve University Course Timetabling Problem (UCTP), American Scientific Publishers, vol. 4, pp. 400-407 (2016).