Asian Journal of Research in Computer Science

1(4): 1-9, 2018; Article no.AJRCOS.43443

Implementation of Stable Marriage Algorithm in Student Project Allocation

Salisu Modi[1], Nura M. Shagari[1*] and Buhari Wadata[1]

[1]Department of Computer Science, Sokoto State University, Sokoto, Nigeria.

Authors' contributions

This work was carried out in collaboration between all authors. Author SM designed the study, performed the statistical analysis, wrote the protocol, and wrote the first draft of the manuscript. Authors NMS and BW managed the analyses of the study and managed the literature searches. All authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AJRCOS/2018/v1i424749
Editor(s):
(1) Dr. Stephen Mugisha Akandwanaho, Department of Information Systems and Technology, University of KwaZulu-Natal, South Africa.
Reviewers:
(1) Suneeta Agarwal, Motilal Nehru National Institute of Technology Allahabad, India.
(2) Manish Mahajan, IKG- Punjab Technical University, India.
(3) Utku Kose, Suleyman Demirel University, Turkey.
Complete Peer review History: http://prh.sdiarticle3.com/review-history/26233

Short Research Article

Received 03 June 2018
Accepted 29 August 2018
Published 15 September 2018

ABSTRACT

Project allocation is an annual challenge for lecturers and students. The process of allocating project involves matching preferences of students over project and with of staff over the student, and is thus an instance of stable marriage problem from theoretical computer science aspect. The aim is to find a stable allocation of project to students, such that it is impossible to find a project swap that would make all involved parties (both students, both staff) happier. This paper investigated efficacy of stable marriage algorithm and deployed basic Gale Sharply Algorithm into the process of allocating student project. A system was developed using ruby and MySQL to handle the task. The result showed that the algorithm was able to improve the process by enhancing the stability involved.

Keywords: Stable marriage; preferences; allocation; algorithm and project.

**Corresponding author: E-mail: nuramodi@yahoo.com;*

1. INTRODUCTION

The allocation of final year student a project is continuous process that attracts a lot of attention at the end of every academic session. The task involves assigning each student a project topic for their research work as part of the requirement for graduation. The projects are proposed either by the student or the tutor, after which both parties negotiate on the scope of the project. This paper is about deploying basic Gale Sharply stable marriage algorithm in the process of allocating student project. Where each supervisor and a student, will develop preference list from which project are allocated automatically when the algorithm is run.

1.1 Statement of the Research Problem

At present, most institutions do not have complete resources for managing process of final year student project allocation. The current manual system of allocating project to student by the project coordinators tend to be inefficient as a student can be allocated to a supervisor that they do not prefer. Equally, a supervisor might not be able to select the student that they can work with effectively. Therefore, students or supervisors proposed a project whilst project coordinator handles the allocation process. It is most likely that a student might be allocated to a topic or a supervisor in an area that he/she is not interested in. Similarly, supervisor's proposed topic might be allocated to a student who is not capable of undertaking it, thus posing a great challenge in the process.

1.2 Brief Overview of Basic Gale Shapley Stable Marriage Algorithm

Matching between two set of elements is a natural phenomenon that is of significant interest to researchers. The most aspect of human nature involves pairing between two set such as man to women, doctors to a hospital, student to a project and so on. And thus, this matching needs to be smooth and stable. The concept of stable marriage was initially studied in 1962 by Gale and Shapley (Gale and Sharply) [1]. The aim was to solve the problem of matching between equal number of men and women [2]. The stable marriage problem deals with finding a stable pairing between two equally sized sets of groups, from preference order for each element in the group [3]. The Gale-Shapley algorithm requires each element from one set in the matching to provide a complete set of preference ordered list of other opposite set in the matching. In Gale-Shapley algorithm, no incomplete preference is accepted, which means both the two sets most, be of the same size and are ranked to each other [4]. Generally, it can be argued that stability is the key aspect that determines the success of each matching, and according to Gale and Shapley (Gale and Shapley), there always exist at least one stable matching in an instance of the stable marriage algorithm [5].

Sanfoundry [3] argued that the Gale Shapley algorithm could be implemented programmatic-cally as shown in the Fig. 1 below:

```
function stableMatching {
    Initialize all m ∈ M and w ∈ W to free
    while ∃ free man m who still has a woman w to propose to {
       w = m's highest ranked such woman to whom he has not yet proposed
       if w is free
         (m, w) become engaged
       else some pair (m', w) already exists
         if w prefers m to m'
           (m, w) become engaged
           m' becomes free
         else
           (m', w) remain engaged
    }
}
```

Fig. 1. A pseudocode of gale shapley algorithms [3]

1.3 Stable Marriage Problem and Student Project Allocation

It is obvious that, the criteria for allocating projects to students are much similar to the stable marriage pairing. Matching different entities from two set of elements to each other usually invokes the need for stability since individuals' show preferences over one another. Allocating fixed number of student to a fixed number of the project has much in common to the coupling of n men and n women, in terms of the problems that may evolve. To this vein, it is apparent that deploying stable marriage problem and some of its solutions will have a great impact on the process of allocating a student a project. During the process of allocating project the main aim for both student and staff is to have a happier working partner, it is argued that the basic Gale and Shapley algorithm terminates with stable set of engaged couple in which each pair is happy with one another and no any possibility for any swap that could result to happier couple than initially formed [6].

The convention in the process of allocating student project was, student always makes a request to the supervisor's project and a supervisor response to the request with an offer. This is exactly in line with the idea of basic Gale-Shapley algorithm, which involves sequential proposal from men part to the women (Gale and Shapley, 1962). However, some extension of Gale-Shapley algorithm has the view that a woman can make a request to a man and can accept two or more men with the same rank [7]. Never the less, it can be said that stable marriage problem and its solutions, is still feasible to project allocation problem.

Moreover, allowing individual's (both students and staff) to create preferences, in the process of the allocating project, is vital for the performance of the student during the research. Stable marriage problem is strictly based on ordered list of preferences for the two parties involved. It is argued that matching is always stable between two set if it's resulted from their preferences to each other [8].

Therefore, it is said that stable marriage problem have much in common to the process of allocating project and the algorithm will provide the best solution to this process. Finally, it is evident that Basic Gale-Shapley algorithm is applicable to the process of allocating project.

2. METHODOLOGY

The method adopted in this paper was to design an allocation algorithm based on the criteria and the requirement of Gale Sharply algorithm. We start with creating the student and the supervisor's preferences followed by algorithm design. The final system was developed using the Ruby Programming Language and MySQL as the database and local server for implementation.

3. SYSTEM DESIGN

3.1 Student Preferences Design

This involves allowing the student to enter their preferences to the projects of their interest. To achieve this goal, it is also necessary to consider the requirement of the stable marriage algorithm that was deployed in the design of this system. The algorithm requires that each student in the system should rank each project available in a strictly ordered way [9]. This implies a student preferences list is required to include all available project ranked in a decreasing order of importance. So that first project in their list is more preferable compared to subsequent one in that order [10].

3.2 Supervisor Preferences Design

From the reviewed system, it is understood that, during the period of project allocation, the project coordinator or admin allocate a number of the project to be proposed and supervised by each staff. Supervisors also show interest and need to create a preference of the student requesting to take their project. This resulted in staff making preference list of students willing to accept their project.

The design of supervisor's preferences list also fulfils the requirement of stable marriage algorithm, as the second entity in the matching. The algorithm requires that all the other entities (students) must be ranked by each supervisor (project).

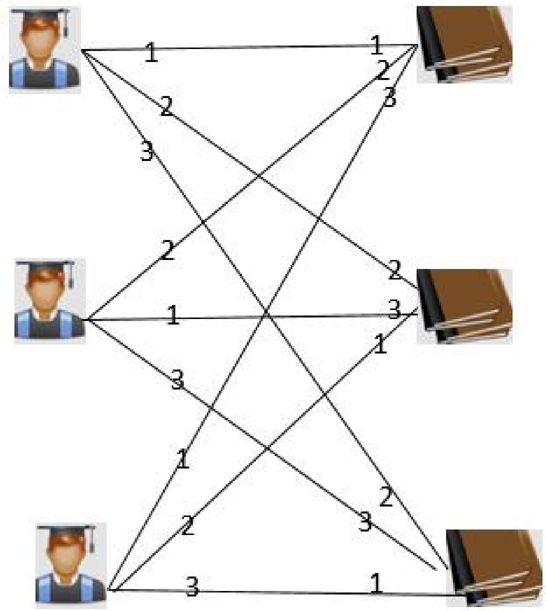

Fig. 2. Student and supervisors (project) preferences

3.3 Student-project Allocation Algorithm

The student project allocation algorithm was designed based on the basic Gale-Shapley stable marriage algorithm and some other related stable marriage problems derived from the review of other extensions of the Gale-Shapley algorithm. Some of the extension from the basic algorithm require that no complete ranking of both partners is needed [11,12].

The final system was developed with little extension of the basic Gale algorithm. The system is implemented such that it automates the ranking process when the number of involved parties grow larger. When student rank some number out of a large number of project or supervisor rank some students out of the list, the system upon the execution of the algorithm automates the ranking and adds to each user preference.

3.4 Pseudocode for Student-project Allocation Algorithm

Begin:
Initialization:
    Each student=nil project
    Each project= nil student
    While some student S is unmatched from student list
     (Students making request to   projects)
    P= 1^{st} project in S preference list not requested
    S = P for each s and p (s and p could be set of students and projects respectively)
    Add pair S-P to the matching set
            (Project making decision of acceptance)
If P is matched to two or more S || P prefer S' than S from s
            Remove pair S-P from matching set and add to the Unmatched set
            Match S'-P add the pair to the matching set
            Add S to the unmatched set
      End if
            Iterate the loop again
    End while
Return the matched set
Stop.

3.5 Flowchart for Student-project Matching Algorithm

A flowchart represents pictorially the step-by-step follow in the execution of an algorithm. Fig. 3 shows the steps of execution of a students project algorithm, the flowchart starts by initializing both student and project to be unpaired. The next step involves pairing. For a pairing to be successful and added to the matching set, it must satisfy the condition which checks that no single project allocated to two students. If the condition failed, the pair is added

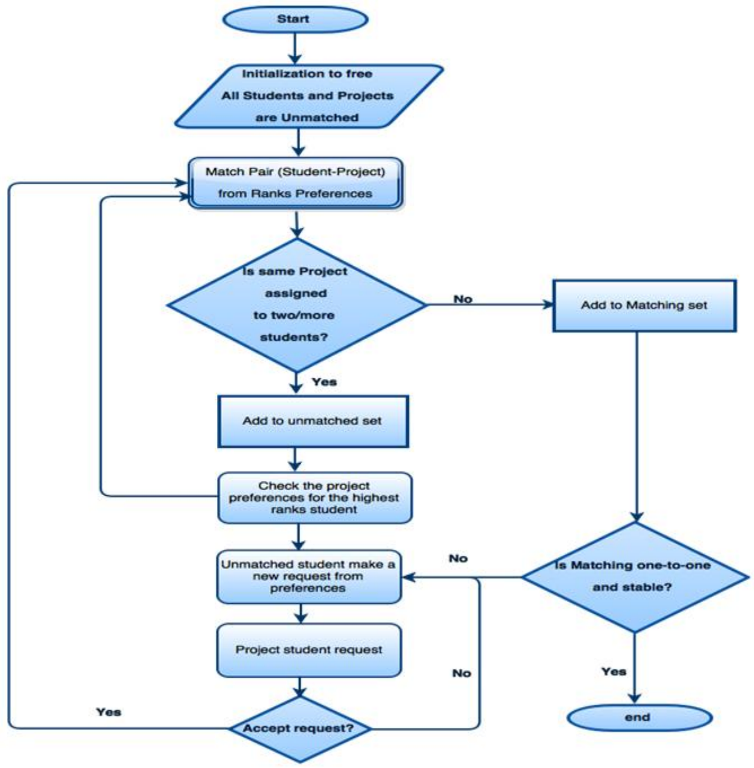

Fig. 3. Flowchart for student-project allocation algorithm

to the unmatched set. Then unmatched student makes a new request from the unassigned project. If the request is accepted the set are paired and added to the matching set. The cycle continues until all pairing is stable and one-to-one before the algorithm terminate.

3.6 Input for Student Project Allocation Algorithm

The students in the system individually create their preferences, from the available project of their interest. Similarly, the staff create their rank preferences from the student in the system. Student allocation algorithm requires those overall preferences as input, in a certain constraint order. This requires n number of students and n number projects to be ranked to each other.

The developed system consists of three (3) dashboard: The student, project supervisor and administrator. The student login into the system to submit project topic, create ranking preferences, and to receive update about the allocation. The supervisor logs in to submit a propose topic and also create preference of the student. The administrator manage the allocation process as well as run the allocation algorithm.

The final system was implemented and the admin dashboard is shown below in Fig. 4.

4. RESULTS

To test the feasibility of the algorithm in the allocation process, a system is being developed for the allocation with the algorithm implemented in it. The system provided an interface for the students to enter their preferences to the available projects and supervisor (project) to the available students. The algorithm take as input the two preference list and allocates each student to his/her most appropriate project from the perspective of both ranking. The system is tested with the data and result below:

Example:

Sample students (University username)
Acp14jlr, acp14sh, acp14msa, acp14xw, and acp14hat
Sample project topics

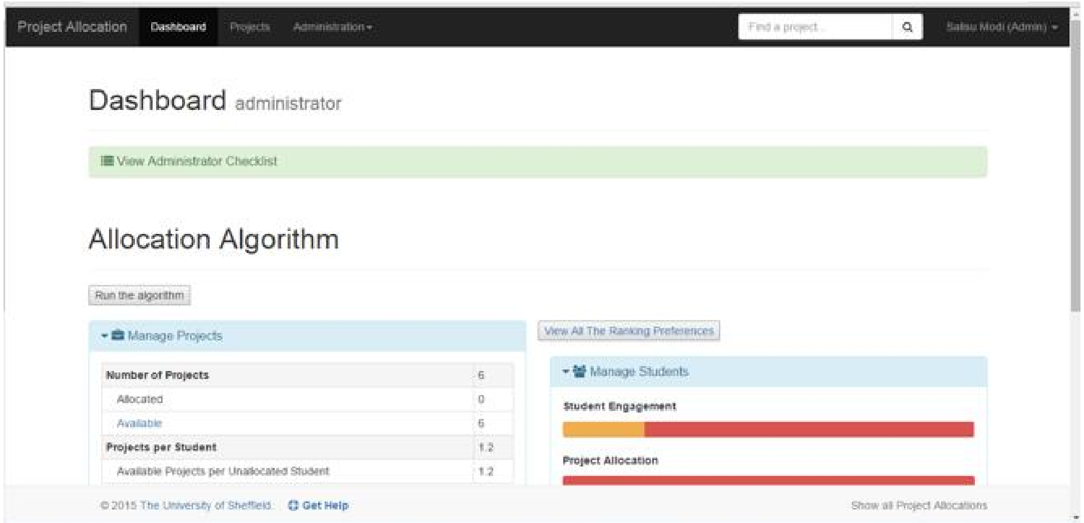

Fig. 4. Admin dashboard of the implemented system

Listening to Sheffield (LTS), decision support system (DSS), student placement portal (SPP), privacy of information (PI), and project allocation system (PAS).

4.1 Ranking Preferences

Each student rank the available project from highest to the lowest left to right. Likewise, the supervisor (project) rank the students from highest to the lowest in same order in the Table 1 below.

This page shows the sample ranking from the implemented system. The student created a rank preferences of the available project in Fig. 5.

The case where the number of students or the project grow large, and the student or the supervisor could not rank all the other partners. The system implements a function which automate the ranking of unranked partner.

The result after running the system with the above data is shown in Fig. 6.

From the Fig. 6 result of matching, student acp14jlr was allocated to LTS project which happened to be his/her second choice. The student cannot get his /her first choice because the student was rank fourth by the supervisor of the project. And LTS supervisor cannot get his/her first choice student (acp14xw) because was ranked the last by the student. In order, the algorithm makes all the remaining allocation.

At the end of matching the student to project from the preferences from both sets, a set containing each student with allocated partner was returned. It was argued that matching entities from two set of the element with preferences from both set always resulted to individuals in the set been paired with one another [8].

Table 1. Students and supervisors ranking preferences

Student preferences	Supervisor (project) preferences
Acp14sh =>DSS, SPP, PI, LTS, PAS Acp14msa=>SPP, PAS, LTS, PI, DSS Acp14xw=> PI, SPP, PAS, DSS, LTS Acp14hat=> PI, PAS, LTS, SPP, DSS	PAS=>acp14jlr, acp14hat, acp14sh, acp14msa, acp14xw DSS=>acp14msa, acp14sh, acp14hat, acp14jlr, acp14xw PI=>acp14msa, acp14xw, acp14hat, acp14sh, acp14jlr LTS=>acp14xw, acp14msa, acp14jlr, acp14sh, acp14hat

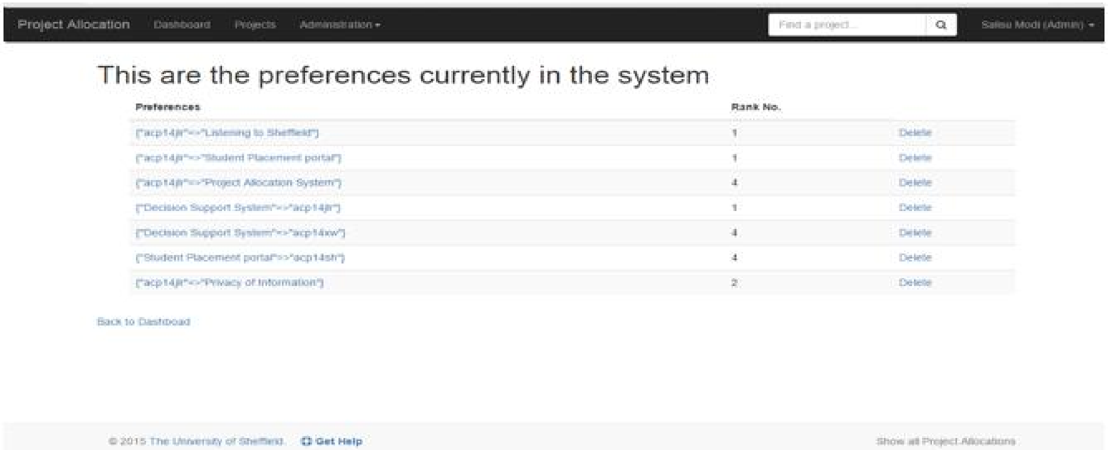

Fig. 5. Sample ranking from the system

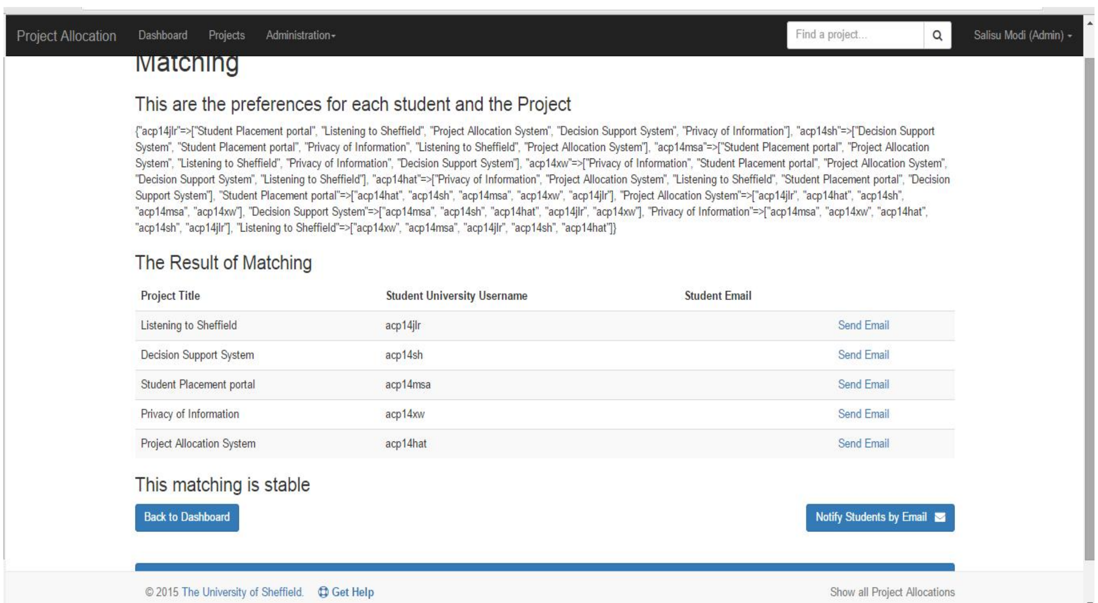

Fig. 6. Sample test result from implemented system

The result of running student project allocation algorithm, student and project instances are returned. Those instances have a number of properties which include allocated partners. Each student has a partner (project) assigned to him/her. This project was at least the first or at most the last project from the preference list of the student, depending on the rank position the student was in the preferences order of the project.

From the result of running the algorithm as applied to some number of student and project, it can be concluded that, no swap between any pair will result to happier matching than the initial one since all pairing resulted from the preferences that the student or the project created and accepted before the pairing [13]. The final result of student project allocation algorithm returns a matching set with each student in the system allocated to one project.

4.2 Performance of the Algorithm

The strength of this algorithm is in its ability to provide stability in matching n equal number of involved parties [14,15] argued that in any matching that is two-sided and both the involved parties need to rank one another, then the solution that handled the matching, the stability is the main requirement for such solution. The Gale Shapley algorithm always result to a matching set when two parties are matched from their preferences [16].

4.3 Time Complexity and Correctness of the Algorithm

The whole idea in Gale Shapley algorithm is matching n number of one party with n number of other party (where n is any counting number). In this implementation, n number of student is matched with n number of projects from their preferences. The process involved iterating through all free students while there is still unallocated student. Each free student goes to all project topic in his preference list in an orderly manner. And for each supervisor (project) student goes to, he checks if the project is unallocated, if yes, the allocation (student-project) is performed. If the project is allocated to someone else, then the project (supervisor) chooses either to remain allocated (reject the current student request) or dumps the current allocation (reject the current student request) from preference list of the project. The process continues iteratively until all students are unallocated. Intuitively this algorithm involve matching N X N items iteratively, and hence the time complexity Gale Shapley algorithm is O (n^2).

The correctness of this algorithm can be viewed from two perspective; stability and perfection. In the stability aspect, Gale Shapley claim that at the end of matching no swap between pairs that will make a happier match than the initial matching will be possible. And the perfection of the algorithm is such that all the member of the two parties involved must be matched to a partner [14].

5. CONCLUSION

The goal has been to investigate the different concept of stable marriage problem algorithm and how they can be deployed in student project allocation process. This goal has been achieved through developing a system using an algorithm base on Gale Sharply algorithm that is capable of handling project allocation.

The system was tested with some sample data of students and supervisors. The algorithm was supplied with input (students and supervisors preferences) and the output was produced by running the algorithm as shown in the previous sections.

Therefore, it can be deducted from this research that, stability in allocating student project will result to a quality of the research since student are allocated project from their preferences. We also show that deploying stable marriage algorithm in student project allocation could help in reducing the challenges during project allocation.

Further research can be conducted to extend the allocation algorithmm to enable monitoring of the number of project that each supervisor could be assign to avoid overloading, since problem arise when the number of students who have chosen the supervisor as their first choice exceeds the supervisor’s supervision capacity.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Gonczarowski YA, et al. A stable marriage requires communication in proceedings of the twenty-sixth annual. ACM-SIAM Symposium on Discrete Algorithms. SIAM; 2014.
2. Teo CP, Sethuraman J, Tan WP. Gale-shapley stable marriage problem revisited: Strategic issues and applications. Management Science. 2001;47(9):1252-1267.
3. Sanfoundry. Java program to implement gale shapley algorithm – sanfoundry; 2013.
4. Iwama K, Miyazaki S. A survey of the stable marriage problem and its variants. International Conference on Informatics Education and Research for Knowledge-Circulating Society. IEEE; 2008.
5. Lightfoot S, Pratt-Hartmann I. Solving the student project allocation problem; 2016.
6. Irving WR, GD. The stable marriage problem: Structure and algorithms; 1989.
7. Hattori T, Yamasaki T, Kumano M. New fast iteration algorithm for the solution of generalized stable marriage problem. Systems, Man, and Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference. IEEE; 1999.
8. Manlove DF, O'Malley G. Student-project allocation with preferences over projects.

Journal of Discrete Algorithms. 2008;6(4): 553-560.
9. Wan-Hong I. A web-based project allocation system; 2017.
10. El-Atta AAMM, Student project allocation with preference lists over (Student, Project) Pairs. Second International Conference on Computer and Electrical Engineering; 2009.
11. Fleiner T. On stable matchings and flows. Algorithms. 2014;7(1):1-14.
12. Salami HO, Mamman EY. A genetic algorithm for allocating project supervisors to students. International Journal of Intelligent Systems and Applications. 2016; 8(10):51.
13. Aderanti FA, Oluwatobiloba RTAAA. Development of student project allocation system using matching algorithm. Technology (ICONSEET). 2016;1(22):153-160.
14. Chiarandini M, Fagerberg R, Gualandi S. Handling preferences in student-project allocation. Annals of Operations Research. 2017;1-40.
15. Moussa MI, El-Atta AHA. A visual implementation of student project allocation. International Journal of Computer Theory and Engineering. 2011; 3(2):178.
16. Deng Y, Panigrahi D, Waggoner B. The complexity of stable matchings under substitutable preferences. AAAI; 2017.

Peer-review history:
The peer review history for this paper can be accessed here:
http://prh.sdiarticle3.com/review-history/26233