Improvement sub-phase

Finally, the initial solution is improved by applying algorithm that may try to minimize the violation of soft constrains while ensuring the feasibility. Recently Adaptive Acceptance Criterion algorithm (AAC) [20] investigated over different optimization problems and obtained good quality solution results outperformed different approaches results in the literature, which motivates to use it in the improvement phase for the system.

AAC is an algorithm employs a non-parametric acceptance criterion that does not rely on user defined. AAC accepts a little worse solution based on stored estimated value. In AAC, an estimated value added to the threshold (when the search is idle) to increase the search exploration. The estimated value is generated based on the frequency of the solutions quality differences, which are stored in an array. Pseudo code for AAC is described in Figure 3 (please read [20] for more details).

Procedure Adaptive Acceptance Criterion Algorithm (AAC)

Step-1: Initialization Phase
Set EV, $C_{idle\text{-}iterations}$, equal to zero ;
Set L_{EV} equal to zero value with zero frequency;
Set idle-iterations;

Step-2: Improvement (Iterative) Phase
repeat (while termination condition is not satisfied)
Add Iterations by one;
Step-2.1: Selecting candidate solution $S_{working}$

Step-2.2: Accepting Solution
if $f(S_{working}) < f(S_{Arranged})$
$C_{idle\text{-}iterations} = 0$;
$EV = f(S_{Arrange}) - f(S_{working})$;
Update the frequency of EV in the L_{EV} (or add EV in L_{EV} if it is not exist);
Sort the EV elements in L_{EV} in descending order based on their frequency or if equivalence then Ascending based on their value;
$S_{Arrange} := S_{working}$; $f(S_{Arrange}) = f(S_{working})$;
$S_{source} = S_{working}$; $f(S_{source}) = f(S_{working})$;
else
if $SV <> 0$
if $AC >= SV$ then
else
$C_{idle\text{-}iterations} = C_{idle\text{-}iterations} + 1$;
end if
end if
end if

Step 2.3: Idle Iteration
if $C_{idle\text{-}iterations} >= idle\text{-}iterations$
$EV = L_{EV}[0]$; // get the first element in A
Rotate left all elements in L_{EV};
Calculate AC by adding EV to $f(S_{Arrange})$;
if $AC >= SV$ then
$S_{source} = S_{working}$;
$C_{idle\text{-}iterations} = 0$
end if
until $Iterations > N_{iterations}$ (termination condition is met)

Step-3: Termination phase
Return the best solution found $S_{Arrange}$

Figure 3. Pseudo code for (AAC)

IMPLEMENTATION AND DISCUSSION

In this work, the system implemented by using visual basic language on a PC with a processer of Intel core I5- 3.2GHz and 4GB RAM. The system is described for the male section data. Figure 4 and Figure 5; show the main screen for the system.

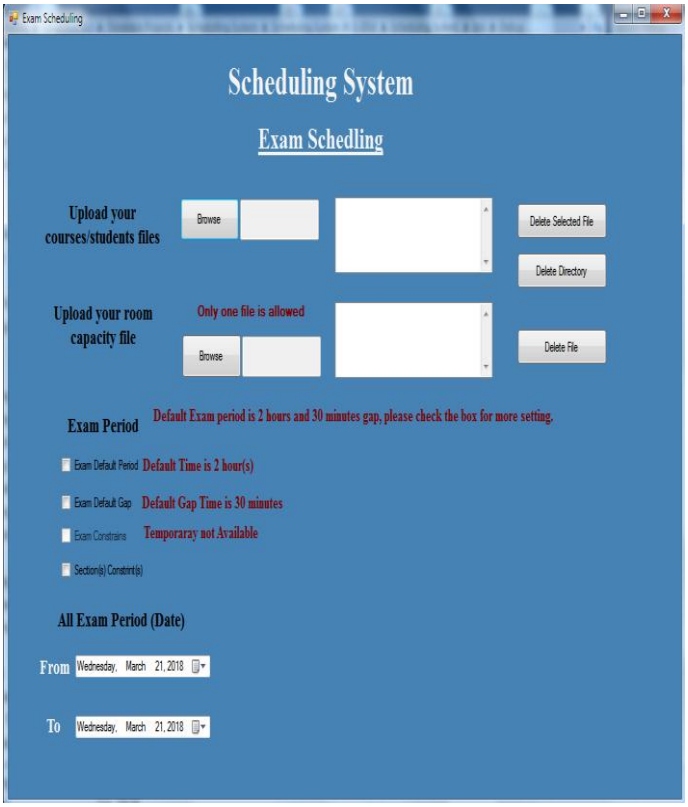

Figure 4. Main System Screen

Figure 4 shows, the main screen for the proposed examination system, where you can:

- Upload the file of student's id's numbers for each courses section or upload the file of the student's id's numbers by combining all sections for the similar course in one file. However, you can upload whole directory directly to the system file location (which are created in C: drive) as in Figure 5.
- Upload room files (boys and girls sections) that stores each room name with the student capacity number as in Figure 5.
- Parameter settings can be assign before creating the exam timetable, for example, exam default period as in Figure 6, exam default gap as in Figure 7, and sections constrains as in Figure 8.
- Choose the exam starting and ending days as in Figure 9.

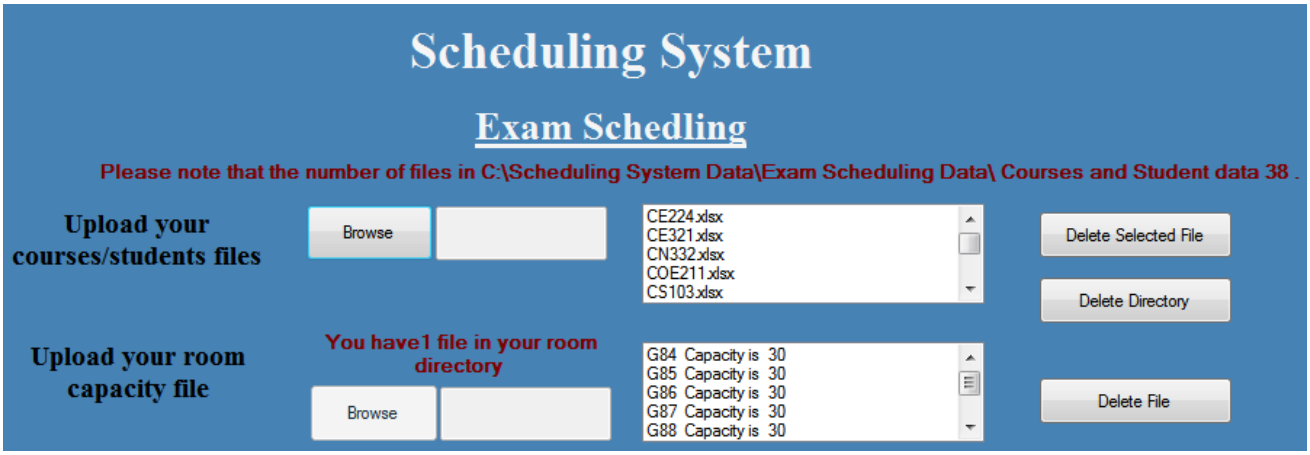

Figure 5. Uploading Courses/Students and rooms files Screen

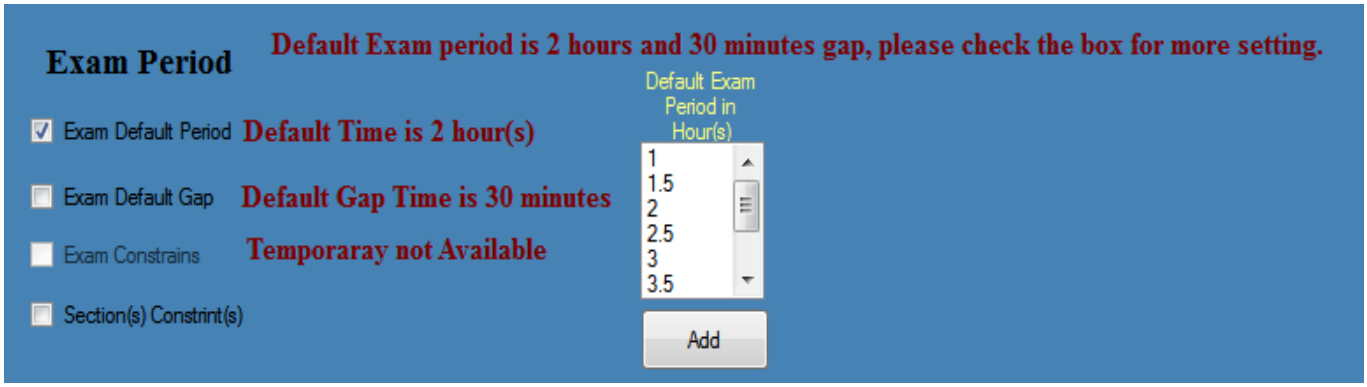

Figure 6. Exam default period setting

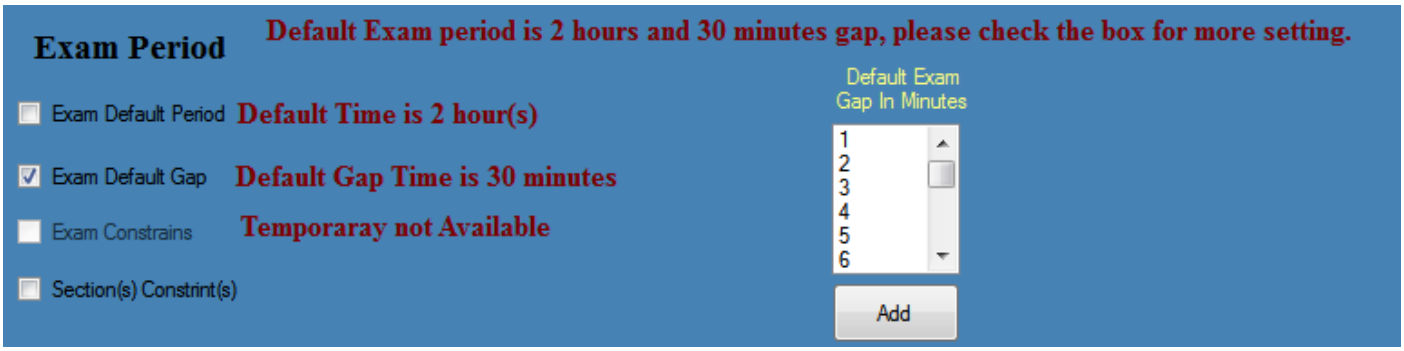

Figure 7. Exam default gap setting

Figure 5 shows that, the student in each course and rooms file are uploaded into the system with clarifying the path of it. In case of not uploading the correct files (courses and rooms), the system will not allow to use any of the system settings, otherwise, the settings will be available for use.

The system initialized with default exam time as two hours and the gap between each two exams is 30 minutes, however, the system allow the user to modify the initialized setting in Figure 6 and Figure 7. Figure 6 shows that, in case of the user checked for the exam default period setting, then the system will allow the user to choose different exam period to set as default period for all exams in hours. Also Figure 7 shows that, the system will allow changing the default gap time in minutes, and then set it as default gap time between two consecutive exams. Later on, the system allow the user to assign sections setting as in Figure 8, where in case of the user upload the files as courses section, then the system will allow to add a constraint to consider it as one course, or in case there are different plan in the departments and there is two different course number for the same course, then can combine it as one course.

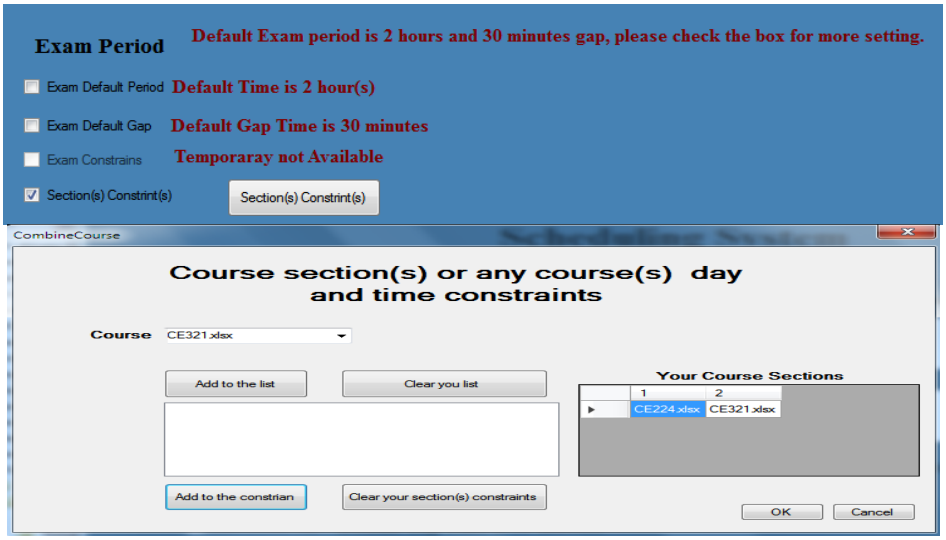

Figure 8. Exam sections constrains setting

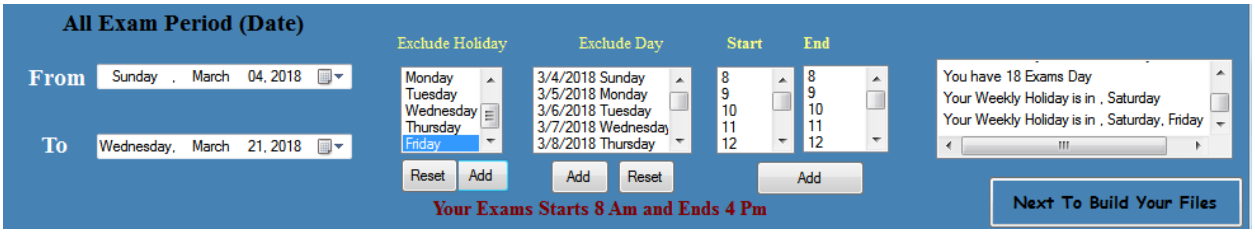

Figure 9. Exam starting and ending days setting

Figure 9 shows that the system initialized with default staring exam time at 8AM and ending exam time 4PM in case of choosing exam days. The setting for the exams starting and ending days shows that you can:

- Choose the starting day date and ending day date.
- Exclude any day as a weekly holiday for all chosen days between starting and ending day.
- Exclude specific day as holiday between starting and ending day.
- Changing the default starting and ending exams time for all days.
- Build your files settings to construct the conflict matrix as in Figure 10.

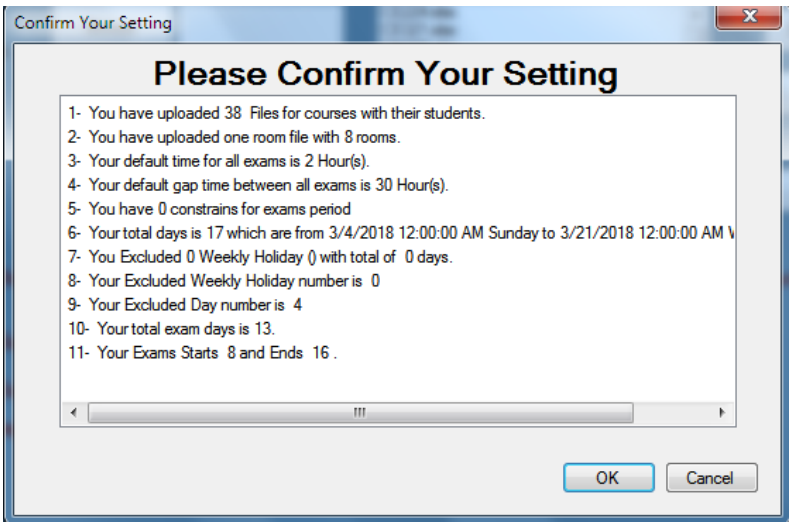

Figure 10. Exam setting Confirmation

After confirming all settings as in figure 10, the courses/students conflict matrix is built as in Figure 11, then you can see the students/courses conflicts matrix in Figure 12, and for more clarification you can see students/courses in conflicts after optimize and exclude all conflicts duplications as in Figure 13. Please be noted, that the matrix is optional to follow to understand what happened, where you can directly move to the next step to construct feasible exam table as in Figure 14.

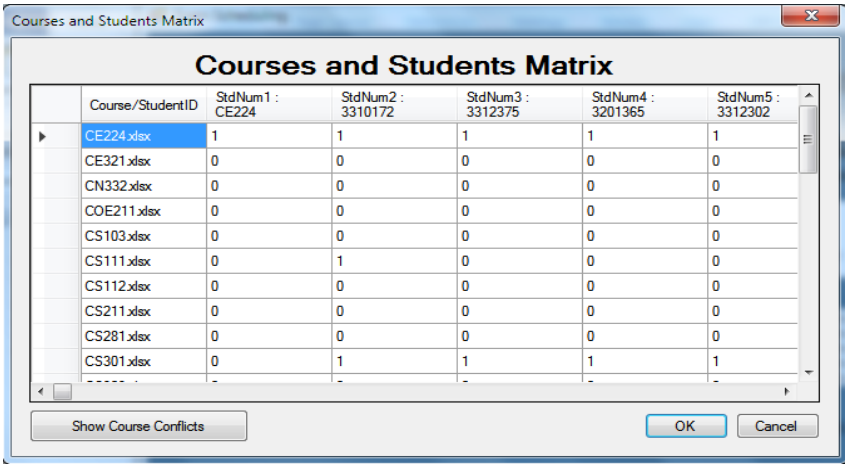

Figure 11. Courses/Students Conflict Matrix

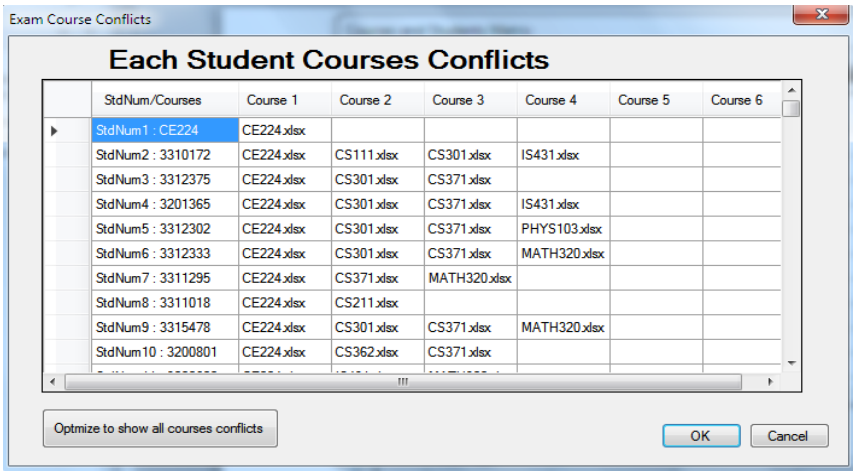

Figure 12. Students/Courses Conflict Matrix

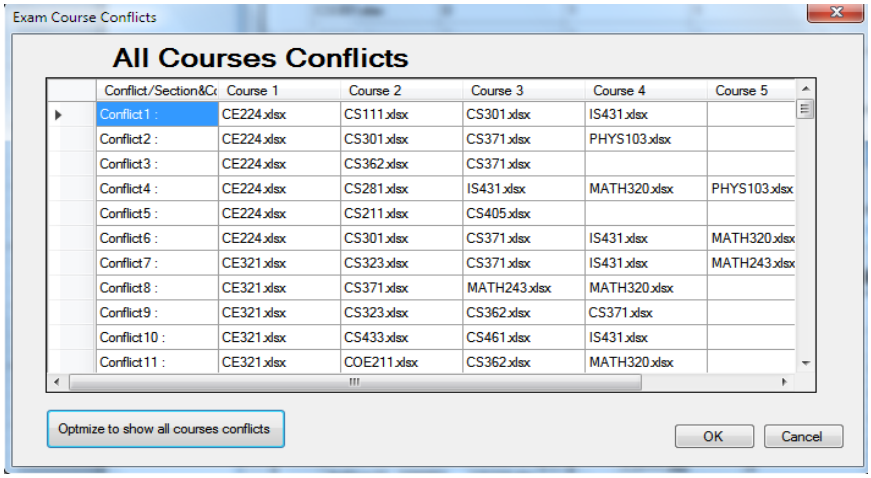

Figure 13. Optimize Students/Courses Conflict Matrix

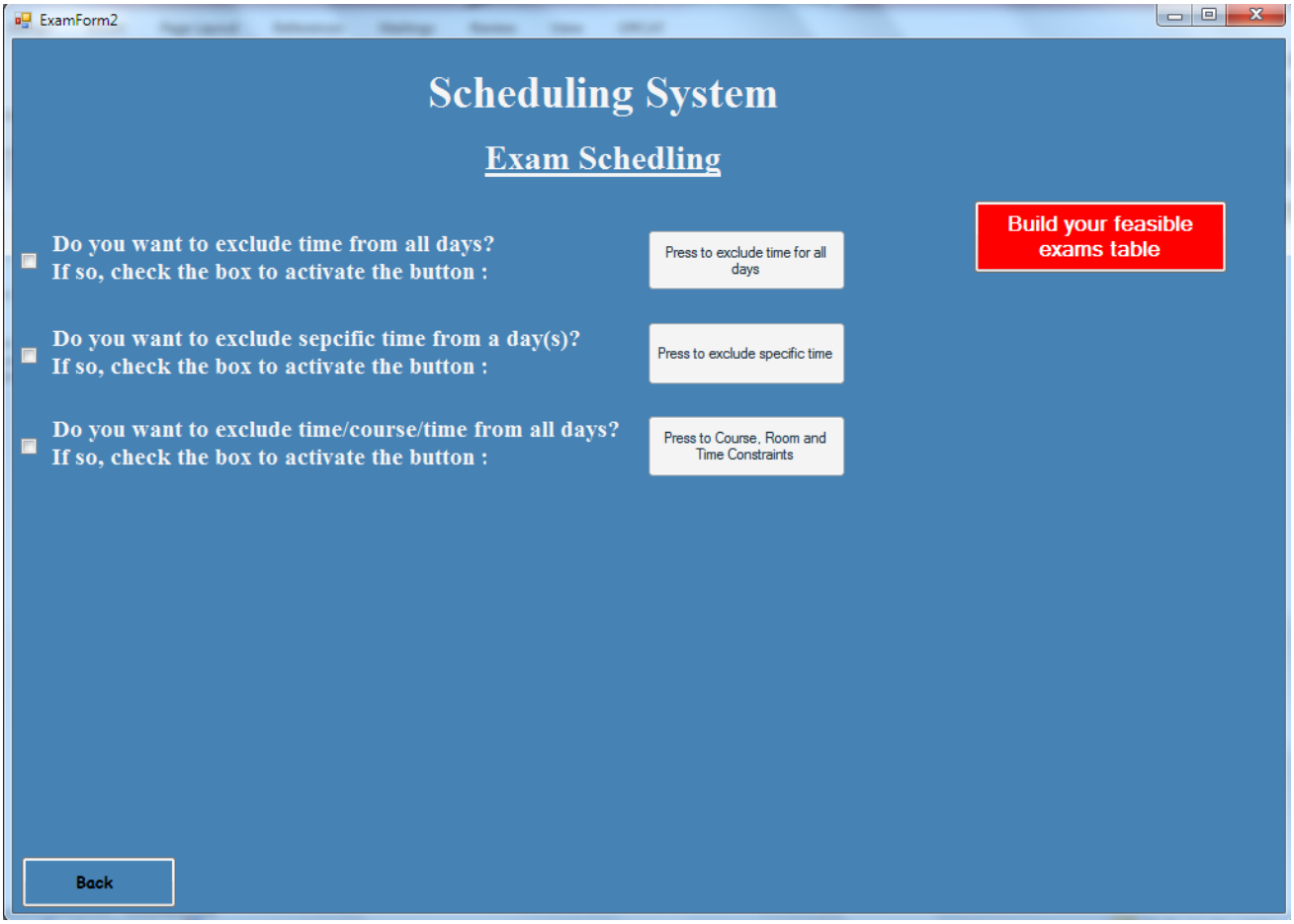

Figure 14. Constructing feasible timetable screen

Figure 14 show that, there are three constrains options before building the feasible exam timetable as follows:

- Exclude specific time from all days as in Figure 15.
- Exclude specific time from a specific day as in Figure 16.
- Exclude specific time or course for all days as in Figure 17.

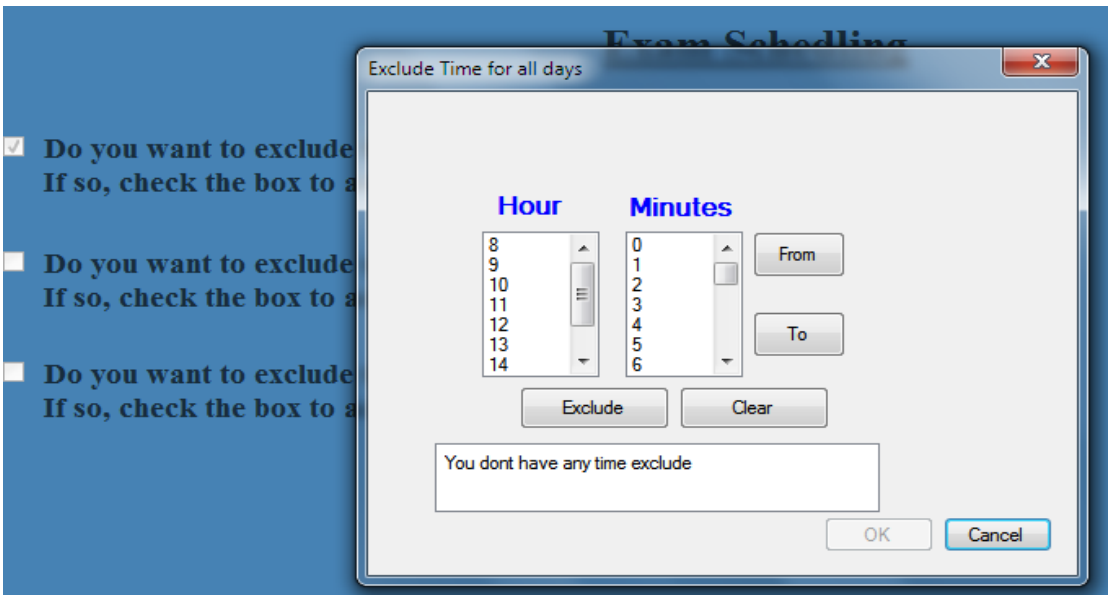

Figure 15. Exclude specific time from all days

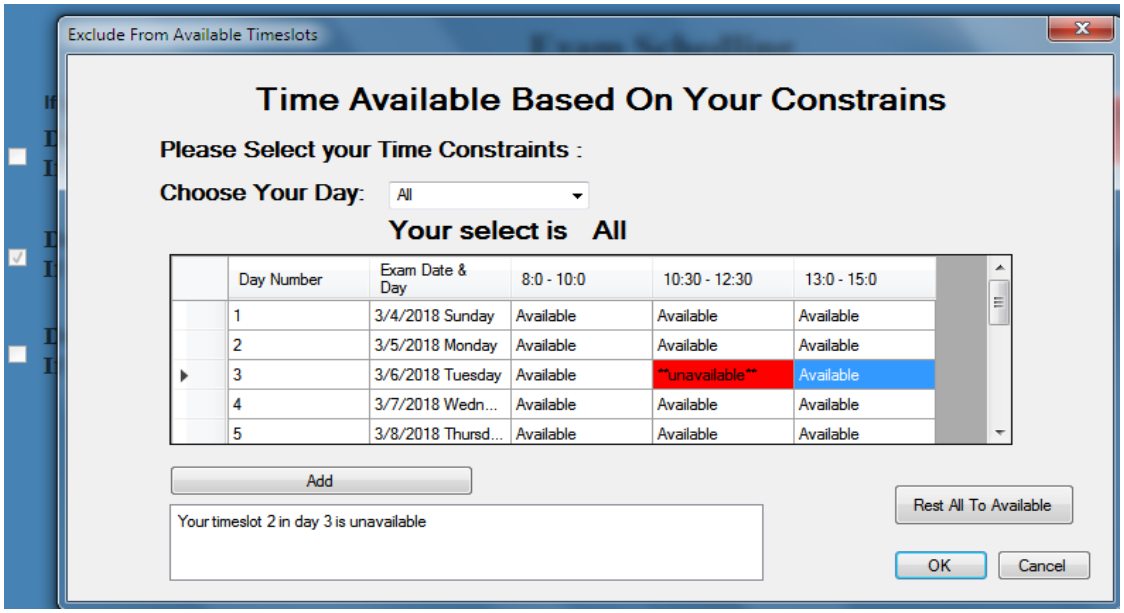

Figure 16. Exclude specific time from a specific day

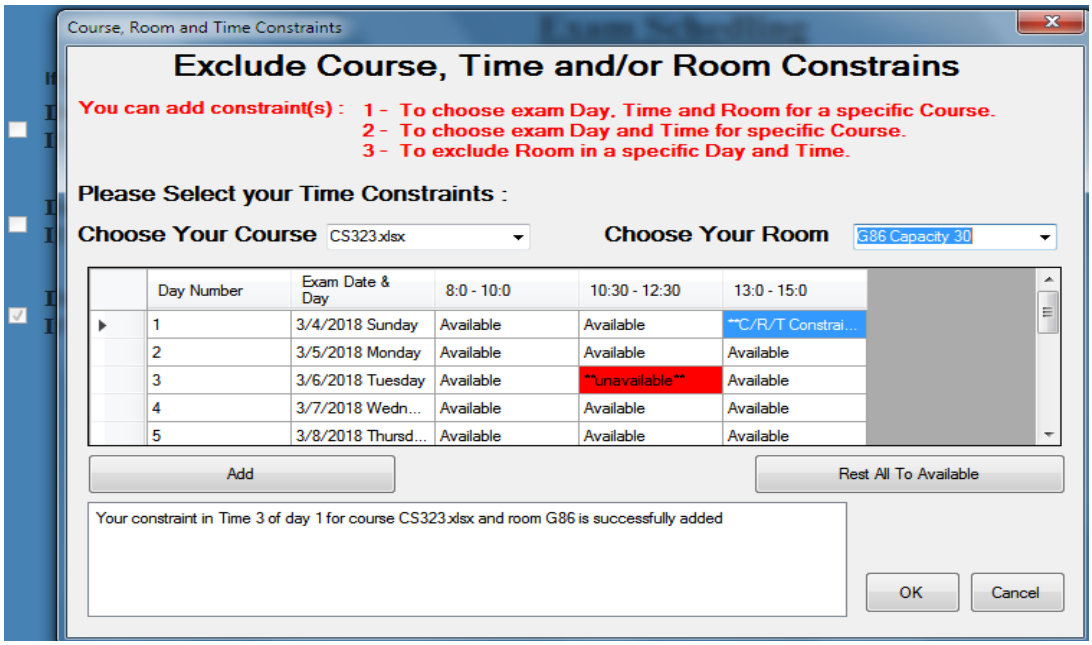

Figure 17. Exclude specific time or course for all days

Figure 15 shows that, the system allow excluding a specific time (e.g. Pray) for all days, where the timeslots reinitialized based on the new setting chosen. In addition, Figure 16 shows that it is possible to exclude a specific time for a specific day (e.g. 3/6/2018 Tuesday at 10:30-12:30). Furthermore, it's allowed to choose a specific course for a specific day/time/room (e.g. 3/4/2018 Sunday at 13:00-15:00, CS323 course in room G86) as in Figure 17. After that, the system settings are specified and can move to the next step to build a feasible exam timetable as in Figure 18.

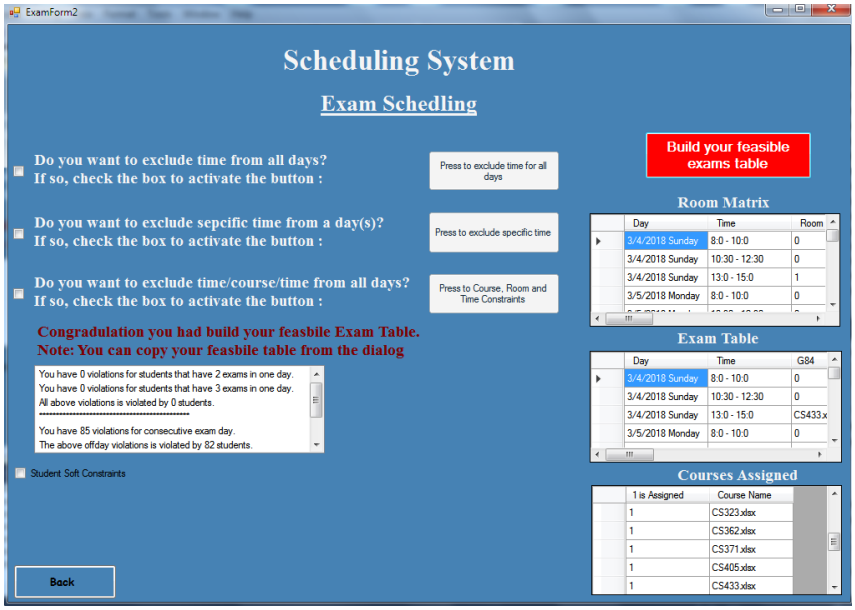

Figure 18. Building Exam Timetable

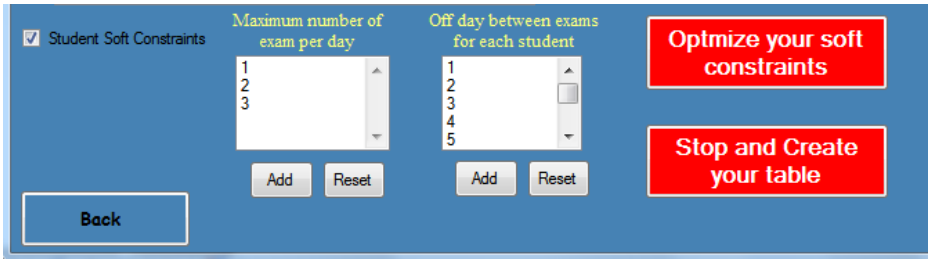

Figure 19. Additional soft constrains for exam timetable

Figure 18 shows the room matrix, exam timetable, and assigned course list after building the feasible table and an improvement is done. Meanwhile, there is description for the exam timetable violations. For instance, in room matrix there is "1" in 3/4/2018 Sunday at 13:00-15:00, which reflect to the CS433 course exam in exam table and value of "1" in the course assigned list. However, the timetable description shows, that there is 0 students have three consecutive exams in one day, 0 students have two exams at same time in one day, and 85 students have two consecutive exams in one day.

In case of the user satisfied with exam timetable violations, then it is possible to be taken as it is, or rebuild the exam table to obtain a new exam timetable and son on. However, there is additional constrains can be apply to the exam timetable as in Figure 19, where can modify the maximum number of exams per day for all students and specify a number of days for the students as a break between exams if possible. These constrain, will reflect on the violation of the soft constraints and calculate it with the previous soft constraints and when we satisfy, we can stop the timetable optimizing and create the exam timetable.

CONCLUSION

Exam timetable involve of allocating some students in courses to predefined rooms and timeslots subject to some constraints that must satisfy or others may satisfy. Several institution have their own way in generating the exam timetable that will suit their population size, which often may initially generating with some errors. In this paper, a dynamic real system is proposed by using the settings for day, time, and room dynamically, since it allows the used to input the resources available. Largest color degree first is used to generate a feasible timetable and Adaptive acceptance criteria algorithm is used for improvement. Pervious data for the timetable in CCSE- Taibah University is used as a test domain. The system show a good quality timetable that meets the regulations imposed while comparing it to the original timetable proposed that term, in which save time and money.

Generally, the construction phase plays an important role in initializing the solution for improvement. In the proposed system, largest color degree first is used in the construction phase to obtain initial solution, where there are different graph coloring techniques can be used to obtain different solution structure to intensifying the search space that may help to obtain better solution in different universities data problem

such as: largest Degree first, least Saturation degree first, largest Weighted degree first, and largest Enrolment first for obtaining different solutions structures. Also, the proposed system deals with the room capacity; however different university may needs rooms' features for different exams Which pose a future works, to enhance the system by intensifying the search space by multiple graph coloring techniques and add the rooms features to the system, then evaluate the effectiveness of the system in different datasets.

REFERENCES

[1] Even, S., Itai, A., and Shamir, A. 1976, "On the complexity of timetable and multicommodity flow problems," SIAM Journal on Computing, 5(4), pp. 691-703.

[2] Cooper, B., and Kingston, J.H., 1996, "The Complexity of Timetable Construction Problems," In the Practice and Theory of Automated Timetabling, ed. E.K. Burke and P. Ross, Springer-Verlag, pp. 283- 295,

[3] Bardadym, V.A., 1995, "Computer-aided school and university timetabling: The new wave," Practice and Theory of Automated Timetabling I: Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling (PATAT I), Edinburgh, UK,, Lecture Notes in Computer Science, Springer-Verlag, 1153/1996, pp. 22-45.

[4] Wren, A., 1996, "Scheduling, Timetabling and Rostering – A special relationship?," The Practice and Theory of Automated Timetabling I: Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling (PATAT I), Edinburgh, UK, Lecture Notes in Computer Science, Springer-Verlag, vol 1153, pp. 46-75.

[5] Burke, E.K., Kingston, J., Jackson, K., and Weare, R., 1997, "Automated University Timetabling: The State of the Art," The Computer Journal, 40(9), pp. 565-571.

[6] Schaerf, A., 1999, "A survey of automated timetabling," Artificial Intelligence Review, 13(2), pp. 87-127.

[7] Carter, M.W., 1983, "A decomposition algorithm for practical timetabling problems," Technical Paper 83-06, Department of Industrial Engineering, University of Toronto.

[8] Carter. M.W., 1986, "A survey of practical applications of examination timetabling," Journal of Operations Research, 34(2), pp. 193-202.

[9] Carter, M.W., Laporte, G., and Chinneck, J.W., 1994, "A general examination scheduling system," Interfaces, 24(3), pp. 109-120.

[10] Carter, M.W., Laporte, G., and Lee, S., 1996, "Examination timetabling: Algorithmic strategies and applications," Journal of the Operational Research Society, 47(3), pp. 373-383.

[11] Hertz, A., 1991, "Tabu search for large scale timetabling problems," European Journal of Operational Research, 5 September, 54(1), pp. 39-47.

[12] Asmuni, H., 2008, "Fuzzy Methodologies for Automated University Timetabling Solution Construction and Evaluation by Hishammuddin Asmuni," MSc Thesis submitted to the University of Nottingham for the degree of Doctor of Philosophy, April 2008, pp.16-24.

[13] Carter, M.W., and Laporte. G., 1998, "Recent developments in practical course timetabling," The Practice and Theory of Automated Timetabling II: Selected Papers from 2nd International Conference on the Practice and Theory of Automated Timetabling (PATAT II), 1997, Toronto, Canada, Lecture Notes in Computer Science, Springer-Verlag, vol 1408, pp. 3-19.

[14] Petrovic, S., and Burke, E.K., 2004, "University Timetabling. Handbook of Scheduling: Algorithms, Models and Performance Analysis," Chapter 45, Chapman Hall/CRC Press.

[15] Lewis, R., 2008, "A survey of metaheuristic-based techniques for university timetabling problems," OR Spectrum, 30(1), pp. 167-190.

[16] Qu, R., Burke, E. K., McCollum, B., Merlot, L. T. G., and Lee, S. Y., 2009, "A Survey of Search Methodologies and Automated System Development for Examination timetabling," Proceeding of Journal Scheduling, 12(1), pp. 55-89.

[17] Burke, E.K., Bykov, Y., 2008. A Late Acceptance Strategy in Hill-Climbing for Exam Timetabling Problems, in the Proceeding PATAT '08, Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling, Universit de Montral, Montreal, Canada, August (http://www.cs.nott.ac.uk/~yxb/LAHC/).

[18] Abuhamdah, A., and Ayob, M., 2010, "Adaptive Randomized Descent Algorithm using Round Robin for solving course timetabling problems," In: International Conference on Intelligent Systems Design and Applications (ISDA), IEEE, pp. 1201-1206

[19] Abuhamdah, A., Ayob, M., Kendall, G., and Sabar, N., 2013, "Population based Local Search for university course timetabling problems," Proceeding of Applied Intelligence Journal, Springer link, Online ISSN:1573-7497, doi:10.1007/s10489-013-0444-6

[20] Abuhamdah, A., 2015, "Adaptive Acceptance Criterion (AAC) algorithm for Optimization Problems," Journal of Computer Science, DOI: 10.3844/jcssp.2015. 11(4), pp. 675-691.

[21] Ghaith, Jaradata., Masri, Ayobb., and Ibrahim, AlMarashdeh., 2016, "The effect of elite pool in hybrid population-based meta-heuristics for solving combinatorial optimization problems," Applied Soft Computing journal, vol 44, pp. 45-56.

[22] Ghaith M.Jaradat., Anas, Al-Badareen., Masri, Ayob., Mutasem, Al-Smadi., Ibrahim, Al-Marashdeh., Mahmoud, Ash-Shuqran., and Eyas, Al-Odat., 2018, "Hybrid Elitist-Ant System for Nurse-Rostering Problem," Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2018.02.009. XXX, XXX-XXX.

[23] MOREIRA, José Joaquim., 2008, "A System for Automatic Construction of Exam Timetable Using Genetic Algorithms," Tékhne [online], ISSN 1645-991, n. 9, pp.319-336

[24] Lukas, S., Aribowo, A. , and Muchri, M., 2012, "Solving timetable problem by genetic algorithm and heuristic search case study: Universitas pelita harapan timetable," In O. Roeva, editor,Real-World Applications of Genetic Algorithms, InTech, 2012. 378, pp.303–316.

[25] Arogundade, O.T., Akinwale, A.T., and Aweda, O.M., 2010, "A Genetic Algorithm Approach for a Real-World University Examination Timetabling Problem," International Journal of Computer Applications, December 2010, 12(5), pp.1-4.

[26] Rodriguez, N., Martinez, J., Flores, J. J., and Graff, M., 2014, "Solving a Scholar Timetabling Problem Using a Genetic Algorithm - Study Case: Instituto Tecnologico De Zitacuaro," Proc. 13th Mexican International Conference on Artificial Intelligence (MICAI), pp.197-202.

[27] NASA, 2000, "Management Concepts: Software Cost Estimating for NASA Ames Research Center," Course Handout, NASA Ames Research Center, Moffett Field, CA, Oct.

[28] Silva, D.L, and Obit, J.H., 2008, "Great Deluge with Nonlinear Decay Rate for Solving Course Timetabling Problems," In. Proceedings of the 2008 IEEE Conference on Intelligent Systems (IS 2008), IEEE press, September, 2008, pp.8.11-8.18.