International Journal of Computer and Information Technology (ISSN: 2279 – 0764)
Volume 07– Issue 04, July 2018

An Innovative Computerized Method For Creating an Exam Schedule

Mutaz Rasmi Abu Sara
Department of Computer Science
Taibah University, TU
Al-Medina, Saudi Arabia
Email: mabusara [AT] taibahu.edu.sa

***Abstract*— The process of electronically establishing and organizing exam schedules at universities and academic institutes has become urgent in light of the rapid development of computer systems and technology. This study offers an innovative computerized method of creating an exam schedule by completing what was suggested in our previous research, which presented the first step of this procedure. In this study, the program has been improved to perform the rest of the steps to produce an integrated exam schedule of the courses that students are currently enrolled in in a set of groups, with each group in a distinct period. The new proposal enables the rearrangement of the results so that different tables can be produced by rearranging the sets of courses, guaranteeing the absence of scheduling conflicts for all the targeted students.**

The main result is that the new program produces an integrated exam schedule that solves several key constraints related to scheduling, students, conflicts, and courses. The program design is user-friendly and allows users to see, with 100% accuracy, which courses do not conflict. For courses that do conflict, the program outputs the conflict ratio in a flexible and easy process. It also arranges all the nonconflicting courses into sets of groups with each group in a distinct period; that is, the program produces a schedule that is ready for assignment to the days specified for the exam.

Keywords- Exam-scheduling computer program; nonconflicting groups; shuffling

I. Introduction

The process of building conflict-free exam schedules at institutes and colleges can be difficult, mainly because students' schedules are irregular according to their study plan. Thus, it is necessary to build a program that can analyze a list of currently enrolled students to find the conflicts and propose a suitable schedule for all the students.

Our new program offers a way to create an exam schedule with no conflicts. The input data are the numbers of students enrolled in each course, and the output is a schedule containing groups of courses that are not in conflict and are suitable for being the same interval. In addition, the user can obtain a list of all the conflicting courses that shows the number of students with a conflict.

In Sections II and III, we will discuss the importance of this and previous studies. Then, the algorithm will be introduced in detail in Section IV. The program is detailed in Section V, followed by the conclusion and future work.

II. Importance of the Study

The significance of this study can be summarized by the following steps: first, the program produces an integrated exam schedule in which the courses are divided into a set of periods; second, the program gives the user five options for viewing the results, which allows him to produce a conflict-free exam schedule; third, the program has the ability to produce many different versions of the schedule by allowing shuffling.

III. Previous Studies

In [1], the authors proposed an exam-scheduling computer program that allows users to generate a course list with no conflicts with 100% accuracy. In addition, it can report the conflicting courses in a fast, easy, and flexible way by giving the conflict ratios. In [2], the authors used colored-graph scheduling technique to present an exam schedule that emphasized two things: first, the constraints of their system handles, and second, a user-friendly interface. In [13], the authors colored a graph representing different sections using the widely known recursive largest first (RLF) algorithm. The authors in [7] solved the exam-scheduling problem by presenting a survey of different particle swarm techniques. The authors in [5] generated exam schedules using a genetic algorithm; they considered a two-dimensional chromosome consisting of days as one dimension and exams as another dimension. The genetic algorithm they used relies on a mutation operator and excludes a crossover operator. The work in [16] used an ant colony approach to generate exam schedules, relying on the construction of an initial solution comprising days, rooms, slots, and exams; then, an exam schedule was developed by tracking the pheromones of ants trying to make paths to find the optimal exam schedules. In [14], a schedule was generated by searching among heuristics; this was achieved using an iterative local search and a set of movable operators that tended to improve the quality of the outcome schedule. A survey of different exam scheduling techniques can be found in [3] and [10]. A clonal selection algorithm that produces exam schedules was proposed in [15], in which a set of solutions (antibodies) is developed, and the affinity (fitness) of those solutions is calculated based on a fitness function; after that, the fittest antibodies are chosen to be cloned with a certain degree of mutation in order to find better solutions. The authors of [12] discussed a honey-bee mating optimization algorithm used to find near-optimal exam schedules. The algorithm relies on a queen (the current best solution), drones (trial solutions), workers (heuristics), and broods (new solutions). The algorithm first generates a pool of solutions from which the best is chosen as the queen, and the others are considered drones. Drones mate with the queen using crossover, thus generating new solutions.

Our new program has been built to complete what was started in the our previous research [1]—to provide a set of courses that do not conflict and to show them in the form of periods that can be transformed into an exam schedule directly.

IV. THE ALGORITHM

In order to apply the algorithm, the following structured or class will be used [1]. The first three steps can be used independently to find a matrix of conflicts:

```
public class Subject
{
  public string name; // course name
  public string[] students;
// number of students enrolled in the course
 public string values = "";
// list of courses that can be safely offered with the course
  public int[] conflicts;
// indicater (0)- has a conflict, (-1)- has no conflict with
  current course
  public int whight; // number of conflicts the course has
  }
```

This is illustrated as following:

Step1:

```
int colCount -> count Range_Columns_Count
// which refer to the number of courses
int rowCount -> count the Range_Rows_Count
        // which refer to the max number of students
```

Step2:

```
Subject[ ] subject -> Array of Subject with length [colCount];
        // initialize the array containing courses and students enrolled in these courses
```

Step3:

```
  Boolean noConflict = true;
// to decide whether there is a conflict
  for (int fcourse = 0; fcourse < subject.Length; fcourse++)
// fcourse - first course to be compared
  {
   for (int scncourse = fcourse + 1; scncourse < subject.Length; scncourse++) {
  // scncourse - socened course to be compared
    noConflict = true;
    for (int frow = 0; frow < rowCount AND
     subject[fcourse].students[frow] != null; frow++){
    if (subject[fcourse].students[frow].Equals("-1"))
     continue;
    for (int scndrow = 0; scndrow < rowCount AND
      subject[scncourse].students[scndrow] != null; scndrow++) {
     if (subject[scncourse].students[scndrow].Equals("-1"))
      continue;
     if(subject[fcourse].students[frow].
      Equals(subject[scncourse].students[scndrow])) {
      noConflict = false;
      break;
      }
     }
    if (noConflict == false) {
     break;
    }
   }
   if (noConflict == true) {
    subject[fcourse].values += subject[scncourse].name + ", ";
    subject[scncourse].values += subject[fcourse].name + ", ";
   }
  }
```

Step4:

```
    if (!btnShuffling.Enabled) {// for first time the input file is read
     getResultWithoutConflict();
    }
    else // shuffling the array of courses to get new groups for same input file without repeating the previous steps, which improves the algorithm performance
    {
     Random random = new Random();
     int r;
     Subject t;
     int tt;
     for(int i = 0; i < subject.Length; i++) {
       r = random.Next(subject.Length);
       t = subject[i];
       subject[i] = subject[r];
       subject[r] = t;
       for (int j = 0; j < subject.Length; j++) {
         tt = subject[j].conflicts[i];
         subject[j].conflicts[i] = subject[j].conflicts[r];
         subject[j].conflicts[r] = tt;
       }
     }
    }
    result = "";
    List<List<int>> list = new List<List<int>>(); // save the groups that are not conflicting
    List<int> temp = new List<int>(); // save the list of current groups to be compared later
```

```
      for (int i = 0; i < subject.Length; i++) {
        list.Add(new List<int>());
        for(int j = i + 1; j < subject[i].conflicts.Length; j++)
          if (subject[i].conflicts[j] == -1) { // noconflict
            list[i].Add(j);
          }
      }
   // at this point the list is initialized with courses (the course
and list of courses not conflicting with it)
      int period = 1;
      int[] done = new int[list.Count];
      Boolean hasConflict = false;
      for (int i = 0; i < list.Count; i++) {// start searching for
groups
        if (list[i].Count < 1)
          continue;
        temp.Clear();
        result += "Period " + period + ": {" + subject[i].name +
"; " + subject[list[i][0]].name + "; ";
        day++;
        //result += "{" + i + "; " + list[i][0] + "; ";
        temp.Add(list[i][0]);
        done[list[i][0]] = 1;
        done[i] = 1;
        for (int j = 1; j < list[i].Count; j++) {
          for (int k = 0; k < temp.Count; k++) {
            if (subject[list[i][j]].conflicts[temp[k]] == -1)
            {
              hasConflict = false;
            }
            else {
              hasConflict = true;
              break;
            }
          }
          if (!hasConflict) {
            result += subject[list[i][j]].name + "; ";
            temp.Add(list[i][j]);
            done[list[i][j]] = 1;
          }
        }
        for (int t = 0; t < temp.Count; t++) {
          list[temp[t]].Clear();
          for (int s = i; s < list.Count; s++) {
            list[s].Remove(temp[t]);
          }
        }
        result += "}\r\n\r\n";
      }
      for (int i = 0; i < done.Length; i++) {
        if (done[i] == 0) {
          result += "Period " + period + ": {" + subject[i].name
+ "}\r\n\r\n";
          day++;
        }
      }
    }
```

V. Pseudocode

The pseudocode contains the steps shown as follow:

First: Prepare a file (such as an Excel sheet) that contains the course names and the IDs of the students enrolled in those courses. Each course is loaded into a column; the first cell is the course code or name, while the other cells in the column will be the students' IDs with no empty cells. Empty cells, if any, can be converted to a value of -1 to be omitted during the algorithm execution. Then, the following steps are applied.

Second: Read the input file to determine the number of courses and the maximum number of students.

Third: Initialize the array with the courses and the students enrolled in those courses.

Fourth: Compare each course with all the remaining courses and identify the courses that do not conflict with that course.

Five: Determine the groups of courses that do not conflict with each other.

Sixth: The report shows the groups as numbered periods.

The overall algorithm is illustrated in Figure 1.

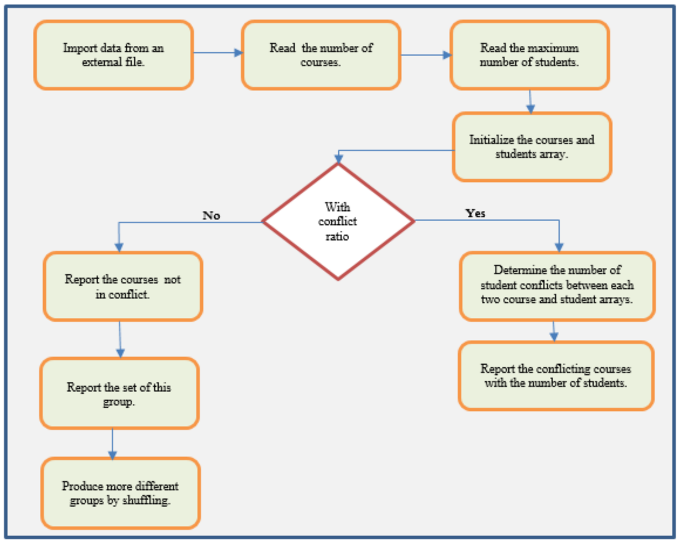

Import data from an external file.
Read the number of courses.
Read the maximum number of students.
Initialize the courses and students array.
With conflict ratio
No
Yes
Report the courses not in conflict.
Determine the number of student conflicts between each two course and student arrays.
Report the set of this group.
Report the conflicting courses with the number of students.
Produce more different groups by shuffling.

Fig. 1. Algorithm methodology.

VI. The Program

The user interface was designed to be attractive and easy to use so that the user can import data from another program such as Microsoft Excel. The user interface is illustrated in Figure 2.

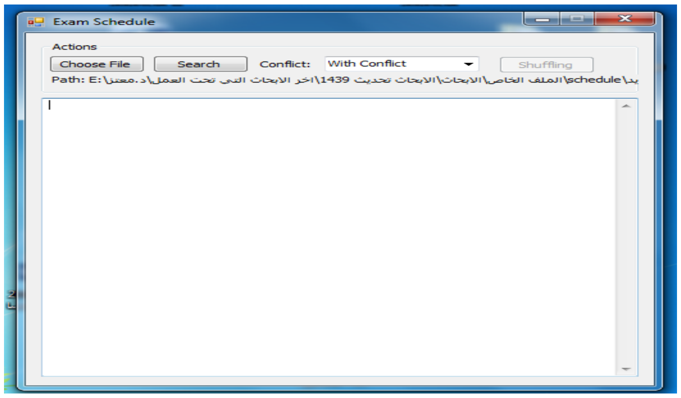

Exam Schedule
Actions
Choose File
Search
Conflict:
With Conflict
Shuffling
Path: E:\د.معتز\العمل تحت التي الابحاث اخر\1439 تحديث الابحاث\الابحاث\الابحاث\الملف الخاص\schedule\بد

Fig. 2. User interface.

The program provides five options, as shown in Figure 3, in a drop-down menu:

- With conflict: produce the courses that conflict and the number of students having a conflict in each course.
- Without conflict: produce the courses that do not conflict.
- With all conflict: produce a list of each course with all the conflicting courses and number of students having a conflict.
- Table of conflict: produce a table of all the courses, showing the courses that conflict with each.
- Groups: produce a schedule containing groups of courses that do not conflict and are suitable for being in the same interval.

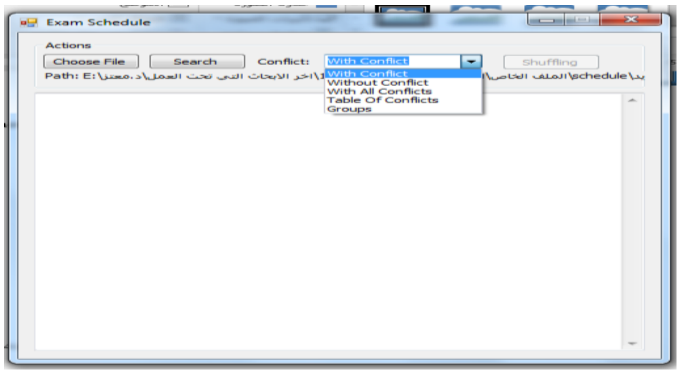
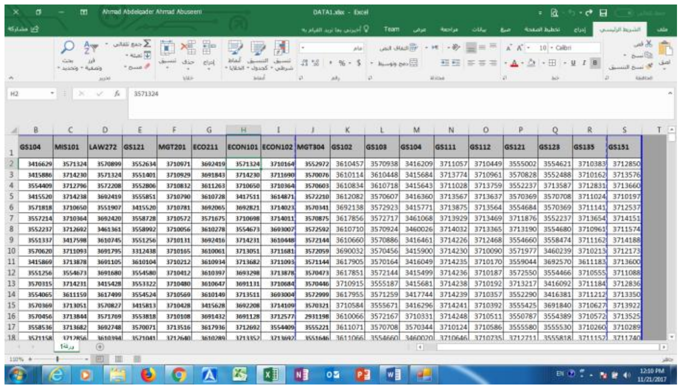


Fig. 3. Five options in a drop-down menu.

The program can import data from an external file, as shown in Figures 4 and 5.



1	GS104	MIS101	LAW272	GS121	MGT201	ECO211	ECON101	ECON102	MGT304	GS102	GS103	GS104	GS111	GS112	GS121	GS123	GS135	GS151
2	3416629	3571324	3570899	3552634	3710971	3692419	3571324	3710164	3552972	3610457	3570938	3416209	3711057	3710449	3555002	3554621	3710383	3712850
3	3415886	3714230	3571324	3551401	3710929	3691843	3714230	3711690	3570076	3610114	3610448	3415684	3713774	3710961	3570828	3552488	3710162	3713576
4	3554409	3712796	3572208	3552806	3710832	3611263	3710650	3710364	3570603	3610834	3610718	3415643	3711028	3713759	3552237	3713587	3712831	3713660
5	3415520	3714238	3692419	3555851	3710790	3610728	3417511	3614871	3572210	3612082	3570607	3416360	3713567	3713637	3570369	3570708	3711024	3710197
6	3571818	3710650	3551907	3415520	3710781	3692065	3692821	3714023	3570341	3692138	3572923	3415771	3713875	3713564	3554684	3570369	3711141	3712537
7	3557214	3710364	3692420	3558728	3710572	3571675	3710698	3714011	3570875	3617856	3572717	3461068	3713929	3713469	3711876	3552237	3713654	3714151
8	3552237	3712692	3461361	3558992	3710056	3610278	3554673	3693007	3572592	3610710	3570924	3460026	3714032	3713365	3713190	3554680	3710961	3711574
9	3551337	3417598	3610745	3551256	3710131	3692416	3714231	3610448	3572144	3610660	3570886	3416461	3714226	3712468	3554660	3558474	3711162	3714188
10	3570620	3711093	3691795	3312438	3710165	3610061	3713051	3711681	3572059	3690032	3570456	3415900	3714230	3710090	3571977	3460239	3710213	3712173
11	3415869	3713878	3691105	3610104	3710212	3610934	3713682	3711093	3571144	3617905	3570164	3416049	3714235	3710170	3559044	3692570	3611183	3713600
12	3551256	3554673	3691680	3554580	3710412	3610397	3693298	3713878	3570473	3617851	3572144	3415499	3714236	3710187	3572550	3554466	3710555	3711088
13	3570315	3714231	3415428	3553322	3710480	3610647	3691131	3710684	3570446	3710915	3555187	3415681	3714238	3710192	3713217	3416092	3711184	3712836
14	3554065	3611159	3617499	3554524	3710569	3610149	3713511	3693004	3572999	3617955	3571259	3417744	3714239	3710357	3552290	3416381	3711212	3713350
15	3570369	3713051	3570827	3415813	3710428	3415628	3692208	3714109	3570321	3710584	3555671	3416296	3714241	3710392	3555425	3691840	3710627	3713922
16	3570456	3713844	3571769	3553818	3710108	3691432	3691128	3712577	2931198	3610066	3572167	3710331	3714248	3710511	3550787	3554389	3710572	3713525
17	3558536	3713682	3692748	3570071	3713516	3617936	3712692	3554409	3555221	3611071	3570708	3570344	3710124	3710586	3555580	3555530	3710260	3710289
18	3521158	3712856	3610394	3571041	3712640	3610289	3713352	3713692	3551646	3611066	3554660	3460020	3710646	3710735	3712711	3555818	3711152	3711740

Fig. 4. Imported data.

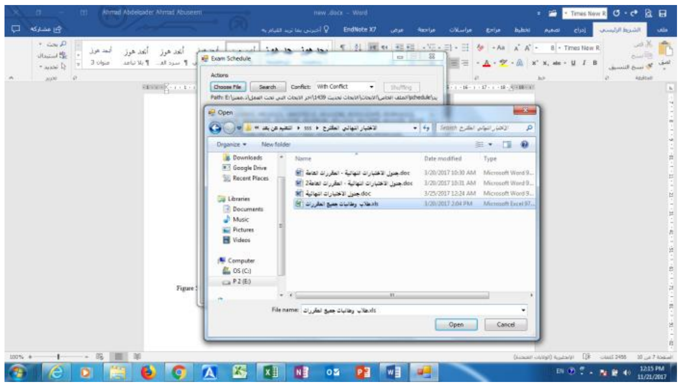
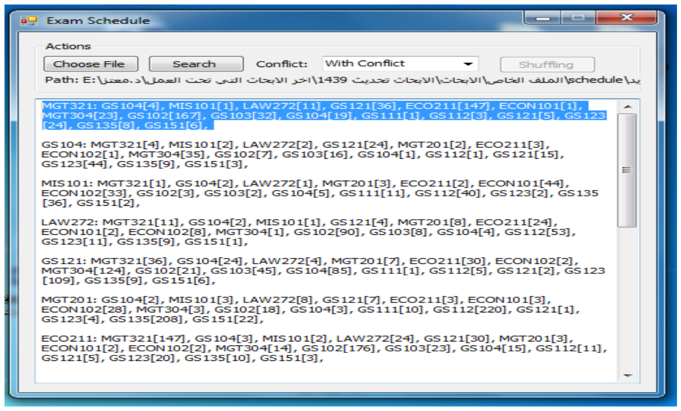


Fig. 5. Importing from an external file.

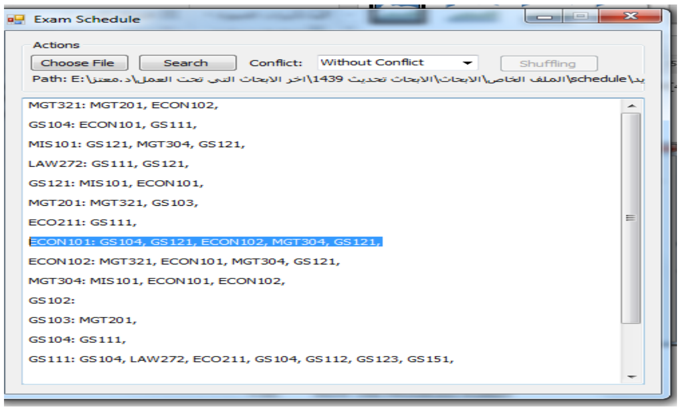
The first option gives the results with conflict ratios; the example shown in Figure 6 (MGT321: GS104[4]) means that the course MGT321 conflicts with the course GS104 for four students, and so on.


Exam Schedule
Actions
Choose File
Search
Conflict:
With Conflict
Shuffling
Path: E:\د.معتز\العمل تحت التي الابحاث اخر\1439 تحديث الابحاث\الابحاث\الابحاث\الملف الخاص\schedule\بد

```
MGT321: GS104[4], MIS101[1], LAW272[11], GS121[36], ECO211[147], ECON101[1],
MGT304[23], GS102[167], GS103[32], GS104[19], GS111[1], GS112[3], GS121[5], GS123
[24], GS135[8], GS151[6],

GS104: MGT321[4], MIS101[2], LAW272[2], GS121[24], MGT201[2], ECO211[3],
ECON102[1], MGT304[35], GS102[7], GS103[16], GS104[1], GS112[1], GS121[15],
GS123[44], GS135[9], GS151[3],

MIS101: MGT321[1], GS104[2], LAW272[1], MGT201[3], ECO211[2], ECON101[44],
ECON102[33], GS102[3], GS103[2], GS104[5], GS111[11], GS112[40], GS123[2], GS135
[36], GS151[2],

LAW272: MGT321[11], GS104[2], MIS101[1], GS121[4], MGT201[8], ECO211[24],
ECON101[2], ECON102[8], MGT304[1], GS102[90], GS103[8], GS104[4], GS112[53],
GS123[11], GS135[9], GS151[1],

GS121: MGT321[36], GS104[24], LAW272[4], MGT201[7], ECO211[30], ECON102[2],
MGT304[124], GS102[21], GS103[45], GS104[85], GS111[1], GS112[5], GS121[2], GS123
[109], GS135[9], GS151[6],

MGT201: GS104[2], MIS101[3], LAW272[8], GS121[7], ECO211[3], ECON101[3],
ECON102[28], MGT304[3], GS102[18], GS104[3], GS111[10], GS112[220], GS121[1],
GS123[4], GS135[208], GS151[22],

ECO211: MGT321[147], GS104[3], MIS101[2], LAW272[24], GS121[30], MGT201[3],
ECON101[2], ECON102[2], MGT304[14], GS102[176], GS103[23], GS104[15], GS112[11],
GS121[5], GS123[20], GS135[10], GS151[3],
```

Fig. 6. The first option, showing the results with the conflict ratios.

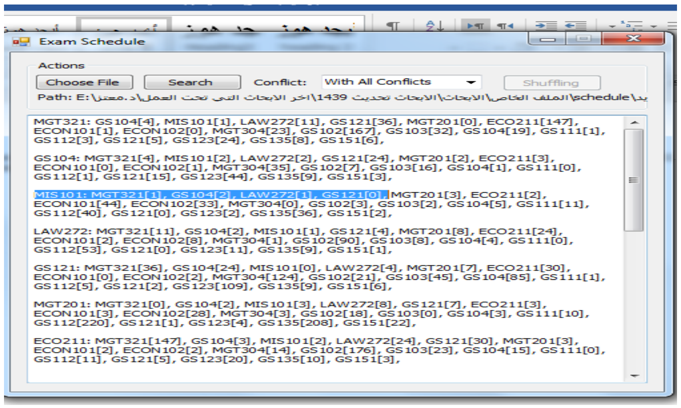
The second option gives the results without conflict; the example shown in Figure 7 (ECON101: GS104, GS121, ECON102, MGT304, GS121) means that the course ECON101 did not conflict with GS104, GS121, ECON102, MGT304, or GS121, so all these courses can be scheduled in the same interval. The result "GS102:" means that the course GS102 conflict with all the courses, so it must be scheduled alone in an interval, and so on.



```
Conflict: Without Conflict
Path: E:\د.معتز\اخر الابحاث التى تحت العمل\1439 تحديث الابحاث\الابحاث\الملف الخاص\schedule\يد

MGT321: MGT201, ECON102,
GS104: ECON101, GS111,
MIS101: GS121, MGT304, GS121,
LAW272: GS111, GS121,
GS121: MIS101, ECON101,
MGT201: MGT321, GS103,
ECO211: GS111,
ECON101: GS104, GS121, ECON102, MGT304, GS121,
ECON102: MGT321, ECON101, MGT304, GS121,
MGT304: MIS101, ECON101, ECON102,
GS102:
GS103: MGT201,
GS104: GS111,
GS111: GS104, LAW272, ECO211, GS104, GS112, GS123, GS151,
```

Fig. 7. The second option, showing the results without conflicts.

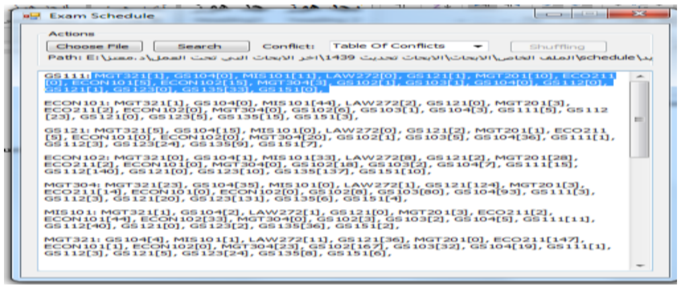
The third option gives the results with all the conflict ratios. If the courses do not conflict, the program gives a ratio equal to zero. The example shown in Figure 8 (MIS101: MGT321[1], GS104[2], LAW272[1], GS121[0], MGT201[3], ECO211[2], ECON101[44], ECON102[33], MGT304[0], GS102[3], GS103[2], GS104[5], GS111[11], GS112[40], GS121[0], GS123[2], GS135[36], GS151[2],) means that the course MIS101 conflicts with the course MGT321[1] for one student, but it does not conflict with the course GS121, and so on.



```
Conflict: With All Conflicts
Path: E:\د.معتز\اخر الابحاث التى تحت العمل\1439 تحديث الابحاث\الابحاث\الملف الخاص\schedule\يد

MGT321: GS104[4], MIS101[1], LAW272[11], GS121[36], MGT201[0], ECO211[147],
ECON101[1], ECON102[0], MGT304[23], GS102[167], GS103[32], GS104[19], GS111[1],
GS112[3], GS121[5], GS123[24], GS135[8], GS151[6],

GS104: MGT321[4], MIS101[2], LAW272[2], GS121[24], MGT201[2], ECO211[3],
ECON101[0], ECON102[1], MGT304[35], GS102[7], GS103[16], GS104[1], GS111[0],
GS112[1], GS121[15], GS123[44], GS135[9], GS151[3],

MIS101: MGT321[1], GS104[2], LAW272[1], GS121[0], MGT201[3], ECO211[2],
ECON101[44], ECON102[33], MGT304[0], GS102[3], GS103[2], GS104[5], GS111[11],
GS112[40], GS121[0], GS123[2], GS135[36], GS151[2],

LAW272: MGT321[11], GS104[2], MIS101[1], GS121[4], MGT201[8], ECO211[24],
ECON101[2], ECON102[8], MGT304[1], GS102[90], GS103[8], GS104[4], GS111[0],
GS112[53], GS121[0], GS123[11], GS135[9], GS151[1],

GS121: MGT321[36], GS104[24], MIS101[0], LAW272[4], MGT201[7], ECO211[30],
ECON101[0], ECON102[2], MGT304[124], GS102[21], GS103[45], GS104[85], GS111[1],
GS112[5], GS121[2], GS123[109], GS135[9], GS151[6],

MGT201: MGT321[0], GS104[2], MIS101[3], LAW272[8], GS121[7], ECO211[3],
ECON101[3], ECON102[28], MGT304[3], GS102[18], GS103[0], GS104[3], GS111[10],
GS112[220], GS121[1], GS123[4], GS135[208], GS151[22],

ECO211: MGT321[147], GS104[3], MIS101[2], LAW272[24], GS121[30], MGT201[3],
ECON101[2], ECON102[2], MGT304[14], GS102[176], GS103[23], GS104[15], GS111[0],
GS112[11], GS121[5], GS123[20], GS135[10], GS151[3],
```

Fig. 8. The third option, showing the results with all the conflict ratios.

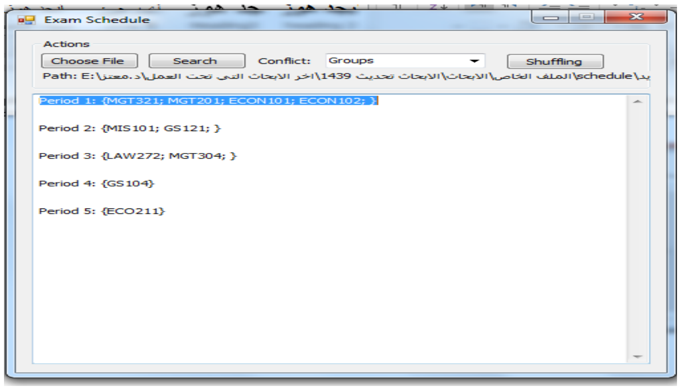
The fourth option gives the results as a table of conflict, as shown in Figure 9.



```
Conflict: Table Of Conflicts
Path: E:\د.معتز\اخر الابحاث التى تحت العمل\1439 تحديث الابحاث\الابحاث\الملف الخاص\schedule\يد

GS111: MGT321[1], GS104[0], MIS101[11], LAW272[0], GS121[1], MGT201[10], ECO211
[0], ECON101[5], ECON102[15], MGT304[3], GS102[1], GS103[1], GS104[0], GS112[0],
GS121[1], GS123[0], GS135[33], GS151[0],

ECON101: MGT321[1], GS104[0], MIS101[44], LAW272[2], GS121[0], MGT201[3],
ECO211[2], ECON102[0], MGT304[0], GS102[6], GS103[1], GS104[3], GS111[5], GS112
[23], GS121[0], GS123[5], GS135[15], GS151[3],

GS121: MGT321[5], GS104[15], MIS101[0], LAW272[0], GS121[2], MGT201[1], ECO211
[5], ECON101[0], ECON102[0], MGT304[20], GS102[1], GS103[5], GS104[36], GS111[1],
GS112[3], GS123[24], GS135[9], GS151[7],

ECON102: MGT321[0], GS104[1], MIS101[33], LAW272[8], GS121[2], MGT201[28],
ECO211[2], ECON101[0], MGT304[0], GS102[18], GS103[2], GS104[7], GS111[15],
GS112[140], GS121[0], GS123[10], GS135[137], GS151[10],

MGT304: MGT321[23], GS104[35], MIS101[0], LAW272[1], GS121[124], MGT201[3],
ECO211[14], ECON101[0], ECON102[0], GS102[8], GS103[80], GS104[93], GS111[3],
GS112[3], GS121[20], GS123[131], GS135[6], GS151[4],

MIS101: MGT321[1], GS104[2], LAW272[1], GS121[0], MGT201[3], ECO211[2],
ECON101[44], ECON102[33], MGT304[0], GS102[3], GS103[2], GS104[5], GS111[11],
GS112[40], GS121[0], GS123[2], GS135[36], GS151[2],

MGT321: GS104[4], MIS101[1], LAW272[11], GS121[36], MGT201[0], ECO211[147],
ECON101[1], ECON102[0], MGT304[23], GS102[167], GS103[32], GS104[19], GS111[1],
GS112[3], GS121[5], GS123[24], GS135[8], GS151[6],
```

Fig. 9. The fourth option, with the results showing a table of conflicts.

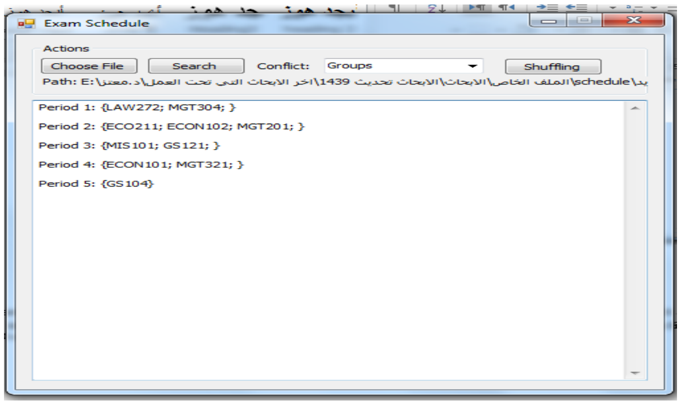
The fifth option produces a schedule containing groups of courses that are not in conflict and are suitable for being in the same interval. The example shown in Figure 10 (Period 1: {MGT321; MGT201; ECON101; ECON102;}) means that the courses MGT321, MGT201, ECON101, and ECON102 can be scheduled in the same interval, which the program calls Period 1.



```
Conflict: Groups
Path: E:\د.معتز\اخر الابحاث التى تحت العمل\1439 تحديث الابحاث\الابحاث\الملف الخاص\schedule\يد

Period 1: {MGT321; MGT201; ECON101; ECON102; }

Period 2: {MIS101; GS121; }

Period 3: {LAW272; MGT304; }

Period 4: {GS104}

Period 5: {ECO211}
```

Fig. 10. The groups of courses that did not conflict.

An important option is the shuffling, which produces another group of courses that are not in conflict and are suitable for being in the same interval. The user can change the groups many times by using the shuffling option. The example in Figure 11 shows the same data as Figure 10 after shuffling.



```
Conflict: Groups
Path: E:\د.معتز\اخر الابحاث التى تحت العمل\1439 تحديث الابحاث\الابحاث\الملف الخاص\schedule\يد

Period 1: {LAW272; MGT304; }
Period 2: {ECO211; ECON102; MGT201; }
Period 3: {MIS101; GS121; }
Period 4: {ECON101; MGT321; }
Period 5: {GS104}
```

Fig. 11. The groups of courses after shuffling.

VII. Conclusion and Future Work

In this study, a program was developed that can build an error- and conflict-free exam schedule. It can import data from other programs such as Excel and gives the results in the form of sets of courses distributed at intervals. The user can change the distribution using the shuffling option until he finds the distribution that fits the exam conditions. This program also gives the user five options for viewing the results, which enables him to obtain more accurate results.

This program can integrate with any academic program used in universities and colleges, and if it is linked with these programs so that it has up-to-date course registration lists, an exam schedule can easily be created that is immediately accessible to all the enrolled students and can solve many problems due to scheduling conflicts.

References

[1] Abuseeni, AA and Abu Sara, MR (2017) A software program to facilitate the construction of exam scheduling in an effective manner. Intl. J. Engineer. Sci. Tech. 9(8):846-852.

[2] Hasan, MA and Hasan, OA (2016) Constraints-aware and user-friendly exam scheduling system. Intl. Arab J. Inform. Tech. 13(1A):156-162.

[3] Babaei, H, Karimpour, J, and Hadidi, A (2015) A survey of approaches for university course timetabling problem. Computers & Industr. Engineer. 86: 43-59.

[4] Cavdur, F and Kose, M (2016) Fuzzy logic and binary-goal programming-based approach for solving the exam timetabling problem to create a balanced exam schedule. Intl. J. Fuzzy Systems, volume 18, issue 1, pp 119–129.

[5] Gonsalves, T and Oishi, R (2015) Artificial immune algorithm for exams timetable. J. Inform. Sci. Computing Tech. 4(2): 287-296.

[6] Hosny, M and Al-Olayan, MA (2014) Mutation-based genetic algorithm for room and proctor assignment in examination scheduling. Sci. Inform. Conf. 260-268.

[7] Kalender, M, Kheiri, A, Ender, A, and Burke, E (2013) A greedy gradient-simulated annealing selection hyper heuristic. J. Soft Computing 17(12): 2279-2292.

[8] Larabi, S and Sainte, M (2015) A survey of particle swarm optimization techniques for solving university examination timetabling problem. Artificial Intell. Rev. 44(4): 537-546.

[9] Lei, Y, Gong, M, Jiao, L, and Zuo, Y (2015) A memetic algorithm based on hyper-heuristics for examination timetabling problems. Intl. J. Intell. Computing and Cybernetics 8(2): 139-151.

[10] Malkawi, M, Al-Haj, HM, and Al-Haj HO (2008) A new exam-scheduling algorithm using graph coloring. Intl. Arab J. Inform. Tech. 5(1): 80-86.

[11] Pillay, N (2014) A survey of school timetabling research. Ann. Operations Res. 218(l): 261-293.

[12] Pinedo, M, Zacharias, C, and Zhu, N (2015) Scheduling in the service industries: An overview. J. Syst. Sci. Syst. Engineer. 24(1): 1-48.

[13] Sabar, N, Ayob, M, and Kendall, G (2009) Solving examination timetabling problems using honey-bee mating optimization (ETP-HBMO). Proc. Multidisc. Intl. Conf. Scheduling: Theory and Applications (MISTA), Dublin, Ireland, 399-408.

[14] Selemani, M, Mujuni, E, and Mushi, A (2013) An examination scheduling algorithm using graph colouring: The case of Sokoine University of Agriculture. Intl. J. Computer Engineer. Appl. 3(1): 116-127.

[15] Soria-Alcaraz, J, Ochoa, G, Swan, J, Caipio, M, Puga, H, and Burke, E (2014) Effective learning hyper-heuristics for the course timetabling problem. Euro. J. Operational Res. 238(1): 77-86.

[16] Thepphakorn, T, Pongcharoen, P, and Hicks, C (2014) An ant colony-based timetabling tool. Intl. J. Product. Econ. 149: 131-144.