Journal of Advances in Mathematics and Computer Science

25(6): 1-12, 2017; Article no.JAMCS.32569

ISSN: 2456-9968

(Past name: British Journal of Mathematics & Computer Science, Past ISSN: 2231-0851)

A Conceptual Approach to Resources Allocation Scheduling

A. A. Eludire[1*] and C. O. Akanbi[2]

[1]*Department of Computer Science, Joseph Ayo Babalola University, Ikeji Arakeji, Nigeria.*
[2]*Department of Information Communication Technology, Osun State University, Osogbo, Nigeria.*

Authors' contributions

This work was carried out in collaboration between both authors. Author AAE designed the study, performed the conceptual analysis and wrote the first draft of the manuscript. Author COA managed the analyses of the study and the literature searches. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JAMCS/2017/32569

Editor(s):
(1) Junjie Chen, Professor, Department of Electrical Engineering, University of Texas at Arlington, USA.

Reviewers:
(1) Mohammed Aboud Kadhim, Middle Technical University, Iraq.
(2) Ankur Singh Bist, Krishna Institute of Engineering and Technology, India.
(3) C. A. Oyeleye, Ladoke Akintola University of Technology, Nigeria.
(4) Shinsuke Tamura, University of Fukui, Japan.

Complete Peer review History: http://www.sciencedomain.org/review-history/22389

Original Research Article

Received: 2nd March 2017
Accepted: 12th December 2017
Published: 20th December 2017

Abstract

The problem of scheduling and allocation of resources, both space and time, has generated a lot of issues and has become of great concerns in daily human activities. The choice of best or appropriate computational algorithms for solving this problem has been the subject of several seminars and conferences organized to discuss the allocation of resources in different fields. The problem of resource scheduling deals with various criteria that are involved in the allocation of resources and associated variation on how the resources are to be allocated. The scheduling of an allocation of resources requires proper management of resources and time for various users to avoid clashes in timing of events. This paper presents a conceptual framework for solving resource allocation scheduling problems using timetabling as a representative scheduling problem requiring proper management of resources and time for various users to avoid clashes in timing of events. The work identifies the constraints involved in algorithms that have been used to solve several timetabling problem like airplane roster, lecture schedules, etc. The paper attempts to identify a good framework for the use of fuzzy algorithm implementation that will be of great value to those resolving timetabling oriented problems.

Keywords: Scheduling; concepts; framework; resource allocation; timetabling; algorithms.

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

1 Introduction

Scheduling problem is an assignment problem, which can be defined as the assigning of available resources (machines) to the activities (operations) in such a manner that maximizes the profitability, flexibility, productivity, and performance of a production system [1]. Scheduling of operations is one of the most critical issues in the planning and managing of manufacturing processes. The literature review indicates that meta-heuristics may be used for the advanced scheduling in manufacturing systems and the genetic algorithm is one of the meta-heuristics that has attracted many researchers. Therefore, one of the main objectives of this paper is to present a survey of the recent developments of evolutionary-based methods for the advanced scheduling.

The review of existing literature showed the availability of special algorithms that will suit the achievement of the objective of this work. A timetable is a placement of a set of meetings in time where a meeting is a special combination or scheduling of resources (e.g. rooms, people and items of equipment), some of which may be specified by the problem, and some of which must be allocated as part of the solution [2]. Resource allocation scheduling has long been known to belong to the class of problems called NP-complete, i.e. no method of solving it in a reasonable (polynomial) amount of time is known.

With respect to timetabling, Wren in [3] defined timetabling as the allocation of subjects to constraints of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible some set of desirable objectives. Various papers in this research area had been published in conference proceedings such as [3] and [2].

The objectives of this work are to study existing scheduling constraints in resource allocation, gather the necessary constraints that will be needed for efficient resource allocation and propose a framework for automated resource allocation according to the constraints gathered or defined.

Timetabling problems are usually subjected to many constraints but for ease the constraints are divided mainly into two according to [3] which includes Hard and Soft Constraints. In this work, after considering the possible constraints, an algorithm was developed for the design of course timetable. The result from the designed timetable was implemented in order to ensure correct or proper co-ordination of scheduling resources in the organisation.

The major problems in timetabling are the occurrence of conflicts in time and place of event during the process of scheduling the available resources and the inefficient use of the resources. The occurrence of conflicts in course timetabling usually is at;

I. Venues where activities clash.
II. Clashes of compulsory courses of the same level at the same period of time.
III. Same lecturer assigned to two or more courses of different level at the same period (hour) of the same day.

Timetabling as a topic has an history of several work done on it especially in the area of the education and of which a perfect solution has not been given simply because there are several constraints that needs to be considered to avoid conflict in resources (e.g. Classrooms, Lecturers, etc.) even after considering what seem to be the complete or perfect constraints.

This paper consists of four sections. Section I presents a general overview of the whole concept of resource allocation scheduling. Section II deals mainly with the literature review of other related works done in the past, also relevant materials are considered with the methodological approach used in carrying out the work.

Section III details the design and implementation segment of the proposed a working framework for resource allocation including the program coding, documentation and other tools used in carrying out the work are shown in this section. The last section presents the conclusion and recommendation for further studies arising from the implementation of the design.

1.1 Resource allocation scheduling constraints

Some views on the term scheduling are found in literatures. Wren [3] emphasized that scheduling often aims to minimize the total cost of resources used, while trying to achieve the desirable objectives as nearly as possible. For example, the process of generating a university course timetabling does not usually involve specifying which lectures will be allocated to a particular course. This information is usually described upon well before the timetabling is actually constructed.

In a school environment, resource allocation problem exist in scheduling a sequence of lectures between teachers and student in a prefixed period of time (typically a week) satisfying a set of constraints of various types.

Resource allocation scheduling is subject to many constraints. These constraints are mainly divided into two main categories according to [2]. Each constraint defines a subset of the set of all possible designs in which it is satisfied. When several constraints are specified, it is only the possibilities within the intersection of all the subsets that we are interested in. The two main classifications are hard constraints and soft constraints.

1.1.1 The hard constraints

Hard constraints are those which we definitely want to be true or rigidly enforced when considering resource allocation constraints. Each hard constraint, is an element of the timetable that is equally weighted and must be abided by. Simply put – if they are broken, the timetable will not work. Examples of such constraints are:

i. No resources (Student or staff) can be demanded to be in more than one place at a time.
ii. For each time period there should be sufficient resources (e.g. lecture room, lecture etc) available for all the events that have been scheduled for that time period.

1.1.2 The soft constraints

The soft constraint are those we would like to be true or desirable - but not absolutely essential at the expense of the others. The soft constraints, are the other side of the coin and are made up of the elements which can turn or enhance a working timetable into a great timetable. Examples of soft constraints are:

i. Time assignment:- An event may need to be scheduled in a particular period of time
ii. Time constraint between event:- One event may need to be scheduled before / after the other
iii. Spreading Events out in time: - Users should not make use of resource in consecutive period or two resources on the same day.
iv. Coherence: - Resources may be used in a number of days consecutively in order to have a number of free days.
v. Resources assignment: - Resources may be preferably used in a particular location or venue.
vi. Room Capacity: - the number of users in a location/venue should not exceed the venue's capacity.

The basic quality of these feasibility studies can be accessed on the basis of how the constraints are satisfied. However, some problems are so complex that it will be difficult to find a feasible solution. The resource allocation scheduling process is made difficult by the fact that so many entities are affected by the outcome.

Romero [4] identified three main stakeholders in this process each with their own set of aims and wants.

i. The administration sets the minimum standards that the timetable must conform to for example, some universities specify that no student should have to take two exams in consecutive periods.

ii. The department concerns are more likely to be prominent in the course timetable. They will want the “schedule to be consonant with the development of the subject taught” as well as making specific demands for particular classrooms or labs. In an examination context, they are likely to request that large exams be placed early in order to allow more time for marking.
iii. The third groups of stakeholders are the students, whose view of the timetable will be restricted to the parts that affect them. Given the number of students involved, it is difficult to obtain specific criteria as to what is the best timetable for students. Many student prefers not to have lectures on Friday, and to have a break between consecutive courses/exams.

The set of constraints considered earlier varies significantly from university to university due to several feasibility researches. The overviews of different constraints imposed by most universities are stated in literature [5,2].

2 Materials and Methods

2.1 Methodology of conceptual framework for models

Resource allocation scheduling problems at many organisations share characteristics with the general models. Consequently, attention will be focused more on the type of decisions included in this model than the institution involved. Although, the objective of this work is to determine an acceptable resource sharing, the criteria for an acceptable resource scheduling are usually modelled by means of constraints and objective function. This then reflect the division of criteria into requirements that must be fulfilled as well as possible.

There are basically two types of resource allocation scheduling models according to [1] which are Dynamic and Functional models.

2.1.1 Dynamic model

Dynamic models are expressed in such a way that at the beginning of the scheduling time, relevant events are collected and assigned venues and activities period as depicted in Fig. 1.

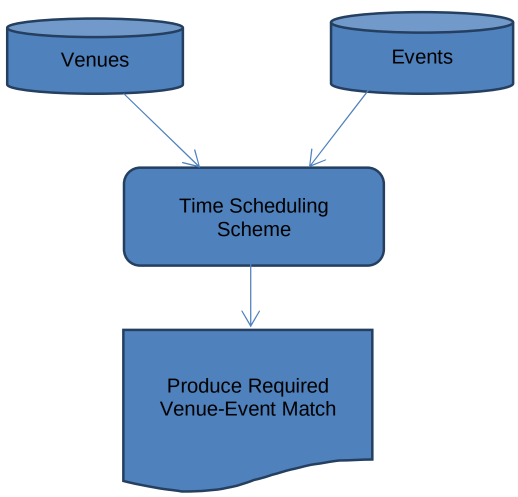

Fig. 1. Dynamic model diagram

2.1.2 Functional model

The functional or generic model was proposed by [6] for timetabling problems consisting of a set of resources, a set of activities, and a set of dependencies between the activities.

According to this model, time is divided into the time slots with the same duration. Every slot may have been assigned constraints either hard or soft one: a hard constraint indicates that the slot is forbidden for any activity, while a soft constraint indicates that the slot is not preferred. This type of constraints is called "time preference". Every activity and every resource may have been assigned a set of time preferences, which indicates forbidden and not preferred time slots.

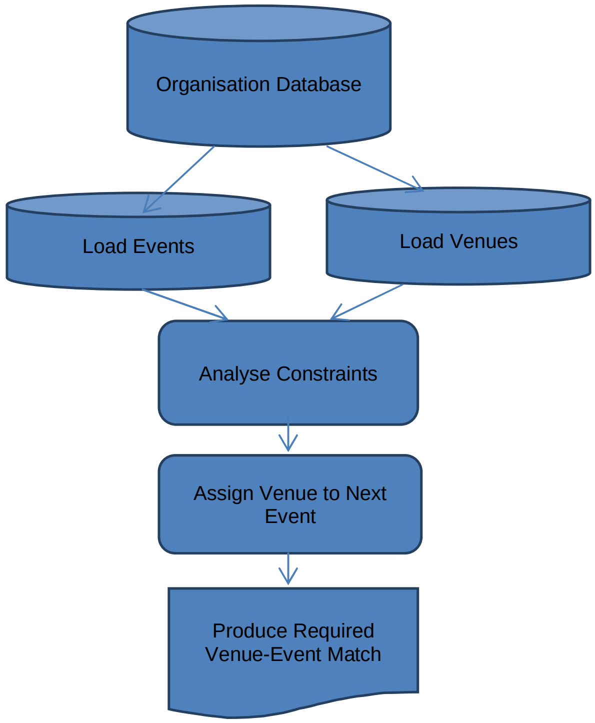

Fig. 2. Functional model diagram

Activity (which can be directly mapped to a lecture) is identified by its name. Every activity in this model is described by its name. Every activity in this model is described by its duration (expressed as *s* number of time slots), by time preference, and by a set of resources. This set of resources determines which resources are required by the activity. To model alternative as well as required resources, we divide the set of resources into several subsets-resources groups. Each group is either conjunctive or disjunctive. The conjunctive group of resources means that the activity needs all the resources from the group, the disjunctive group means that the activity needs just one of the resources (we can select among the alternatives). An example can be a lecture which will take place in one of the classroom and it will be taught for all of the selected classes. However, usage of both conjunctive and disjunctive groups simplifies modelling for the users. This assist in generating a functional dependency diagram where the realization of one event is dependent on the availability of a previous event as depicted in Fig. 3.

The example of the problem defined by the above model is a course timetable, where every scheduled activity is assigned its start time and a set of reserved resources, which are needed for its execution (the activity is allocated to respective slots). The timetable model proposed by Muller in [7] satisfies all the hard constraints namely:

i. Every scheduled activity has all required resources reserved, i.e. all resources from the conjunctive groups and one resource from each disconjunctive group of resources.
ii. Two scheduled activities cannot use the same resource at the same time.

iii. No activity is scheduled into a time slot where the activity or some of its reserved resources has a hard constraint in the time preferences.
iv. All dependencies between the scheduled activities must be satisfied.

In the majority of these models, the problem under consideration is decomposed into course timetabling. In further consideration we then have the student sectioning sub-problem.

In [8] students were first assigned while roughly minimizing the number of lectures that are manually conflicting, then a timetable can be constructed and finally the section assignment is then revised.

In [9], sections which are taught simultaneously are first determined and then student sectioning is carried out. This then made it possible to start with the design of course timetabling and improve solving the student sectioning sub-problem for each timetable they examine.

In [1] students were first assigned to sections and then tried to cluster sections that are taught at the same time.

In practice, each organisation will have a different way of evaluating the quality of developed timetable in many clauses. The quality is calculated based on a penalty function which represents the degree to which the constraints are satisfied.

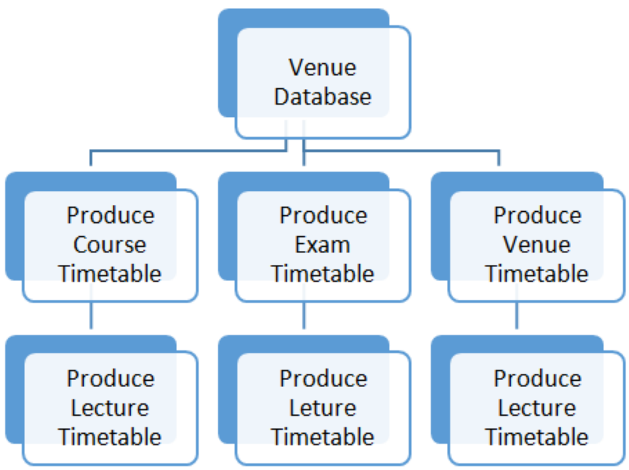

Fig. 3. Functional dependency diagram

2.2 Course timetabling problem statement

From the review of existing literature the algorithms that will suit the achievement of the objective were identified. During the construction of timetable, there is a need to take different constraints into account simultaneously. These constraints may be conflicting in the sense that an attempt to satisfy one of the constraints might lead to the violation of another. For example, suppose there is a situation where trying to schedule two lecturers which needs to be scheduled immediately before/after each other. Suppose also that there is a constraint that requires an even spread of lecture period, in such a situation, it can be seen that satisfaction of the first constraints can probably lead to the violation of the second. In addition, the violations of different constraints are measured by different unit with different scale (the number of students involved in the violation, the number of courses involved in the violation etc.).

Such observations motivated a multi-criteria statement of timetabling problems where criteria measure the violations of the corresponding constraints. A university may access the quality of a timetable from very different points of view (expressed by departments, students, a timetable officer etc.). Most departments often assign importance to the constraints that are imposed on the timetabling problem. For example, a department may have strong demand for a specific classroom for certain course, while students are usually concerned with the order in which courses are scheduled and their proximity, etc.

2.3 The algorithm

Over the years, numerous approaches have been investigated and developed for resource allocation , such approaches includes Constraints programming, Graph colouring, and various meta-heuristic approach which include Direct heuristic approach, Genetic Algorithm, Tabu-search, simulate-annealing, the Great Deluge Algorithm. Some recent important research works that reflect these broad ranges of activities are [10] with [11,12].

Heuristic approaches has been used in trying to solve the issue of scheduling problems using an inductive reasoning by evaluating past experience and moving trial and error to a solution as shown in [13].

Direct heuristic approach is based on successive augmentation. Using direct heuristic approach a partial timetabling is extended until all lectures are scheduled. The most important aspect of this approach is to schedule the most constrained classes first until all classes are scheduled. Direct heuristics usually fill up the complete timetable with one lecture or group of lecture at a time as far as no conflict arises. At the point they start making some swapping so as to accommodate other lectures. A typical example of this method is the system SCHOLA described in [14]. The system is based on the following three strategies;

I. Assign the most urgent lecture to the most favourable period for that lecture.
II. When a period can be used only for one lecture, then assign the period to that lecture.
III. Move an already-scheduled lecture to a free period so as to leave the period for lecture that we are currently trying to schedule.

The system SCHOLA schedules the lectures alternating strategies I and II as much as possible. When no more lectures can be scheduled in this way, it starts using strategy III.

Genetics Algorithm (GAs) is analogous to Darwinian evolution. A “Population” of feasible timetable is maintained. The “Fittest” timetable (those with the lowest penalty values) are selected to form the basis of the next iteration. The most common genetic representation for its timetable is a long bit string encoding when and where each meeting is to take place, also approaches which order courses prior to assignment of a timeslot has been discussed by several author as mentioned earlier and also including [14] and [15].

Fuzzy Methodologies has been successfully applied in a wide range of real world application since it has been introduced in 1965. Example of scheduling, planning and timetabling problem domains include fuzzy evaluation functions utilized in generator maintenance scheduled [7] and who used fuzzy target gross sale to find optimal solution to manpower allocation problems.

Fuzzy methodology is an algorithm that has been investigated for other timetabling problems such as aircrew rostering. Furthermore, fuzzy technique can represent and deal with multi-criteria decision making described by [16,17]. As far as we are aware, fuzzy method have not yet been implemented in the context of examination timetabling.

In this research a fuzzy system can be used to rank courses based on an assessment of how difficult it is to schedule the course taking into account multiple criteria.

By considering more than one criterion to run the courses, ranking are produced better by reflecting the actual difficulty of placing the courses as several factors are simultaneously taken into account.

2.4 Generalised algorithm

As it was stated in the beginning of this work, a set of constraints must be satisfied in order to have a valid timetable. The number and the type of constraints vary from organisation to organisations, and it also varies from one department to another. As it can be easily seen, to get a complete solution for a particular timetabling problem with all the constraints satisfied is very difficult, possibly sometimes, even impossible to accomplish [18]. A generalised algorithm is proposed as follows:

1. Determine the total number of Level to be considered
2. Determine the total number of courses to be assigned
3. Pick a course based on stated conditions
4. Check the number of possible slots possible
5. Run the constraint(s) on the course(s)
6. Save the result in a target table.
7. Check if a slot is still remaining or finished.
8. If still remaining run (5) else run (9)
9. Check if all course(s) have been assigned
10. If not GOTO (2)
11. STOP

2.5 Information flow in timetable production

Firstly, the process of timetable production can be defined as a process that starts when the data for the timetable period p of validity are entered. The period p is a semester, period p data are essential to the construction of a timetable for a given period. They contain courses offered during that period, the availability of lecturers for that period etc. These data are prerequisite for timetable construction, and must be entered for each construction.

The process ends when all the concerned people receive the schedules on paper and the period of the validity of the schedule comes to an end. Multiple data will be entered into the system, for example, courses offered during period p, format of courses (2 hour or 2 hour + 1 hour or 3 hour), assigned or potential lecturers for certain courses, assigned or potential or class rooms, availability of lecturers and classrooms, etc. It was found that in some institution pre-selection was used while in others the timetables were made with no pre-selection.

3 Framework System Design

This comprises of several Forms, Modules, Database and Software which will work in autonomous and coordinated way or pattern. This system includes two main components which are ability to save data and the timetable construction software.

3.1 Data

The data required to produce the timetable were provided. All necessary data were saved in a data back-up system. It was indicated that modules of the timetable production system can handle data. For example, the pre-choice acquisition modules will save the courses offered by students, data construction module then take its inputs from the database, and when construction are performed, the data obtained are saved in a data warehouse. Once the timetable had been created, the output of the system will be saved in a file in order to output to appropriate media. The data flow process is shown in Fig. 4.

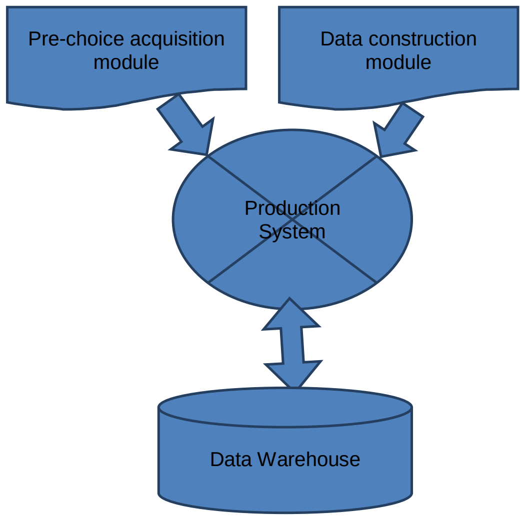

Fig. 4. Data flow implementation

3.2 The architecture

Further to the Implementation of data flows in timetable production, architecture for implementing the system was developed. Abstractly speaking, software architecture describes the elements of a system. It also shows the interaction between these elements, the model governing its composition and constraints of these models [10].

Generally, when facing a complex problem, the best approach is to break it down into parts that become easier to solve with simple solutions, then all these small solutions were combined to allow finding solution to complex problems [4].

Of all architecture now available for the development of software productions, the one most appropriate to timetable production system is that in layers. The architecture shown in Fig. 5 divides the system into three layers, each one with a well-defined function.

- **The Interface:** This Layer presents the data to the user in order to allow data input and ensure exchanges with other layer.
- **The business logic Layer:** This layer ensures data exchanges with the interface layer, checks and validates the data input and sends the data to be presented in an adequate format. It also ensures data exchanges with the data persistence layer. Business rule are used to ensure the coherence of the system.
- **The data persistence layer:** This layer manages the physical storage of the data in files with a certain format, or in a traditional database (Microsoft Access/MySQL) system, or in other persistence models able to manage complex databases.

3.3 Instances of the architecture

There are three options of architecture instances stated (see Table 1). For the implementation, the third instance (i.e. Instance 3) in Table 1 was chosen.

Using Instance 3, element are opened if built approve standards, or if built with a private specification which may be made public by the developers.

Table 1. Possible instances of architecture

Technology	Instance 1	Instance 2	Instance 3
Language for the business logic	Java	Java /Applet Java	Visual Basic
Logic	———	———	———
Data	Database	Database	Database

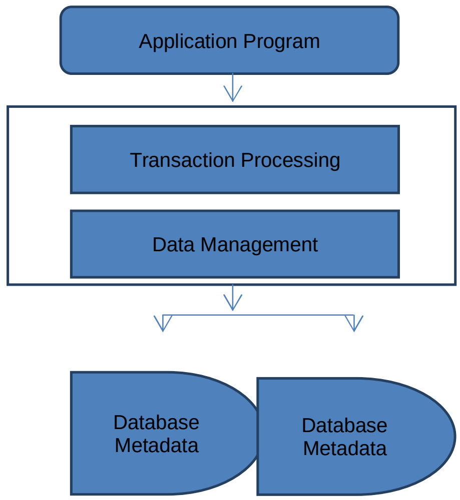

Fig. 5. System architecture

It is important to consider that an element is extensible if it is easily adaptable to specific changes [19,20]. Bearing this consideration in mind, there are analytical explanations that prompted the third Instance architecture as one of the choices. These are:

- **Language to Code the Business Logic:** The Microsoft Visual Basic was chosen. Visual Basic (VB) is an interpreter / compiler language that uses the runtime Libraries of windows operating system.
- **Data persistence:** For the Implementation of the data persistence Layer, a relational database was considered. A database is an entity in which it is possible to store data in a structured way with minimum redundancy. This data must be used by the programs written in different languages by resorting to ODBC standard technology (Open Data Base Connectivity).

3.4 Technology requirements

To implement the architecture, the technological elements required were first considered. The needs are as follows:

- A language to facilitate data display by the user, and the creation of forms to enter data.
- A set of rules to define how data exchanges are established with other systems.
- A language to code the business logic.
- A tool to manage data persistence.

Through the analysis, it was found out that there was more than one option. There are actually several ways of implementing this architecture. Table 1 shows an open-ended list of possible instances.

For the construction of each instance, we put together the element sharing common characteristics, or belonging to the same software family, and are open. For the database section, there is no need to mention any specific name of a database management system. The choice remains open to the designer but in this work MySQL database was used.

The architecture is an open one since it accepts several instances. The architecture is also extensible, because the module could easily be changed [21]. For example, one can start with an element of data persistence like a file in text format; also it would be necessary to make changes in the interface layer and to program the data persistence element.

4 Conclusion

This work has been able to achieve its major objective which is the proposal of a conceptual framework for the design and implementation of resource allocation scheduling. It used the identified concepts to build a foundation for the generation of timetable for a department of an educational system. It also gives opportunity to view some resources allocated and those that are yet to be allocated.

The needed future works include implementation of these framework concepts for the grouping of resources when there are no venue capacities to handle such resources. A web implementation of the framework is highly desirable and also the development of other constraint-based scheduling activities like examination timetable could be incorporated into it.

The following are strongly recommended to the proper usage of the framework:

I. All system requirements must be strictly satisfied.
II. All necessary information must be provided before attempting to generate needed timetable.
III. Program must be properly installed by following the list of instructions provided in the course of installation.
IV. Only authorized user should be allowed in the management of the timetable software.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Prakash A, Felix T. S. Chan, Deshmukh SG. FMS scheduling with knowledge based genetic algorithm approach. Expert Systems with Applications. 2011;38:3161–3171.

[2] Gonzalez, R. Rubio. Generating university timetable in an interactive system: DIAMANT. In PATAT 2000, Konztanz, Germany. Pattern-Oriented Software Architecture. A System of patterns, Wiley, London. 2000;1.

[3] Wren A. Scheduling, Timetabling and Rostering. A special relationship, in (Burke and Ross). 1996a;46-75.

[4] Carter MW, Larporte G. Recent developments in practical course timetabling in (Burke and Carter) 1998;3-19.

[5] Edmund Kieran Burke, Sanja Petrovic. Automated scheduling, optimization and planning (ASAP) Research group school of computer science and information technology university of Nottingham; 2000.

[6] Romero BP. Examination scheduling in large engineering school: A computer – assisted participative procedure. Interfaces. 1995;12(2):17-23.

[7] Boizumault P, Delon Y, Peridy L. Constraint logic programming for examination and course timetabling. The Journal of Logic Programming. 1996;26(2):217-133.

[8] Carter MW, Laporte G, Lee SY. Examination timetabling and algorithmic strategies and Application of the Operational Research Journal. 1996;47:372-383.

[9] Thomas Muller, Roman Bartak. Interactive timetabling in proceedings of ERCIM workshop on constraints, Prague. 2001;4-5.

[10] Buschmann F, Meunier R, Rohnert H, Sommerlad P, Stal M. (1996). Timetabling in encyclopedia of operation research and management science. Gass S. and Harrris C., Eds. Kluwer Academic Publishers. 2001;833-836.

[11] Shaw M, Garlan D. Software Architecture on an emerging discipline. Prentice Hall London; 1996.

[12] Thompson JM, Downsland KA. A robust simulated annealing base examination timetabling system. Computers and Operations Research. 1998;637-648.

[13] Deris S, Omatu SH, Saad P. Incorporating constraints propagation in generic Algorithm for University Timetabling Planning; 1999.

[14] Owoseni BO. Generating University timetable system; 2000.

[15] Junginger W. Timetabling in Germany – a survey. Interfaces. 1986;16:66–74.

[16] Müller T, Rudová H, Barták R. Minimal perturbation problem in course timetabling in E. Burke and M. Trick (Eds.): Practice and Theory of Automated Timetabling V: 5th International Conference, PATAT 2004, Pittsburgh, PA, USA, August 18-20, 2004, Revised Selected Papers, LNCS. 2005; 3616:126-146.

[17] Fleurent C, Ferland JA. Genetic and hybrid algorithm for graph coloring. Technical Report - Université de Montréal; 1994.

[18] Raj PA, Nagesh Kumar D. Ranking alternatives with fuzzy weights using maximizing set and minimizing set. Fuzzy Sets and Systems. 1999;105:365–375.

[19] Joseph Y-T. Leung. Handbook of scheduling: Algorithms, models, and performance analysis. Chapman & Hall / CRC, Boca Raton-London-New York; 2004.

[20] Bertrand Meyer. Object oriented software construction. Prentice Hall, 2nd Edition, London; 1997.

[21] Schaerf A. A survey of automated timetabling. Artificial Intelligence Review. 2000;13(2):87-127.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://sciencedomain.org/review-history/22389