JOURNAL OF SCIENCE TECHNOLOGY AND EDUCATION 9(3), SEPTEMBER, 2021
ISSN: 2277-0011; Journal homepage: www.atbuftejoste.com

Development of an Integrative Timetabling IS for Generation and Management using Genetic Algorithm

Bara'u Gafai Najashi, Oluwatobi Samuel Sholanke
Department of Electrical & Computer Engineering,
Baze University, Abuja, Nigeria

ABSTRACT
The timetable generation problem, as reported time and time again in the literature, has been solved using several techniques. Such claims lead to a benchmark that is proposed to enable an initial comparison of the effectiveness of proposed solutions by different researchers. Despite over sixty years of research conducted into this area, there still exist a significant number of research avenues still to be pursued. In this work, an information system based upon the need to support timetable production and maintenance is presented. Given the very practical outcomes expected of timetable research, the information system was designed to enable the whole range of administrative functions performed by lecturers to be either directly supported or readily modified to prove such support. The implementation of this particular system is given and results are presented and discussed. The system generated manual and automated timetables and these were produced by trailing several objective functions. It was noted that the determination of the optimal objective function is dominated by specific individual institutional criteria. It is suggested that this would make a more than a significant project for future information systems research.

ARTICLE INFO
Article History
Received: June, 2020
Received in revised form: July, 2021
Accepted: October, 2021
Published online: December, 2021

KEYWORDS
Timetabling, scheduling, genetic algorithm

INTRODUCTION

For several decades, the area of scheduling has been a prominent domain with various interesting research topics. Although the term scheduling is mostly used generically to refer to both scheduling and timetabling, however, they are viewed as two separate activities; scheduling typically is used when dealing with some special type of problems. Thus, the development of timetable algorithms can be regarded as a special case of a typical scheduling activity.

Generally, we can define scheduling as the process of allocating resource objects to time-space while satisfying some constraints to minimize wastage of resources to the barest minimum [1]. Timetable construction can then be defined as the allocation of the subject of given resources to time-space objects which must satisfy some constraints to reach satisfaction or near-satisfaction [1]. Examples of these kinds of timetables are lecture and examination timetables because all hard constraints must be satisfied to arrive at a reasonable solution.

Thus, the term scheduling covers all aspects of the activity of allocating resources and, at the same time, satisfying some predetermined objective. However, due to the enormity of the problem, it becomes necessary to classify the scheduling problem into specialized activities such as timetabling. Consequently, the timetabling problem can be said to be the process of scheduling a group of lectures between lecturers and students in a fixed period and at a particular location, without violating some set of constraints [2]. Because of the nature of the timetabling problem, its requirements differ from one university to another; thus, a lot of models were been developed to deal with the problems [3]. Some of these problems include the construction of semester or annual timetables and examination timetables at the end of each semester in universities.

Human approaches in solving timetable puzzles were simulated and applied as techniques to the timetable problem during the early days. These included techniques known as direct heuristics which are some forms of sequential reinforcement. The idea

Corresponding author: Bara'u, G. N. ✉ *oluwatobi2528@bazeuniversity.edu.ng* ✉ *Department of Electrical and Computer Engineering, Baze University, Abuja.*

behind these techniques is to create a partial timetable by scheduling the entire lectures one after the other, starting from the most constrained lecture to the least constrained lecture [2]. Afterwards, researchers were then applying some techniques such as integer and linear programming, graph colouring and network flow for solving the timetable problem. Consequently, the first two papers published on timetable construction using these general techniques are generally attributed to [4] and [5].

From the literature, there emerged a trend that in the last decades the topics of timetabling were mainly confined to the Operational Research community. The early techniques used were purely mathematical, of which linear programming (LP) and integer programming (IP) were the two most common ones. Recently, the advent of artificial intelligence brought about a much better approach to solving the timetable problems, modern heuristic algorithms such as Genetic Algorithms (GA) [6][7], Simulated Annealing (SA) [8][9][10] and Tabu Search (TS) [11][12] were introduced.

Constraint-based reasoning has proven to be a productive research method for researchers in the areas of artificial intelligence, operational research and logic programming [13]. Gradually, a theoretical framework called Constraint Satisfaction Problems (CSP) has evolved [14]. The process of Constraint Satisfaction involves finding values for variables within the framework of the problem, subject to constraints on acceptable combinations of values. As in some cases, it is impossible to solve the problem completely, a subset of the problem is solved that is called partial constraint satisfaction [15]. CSPs are solved by different versions of a backtracking search that attempts to improve the search efficiency and generally deals with the binary CSPs [16]. The CSP approach allows for greater flexibility in the formulation of the problem, which is broadly recognized as a crucial issue in generating successful timetables. In most cases, it is difficult to define the constraints of the problem as they are unclear and require many refinements and interactions with the user. This level of flexibility allows CSP systems to address at a practical level the problems that are encountered at the implementation stage. Furthermore, due to the very nature of CSP definitions, the problem of uneven time-slots can be easily accommodated.

In this work, a design and development of an integrative timetabling IS for the generation and management of university timetables is presented. This included support for both automatic and manual development of semester timetables. The system developed would allow for plug and play of any solution generation technique into the IS. A constraint-based technique has been adopted but this could easily be substituted with a Genetic Algorithm (GA) or Simulated Annealing (SA) algorithms. The rest of the paper is organized as follows; Section II discusses the timetabling problem in Baze University, Section III provides the materials and methods used in solving this problem. Section IV provides the platform and procedure used for generating the timetable; results and discussion are in section V while section VI is the conclusion and recommendation.

TIMETABLING PROBLEM

Several works have been presented in the literature in the implementation of various software to test cases of different university settings. However, changes in operations from one university to another constitute a considerable gap between a large number of the implemented solutions and the requirements needed for an optimal or near-optimal timetable construction. This gap can be associated with unforeseen problems but is usually encountered in reality.

Some of the problems documented in Baze University timetabling are:

1. Students' enrolments for courses are usually changing sometimes up to 3 to 4 weeks into the semester, which can cause clashes that may require the reconstruction of the entire timetable.
2. The availability of sessional lecturers (otherwise known as visiting lecturers), who generally have to juggle between two or more jobs, can provide an extra complexity to the timetable problem even after it has been generated.
3. The majority of the implemented solutions do not provide relaxation periods between successive lectures for both students and lecturers. This is most cases leads to unproductive learning, especially in a situation

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

where the students or lecturers have more than two lectures in a row consecutively.

4. Types of the venue is another problem, most of the algorithms do not have techniques to check whether a subject is fit for a venue. This can also cause a dissatisfied solution because a practical based subject that is supposed to hold in a laboratory can be assigned to a non-laboratory venue, which will require a manual re-allocation of venue.
5. Muslims' Friday Prayer time is another observed problem that needs to be resolved; especially in Nigeria where half of the citizens are Muslims [17].
6. A lot of the implemented solutions assigned venues from any block in the university randomly. This becomes an issue when a subject that belongs to a group of students and lecturer in a certain faculty block is randomly allocated to hold in another faculty block.
7. Another problem that complicates the timetable problem is the university mode of admission. This is whether the university admits students on a semester basis or a sessional basis. Universities that admit students on a semester basis will have a more complicated timetable because for every semester in these universities, there are two types of students – those in their first semester and those in their second semester.

Therefore, to generate a more realistic and effective timetable it needs to be flexible enough so that it can facilitate and overcome the problems listed above as they occur.

MATERIALS AND ANALYSIS TECHNIQUES

The Timetabling Problem at Baze University

At the time of this work, Baze university is comprised of six faculties with a total of thirty-four departments in all. These faculties sometimes share lecturers, classrooms and modules, which divides the timetabling process into levels: creating a valid timetable for every level in each department of each faculty and eliminating conflicts between all the faculties. The timetabling process is centralized at the university i.e. all faculties send modules' schedules to the IT department where the final schedule is done through a timetable management system built by the IT department. Because the system schedules classes manually, conflicts occur between modules sometimes, which takes a lot of time for operators to fix due to the complexity of the problem[28].

Some other factors that complicate the timetabling process at Baze university are visiting lecturers' issues and admission methods. A visiting lecturer is a lecturer from another university who visits a host university (Baze University) to teach at least one module on a particular day(s). Sometimes these visiting lecturers' modules are scheduled on days that they are not available on campus resulting in wrong scheduling[27].

Students enrolment for courses is done every semester in the university, the effect of this on the timetable is that there are two sets of levels for every department in each faculty at every given time which multiplies the complication by two. For example, for every semester there are two sets of 500 level students in the Department of Computer Engineering – those in the 1st semester and those in the 2nd semester.

In an attempt to solve the timetable problem at Baze University we developed a prototype of a system that automates the scheduling process as well as provides functions for management of the entire university timetable. This system is tested with model data of student enrolment from the department of Computer, Electrical & Electronics and Telecommunications Engineering.

Materials

Because of the nature of the problem at hand, which involves manipulations of the huge amount of data, stringent data connectivity and analysis we have chosen Python as our primary development language and web as a platform for easy accessibility.

Python

Python is one of the leading programming languages, it is a general-purpose object-oriented language and thus, it is widely used for several applications [18]. Python comes with a lot of useful and

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

interesting features, this includes dynamic binding and typing, a large standard library as well as features that make it very useful for solving complex problems easily.

Another interesting thing about python is its extensibility feature, it allows for system calls to almost all operating systems and it is easily integrated with other languages such as C, C++, Java etc. It has a huge community that provides unlimited supports and its syntax is very easy to use and learn, consequently, is it the best choice for a rapid development project.

Django

To develop a web application with python, a framework is needed. Django is one of the web frameworks used for python web application development. Django is a powerful python framework, that is used for developing python web applications. It is licensed under the Berkeley Software Distribution (BSD) License, and a registered trademark of the Django Software Foundation. It encourages rapid development with a clean professional-looking design. Like other modern web frameworks such as Laravel (for PHP) and Express.js (for Node.js) that support the MVC architecture, Django also supports the architecture but with a slight difference [19].

MVC Architecture

The Model-View-Controller (MVC) pattern is a paradigm for building applications with rich user interfaces (UI) by dividing related modular logic into three interconnected components – the model, the view and the controller. The basic importance of this is to completely separate the way information is presented to the user from the logic behind the presentation [19].

DJANGO MVC - MVT Pattern

As mentioned earlier, Django uses the Model-View-Template (MVT) pattern which is slightly different from the MVC pattern. The only difference is a change of the Controller to Template, Django itself controls the interactions between the Model and View which is the job of the Controller and leaves software developers with only the template [19]. The Template is mainly an HTML file improved with Django Template Language (DTL) for information presentation.

So what developers are left to do is to provide the model, the view, the template, and map everything to a URL for server request and Django does all the processing in-between to get the information presented to the user.

The diagram below shows how the components of the MVT architecture work together to serve a user request:

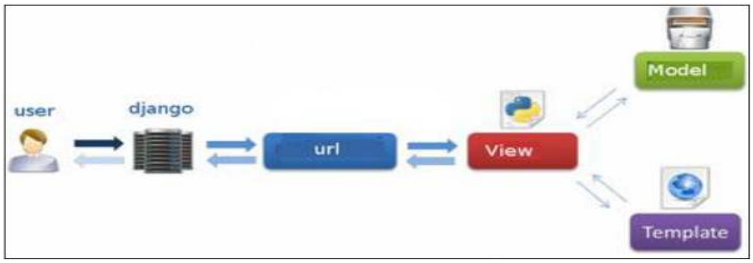

Figure 1: Components of MVT Architecture

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

Database

A database is a collection of data. It organizes and stores data (such as Boolean, integers, strings, floating-point data etc.) in well-structured tables and provides at least one unique application programming interface (API) for manipulating the data – such as creating, accessing, updating, and deleting and other database operations. Files can also be stored in a database, examples are files on the file system or large hash objects in memory; however, accessing and manipulating these types of data may not be fast and sometimes requires some other application/API for handling [20].

In reality, data are usually related to one another, a typical example of this is a situation where all students' results data in one table of a database are related to each of the students in another table of the same database by student registration number. Thanks to the type of database system called Relational Database Management Systems (RDBMS).

RDBMS provides powerful functionalities for storing and managing a huge amount of data. Data are stored in different tables with the well-established relationship between the tables through their primary keys (the primary key of a table is referenced as a foreign key in another table) [21]. It provides data integrity between the rows and columns of various tables and performs automatic updates on all the indexes. It also provides a command-line for Structured Query Language (SQL) query execution. Examples of RDBMS are Oracle-DB, SQLite, PostgreSQL, MySQL DB, Microsoft Access DB etc. For further details on the Database and especially MySQL used here, kindly refer to [21].

Python Pandas

Python Pandas or simply Pandas is an open-source python library licensed under the BSD-License that provides high-performance data management and analytical tools using its powerful data structures.

In 2008, McKinney started the development of pandas after noticing the need for it while working on a large scale data project [22]. He soon named it Pandas after the word Panel Data (also called longitudinal data) – a multi-dimensional data in statistics and econometrics [23].

Before then, python was not famous for analysis because data wrangling and preparation was majorly done using pure python, which was a really difficult task. The advent of pandas brought the solution to this problem[27]. With pandas, data processing and analytical operations such as loading, preparation, manipulating, modelling and analysing are now very easy to perform regardless of the data origin [23].

Analysis Techniques

The procedures for analysis, design and implementation of an information system are well-known. The systems development life cycle is considered to be the set of activities that analysts, designers, and users carry out to develop and implement an information system. In most situations, all activities are closely related and often inseparable. Even the order of the steps in these activities may be difficult to determine. Different parts of a project can be in different phases at any one point in time [24].

Generally, SDLC methodology helps designers and developers to avoid any flaws in the final system. The techniques used during the analysis and design phases facilitate the smooth production of working programs that perform the appropriate functions for the information system [28]. However, there are three major disadvantages of SDLC methodology. Firstly, SDLC requires more time to develop and integrate an information system. Secondly, the costs of developing a system using the SDLC approach are very high. Finally, the use of the SDLC approach demands heavy documentation of the various processes used [25].

Applying this methodology to the timetable system is justified on the grounds of the well-documented features of the system that should be modelled using the SDLC approach. It is important to have a *clearly defined problem* as this will facilitate and save development time about gathering accurate information from the client [26].

The features of the timetable system include systematic inputs and outputs in a well-defined, well-structured system that produces all necessary reports, and the implementation of the system can be done on completion and not in a phase-in method. These features collectively enable the application of the SDLC methodology [24].

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng Department of Electrical and Computer Engineering, Baze University, Abuja.

SYSTEM ANALYSIS AND DESIGN

During the design of the database for the timetabling system, there were various entities identified. These entities were further classified into major and related entities. The major entities were identified as the primary source of data, which in turn post relation setting with related entities, generated a complete and robust database design for the system.

This taxonomy of entities is comprised of Faculties, Departments, Lecturers, Staff Types, Venues, Courses, Enrolments, Table Registry, Generated Tables, Constraints, and Settings [27]. The tables below give a brief description of each of the above-mentioned major entities and the entities related to them, which form the database structure of the Timetable System.

Table 1: Data Structure - Faculties, Departments, Lecturers, Staff Types, Venues, Courses, Enrolments, Table Registry, Generated Tables, Constraints, Settings

Major Entities	Description	Related Entities
Faculties	This entity stores information of faculties within the university.	Departments
Departments	Stores information of each department and faculties in which they belong as well.	Faculties
Lecturers	This entity stores information about lecturers, lecturer details, faculty and department of the lecturer as well as days and time availability.	Departments Staff Type
Staff Type	This entity stores information (designation, maximum contact hours per week) regarding the different types of staff in the university.	Lecturers
Venues	This entity stores details of lecture venues within available, this includes the capacity, type (classroom, laboratory, Club or lecture hall) and reservation details (faculties, departments, levels or days reservation).	Generated Tables
Courses	Stores data of all modules/subjects offered by all departments. This includes course hours per week, type (practical or non-practical based), departments of students who can register for the course and lecture(s) in charge.	Generated Tables Enrolments
Enrolments	Stores information of students' registrations for courses; such as module, student and semester's details.	Departments Courses
Table Registry	This entity stores data in context to the group of timetables generated by the system. This includes the session and semester of the timetable.	Table Registry
Generated Tables	This entity stores the generated timetables, this includes the modules/courses, students, lecturer, time and venue for the lecture.	Table Registry Courses Venues Departments Lecturers
Constraints	This stores information regarding some user customizable constraints such as lecture start and end time, lunch break, Muslim prayer time, free space between successive lectures (both lecturers and students), the maximum number of lecturers per day for lecturers and students, as well as Saturdays lecture on and off.	
Settings	Stores the general information about the university such as the method of admission, current semester and session.	

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

In the context of the timetabling problem, the major difficulty has been with identifying and preventing various clashes within the generated timetable. This issue within the system is prevented by a unique Artificial Intelligence based algorithm which takes into consideration all the historical data of the timetable and generates best-fit results avoiding all possible clashes. In a similar pattern, the timetable clashing for courses/modules, venues, students, and lecturers is completely prevented

.

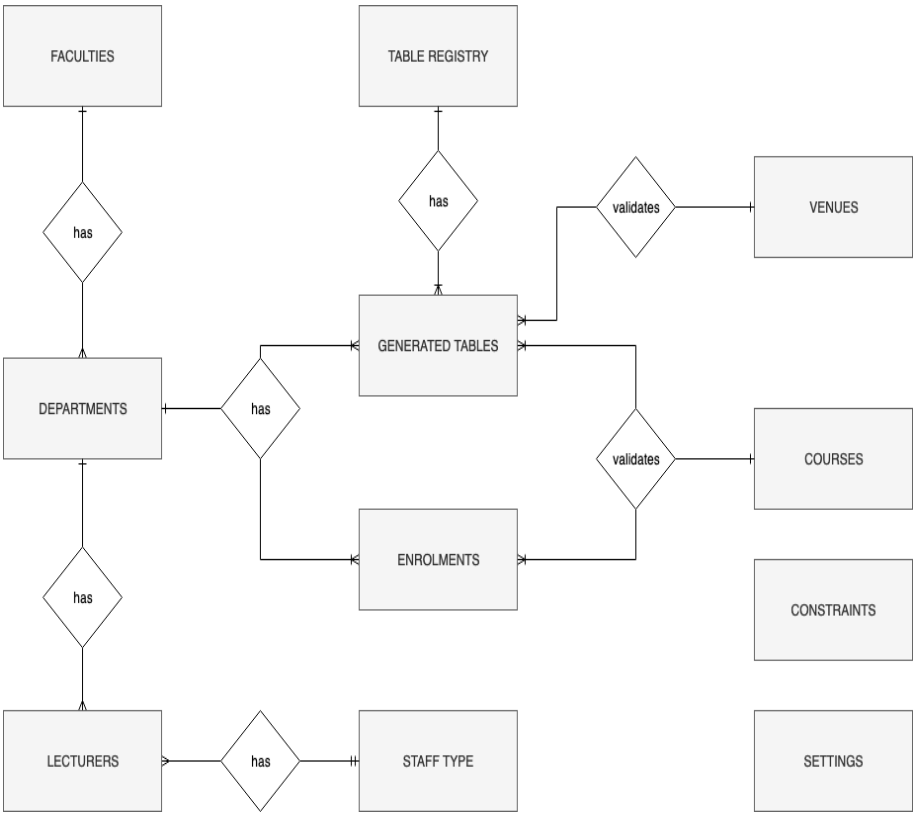

Figure 1 Er Diagram - Faculties, Departments, Lecturers, Staff Types, Venues, Courses, Enrolments, Table Registry, Generated Tables

The Entity Relationship Diagram

The ER diagram above shows the relationship amongst all the entities. *Constraints* and *Settings* entities are not directly related to any other entities because the two are only used by the timetable generation algorithm for dynamic constraints adjustment.

The *faculties* within the university system come at the first level in terms of course schedules organization structure. Various departments' information is stored in the Departments entity which comes at the second level of the structure. Each department has its various lecturers whose data are stored in the Lecturers entity. A lecturer must be of a certain type e.g. Head of Department, Lecturer I, and Lecturer II etc. which are contained in the Staff Type entity where we specify the attributes for a lecturer such as maximum contact hour per week.

The departments also have their various students' courses enrolment data stored in the Enrolments entity which are validated by the courses/modules stored in the Courses entity. The Table Registry entity holds information about a group of generated timetables e.g. timetables for the semester 19B of 2019/2020 session (if the admission method

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

is per semester) or 1st Semester of 2019/2020 session (if the admission method is per session). The timetable contents are stored as schedules in the Generated Tables entity whose data comes from the enrolment data for each department in the Department entity and are validated by the Courses entity.

Finally, each schedule in the Generated Tables entity is assigned to a valid venue in the Venues entity.

Time Table Generation

The generation of the university courses timetables starts by creating a *scheduler*. A scheduler in this context is the module of the system that is responsible for creating and initiating a timetable group. It assigned a unique ID for each group of timetables, the information required by the scheduler to create a timetable group ID is the session and semester name (if student enrolment is per semester) or semester type (if student enrolment is per session).

Once a scheduler is created, the next phase is to create the timetables that belong to this group i.e. timetables that belong to the scheduler session and semester.

Generation Algorithm

The generation algorithm is composed of four distinct stages – Data collection, Data Wrangling, Data Processing and Data Storage. The input is a scheduler ID and the output is the equivalent timetables that are stored in the Generated Tables entity after generation.

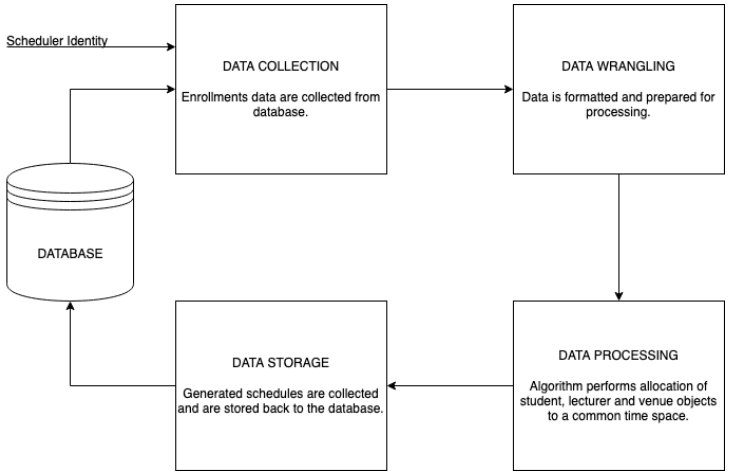

Figure 2: Generation Algorithm

Data Collection

This is the first stage of the algorithm; the input is a scheduler ID. The algorithm checks the validity of the ID and proceeds to collect all the enrolment data regarding the ID if the supplied ID is valid or return a corresponding error message to the user if not. Also, if no enrolment data is found, it returns an error message otherwise proceed to the data wrangling stage of the algorithm.

Data Wrangling

This is the second stage of the algorithm; its input is the raw data of students' enrolments and the output is a data frame containing rows of each unique module/course that needs to be scheduled in the processing stage. Some of the major operations performed here include Formatting of the raw enrolment data into scheduling data and generation of unique student group IDs for each module/course. This is done according to the type of student enrolment the university uses i.e. Baze University enrol students every semester so the corresponding group ID for PHY100 in semester 19B will be 2019/202019BPHY100.

The scheduling data is a data frame containing rows and columns of the various courses/modules grouped by the course group ID.

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

Data Processing

This stage is the third stage of the algorithm and it is where allocation of students, lecturers, and venue objects to time-space is done. The algorithm is in two different phases:

1. Students and lecturer time-space assignment:

This is the first phase of the data processing stage; it involves checking for available time-space for both students and lecturers. This is made possible and faster through the scheduling data frame from the data wrangling stage. Python Pandas library provides sophisticated tools for querying and manipulating the data frame instead of the traditional database querying that causes time delay and some other problems relating to database querying. If a time-space is found, it is allocated to the objects and the next phase is initiated; otherwise, the reason why no time-space is found is logged in the status column of the data frame.

2. Students-Lecturer and Venue assignment:

If a group of students and a lecturer is successfully assigned a time-space, the algorithm proceeds to find an appropriate venue that best fits the group based on a simple heuristic algorithm.
First, it checks for all venues that are free between the start and end time and then picks the venue with the highest weight based on the venue and home department attributes.

Data Storage

This is the final stage of the timetable generation algorithm; its input is the resulting data frame of scheduled courses/modules from the data processing stage. Here the algorithm makes sure that entries are not repeated by checking if group ID exists or not. If it exists it updates the existing row with the new scheduled data otherwise, it creates a new schedule record.
Finally, it returns a success message and instructions to the user.

Procedure to Generating Timetables

The procedure to generate the entire university timetable is very simple. Generation module is built-in with the home page and the steps to generate time tables are listed below:

1. From the home page, the user clicks create a scheduler instance by clicking on the plus button to add one.
2. User selects scheduler session and semester and clicks the create scheduler button to create the scheduler.
3. On successful creation of scheduler, user proceed to the scheduler table on the home page, and the scheduler is seen listed on the table. From there, the user clicks on the Open schedule button to open the scheduler, from which the button to generate all timetables for courses associated with the scheduler is accessed.
4. After generation, the user can view all generated timetables by clicking on the "View Tables" button.

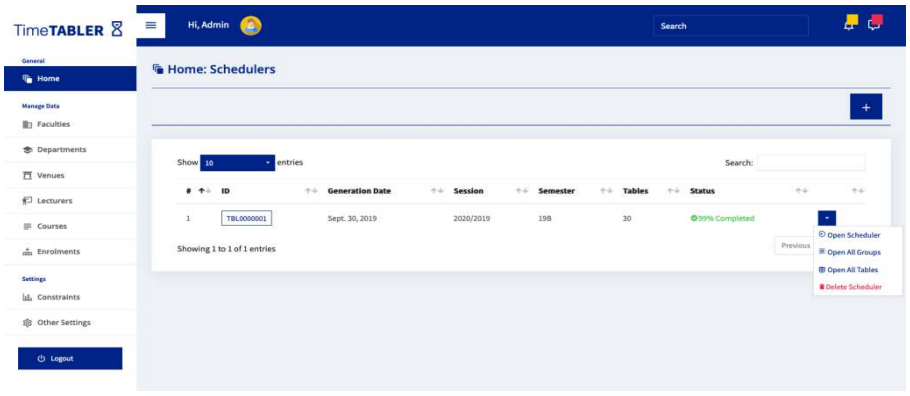

Figure 3: Scheduler Options

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng Department of Electrical and Computer Engineering, Baze University, Abuja.

RESULTS AND DISCUSSION

The system as developed and described in detail above may not be used for timetable generation in universities for very practical reasons. However, the level of efficiency of its implementation is needed to be established. The system worked smoothly and provided reasonable results, thus, the objective to automate the timetable generation process and provide effective management tools for timetable management for both universities that admit students per semester and per session was achieved.

Modules from departments of Computer, Electrical/Electronics and Telecommunication Engineering were used to test the system. Lecturers' data and students' enrolment data were modelled by a custom-designed algorithm that randomly generates simulated enrolment data based on the courses in the departments.

We can see that the system provides a module for the configuration of some constraints. Below we present and analyse the result of a group of generated timetables by the system based on the configuration below.

Table 1: Enrolments data for timetable

Data	Value
Total Courses	111
Total Courses with Lecturer	108
Total Courses without Lecturer	3
Total Course Enrolments	4710
Total Lecturers	24
Total Departments	3
Total Faculties	1
Total Group of Students	30
School Method of Admission	Per Semester

Table 2: System Configuration for Timetable Generation

Constraint		Value
Lecture Starts		09:00
Lecture Ends		18:00
Allow Saturday lecture?		False
Observe a compulsory break daily		True
Break Starts	**Break Duration in minutes**	
13:00	30	
Observe a compulsory Jumat Prayer on Fridays		True
Prayer Starts	**Prayer Duration in minutes**	
13:00	60	
Schedule visiting lecturers' courses first.		True
Restrict visiting lecturers to only their available days.		True
A maximum number of lectures a lecturer can take per day.		2
A minimum break between successive lectures for lecturers (in minutes).		30
The maximum number of lectures a student can take per day.		2

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

A minimum break between successive lectures for students (in minutes).	30

Table 3: Generated Timetable analysis

Data	Value	Remarks
Generation Starts	18:42:16	Total Time Spent for generation = 37seconds
Generation Finished	18:42:53	
Total Objects (Students and Lecturers) Filtered for Schedule	109	Average Time Per Processing = 0.35seconds
Total Objects (Students and Lecturers) Assigned to Time Space	106	Percentage Assigned to Time Space = 97.25%
Total Objects Assigned to Venue Space	100	Percentage Assigned to Venue Space = 100%
Percentage of Total Objects (Students and Lecturers) Assigned to Time-Space with Clash	0%	Clash Avoidance Percentage = 100%
Percentage of Objects Assigned to Venue Space with Clash	0%	

System efficiency can be calculated to be
= (Percentage Assigned to Time Space + Percentage Assigned to Venue Space + Clash Avoidance Percentage) *100/300
= (97.25 + 100 + 100)*100/300
= 99.08%

From the presentation above, it is found that the system processed 109 schedules within 37seconds which is extremely negligible to the several days or hours that would have been spent by timetable officers to process an equivalent number of schedules manually. Also, the approximated efficiency of the system is calculated to be 99.08%, the remaining 0.92% found to be 98% lack of resources and 2% system error.

The objective function of the system is designed to completely avoid conflicts while adjusting itself based on the user-specific constraint configuration. The system can determine optimal combinations for both school systems where admission is per semester as well as schools where admission is per session.

In addition to this project, the system can address problems regarding carry-over students and timetables. This is made possible because generation data is from student enrolment. So as far as a student is enrolled for a module whether fresh or carry-over, he or she is getting considered as part of the group. Also, the system can provide the school administrator overview of resources available, under resources and over resources can be easily accessed.

Nonetheless, these evaluations were almost objective, and are used to primarily eliminate the possibility that the system was not user-friendly or too technical.

CONCLUSION AND RECOMMENDATION

The objectives of studying the various timetable generation algorithms and developing an information system that can be used to automate the process of timetable generation as well as managing the timetable system; in both universities and colleges that admit students per semester and session has been achieved. The system can accommodate and synchronize all lecture sessions across the school and also solves the problem of shared resources. The project which was inspired to solve the various timetable related problems at Baze University has gone from just a simple idea into a complete timetable generation solution. Universities will be able to generate timetables automatically in a negligible length of time with the system instead of the laborious manual way of assignment that involves serious brainstorming on how to avoid course or room clashes by a group of students or lecturers. Given the amount of the task already undertaken, there is still a little number of changes or adjustments that are required to be completed before a production run of the

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng Department of Electrical and Computer Engineering, Baze University, Abuja.

system for widespread use could be achieved. Nevertheless, the current version could still be used for in-house usage for any university, polytechnic, college education or even a secondary and primary school.

Finally, it is hoped that the knowledge acquired in undertaking this project will be put to good use and also be applied in solving real-life problems in the academic industry and hence ultimately improving learning and teaching.

RECOMMENDATION

Although the system generation algorithm contains an in-built algorithm that verifies that each schedule made is not clashing with another one. Final verification is done manually by inspecting the generated timetables one by one, this process of verification is made faster by filtering the tables. However, there is a need for an automated verification algorithm that will take the entire generated timetables and verify if there is any clash since manual verification is not completely reliable and also time-consuming.

Recently, the use of quantum computing algorithms has evolved. These algorithms have more tendencies in solving optimization problems correctly than the usual computing algorithms. They could be applied to optimization problems such as in timetabling.

REFERENCES

[1] A. Wren, "Scheduling, timetabling and rostering — A special relationship?" Springer, Berlin, Heidelberg, 1996, pp. 46–75.J.

[2] Schaerf A., "Local Search Techniques for Large High School Timetabling Problems," *IEEE Trans. Syst. Man Cybern.*, 1999.

[3] V. . Bardadym, "Computer-aided school and university timetabling: the new wave," *First Int. Conf. Ger.*, 1996.

[4] H. Kuhn, "The Hungarian method for the assignment problem," *Nav. res Logist quart*, 1955.

[5] J. Haynes, "The conference scheduling problem," *Oper. Res. Soc. Am. 16th Annu. Meet. Pasadena Calif.*, 1959.

[6] P. Ross and D. Corne, "Comparing genetic algorithms, simulated annealing, and stochastic hill climbing on timetabling problems," *Evol. Comput. AISB Work. Sel. Pap. Springer-Verlag, Berlin, Ger.*, 1995.

[7] C. Maxfield, "Genetic algorithms: programs that boggle the mind.," *EDN*, 1997.

[8] D. Abramson, "Constructing School Timetables Using Simulated Annealing: Sequential and Parallel Algorithms," *Management-Science*, 1991.

[9] R. Dikmann, R. Luling, and J. Simon, "Problem independent distributed simulated annealing and its applications," *Tech. Rep. -- Paderborn Cent. Parallel Comput.*, 1993.

[10] D. Abramson and H. Dang, "School Timetables: A Case Study in Simulated Annealing," Applied Simulated Annealing," *Lect. Notes Econ. Math. Syst. Springer-Verlag, Ed. V. Vidal*, 1993.

[11] F. Glover and M. Laguna, "Tabu Search," *Kluwer Acad. Publ. Massachusetts*, pp. 0-7923-8187–4, 1997.

[12] F. Glover, *Tabu Search – Part II*, vol. 2, no. 1. 1989.

[13] E. F and M. A, "The complexity of some polynomial search algorithms for constraint satisfaction problems," *Artif. Intell.*, 1994.

[14] O. Lhomme, "Consistency techniques for numeric CSPs," *13th Intl. Jt. Conf. Artif. Intell.*, 1993.

[15] E. P. K. Tsang, P. Mills, R. Williams, J. Ford, and J. Borrett, "A computer-aided constraint programming system," 1999.

[16] J. P, J. P, N. B, and V. M.C, "A Filtering Process for General Constraint-Satisfaction Problems: Achieving Pairwise-Consistency Using an Associated Binary Representation," *IEEE Work. Tools Artificial Intell.*, 1989.

[17] B. J. Grim *et al.*, "Boko Haram Beyond the Headlines: Analyses of Africa's Enduring Insurgency," no. March 2017, pp. 2–6.

[18] Michael Dawson, *Python Programming - For Beginners*. 2010.

[19] A. Ravindran, *Django Design Patterns and Best Practices*. Packt Publishing, 2015.

[20] B. Schwartz, P. Zaitsew, and V. Tkachenko, *High Performance MySQL*. 2012.

[21] S. M.M and H. E. Williams, *Learning MySQL*. 2007.

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng *Department of Electrical and Computer Engineering, Baze University, Abuja.*

[22] W. McKinney, *Python for Data Analysis*. 2013.
[23] Daniel L. Chen, *Pandas for Everyone*. 2018.
[24] S. J.S, *Analysis and Design of Information System*. McGraw-Hill Publishing Company, 1989.
[25] W. J.L, B. L.D, and K. . Dittman, *System Analysis and Design Methods*. McGraw-Hill Publishing Company, 2001.
[26] W. S.H and W. M.S, *System Analaysis and Design*. West Publication Company, 1994.
[27] S. Su, X. Wang, Y. Cao and J. Yin (2019) ''An Energy-Efficient Train Operation Approach by Integrating the Metro Timetabling and Eco-Driving'' IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. DOI 10.1109/TITS.2019.2939358
[28]J. Tan, S. Goh, G. Kendall and N. Sabar (2021) ''A survey of the state-of-the-art of optimization methodologies in school timetabling problems'' Expert Systems With Applications 165 (2021) 113943

Corresponding author: Bara'u, G. N. oluwatobi2528@bazeuniversity.edu.ng Department of Electrical and Computer Engineering, Baze University, Abuja.