Jatilima : Jurnal Multimedia Dan Teknologi Informasi
Volume 06, Number 02, 2024, doi.org/10.54209/jatilima. v6i02.705
E.SSN : 2721-1800 (Online)
https://journal.cattleyadf.org/index.php/jatilima/index

APPLICATION OF THE A* METHOD IN MAKING COURSE SCHEDULING DECISIONS AT STMIK MULIA DARMA

Muhammad Iqbal Panjaitan[1], **Denni M. Rajagukguk**[2], **Mamed Rofendi Manalu**[3], **Monang Juanda Tua Sihombing**[4], **Kristian Siregar**[5]
[1,2,3,4,5] STMIK Mulia Darma, Labuhanbatu, Indonesia

Article Info
Keywords:
Course Scheduling,
A* Method,
Search Algorithm,
Optimization, Decision Making

ABSTRACT
One of the biggest issues in managing higher education is course scheduling, where a variety of criteria need to be taken into account in a balanced way, including lecturer availability, classroom size, and student preferences. This procedure can be greatly optimized by using the A* method, also referred to as the optimal path finding algorithm. Applying the A* technique to course scheduling decisions is the goal of this research, which attempts to improve efficiency and happiness for all stakeholders.A customized A* algorithm is designed and implemented as part of the research process to manage scheduling characteristics such lecturer schedule compatibility, space allocation, and time slots. We tested this technique on scheduling data from an Indonesian university. The results of the tests indicate that the A* approach can yield a more optimal.

Corresponding Author:
Muhammad Iqbal Panjaitan
STMIK Mulia Darma
Labuhanbatu
Iqbalpj87@gmail.com

INTRODUCTION

Scheduling is a very important issue for an institution education. There are many obstacles that make scheduling difficult made. This is also what STMIK Mulia Darma experienced, making a schedule usually takes a lot of time very long because STMIK Mulia Darma is still using the application which is manual and not yet automatic, while the lecture schedule must be ready quickly because it will immediately be used for each Study Plan Activity (KHS). Another thing that makes the scheduling system problems even more complex is a lecture schedule activity at the time, class, lecturer that has been determined (Lestari, U., Widyastuti. N., and Listyaningrum, D.A., 2014). Problem Scheduling in higher education is a particular problem optimization found in real situations. This problem takes time computing is quite high for finding a solution, especially if it is large The problem gets bigger as the number of components and constants increases or conditions determined by the institution where the schedule is used (Uning L., Naniek, and Desti,, 2014). The process is still manual and the accuracy factor Sometimes this causes the resulting schedule to be less than perfect. This is because there are still violations of existing constraints, both soft and hard constraints or

hard constraints, for example schedules that conflict with teaching space or teaching hours, lecturers who teach during hours are unable to teach.

The creation of course schedules is influenced by a number of factors, including time slots, lecture halls, students, and lecturers. There are situations related to each of these elements that could cause issues and disputes while creating course schedules. One issue arising from the lecturer perspective is lecturer conflict, wherein a lecturer is assigned to teach simultaneously in two different places. Alternatively, the issue from the perspective of the student is the high number of classes every generation. Each of these elements has the potential to cause a plethora of additional issues in addition to these ones.

Creating a decision support system is one way to use computerization to solve the aforementioned issues. All of these issues will be recognized by the system as organized issues that can result in effective course scheduling solutions. Structured problems are those in which the solution can be found or comprehended through the use of algorithms or decision rules. It is also feasible to find alternative solutions, assess them, and choose the best one by applying these algorithms or criteria.

A model for resolving issues is necessary for a decision support system. Finding the problem, assessing the problem environment, and figuring out the variables involved in decision-making will help you choose the best model. When it comes to class scheduling, the issues.

George Polya defines heuristics as the study of rules and strategies for discovery. Heuristics are criteria used in course schedule preparation that specify which time period is most likely to result in a workable solution to a problem. Since a time slot is compared to a path that the courses to be scheduled will follow, an algorithm is required to determine the shortest road or path that leads to a solution. The A* algorithm is the best method for generating the shortest path or path to the problem's solution. The Genetic Algorithm (GA) is an algorithm that solves complex problems by applying the ideas of natural development. This technique can be used to find the ideal timetable for course scheduling decisions by taking lecturer, time, and room constraints into account.

METHODS

This research uses a comparative quantitative approach, namely an experimental research method to test the effectiveness of the A* method in course scheduling.

- System Design: Design a scheduling system that uses the A* method to find the optimal schedule.
- Data Collection: Data needed for testing, such as course schedules, lecturers, and rooms.
- Development of the A* Algorithm: Implementation of the A* algorithm in the course scheduling system.
- Testing and Evaluation: Testing the scheduling system using the data that has been collected and evaluating the results to determine the efficiency and quality of decisions (Alfian Bayu Nur Aji, 2024)

RESULTS AND DISCUSSION

Problem Analysis

Course scheduling is a complex optimization problem, where a number of variables and constraints must be considered to produce an efficient schedule. Some of the main factors that need to be analyzed in this context include:

1. Lecturer Availability: Each lecturer has a specific schedule that must be accommodated, including certain times when they are unable to teach.
2. Classroom Allocation: Classrooms have certain capacity and equipment which must be in accordance with the needs of the courses to be implemented.
3. Time Slot: Courses must be scheduled in a time slot that does not conflict with other course schedules, both for lecturers and students.
4. Student Preferences: Preferences regarding time and courses that students want to take are also taken into consideration in preparing an optimal schedule.

Analysis and Design of System Components

In order for this system to be able to schedule courses, it requires the ability to allocate all course resources, lecturers, students and rooms into time slots with predetermined constraints. To achieve this, this ability is filled with knowledge of the constraints that exist in hard constraints and soft constraints so that the system can overcome the problems faced when scheduling courses. All scheduling criteria before being scheduled must meet the constraints of hard constraints and continue with checking the constraints. which exist in soft constraints after the course schedule solution is obtained. Each constraint, both hard constraints and soft constraints, has different limitations. In hard constraints, each criterion must meet the constraints contained in the constraints. If any of the constraints on the constraints cannot be met, it is certain that the criteria cannot be scheduled in the selected time slot. In contrast to soft constraints, the constraints provided do not have to be met by every criterion, but only as a reference for determining the quality of the scheduling solution produced by the system later. Even though all the scheduled criteria cannot meet the soft constraints, these criteria are still scheduled in the selected time slot. Apart from knowledge of hard constraints and soft constraints, this subsystem also contains knowledge of a tree, looking for conflict values for courses, looking for values for the quality of course schedule solutions and finding values for the effectiveness of course scheduling. This tree will be used as a medium for the A* algorithm to find time slots. The course conflict value is used to determine which courses will be scheduled first. The quality value of the course schedule solution is used to determine the average fulfillment of each course that has been scheduled against the soft constraints. Meanwhile, the effectiveness value of course scheduling is used to determine the percentage of scheduled courses

Hard Constraints

Based on the identification of the case study in this thesis and the criteria problems explained in the previous subchapter, the hard constraints used in this system and which must be met by all criteria to build a course schedule are as follows.

H(1). A lecturer can only give lectures for one location at a certain time.
H(2). A lecturer can only give lectures according to the lecturer's availability to teach.
H(3). A student can only attend lectures for one location at a certain time.
H(4). A location (room) can only be used for one course at a certain time.
H(5). Students cannot be allocated to one location where the number of students exceeds the capacity of that location.
H(6). Class times are held every Monday to Saturday from 08.00 WIB to 16.20 WIB.

H(7). There are no lectures for all locations on Fridays from 12.10 WIB to 13.00 WIB.
H(8). Courses of the same level cannot be scheduled at the same time.
H(9). Courses that use infocus are allocated to one location that has infocus facilities

Soft Constraints

The soft constraints used as a reference for determining the quality of course schedule solutions in this system are as follows.

H(1). Minimize the occurrence of consecutive course schedules for courses that have the same student class. This constraint is used to avoid boredom and exhaustion experienced by students when attending lectures. This constraint is also used to anticipate moving lecture rooms from one course to another where the distance between the rooms is far enough so that it does not cause hassle for students when changing courses.

H(2). As far as possible, a course is not scheduled with courses that are one level below and/or one level above that course at a certain time together. This constraint can allow a student who wants to take a course one level above it, thereby making the student graduate more quickly or allow a student who wants to take a course below it to repeat a course they failed previously.

H(3). Minimize the occurrence of course schedules that cause a lecturer to give lectures in a row. This constraint is intended so that a lecturer does not experience boredom and fatigue when giving lectures. In the system to be built, the constraints of the three soft constraints above are only used to determine the quality of the course schedule solution produced after the hard constraints are applied to all course resources, lecturers, students and rooms. The quality value of the course schedule is between 0 and 1 which will be expressed in the form of a percent (0-100%).

Tree

In the lecture search process, the system uses trees as a search medium. A tree is defined as a collection of nodes with one special element called the root and the other nodes are divided into sets that are not related to each other or are called subtrees (Sanjaya, 2001). This tree has a size of 36 nodes with a depth of up to 8 levels. Each node contains different time slots representing class days and hours. The A* algorithm will look for the best time slot (lowest cost) for each course to be scheduled by tracing the nodes in the tree to get the desired solution.

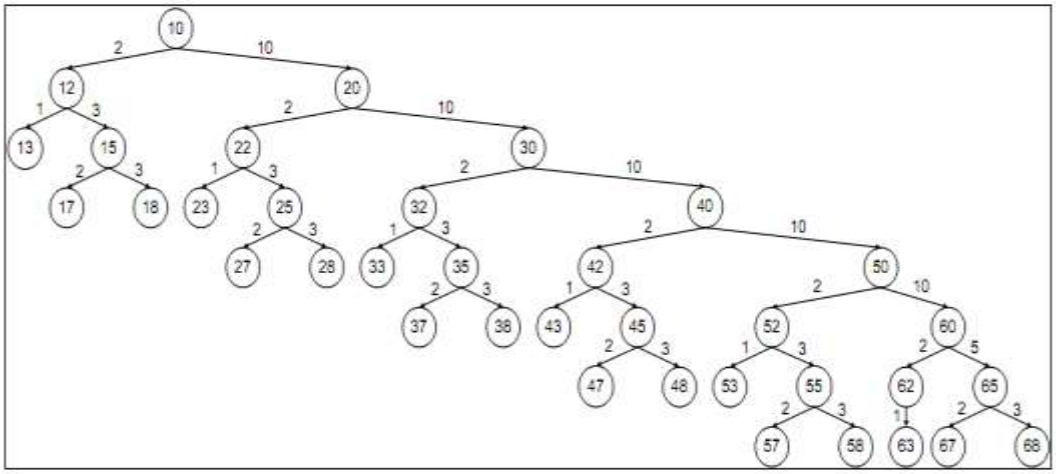

Figure 1. Tree Of Time Slots

Tables only use horizontal lines in the header and closing sections with a line thickness of 1 pt (Table 1. Example of table format). Table titles use Nunito 8pt font, while table contents use Nunito 8pt font. Information can be added at the bottom of the table with the provisions in the example Table 1.

Each node contains a time slot that represents the day and time of the lecture. The tens number on the node represents the lecture day. The following is an explanation for the tens number on the time slot node.

1. The number '1' represents Monday.
2. The number '2' represents Tuesday.
3. The number '3' represents Wednesday.
4. The number '4' represents Thursday.
5. The number '5' represents Friday.
6. The number '6' represents Saturday.

Meanwhile, the unit number at the node represents lecture hours. The following is an explanation for the unit digits on the time slot node.

1. The number '0' represents the first hour of lectures (08.00 to 08.50 WIB).
2. The number '2' represents the third hour of lectures (09.40 to 10.30 WIB).
3. The number '3' represents the fourth hour of lectures (10.30 to 11.20 WIB).
4. The number '5' represents the sixth hour of lectures (12.10 to 13.00 WIB).
5. The number '7' represents the eighth hour of lectures (13.50 to 14.40 WIB).
6. The number '8' represents the ninth hour of lectures (14.40 to 15.30 WIB).

Apart from being an indication of the days and hours of lectures, the contents of the node also function as the weight of the node. And each interconnected node has a value that indicates the distance between the nodes. For example, the distance between node '10' and node '12' is 2. The weight of the nodes and the distance between these nodes will be used to find the value of f(n) in the ma A* algorithm.

Course Conflict Value

It was mentioned above that there are nine hard constraints that must be met. To avoid conflicts between courses, hard constraints will prevent this problem. Hard constraints place restrictions on courses that have the same lecturer, student class, and course level so that they are not scheduled in the same time slot. These constraints are contained in the hard constraints H(1), H(3), and H(8). The H(1) limitation on hard constraints has a function to avoid scheduling courses that have the same lecturer scheduled in the same time slot. This is something that may happen in every department, where a lecturer teaches two or more courses in one department. To find out the conflicts that occur, you can do this by creating a matrix D = [dnm]. This matrix represents conflicts from courses that have the same lecturer. Where N is the number of courses to be scheduled and n, m = 1, ..., N. Mean while d NxN is the conflict value for courses that have the same lecturer. Nm

$$D = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \cdots & d_{1m} \\ d_{21} & d_{22} & d_{23} & \cdots & d_{2m} \\ d_{31} & d_{32} & d_{33} & \cdots & d_{3m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{n1} & d_{n2} & d_{n3} & \cdots & d_{nm} \end{bmatrix}$$

The initial value of d Nm = 0 and will be added by 1 if course n and course m are taught by the same lecturer**.**

The H(3) constraints on hard constraints are the same as the H(1) constraints on hard constraints. Constraint H(3) serves to avoid scheduling courses that have the same student class scheduled in the same time slot. A class of students may not be able to attend lectures in more than one subject at the same time. So that the course can be taken by the class of students concerned, of course it must be scheduled in a different time slot. Thus, it can be said that courses that have the same class of students are conflict courses if they are scheduled in the same time slot. Like the H(1) limitation, to find out the conflicts that occur due to student class, you can do this by creating a matrix K = [knm]NxN. This matrix will represent courses that are in conflict because they have the same student class. N is the number of courses to be scheduled and n, m = 1, ..., N. And k is the conflict value for courses that have the same student class.

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & \cdots & k_{1m} \\ k_{21} & k_{22} & k_{23} & \cdots & k_{2m} \\ k_{31} & k_{32} & k_{33} & \cdots & k_{3m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ k_{n1} & k_{n2} & k_{n3} & \cdots & k_{nm} \end{bmatrix}$$

Likewise with the limitations on hard constraints H(8). The conflict that exists in the H(8) boundary can be identified by creating a matrix. The H(8) constraint was created to avoid scheduling courses of the same level scheduled in the same time slot. This limitation will be able to function properly if one class of students is divided into two or more classes of students in one class. If there is only one student class in one class of students, then the H(8) constraint will have the same function as the H(3) constraint. The H(8) limit was formed with the aim that students have many alternatives in choosing study time. For example, a student wants to take a class one level higher or repeat a course. However, this course conflicts with the class schedule for the courses he is required to take. With another course class, the student can choose another course class as an alternative so that the student can take a course one level above or repeat the course. To find out the conflicts that occur due to the H(8) limitation, you can create a matrix T = [tnm] NxN. This matrix will represent courses that are in conflict because they have the same course level. N is the number of courses to be scheduled and n, m = 1, ..., N and t nm are the conflict values for courses n and m which have the same level.

$$T = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \cdots & t_{1m} \\ t_{21} & t_{22} & t_{23} & \cdots & t_{2m} \\ t_{31} & t_{32} & t_{33} & \cdots & t_{3m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ t_{n1} & t_{n2} & t_{n3} & \cdots & t_{nm} \end{bmatrix}$$

The initial conflict value for all t nm is 0 and will be increased by 1 if there is a conflict for courses n and m because they have the same course level.

After the conflict values from the three matrices are obtained, the three matrices are added together to form a matrix C = [cnm] NxN, where C = D + K + T. This matrix will present the conflict values between courses caused by lecturers, classes and levels. subject. The conflict value from matrix C will be a reference for scheduling. Courses that have the greatest

conflict value will be scheduled first to maximize scheduling solutions. The following is a flowchart algorithm for the matrix C = [cnm]NxN

Heuristic Programming Model

To build a heuristic programming model of a decision support system that can solve course scheduling problems, rules are needed to overcome problems for each scheduling criterion. These rules are made based on existing limitations on hard constraints and soft constraints. These rules will be formed into heuristics which will become decision rules that regulate how a scheduling problem must be solved. Table 4.1 contains heuristics for course scheduling criteria used as a system model.

Table 1 Heuristics of Scheduling Criteria

No	Scheduling Criteria	*Heuristics*
1	Lecturer	Do not schedule lecturers at a certain time if it does not match the lecturer's availability
2	Lecturer/Room	There are no lecturers who teach courses in two or more different rooms at any given time
3	Student/Room	There are no students attending lectures in two or more different rooms at any given time
4	Student/Room	The number of students allocated to one room must be smaller than the capacity of that room
5	Room/Course	The room can only be used for one course at any given time.
6	Subject	Courses of the same level cannot be scheduled at the same time.
7	Courses/Rooms	Courses that use infocus are allocated to one room that has infocus facilities.
8	Time slot	Slot waktu Waktu kuliah dilaksanakan setiap hari Senin sampai dengan Sabtu mulai pukul 08.00 WIB sampai dengan 16.20 WIB
9	Time slot	There are no lectures on Fridays from 12.10 WIB to 13.00 WIB (node '55').

The heuristic above will be combined with the A* algorithm in the process of scheduling courses by looking for time slots that meet the rules in the heuristic. However, not all of the heuristics above can be applied to all courses. There are several courses that are required to be immune from these decision rules. These courses are courses that can be taught together even though they have the same class and course level. The courses in question are religious courses. Based on the case study of this thesis, the religion course consists of five courses, namely, Islamic Religious Education, Catholic Christian Religious Education, Protestant Christian Religious Education, Hindu Religious Education, and Buddhist Religious Education. Each of these courses has the same student class and course level which should not be scheduled at the same time. However, in practice, these courses can be scheduled in the same time slot even though the student class and course level are the same. This is because the students who take these courses are different for each course

(depending on their respective religions) even though the student class is the same. So in the time slot search process, not all heuristics can be applied. To differentiate religion courses from other courses, the system uses the status of the course, namely, 'Ordinary' or 'Joint' status. Courses that have 'Ordinary' status must fulfill all existing heuristics, while religious courses that have 'Joint' status only fulfill several heuristics, namely heuristic numbers 1, 4, 5, 7, and 9 in Table 4.1. In its implementation, the heuristics above will be applied one by one to all course resources that will be scheduled. The steps in applying heuristics that must be fulfilled for courses that have 'Ordinary' status are as follows.

1. Check whether lectures can be held in the time slot or not (H(7)), if yes: continue to number 2.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
2. Check whether in the time slot there are unused/empty rooms (H(4)), if Yes: continue to number 3a.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
3. Check (H(9)):
   a. Does the course require Infocus facilities, if so Yes: continue to number 3b. No: go to number 4a.
   b. Does the room have Infocus facilities, if Yes: continue to number 4a. No: go to number 4b.
4. Check (H(5)):
   a. Is the number of students in the course class smaller than the room capacity, if Yes: go to number 5.
      No: go to number 4b.
   b. Check if there are any other empty rooms in that time slot, if
      Yes: back to number 3a.
      No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
5. Check whether the courses that have been scheduled in the time slot do not have the same level as the courses to be scheduled (H(8)), if
   Yes: go to number 6.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
6. Check whether the student class from the course that has been scheduled in the time slot is not the same as the student class from the course that will be scheduled (H(3)), if
   Yes: go to number 7.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
7. Check whether the lecturer who teaches the course that has been scheduled in the time slot is not the same as the lecturer who teaches the course that will be scheduled (H(1)), if
   Yes: go to number 8.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.

8. Check whether the time slot is within the lecturer's availability time range to teach the scheduled course (H(2)), if
   Yes: Goal.
   No: courses cannot be scheduled into time slots.

Meanwhile, the steps in applying heuristics that must be fulfilled for religious courses that have 'Joint' status are as follows.

1. Check whether lectures can be held in the time slot or not (H(7)), if yes: continue to number 2.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
2. Check whether in the time slot there are unused or empty rooms (H(4)), if
   Yes: continue to number 3a.
   No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
3. Check (H(9)):
   a. Does the course require infocus facilities, if
      Yes: continue to number 3b.
      No: go to number 4a.
   b. Does the room have infocus facilities, if
      Yes: continue to number 4a.
      No: go to number 4b.
4. Check (H(5)):
   a. Is the number of students in the course class smaller than the room capacity, if
      Yes: go to number 5.
      No: go to number 4b.
   b. Check if there are any other empty rooms in that time slot, if
      Yes: back to number 3a.
      No: courses cannot be scheduled in a time slot, move to the next time slot and return to number 1.
5. Check whether the time slot is within the lecturer's availability time range to teach the scheduled course (H(2)), if
   Yes: Goal.
   No: courses cannot be scheduled into time slots.

CONCLUSION

The conclusions obtained from this research areThe course scheduling decision support system with a heuristic programming model using the A* algorithm produces course schedule solutions that can be taken into consideration by decision makers in building course schedules. This decision support system provides an assessment of the quality of the course schedule solutions produced based on the fulfillment of soft constraints. This decision support system provides an effectiveness value which shows the degree to which many courses can be scheduled by the system from the course resources it wants to schedule..

REFERENCE

Mirjalili, S., & Mirjalili, S. (2019). Genetic algorithm. Evolutionary algorithms and neural networks: theory and applications, 43-55.

Ansari, R., & Saubari, N. (2020, April). Application of genetic algorithm concept on course scheduling. In IOP conference series: materials science and engineering (Vol. 821, No. 1, p. 012043). IOP Publishing.

Lestari, U., Widyastuti, N., & Listyaningrum, D. A. (2014). Implementasi algoritma genetika pada penjadwalan perkuliahan. In Prosiding Seminar Nasional Aplikasi Sains & Teknologi (SNAST) (pp. 211-216).

Sri Eniyati, Candra Noor Santi, Perancangan Sistem Pendukung Keputusan Penilaian Prestasi Dosen Berdasarkan Penelitian dan Pengabdian Masyarakat, 2010.

Lambora, A., Gupta, K., & Chopra, K. (2019, February). Genetic algorithm-A literature review. In 2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon) (pp. 380-384). IEEE.

Panjaitan, M. I., Rajagukguk, D. M., & Manalu, M. R. (2023). Implementation of the Decision Support System for the Appointment of Permanent Employees at CV. Armas Suan Sejahatera Using the Analytical Hierarchy Process (AHP) Method. Jurnal Info Sains: Informatika dan Sains, 13(02), 119-128.

Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A genetic algorithm (ga) based load balancing strategy for cloud computing. Procedia Technology, 10, 340-347.

Kusrini, 2007, Sistem Pengambilan Keputusan (SPK), Yogyakarta

Muhtosim Arief, 2006, Pengetahuan dan Faktor Produksi Jasa, Yogyakarta.

Fandi Tjiptono, 2007, Petrovicdan Burke,2004, Penjadwalan Mata Kuliah

Turbanetal,2005, Karakteristik dan Kemampuan Sistem Pendukung Keputusan, Yogyakarta.

Setiawan,1993, Pemrograman Heuristic, Penerbit Bandung.

Panjaitan, M. I., Rajagukguk, D. M., Manalu, M. R., Karo-karo, S., & Sihombing, M. J. T. (2023). Penerapan Algoritma Apriori Untuk Memprediksi Tingkat Kelulusan Mahasiswa (Studi Kasus: Program Studi Manajemen Informatika dan Komputerisasi Akuntansi Universitas Imelda Medan). Jurnal Multimedia dan Teknologi Informasi (Jatilima), 5(02), 157-164.

Shalabi, R. R. (2020). The Importance and Applications of Decision Support Systems (DSS) in Higher Education. doi, 10, m9.