Modeling the Space of Possible States of the Lesson Schedule in Higher Education Institutions

Dmitry Petukhin[1], Stanislav Velykodniy[2] and Valentina Kozlovskaya[3]

[1,2,3] *Odessa State Environmental University, Lvivska str. 15, Odesa, 65016, Ukraine*

Abstract

For the task of automated scheduling of classes in the higher educational institutions, a model of the space of all possible states of the lesson schedule has been developed. The model is developed in terms of relational algebra. The solution to the problem is found by means of a relational DBMS. Using the proposed approach allows you to get an initial solution to the problem of scheduling classes in a higher educational institution. Also, the model of the space of possible states of the lessons schedule allows you to adjust the schedule when it is further optimized or if it is necessary to make changes to the finished class schedule. The initial solution to the problem is found using an iterative process, which at each step chooses the lesson with the least freedom of scheduling or the lesson of the teacher who has the most busy schedule. Freedom of scheduling classes and the density of teachers' schedules are variables that are calculated at each iteration for a subset of classes not yet scheduled.
The optimality of scheduling each lesson is determined by a set of criteria that are summed up using fuzzy logic methods.

Keywords

Relational algebra, database, space of states, subspaces, schedule classes, fuzzy logic.

1. Introduction

The multicriteria task of scheduling classes in the higher educational institutions (HEIs) still does not have a generally accepted solution. R. A. Oude Vrielink, E. A. Jansen, E. W. Hans, J. van Hillegersberg in their article "Practices in timetabling in higher education institutions: a systematic review" made a very detailed review of methods and programs for scheduling classes in higher educational institutions, developed since the 70s years of the 20th century and up to 2017 inclusive [1]. They showed that the problem of automated scheduling of classes still does not have a generally accepted solution. At the same time, the need to obtain an acceptable solution does not decrease over the years, but only increases, since the number of restrictions to the optimal schedule increases.

The works of recent years are mainly devoted to the development of genetic algorithms and swarm algorithms, sometimes modifications of the simulated annealing method are being developed [2] – [7]; in this case, the initial approximation of the solution to the problem is obtained using the Tabu method or graph coloring [8].

"These approaches are defined as heuristics that solve strategies that are used for complex optimization problems. A specific set of algorithms, called heuristic algorithms, have proven to be effective in generating the best non-optimal solution. Such algorithms can give an approximation that is considered an acceptable solution. Thus, heuristic algorithms seem to be superior to traditional methods, and such algorithms are even combined to strengthen each other" [1].

ISIT 2021: II International Scientific and Practical Conference «Intellectual Systems and Information Technologies», September 13–19, 2021, Odesa, Ukraine
EMAIL: tiva2000@ukr.net (A. 1); velykodniy@gmail.com (A. 2); vkozlovskaya20@gmail.com (A. 3)
ORCID: 0000-0002-1588-2275 (A. 1); 0000-0001-8590-7610 (A. 2); 0000-0003-0353-5281 (A. 3)

The purpose of this work is to develop a new method for automated scheduling of classes and its practical implementation by means of relational DBMS. The mathematical model is built in terms of relational algebra; practical implementation involves the development of a database and server software for it.

The proposed method belongs to a variety of simulation methods. An algorithm based on the principles of simulation should have a set of non-formal (heuristic) rules:

1. Choosing the next lesson from the list.
2. Determining the best position for it in the schedule.
3. Evaluating the resulting schedule.

To select the next lesson from the list, a multilevel sorting of the list of unallocated lessons is used according to the criteria of increasing the freedom of the arrangement of the lesson in the schedule and decreasing the density of the teacher's lessons. Fuzzy logic methods with heuristic coefficients determining the weight of each of a large set of criteria for evaluating the optimality of the lesson schedule are used to determine the best position for taking a position in the schedule and assess the resulting schedule.

2. Mathematical model of the problem being solved in terms of relational algebra

The task of automated scheduling of classes is characterized by a large amount of data that must be stored in a database. However, the tools of relational database management systems (RDBMS), together with methods of relational algebra, can be used to solve this problem.

No international standards have been adopted for the operations of relational algebra, so there are currently at least four notations to denote these operations. This paper uses the notation of relational algebra operations, described in detail in the book by Thomas Connolly and Carolyn Begg [9].

2.1. Sections of the space of possible states of the lesson schedule

Consider the formulation of the problem of scheduling classes in a university in terms of the theory of relations and relational algebra. Let's call one academic pair in one academic group in one academic discipline "lesson". Let the relation G describe academic groups; H – possible types of activities (lectures, seminars, laboratory studies); D – academic disciplines; C – University curricula.

The relations G, H, D, C are tables that contain the initial data for the task of scheduling classes. Also, the initial data is contained in the relation T – the list of teachers, and in relation to the A – list of recitation room.

$DC(idc, idd, idh, hour)$ is a derived relation (associative type of entity) that describes the distribution of academic hours between different types of classes in academic disciplines of all curricula:

idc – curriculum identifier;
idd – discipline ID;
idh – activity ID.

The public budget. The main purpose of the public budget is to empower citizens and NGOs to propose their own local development projects and influence the allocation of a certain share of the budget funds by voting for certain projects.

Let the derived relation $GC(idg, idc)$ describe the attachment of groups to curricula; here idg is academic groups ID. Then the classroom load of academic groups is described by the relation $Q(idg, idc, idd, idh, hour)$ (1):

$$Q \leftarrow GC \rhd\lhd DC, \qquad (1)$$

where $R \rhd\lhd S$ – the relational operation of naturally joining a relation R with a relation S over the entire set of common attributes.

The curriculum for each course in each specialty contains data on the number of academic weeks in a semester.

From the relation Q the stored procedure of the database forms a derived relation $W(ids, idg, idc, idd, idh, num)$, which contents data of classroom activities of academic groups; where ids – lesson ID.

In relation W, the num attribute is necessary for unambiguous identification within one academic week of each pair of classes in the case when in a certain discipline for a certain type of classes, for example, lectures, N hours are provided per week, where $N > 2$. Then, for this type of lesson in this discipline, there should be np training pairs in the schedule (2):

$$np = N/2, num = 1..np, \qquad (2)$$

For the relation W the set of attributes $(idg, idc, idd, idh, num)$ is a unique key. Composite key identification is inconvenient, so the primary key ids is added to the W relation.

Let $TL(idt, ids)$ be a derived relation that describes the distribution of classes among teachers, where idt – teacher identifier. Then the activities to be scheduled are described by the relation $S(ids, idg, idt)$ (3):

$$S \leftarrow \pi_{ids,idg,idt}(W \rhd\lhd TL) \quad (3)$$

where $\pi_{a1,..,ak}(R)$ – the relational operation of the projection of a relation R onto a subset of its attributes $a1, \dots ak$.

Derived relations $PT(idt, idp)$, $PG(idg, idp)$ and $PF(ida, idp)$ describe the lists of study pairs allowed for classes, respectively, for teachers, academic groups and classrooms; $SA(ids, ida)$ – lists of acceptable classrooms for lessons. Here idp is the ID of the study pair; ida – classroom ID.

Let $Z(ids, idg, idt, idp, ida)$ be a relation that describes all possible options for a class schedule. The ratio Z is calculated using relational algebra operations by the formulas (4) – (5):

$$TT \leftarrow (SA \rhd\lhd PA \rhd\lhd PT), \quad (4)$$

$$Z \leftarrow \pi_{ids,idg,idt,idp,ida}(S \rhd\lhd TT \rhd\lhd PG) \quad (5)$$

where TT is an auxiliary relation (intermediate relational variable).

The S and Z relations do not include an activity such as streaming lecture – lecture for academic stream (stream of academic groups).

To take into account this type of occupation, it is necessary to enter into the database schema two more additional entities. The basic entity type $R(idr, Rname)$ (stReam) describes the list of streams, and the associative entity type $RG(idr, idg)$ describes the composition of the stream: which groups are included in the stream. Let $DR(idd, idc, idr)$ be a derived relation containing data: on which disciplines of which curriculum lectures are given on streams, and on what streams.

Streaming lectures will be described by the derived relation WRR, which is calculated by the formula (6) – (7):

$$TT2 \leftarrow W \rhd\lhd RG \rhd\lhd DR \quad (6)$$

$$WRR \leftarrow \pi_{ids,idg,idd,idh,num,idr}(TT2) \quad (7)$$

where $TT2$ is intermediate relational variable.

Each streaming lecture in relation WRR takes up as many tuples as there are groups in the stream. Let's introduce an additional key attribute $idsr$ – the stream lesson ID.

By projecting the WRR relation onto the $idsr$, idr, idd,num attributes, we obtain the WR relation – a list of all streaming lectures (8):

$$WR \leftarrow \pi_{idsr,idr,idd,num}(WRR) \quad (8)$$

In relation WR, each streaming lecture is described by one tuple, and the $idsr$ attribute is the primary key of this relation.

For the convenience of fetching data and calculation formulas when setting classes in the schedule, it is desirable that the $idsr$ attribute receive its values from the same domain as the ids attribute, and that the sets of values of these attributes do not overlap. This can be easily achieved if, when assigning a streaming lecture, a database stored procedure is used, which generates the value of the $IDsr$ variable, which is used as the value of the $idsr$ attribute for the new tuple of the WR relationship. $IDsr$ is calculated by the formula (9):

$$IDsr = \text{MAX}\big(\text{MAX}(ids), \text{MAX}(idsr)\big) + 1 \quad (9)$$

Streaming lectures with attached lecturers is described by the SRR relation (10):

$$SR \leftarrow \pi_{idsr,idr,idt}(WR \rhd\lhd TL) \quad (10)$$

Then the relation ZR, which describes all variants of the lesson schedule for streaming lectures, is represented by the formula (11) – (12):

$$TT3 \leftarrow SR \rhd\lhd TT \rhd\lhd PR \quad (11)$$

$$ZR \leftarrow \pi_{idsr,idr,idt,idp,ida}(TT3) \quad (12)$$

where $TT3$ is intermediate relational variable; PR is a derived relation that describes the sets of admissible study pairs for academic streams.

The set of study pairs for each stream is equal to the intersection of the sets of admissible study pairs of all groups included in the stream.

Classes of academic groups can also be conducted with a breakdown of the academic group into subgroups, for example, when conducting laboratory classes in specialized classrooms.

The derived relation $DS(idd, idg, idh)$ stores data: in which disciplines in which groups of which types of classes are conducted with a breakdown into subgroups. Let $SB(sub)$ be an auxiliary relation that contains two tuples: $sub = 1$, $sub = 2$, where sub is the number of the subgroup.

$WS\ (ids, idg, idc, idd, idh, num, sub)$ is a relation that describes group lessons that are carried out by dividing the group into subgroups (13):

$$WS \leftarrow (W \rhd\lhd DS) \times SB, \quad (13)$$

where $S \times R$ – relational operation of the Cartesian product of the relations S and R.

The SS relation contains a list of all classes conducted with the division of groups into subgroups, with teachers attached to them (14):

$$SS \leftarrow \pi_{ids,idg,idt,sub}(WS \rhd\lhd TL) \quad (14)$$

In relation to SS, the primary key is composite: (ids, sub).

Then the ratio ZS, which describes all possible options for the schedule for classes in subgroups, is represented by the formula (15) – (16):

$$TT4 \leftarrow SS \bowtie TT \bowtie PG \quad (15)$$

$$ZS \leftarrow \pi_{ids,idg,idt,sub,idp,ida}(TT4) \quad (16)$$

where $TT4$ is intermediate relational variable.

The relation W contains all of the academic group lessons according to the curriculum, including those highlighted as streaming lectures and laboratory sessions conducted by subgroups. To obtain the final list of classes that are conducted for one whole group, and schedule these classes, we need to subtract from the relation W the tuples that entered the relations WRR and WS (17) – (20) using the relational difference operation:

$$W1 \leftarrow \pi_{ids,idg,idd,idh,num}(WRR), \quad (17)$$

$$W2 \leftarrow \pi_{ids,idg,idd,idh,num}(WS), \quad (18)$$

$$W3 \leftarrow \pi_{ids,idg,idd,idh,num}(W), \quad (19)$$

$$WG \leftarrow (W3 - W2) - W1, \quad (20)$$

where $W1$, $W2$, $W3$ are intermediate relational variables.

Using the WG relation, we get the SG and ZG relations – a list of group lessons with attached teachers, and a list of all possible options for placing academic groups in the lessons schedule (21) – (23):

$$SG \leftarrow \pi_{ids,idg,idt}(WG \bowtie TL) \quad (21)$$

$$TT4 \leftarrow SG \bowtie TT \bowtie PG \quad (22)$$

$$ZG \leftarrow \pi_{ids,idg,idt,idp,ida}(TT4) \quad (23)$$

where $TT4$ is intermediate relational variable.

The curriculum may include disciplines that have an odd number of academic hours per week for some types of classes, for example, 3 hours of lectures, or 1 hour of practical training (seminar). In this case, lesson by number np will be held once every two weeks (2). It is necessary to select these classes from the general list, since the algorithm for setting them in the schedule has its own characteristics.

To distinguish between weekly and biweekly lessons, add an attribute v of type bit (bool) to the W relation.

For lessons that are held weekly, $v = 1$, for the rest – $v = 0$. The values of the v attribute for the tuples of the relation W are assigned by the stored procedure, which forms the relation W from the relation Q.

The v attribute will also be included in all projections of the W relation, that is, in the relations WR, WG, WS, SR, SG, SS, ZR, ZG, ZS.

Then the formulas calculating the relations $ZR1$, $ZG1$, $ZS1$, for lessons that are held weekly, take the following form (24) – (26):

$$ZR1 \leftarrow \pi_{idsr,idr,idt,idp,ida}(\sigma_{v=1}(ZR)) \quad (24)$$

$$ZG1 \leftarrow \pi_{ids,idg,idt,idp,ida}(\sigma_{v=1}(ZR)) \quad (25)$$

$$ZS1 \leftarrow \pi_{ids,idg,sub,idt,idp,ida}(\sigma_{v=1}(ZS)) \quad (26)$$

where $\sigma_F(R)$ – a relational operation of fetching tuples from a relation R satisfying predicate F.

To describe lessons that are held once every two weeks, we introduce an auxiliary relation $WK(w)$ with two tuples: $w = 1$ and $w = 2$ – an odd-numbered academic week and an even-numbered week. Then the formulas calculating the ratios $ZR0$, $ZG0$, $ZS0$, for classes that are held biweekly, take the following form (27) – (32):

$$R0 \leftarrow \pi_{idsr,idr,idt,idp,ida}(\sigma_{v=0}(ZR)) \quad (27)$$

$$G0 \leftarrow \pi_{ids,idg,idt,idp,ida}(\sigma_{v=0}(ZG)) \quad (28)$$

$$S0 \leftarrow \pi_{ids,idg,sub,idt,idp,ida}(\sigma_{v=0}(ZS)) \quad (29)$$

$$ZR0 \leftarrow R0 \times WK \quad (30)$$

$$ZG0 \leftarrow G0 \times WK \quad (31)$$

$$ZS0 \leftarrow S0 \times WK \quad (32)$$

where $R0$, $G0$, $S0$ are intermediate relational variables.

The relations SR, SG, SS contain a complete list of lessons that need to be scheduled. If we write this list of lessons in one relation S, then it must contain all list of the attributes that are present in at least one of these relations: $S(ids, idsr, idg, idr, sub, idt, v)$.

Only two attribute, idt, v from the entire set of attributes is defined for all tuples of the relation S. The remaining attributes are defined for some subsets of the tuples of this relation.

The relations $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$, $ZS0$ make up the fuller space of possible states of the lesson schedule. This relations are sections of a given space. Dividing the common space into several sections is necessary to optimize the work of the class scheduling program.

If the space of possible states of the lesson schedule is described using one relation, then it should include all the attributes that are present in at least one of the 6 relations $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$, $ZS0$, that is, the resulting relation Z will have the following set of attributes: $Z(ids, idsr, idg, idr, sub, idt, idp, ida, w)$

Only 3 attributes from this set: $idt, idp, ida,$ – are defined for all tuples of the relation Z. The remaining attributes are defined for some subsets of the tuples of this relation.

Undefined attribute values can be specified either with the NULL value, or we can provide some specific numeric values for them. Both of these methods lead to complex predicates in the WHERE clause of the SELECT statement when fetching data from a relation. But for relation S, the situation is more complicated, since among the set of attributes there are two keys that must uniquely identify the lesson that needs to be scheduled: $ids, idsr$.

The union of a selection of data (SELECT … UNION SELECT …) from several relations with a simple structure and low cardinality is faster than a selection of data from one large table with several complex predicates in the WHERE clause. Thus, the representation of the space of possible states of the lessons schedule in the form of 6 sections for different types of lessons provides a simpler design of queries to the database, as well as faster execution of them.

2.1.1. Lists of study pairs allowed for lessons

The cardinality of the relations $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$, $ZS0$ depends on the cardinality of the derived relations PT, PG, PA and SA.

The set of admissible study pairs for the lesson is equal to the intersection of the set of admissible study pairs of the group for which the classes are held, the teacher who conducts the classes and the classroom in which the classes are held. In formulas (4), (5), (11), (15), (22), the natural join operations applied to the relations PT, PG, PA give the same result as the applying of the sets intersection operations with a simpler syntax of relational algebra formulas and a simpler notation of the SELECT command. The connection of relations is performed according to the condition of equality of the identifiers of study pairs for a given group, a given teacher and a given classroom.

If classrooms are used for class only and are available on any class, the PA is the Cartesian product of the list of classroom IDs by the list of study pair IDs. The PG relation can also be generated automatically if, for all academic groups, all study pairs of all days of the learning week are valid for classes. If for some groups there is a predetermined set of study pairs for which lessons cannot be assigned, this information should be specified as a source data. For example, on a given day of the week, the group is assigned duty in the laboratory, greenhouse, and the like.

The PT relation is usually formed from the wishes and requirements of teachers for their timetable. The easiest way to fill in the PT data is for the teachers to list the desired pairs for the entire learning week.

It is necessary to distinguish between the wishes and requirements of teachers. All teachers can put forward their wishes. Requirements can only be formulated by teachers with a sufficiently high rating.

If there are wishes and requirements of teachers, the space of all possible states of the lesson schedule is divided into two subspaces: the subspace of desired states (SDSLS) and the subspace of admissible states (SASLS).

Teachers can specify the desired and acceptable study pairs for classes not only in the form of a set of specific training pairs on specific days of the learning week, but also in the form of a range of possible values for the number of pairs during the day and the number of study days during the week. For example, the wishes and requirements of a teacher may look like this: no more than 3 study days a week, but preferably 2 days.

In this case, before the start of the scheduling of classes, all study pairs are considered desirable for the teacher. Therefore, all the lessons of a given teacher fall into the subset of the desired states of the lesson schedule, that is, they are tuples of relations $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$, $ZS0$ – depending on the type of lesson.

Let us denote the relations of the subset of admissible states of the schedule of classes through $YR1$, $YG1$, $YS1$, $YR0$, $YG0$, $YS0$ – also depending on the type of lesson: streaming lecture weekly, a group lesson weekly, a subgroup lesson weekly, streaming lecture biweekly, a group lesson biweekly, a subgroup lesson biweekly.

For the example given above, with the teacher's wishes, after putting at least one study pair in the schedule of this teacher's lessons on any two days of the study week, all study pairs of all other days of the week move from the desired category to the acceptable category. That is, the tuples of relations $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$, $ZS0$, corresponding to the lessons of this teacher on the remaining days of the study week, must be transferred to the relations $YR1$, $YG1$, $YS1$, $YR0$, $YG0$, $YS0$.

When placing each lesson in the schedule, the program first of all considers a subset of the desired states of the lesson schedule, that is, it selects from the relation $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$ or $ZS0$ (depending on the type of lesson), tuples corresponding to the given lesson.

For each variant of placing a lesson in the schedule, the optimality of the position of this lesson in the space of the week's study pairs is calculated – a certain measure of the quality of this state of the schedule for the teacher and for the group (all groups in the case of a streaming lecture). This measure of quality can take into account many factors: the number of study pairs per day; number of study days per week; the appearance of a "window" in the schedule of a group or teacher; closing the "window" in the schedule of the group or teacher; the need for a group or teacher to move from one educational building to another to conduct a lesson; number of lectures during the study day; the number of laboratory lessons during the study day, and more. Particular factors affecting the quality of setting a lesson in the schedule are summed up in different weights using fuzzy logic methods [10].

The optimality of the location of this lesson in the space of study pairs related to the SASLS is also considered. If the quality measure for some option from this subset exceeds all quality measures of lessons from SDSLS, then this option is chosen.

For the example described above, with the teacher's wishes and requirements for their class schedule specified as a range of values, some states of the lessons schedule may go into a subset of inadmissible states. For this example, after assigning classes to a teacher on any 3 days of the study week, all schedule states with teacher's study pairs on the remaining days of the study week become inadmissible.

These scheduling states should not be considered in further sequential scheduling of classes. However, it is not necessary to remove the corresponding tuples from the relations $YR1$, $YG1$, $YS1$, $YR0$, $YG0$, $YS0$, since it may be necessary to change the lessons schedule to optimize it or if a deadlock occurs when scheduling. Therefore, all the corresponding tuples are transferred to the relations $XR1$, $XG1$, $XS1$, $XR0$, $XG0$ or $XS0$ – in the subspace of inadmissible states of the lesson schedule (SISLS). The relations $XR1$, $XG1$, $XS1$, $XR0$, $XG0$, are not used in the planned scheduling of classes.

2.2. Schedule classes

Classes are placed in the schedule in turn, which is formed so that at each step the lesson with the least freedom of setting in the schedule is selected, or the teacher's lesson with the most dense work schedule is selected. The freedom to schedule classes and the density of the teacher's schedule are not constants calculated before the start of scheduling. These values are variables that are recalculated after each scheduled session.

The current freedom to schedule a lesson is determined by two values $K1$ and $K2$. $K1$ – the number of study pairs of the week for which a lesson can be scheduled. $K2$ – the number of places in two-dimensional space (study pair) - (classroom) to which a lesson can be assigned.

The current density of the teacher's schedule is calculated using two values $K3$ and $K4$. $K3$ – the number of free teaching pairs of the teacher, to which at least one of his unallocated classes can be assigned. $K4$ – the number of still unallocated lessons of the teacher. The calculations use the relative density of the work schedule of the teacher $C1$ and the relative freedom of distribution of the lessons of the teacher $K5$ $(33)–(34)$:

$$C1 = K4 \,/\, K3 \tag{33}$$

$$K5 = K3 - K4 \tag{34}$$

The $K1$ and $K2$ values are calculated for each lesson not scheduled. The $K3$ and $K4$ values are calculated for each teacher who has unscheduled lessons.

Since all possible options for setting classes in the schedule are tuples of relations $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$, $ZS0$ and $YR1$, $YG1$, $YS1$, $YR0$, $YG0$, $YS0$, the calculation of values is reduced to calculating the number of tuples in the specified relations with grouping either by class IDs or by teacher IDs.

From the variables $K1$, $K2$, $K5$, $C1$ a single value is not formed, according to which the remaining list of activities is sorted. A five-level sorting of the list of lessons is carried out:

1. Ascending $N1$, where $N1 = K1$, if $K1 < K01$, otherwise $N1 = K01$;
2. Ascending $N2$, where $N2 = K2$, if $K2 < K02$, otherwise $N2 = K02$;
3. Ascending $N5$, where $N5 = K5$, if $K5 < K05$, otherwise $N5 = K05$;
4. Descending CC, where $CC = C1$, if $CC > C01$, otherwise $CC = C01$;
5. Descending Ngr, where Ngr is the number of groups in the stream for which the lesson is held; for lessons in groups and subgroups $Ngr = 1$.

$K01$, $K02$, $K05$, $C01$ are empirical coefficients, the values of which need to be determined in further practical calculations.

When any lesson is added to the schedule, it goes into the category of realized states of the lesson schedule, and the corresponding tuple from the relation $ZR1$, $ZG1$, $ZS1$, $ZR0$, $ZG0$ or $ZS0$ is transferred, respectively, to the relation $RR1$, $RG1$, $RS1$, $RR0$, $RG0$ or $RS0$ – to the subspace of the realized states of the lesson schedule (SRSLS). All other states of the same lesson – all tuples from the same section of the subspaces of desired, admissible and inadmissible states of lesson schedules (SDSLS, SASLS, SISLS) with an identical lesson key ($idsr$, ids or (ids, sub)) are transferred to the subspace of unrealized lesson schedule states (SUSLS), that is into one of the relations $UR1$, $UG1$, $US1$, $UR0$, $UG0$ or $US0$ (depending on the type of lesson). If it is necessary to change the schedule, each tuple from SRSLS can be exchanged with a simple set of operations for any tuple with the same primary key of the lesson from the corresponding section of SUSLS.

It is also necessary to remove from the SDSLS, SASLS, SISLS those states of the lesson schedule that cannot be realized after the current lesson is added to the timetable according to the principle: any teacher cannot conduct more than one lesson at the same time; in any group (stream, subgroup) more than one lesson cannot be conducted at the same time; no classroom can have more than one lesson at a time. All tuples that violate this principle are transferred to the subspace of unrealizable states of the lesson schedule (SVSLS) – in the relations $VR1$, $VG1$, $VS1$, $VR0$, $VG0$, $VS0$. In this case, the search for unrealizable states of the lesson schedule is carried out in all sections of the SDSLS, SASLS, SISLS subspaces.

If the schedule is changed to replace a tuple from some SRSLS section with a tuple from the corresponding SVSLS section, a much more complex set of operations must be performed than when replacing this tuple with a tuple from the corresponding SUSLS section.

3. Conclusions

A mathematical model of the space of all possible states of the lesson schedule in HEIs has been developed. The model is developed in terms of relational algebra. The solution to the problem of automated scheduling of classes is found by means of a relational DBMS.

The initial solution to the problem is found using an iterative process, which at each step chooses the lesson with the least freedom to schedule or the teacher's lesson with the most tight schedule. Freedom of scheduling classes and the density of teachers' schedules are variables that are calculated at each iteration for a subset of classes not yet scheduled. The list of unscheduled lessons is ordered at each step using empirical coefficients, the values of which will need to be determined during a series of practical calculations. The selected lesson is scheduled in the optimal place in a three-dimensional state (day of the school week) - (number of the study pair) - classroom. Optimality is determined by a set of criteria using fuzzy logic methods. An empirical weighting factor is assigned to each criterion. For each possible position of the lesson in the schedule, the criteria are summarized. The value of empirical weighting factors also needs to be determined in a series of practical calculations.

The efficiency of the algorithm based on the proposed model is determined by the developed database schema, which excludes transactions of high complexity.

Several factors contribute to reducing the complexity of transactions:

1. All join operations between database relations are performed prior to starting the scheduling process.
2. Different types of lessons refer to different relations, which allows replacing a data selection from one relationship with a complex structure and high cardinality with several simple selections (connected by the UNION operation) from relations of a simpler structure and lower cardinality; this eliminates NULL-valued attributes and complex predicates in the WHERE clause of the SELECT statement.
3. Storing all possible options for the schedule of classes in the form of physical database tables, and not in the form of their virtual representation – nested queries of various types – increases the volume of the database, but significantly reduces the number of operations required for setting each class in the schedule.

Storing all possible classroom scheduling options in physical database tables also makes it easy to compute characteristics that increase the likelihood of an optimal class schedule, such as the number of unallocated lessons that might "close the window" in a group's or teacher's schedule.

Storing unrealized options for placing a lesson in the schedule, as well as unrealizable options (options that are superimposed on already distributed activities) in the form of separate subspaces of the common space of all possible states of the lesson schedule allows you to adjust the schedule and optimize it if necessary.

4. Acknowledgements

In assessing the current state of the problem of automated scheduling of classes in the HEIs and its relevance, the analysis performed by R. A. Oude Vrielink, E. A. Jansen, E. W. Hans, J. van Hillegersberg turned out to be very useful [1].

5. References

[1] R. A. Oude Vrielink, E. A. Jansen, E. W. Hans, J. van Hillegersberg, Practices in timetabling in higher education institutions: a systematic review, Ann Oper Res, doi 10.1007/s10479-017-2688-8, URL: https://d-nb.info/1149213442/34

[2] Burke, E. K., Cowling, P., Silva, J. L., & McCollum, B. Three methods to automate the space allocation process in UK universities. PATAT 2000, LNCS 2079, pp. 254–273, 2001. URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.720.1897&rep=rep1&type=pdf

[3] Nurlida Basir, Waidah Ismail, Norita Md Norwawi, A Simulated Annealing for Tahmidi Course Timetabling. Procedia Technology 11 (2013) 437 – 445. URL: https://www.sciencedirect.com › science › article › pii › pdf

[4] Astakhova I.F., Firas A.M., Drawing up the schedule of studies on the basis of genetic algorithm, Proceedings of VSU, Series: Systems analysis and information technologies, 2013, №2, pp. 93–99. URL: http://www.vestnik.vsu.ru/pdf/analiz/2013/02/2013-02-17.pdf

[5] Mehdi Yazdani, Bahman Naderi, Esmaeil Zeinali, Algorithms for university course scheduling problems, Tehnički vjesnik 24, Suppl. 2(2017), 241-247. URL: https://hrcak.srce.hr/186061

[6] A. Oner, S. Ozcan and D. Dengi, Optimization of university course scheduling problem with a hybrid artificial bee colony algorithm, 2011 IEEE Congress of Evolutionary Computation (CEC), 2011, pp. 339-346, doi: 10.1109/CEC.2011.5949638.

[7] Sally C. Brailsford, Chris N. Potts, Barbara M. Smith: Constraint satisfaction problems: Algorithms and applications. Eur. J. Oper. Res. 119(3): 557–581 (1999)

[8] Burke, E. K., McCollum, B., Meisels, A., Petrovic, S., & Qu, R. A graph-based hyper-heuristic for educational timetabling problems. European Journal of Operational Research, 176(1), 177–192.

[9] Database Systems: A Practical Approach to Design, Implementation, and Management, 3rd Edition, Thomas Connolly, Carolyn Begg, 2002.

[10] Tutorial on Fuzzy Logic. URL: https://coderlessons.com/tutorials/mashinnoe-obuchenie/osnovy-iskustvennogo-intellekta/11-uchebnik-po-nechetkoi-logike