Informatics in Education, 2009, Vol. 8, No. 1, 69–84

Optimization Algorithms in School Scheduling Programs: Study, Analysis and Results

Lina PUPEIKIENĖ
Institute of Mathematic and Informatics
Akademijos 4, LT-08663 Vilnius
e-mail: linapupe@gmail.com

Denis STRUKOV
Vilnius Gediminas Technical University
Saulėtekio al. 11-424, SRL-I, LT-10223, Vilnius
e-mail: dstrukov@gmail.com

Vytenis BIVAINIS
Kaunas University of Technology
Studentų 50-411, LT-51368 Kaunas
e-mail: vytenis@gmail.com

Received: May 2008

Abstract. To create good and optimal school schedule is very important and practical task. Currently in Lithuania schools are using two programs for making the school schedule at the moment. But none of these programs is very effective. Optimization Department of Lithuanian Institute of Mathematics and Informatics (IMI) has created "School schedule optimization program". It has three optimization algorithms for making best school schedule. A user can choose not only few optimization options and get few optimal schedules, but some subjective and objectives parameters. The making of initial data file is advanced in this program. XML format is used for creating initial data file and getting all optimal results files.

The purpose of this study is to analyze used optimization algorithms used in "School schedule optimization program" and to compare results with two most popular commercial school scheduling programs in Lithuania.

Keywords: optimal scheduling, XML, school timetable, individual schedules.

1. Introduction

1.1. *Optimal School Scheduling Problem*

We all are making a schedule that helps to organize our everyday. Making a schedule for any organized activities, one considers a sequence of tasks and a list of resources. Resources include tools, machines, materials, work force etc. Not only we can make our own schedule. Our employers are making schedule for us. And it is more difficult to

make a schedule for organization, than making a schedule for our everyday life. A failure to make a schedule can bring many negative results. Like example, we are using scheduling of traditional school. It is very close and actual task of real life. Here the sequence of teaching subjects, regarded as tools, can be changed. One needs to reduce the sum of gaps ("empty" hours) in the teacher schedules. There should be no gaps for students. Different classes are considered as different tasks. The classrooms, including the computer and physics rooms and studies, are the limited resources (Gaidukevičienė and Kurilovas, 2005).

The most difficult in today school is to make a schedule for the high school. Here eleventh and twelfth grade pupils are choosing several subjects from the list of available ones (for example, 10–14 from 60). Maximal hours, which pupils can choose for one subject, are described in Table 1. This means that each student works by his own schedule. We search for the most convenient feasible schedule. Penalty points evaluate the inconveniences. We consider the case where the objective is the number of "empty" hours,

Table 1

Subjects and courses table for high school education program

Subjects	Standard course		Expanded course	
	Max lessons per week	Max hours per year	Max lessons per week	Max hours per year
Religion	1	35	–	–
Ethic	1	35	–	–
Native language	4	140	5	175
Lithuanian language	4	140	5	175
Foreign language (1)	3	105	4	140
Foreign language (2)	2	70	3	105
History	2	70	3	105
Geography	2	70	3	105
Integrated social science course	2	70	–	–
Mathematic	3	105	4–5	155
Information technologies	1	35	2	70
Biology	2	70	3	105
Physic	2	70	3–4	124
Chemistry	2	70	3	105
Integrated nature science course	2	70	–	–
Arts	2	70	3	105
Technology	2	70	3	105
Integrated arts and technology course	2	70	3	105
Generic sport	3	105	4	140
Sport branch for choose	3	105	–	–
Subjects for choose	–	–	–	–
Projects	–	–	–	–

when the teachers or pupils wait for the next scheduled lectures. These empty hours in the school are called "gaps" for short. We search for such schedule that reduces the sum of all gaps considering the schedules of eleventh and twelfth class of the comprehensive school. Other factors are school-specific and should be included adapting the software to specific schools (Mockus, 2000).

To make a good schedule for high school, we have five constrains:

- at any given moment a teacher can deliver only one lesson,
- at any given moment a student can take part at only one lesson,
- no "pupil gaps" are allowed,
- the "double" lectures (the sequence of two lectures of the same subject) are not allowed (there are some exceptions),
- the number of lecture hours per day is limited.

The biggest problem is to form regular classes consisting of pupils who have different choices and not to mix their grade levels. We search for the most convenient feasible schedule. The inconveniences are evaluated by penalty points and objectives parameters.

The most popular commercial scheduling software in Lithuania are:

1) MIMOSA by "MIMOSA software OY";

2) "aSc Timetables 2008" by "Applied Software Consultants s.r.o.".

2. School Schedule with MIMOSA Program

"Mimosa Software Ltd." is a privately owned, debt-free Finnish company, which was founded in 1986. This company is dedicated to the development and marketing of high-class school timetable software in all kind of educational and private organizations (MIMOSA). Product of this company is called "MIMOSA". This program is very difficult and not very a user friendly. It is difficult to compile data input required by the program. In addition, it is very tricky to understand where to input just pupils, teachers, classes and where to put pupils choices. It is difficult to understand how to separate some pupils by their selected subjects in grade. If a user makes a mistake, it is difficult to be tracked and corrected. This program has a tricky documentation. It is difficult to understand how to work with the program especially if a user is not IT specialist. If a user wants to make a good use of the software, he must follow some extra courses. About 50% of the schedules produced by MIMOSA depend on the human scheduler operating MIMOSA program. Therefore, comparison with other systems is very difficult. In this sense, the MIMOSA and other similar systems can be regarded mainly as "Support Systems". In the program, it is possible to choose few optimization algorithms (Algorithm1, Algorithm2 or both). We did not felt difference between them. It looks like MIMOSA company is using algorithms similar to "Monte-Carlo" (Pupeikienė, 2005).

3. School Schedule with "aSc Timetables 2008" Program

"aSc Timetables 2008" is very easy to use. Every data group is added step by step: first subjects; then classes; then classrooms; then teachers with selections. When a user adds

a teacher, he must add details such as which subject the teacher teaches and for which classes. If teacher teaches few classes – a user must show it separately (write subject name and how many lessons he has that many times, how many classes he teaches) (Tarptautinis verslo tinklas). However, there are no choices for pupils. If some pupils want to learn other things, it is impossible to specify that in the program. We have used demonstration version. The difference between demonstration and regular versions is that File saves in demonstration version are disabled. It has a good online documentation. It is explained how to use the program step by step. For an average a user, it is not difficult to learn how to work with this program. After some experiments with big high school schedule (there were about 350 pupils and 66 teachers) we got results that are displayed in Fig. 1. In this figure are sowed three charts. Every chart shows three levels of difficulties to make schedule (this is possible to regulate in the program). With every level, we have done three schedules with different numbers of lessons per day. In these charts, we can see how many different schedules have done program and how long it was worked to find the best schedule.

4. School Schedule with "School Schedule Optimization Program"

Recently we have been working on the program called "School schedule optimization program". This program creates an initial schedule of profiled classes. After that, process is followed by optimization of this schedule, which allows minimization of the number of "windows" for pupils and for teachers. As a pupil may choose some of his/her favorite subjects, the schedule must more or less be made individually for every pupil. The schedule must be acceptable not only for pupils; it must also be convenient for teachers. Some physical constraints also should be considered. The schedule can not be considered feasible if:

- a pupil or a teacher participates in two lessons at the same time,
- a pupil or a teacher works a full day with no break (except for breaks between lessons),
- a pupil or a teacher has many breaks during a day (every second or third lesson),
- a teacher has lessons on his "free" day (a lot of teachers have several jobs and their lessons should not overlap).

This program facilitates the schedule making for profiled classes. A user participates only in preparation of the data file, sending the data file to the server and getting the results. The program makes the schedule according to indicated restrictions, performs optimization, calculates penalty points for each schedule variant and displays the best schedule according to the number of penalty points. The schedule of traditional school is optimized using the permutation algorithms. In this program are used three optimization algorithms:

- Monte-Carlo;
- Simulated Annealing;
- Bayes Heuristic Approach.

More about these algorithms describes Section 5. The program is written in the Java programming language, so it could run not only under Windows but also under Linux operating system. Initial data file is created on XML format. This allows "School schedule optimization program" to be used in any operating system.

5. Comparison of Methods and Results in "School Schedule Optimization Program"

"School schedule optimization program" has three optimization algorithms:

1. "Monte-Carlo".
2. "Simulated Annealing".
3. "Bayes Heuristic Approach".

5.1. *The "Monte-Carlo" Optimization Algorithm*

The simplest approximate algorithm is Monte-Carlo (MC):

1) select a current collection I, $N = 1, \ldots, K$ with probability to make new collection from new start point;
2) record the best current collection I^*;
3) stop, if the number of iterations $N = K$,

where N – current iteration, K – number of iterations.

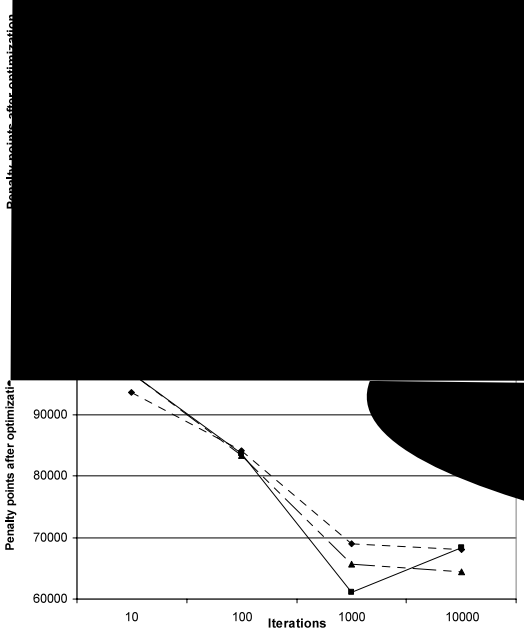
During working time of algorithm, in the memory is recorded the best current collection. It will be the best if the algorithm will not find better. Program should keep in the memory up to 2^N of previous samples. In cases with replacement, it does not know when the optimum is reached. Therefore MC stops, if the time allocated for the optimization is exhausted. This algorithm converges to an exact solution with probability one, if the number of iterations $K \to \infty$. However, the convergence is very slow, nearly logarithmic. Note, that the time $T = CN^m$ is not a limit for MC with replacement, where the (Mockus, 2000).

We have done some experiments with big highs school schedule (there was about 350 pupils and 66 teachers). After some experiments, we got results (Table 2). In the tables, we are showing penalty points number, what we got before optimization. In every table, we have used different fixed number of classrooms and lessons per day. In every table, we have used same number of iterations and "Override". Parameter "Override" shows number of iteration. After these number of iteration algorithm must to stop working if it cannot found better result during this time. In the tables, we marked optimal penalty points after optimization algorithm was finalizing the work. These results are showed in the chats (Fig. 2).

You can see from the tables and charts, that the best results was found, when algorithm was worked 10 000 times (iterations) and with "Override" = 10.

Optimization Al

Override	10
	In
1	18012
5	18230
10	18240
	In
1	93510
5	96750
10	96750



- only one permutation is made, $I(m) = 1$;
- the direct heuristic is used:

$$h_i = c(m^i) - c(m^0), \tag{1}$$

where $c(m^0)$ – best penalty points of collection m, $c(m^i)$ – current penalty points of collection m;

- if $h_i \geqslant 0$, the new collection $I(m^i)$ is selected with probability one;
- if $h_i < 0$, the new collection is selected with probability r_i that declines exponentially:

$$r_{i+1} = \begin{cases} \mathrm{e}^{\frac{-h_{i+1}}{x/\ln(1+(i+1))}}, & \text{when } h_{i+1} > 0, \\ 1, & \text{otherwise,} \end{cases} \tag{2}$$

where N – is the iteration number, x is the "initial temperature".

One difference of this formulation from traditional simulating annealing algorithms is that we optimize the parameter for some fixed number of iterations $N = K$. We disregard the asymptotic behavior. The asymptotic properties are beyond the scope of the Bayesian Heuristics Approach (Mockus, 2000).

Second difference from the traditional SA is that we optimize the parameter x for some fixed number of iterations $N = K$. In this case, the cooling schedule should be optimized, too. A natural way to do it is by introducing the second parameter x_2. This transforms expression

$$r_{i+1} = \begin{cases} \mathrm{e}^{\frac{-h_{i+1}}{x_1/\ln(1+x_2*(i+1))}}, & \text{when } h_{i+1} > 0, \\ 1, & \text{otherwise,} \end{cases} \tag{3}$$

where $x_1 \geqslant 0$ – defines an "initial temperature" of SA, $x_2 \geqslant 0$ – describes a "cooling rate" of SA.

We have done some experiments with the same big school schedule that we were taking for "Monte-Carlo". After some experiments, we got following results (Table 3 and Fig. 3).

We have done experiments when the number of iteration was 100 and 1000. Both times, we used same classrooms numbers and same lessons per day. We have calculating penalty points when number of classrooms is decreasing and number of lessons is most popular as it can be for such big schedule. We marked penalty points before starting the algorithm. As we earlier described, SA algorithm has two optimization parameters – x_1 and x_2. We have fixed optimal penalty points when x_1 – maximal value, x_2 – minimal (column 5) and when x_1 – minimal value, x_2 – maximal (column 6).

You can see from the table and chart, that the better results are always when x_1 is high and x_2 is low. The best results we got when number of classrooms and number of lessons per day match as optimal as possible for such big schedule.

Table 3

Results of SA optimization with different restrictions

Number of iterations	Classrooms	Lessons per day	Initial penalty points	Optimal penalty points, when:	
				$x_1 = 10000$ $x_2 = 1$	$x_1 = 100$ $x_2 = 10$
100	45	8	96720	86770	94880
100	35	7	184300	185355	193975
100	35	8	129000	119305	129695
100	35	9	101000	67670	79840
100	35	10	97500	71060	72630
100	30	8	155455	145395	153830
100	30	9	123955	104215	105835
1000	45	8	96720	72660	87940
1000	35	7	184300	166240	176505
1000	35	8	129000	113580	118785
1000	35	9	101000	68585	71110
1000	35	10	97500	50300	64625
1000	30	8	155455	129570	143290
1000	30	9	123955	95220	96210

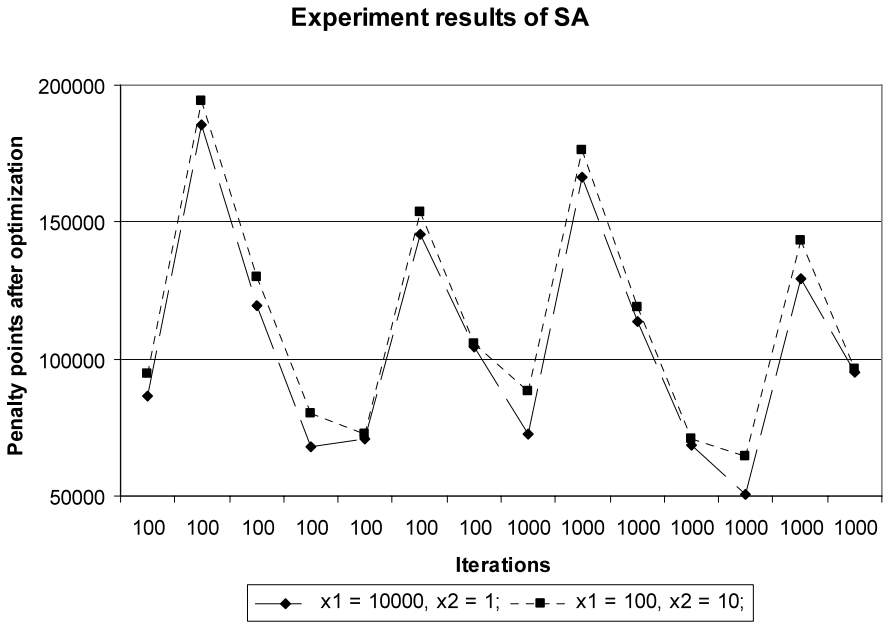
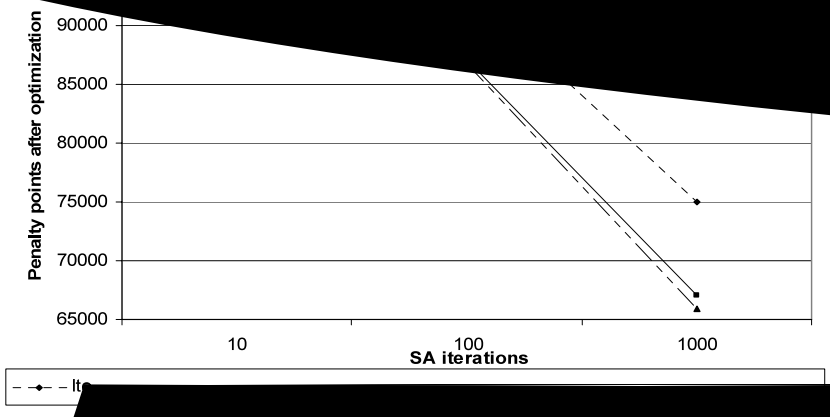


Fig. 3. Received penalty points with SA algorithm, when initial restrictions were different.


Penalty points after optimization
90000
85000
80000
75000
70000
65000
10
100
1000
SA iterations

You can see from the table and chart that the best result of the big schedule will be found when program will search as much as possible.

5.4. *Findings of Comparison All Three Optimization Algorithm*

First and simplest algorithm is MC. Because it is a permutation algorithm, it is the best to use when "Override" is very big. If "Override" is small, algorithm will stop if it will not find better result on the few more steps. So the best schedule can be not found.

Second algorithm is SA. If "Initial temperature" is high and "Cooling rate" is low, then the searching of the best schedule will be fast. Algorithm will search for the best solution in small steps. This algorithm is trickier while it has probability. The probability allows to jump from the point where program found the best schedule on current moment and to search more. If it is better solution – algorithm will found. This algorithm will stop only when it will reach the last iteration. Therefore, to find the best result it should be given as much as possible iteration from the beginning.

Third algorithm is Bayes. It uses SA so many times, as a user will set. For SA algorithm should be given limits of "Initial temperature" and "Cooling rate". This is so, that algorithm should search schedule with different initial parameters ("Initial temperature" and "Cooling rate") in every Bayes iteration to find the best one. Iterations of SA are for searching the best results only with one pare of initial parameters. On every Bayes iteration, initial parameters will change and SA algorithm will search the best schedule with new initial parameters. When program is finalizing the last iteration, it will choose the best schedule from all for save and to show. If Bayes and SA will have very much iteration, program will find the best solution.

Optimal result, what we got with Bayes, we can get with SA too, but it needs more time and more experiments.

6. Visual Results

6.1. *Results Obtained Using "aSc Timetables 2008"*

Since the initial data had groups divided into subgroups, the program displayed the results in subgroups, too. This can be seen in Fig. 5, where the schedules of all thirteen groups are displayed. Also this program allows seeing the whole schedule of pupils (pupil groups), teachers (work schedule of all teachers in one window) and classrooms (schedule of all classrooms in one window). However this program does not display the schedules of individual groups and subgroups. Moreover, in this program we cannot see any penalty points. That makes difficulties to compare results with "School Schedule Optimization Program" and to prove with program is better.

6.2. *Results Obtained Using MIMOSA*

After optimizing high school schedule, we got following results (see Fig. 6). The results are not good, while normally pupil cannot learn sport more then four times per week and

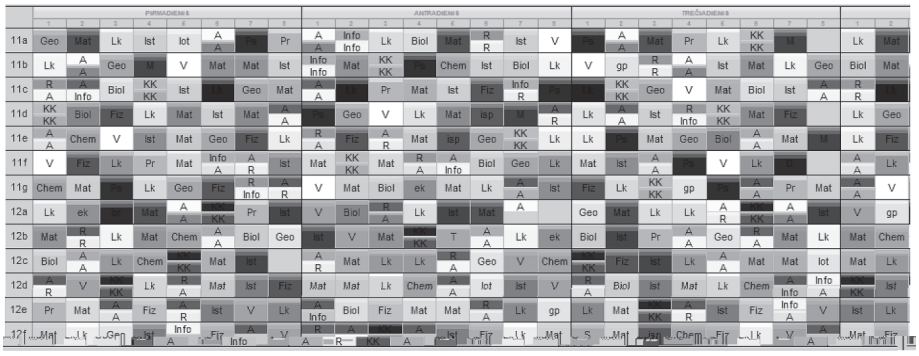


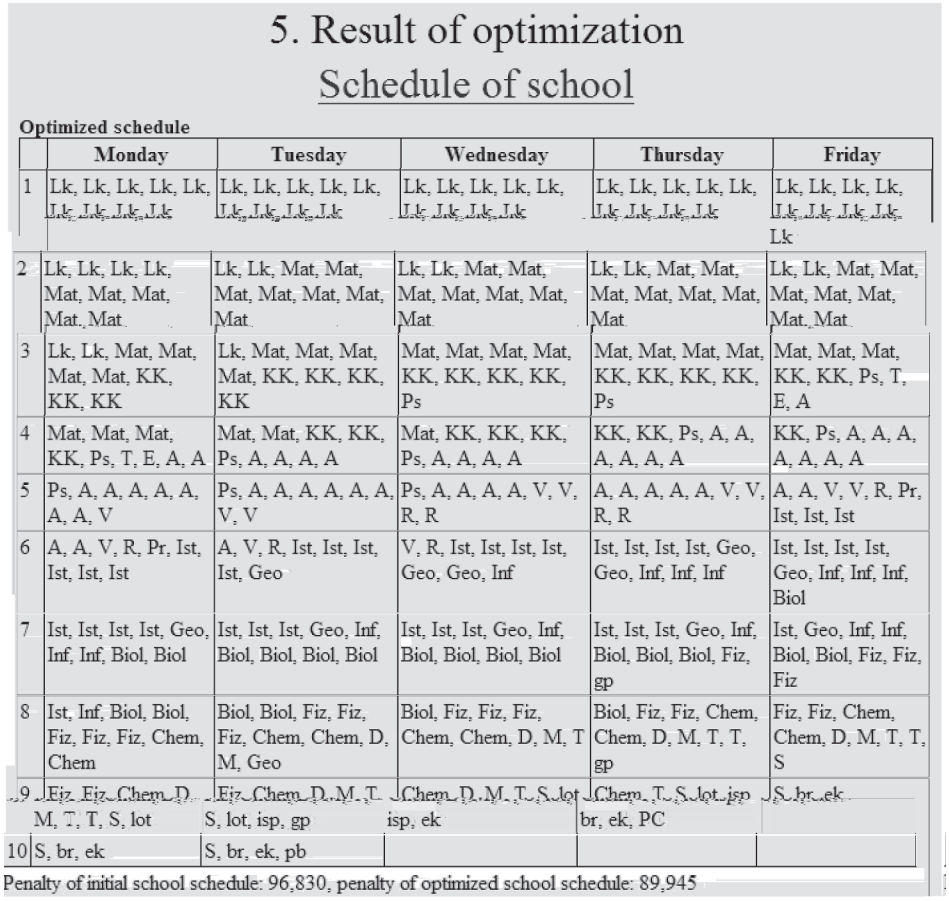
Fig. 5. Results of school schedule making for all groups using “aSc Timetables 2008”.

Fig. 6. Results of class schedule making for all groups using MIMOSA.

cannot learn chemistry more then three times per week. These numbers are described in Table 1. As you see in Fig. 6, pupil learns only sport and chemistry, but in the program it was fixed, that it can be only three lessons per week for each subject. We are experimenting with this program and sill learning how to make correct schedule.

6.3. *Results Obtained Using “School Schedule Optimization Program”*

After optimizing high school schedule, we got following results (see Fig. 7). These results are only for high school. Here every lesson is new group, created for new pupils by them wishes. These wishes were written into XML file and uploaded into the program. Program allows seeing all, optimized and/or initial, schedules of school, of each pupil and of each teacher. However, it still not shows schedule of all pupils and of all teachers on one time. This will be in closest future. On the end of schedule a user can see initial and optimized penalty points. To save all data and results, program allows sending ZIP



5. Result of optimization

Schedule of school

Optimized schedule

	Monday	Tuesday	Wednesday	Thursday	Friday
1	Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk	Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk	Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk	Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk	Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk, Lk
2	Lk, Lk, Lk, Lk, Mat, Mat, Mat, Mat, Mat	Lk, Lk, Mat, Mat, Mat, Mat, Mat, Mat, Mat	Lk, Lk, Mat, Mat, Mat, Mat, Mat, Mat, Mat	Lk, Lk, Mat, Mat, Mat, Mat, Mat, Mat, Mat	Lk, Lk, Mat, Mat, Mat, Mat, Mat, Mat, Mat
3	Lk, Lk, Mat, Mat, Mat, Mat, KK, KK, KK	Lk, Mat, Mat, Mat, Mat, KK, KK, KK, KK	Mat, Mat, Mat, Mat, KK, KK, KK, KK, Ps	Mat, Mat, Mat, Mat, KK, KK, KK, KK, Ps	Mat, Mat, Mat, KK, KK, Ps, T, E, A
4	Mat, Mat, Mat, KK, Ps, T, E, A, A	Mat, Mat, KK, KK, Ps, A, A, A, A	Mat, KK, KK, KK, Ps, A, A, A, A	KK, KK, Ps, A, A, A, A, A, A	KK, Ps, A, A, A, A, A, A, A
5	Ps, A, A, A, A, A, A, V	Ps, A, A, A, A, A, A, V, V	Ps, A, A, A, A, V, V, R, R	A, A, A, A, A, V, V, R, R	A, A, V, V, R, Pr, Ist, Ist, Ist
6	A, A, V, R, Pr, Ist, Ist, Ist, Ist	A, V, R, Ist, Ist, Ist, Ist, Geo	V, R, Ist, Ist, Ist, Ist, Geo, Geo, Inf	Ist, Ist, Ist, Ist, Geo, Geo, Inf, Inf, Inf	Ist, Ist, Ist, Ist, Geo, Inf, Inf, Inf, Biol
7	Ist, Ist, Ist, Ist, Geo, Inf, Inf, Biol, Biol	Ist, Ist, Ist, Geo, Inf, Biol, Biol, Biol, Biol	Ist, Ist, Ist, Geo, Inf, Biol, Biol, Biol, Biol	Ist, Ist, Ist, Geo, Inf, Biol, Biol, Biol, Fiz, gp	Ist, Geo, Inf, Inf, Biol, Biol, Fiz, Fiz, Fiz
8	Ist, Inf, Biol, Biol, Fiz, Fiz, Fiz, Chem, Chem	Biol, Biol, Fiz, Fiz, Fiz, Chem, Chem, D, M, Geo	Biol, Fiz, Fiz, Fiz, Chem, Chem, D, M, T	Biol, Fiz, Fiz, Chem, Chem, D, M, T, T, gp	Fiz, Fiz, Chem, Chem, D, M, T, T, S
9	Fiz, Fiz, Chem, D, M, T, T, S, lot	Fiz, Chem, D, M, T, S, lot, isp, gp	Chem, D, M, T, S, lot, isp, ek	Chem, T, S, lot, isp, br, ek, PC	S, br, ek
10	S, br, ek	S, br, ek, pb			

Penalty of initial school schedule: 96,830, penalty of optimized school schedule: 89,945

Fig. 7. Results of class schedule making for all groups using “School Schedule Optimization Program”.

file to user computer. In this ZIP file are all initial and optimized schedules what program shows during the working time.

6.4. *Comparison “School Schedule Optimization Program” with Other Programs*

The advantages of this program in comparing it with MIMOSA and “aSc TimeTables 2008”:

- it doesn’t require much effort, computer literacy or a lot of time to get acquainted with this program. It is rather easy to use, and a user does not spend all day in order to learn how to work with the program. All users needs to learn is how to compile the initial file and get familiar with setting requirements;
- the final initial schedule is quite good, while, for instance, two lessons that must go successively are set in the schedule without “windows”. But after optimization this requirement may not be fulfilled, if number of “windows” needs to be reduced;

- schedule is set quite fast even using low performance machine. a user doesn't need to rearrange the lessons manually or think where and what lesson is taking place so that lessons don't take place simultaneously;
- it is convenient to see full name of a subject and the sequence of subjects. Schedule is very easy to read, it is informative (with a name of teacher and classroom number);
- final results will appear not only on the screen but also in files for printing. Moreover, one may check not only the whole school schedule, but also individual schedules of teachers and pupils.

7. Final Conclusions

After analyzing commercial programs we recognized that each of them has some advantages and some disadvantages. However, in our opinion, none of them can be applicable in school. A user needs to set the final program result by him self, no matter if it is MIMOSA or "aSc TimeTables 2008". It is difficult to create individual high school schedule for every pupil. This is a very complicated process (especially with a huge number of teachers and pupils) and it may take a long time. Moreover, program MIMOSA displays only the schedules of teachers and pupils. It did not give me the general school schedule. Program "aSc TimeTables 2008" displayed only the general school schedule and classroom schedule. Both programs have week optimization algorithms. After optimization a schedule, a user manual can close many windows what was left.

"School schedule optimization program" is created in a way that helps a user, who only has to input initial data, choose optimization parameters and optimization algorithm. A user may put initial data into program "MS Excel" easily. Program itself will do all the optimization. If a user modifies the initial data, he doesn't need to worry about resetting the schedule or assuming a more optimal way than the program is displaying. All this may be done by resetting new optimization parameters. After work is finished, program will display the results not only on the screen but also in "zip" file. This "zip" file a user can send to his computer and to analyze it when he wants. It helps to choose one best result from many of them.

The trials of "School schedule optimization program" had showed better results if comparing with commercial programs MIMOSA and "aSc TimeTables 2008", while could make individual schedule for every pupil and optimize them with three algorithms.

References

Alaburdienė, R. *Tvarkaraštis be MIMOSA – tvarkaraštis be ateities?* Švietimo informacinių technologijų centro kompiuterinis leidinys ne tik mokytojams „Veidrodis".

Alaburdienė, R., Dovidauskaitė, S. and Nekiūnienė, V. *Pamokų tvarkaraščiai*. `http://www.soften.ktu.lt/~mockus/mimosa/alaburd.htm`

aSc TimeTables. `http://www.asctimetables.com`

Berger, J. (1985). *Statistical Description Theory and Bayesian Analysis.* Springer-Verlang, Berlin, Heidelberg.

Chorbev, I., Dimitrovski, I., Mihajlov, D. and Loskovska, S. (2007). Hybrid heuristics for solving the constrains modeled high school scheduling problem. In *EUROCON, 2007. The International Conference on "Computer as a Tool"*, Warsawa. 2242–2249.

Cooper, T.B. and Kingston, J.H. (1993). The solution of real instances of the timetabling problem. *The Computer Journal*, **36**(7), 645–653.

DeGroot, M. (1970). *Optimal Statistical Decisions.* McGraw-Hill, New Yourk.

Gaidukevičienė, R. and Kurilovas, E. (2005). Comparative study of profiled school scheduling programs in Lithuania. *Informatics in Education*, **4**(1), 19–42.

MIMOSA. `http://www.mimosasoftware.com`

Mockus, J. (2000). *A Set of Examples of Global and Discrete Optimization*: *Application of Bayesian Heuristic Approach*. Kluwer Academic Publishers. `http://soften.ktu.lt/~mockus/`

Mockus, J., Eddy, W., Mockus, A., Mockus, L. and Reklaits, G. (1997). *Bayesian Heuristics Approach to Discrete and Global Optimization*. Kluwer Academic Publishers, Dordrecht-London-Boston.

Pupeikienė, L. (2005). School schedule optimization program. *Information Technology and Control*, **34**(2), 161–170.

School Scheduling, Discussions About Applications of MIMOSA and GMJ1 systems. `http://www.soften.ktu.lt/~mockus/newalg.pdf`

Simulated Annealing. `http://en.wikipedia.org/wiki/Simulated_annealing`

Tarptautinis verslo tinklas. `http://www.ibn.lt/lit/index.htm`

L. Pupeikienė has graduated from Kaunas Technological University with a bachelor of science in informatics in 2002. She graduated from the same university graduated with a master of science in program engineering in 2004. Currently she is a student of informatics in Institute of Mathematics and Informatics. She is working as assistant in Viliaus Gedimino Technical University.

D. Strukov has graduated from Viliaus Gedimino Technical University in 2007 with a bachelor of informatics (informatics engineering). He also holds a bachelor degree in computer engineering obtained in Athlone Institute of Technology (Ireland) while participating in Socrates/Erasmus program in 2006. The field of research includes optimization theory and programming design patterns.

V. Bivainis has graduated from Kaunas Technological University with a bachelor of science in informatics in 2006. His bachelor work concerned analysis and forecasting of sales data. Currently he is a student of informatics in Kaunas Technological University. He has co-authored or read papers about SWOT modelling with fuzzy cognitive maps, school schedule optimization and integration of object and relational schemas, which is the topic of his master's thesis. He has been working as a programmer at Alna Software since 2004.

Optimizavimo algoritmų analizė mokyklos tvarkaraščių sudarymo programose

Lina PUPEIKIENĖ, Denis STRUKOV, Vytenis BIVAINIS

Labai svarbus, sunkus ir svarbus uždavinys profiliuotoje mokykloje yra mokyklos tvarkaraščio kūrimas ir optimizavimas. Šiuo metu yra sukurta daugelis tvarkaraščių sudarymo programų, tačiau nei viena iš jų nėra efektyvi. Jos teturi menkus optimizavimo algoritmus. Kuriama programa turi keletą optimizavimo algoritmų. Juos panaudodamas vartotojas turės galimybę gauti ne vieną, bet keletą optimalių tvarkaraščių variantų. Programoje įgyvendintas labai patogus pradinių duomenų failo parengimas XML formatu. Šį formatą puikiausia formuoja MS Excel programa. Vartotojui nebereikės suvedinėti ir grupuoti mokinių rankiniu būdu. Atskiruose MS Excel lapuose įvedęs mokytojus ir kiekvienos klasės mokinių pageidavimus, vartotojas naudodamasis šia programa automatiškai gaus suformuotus mokyklos tvarkaraščius. Programoje buvo įvertinti mokyklos parametrai, o tai leidžia programą pritaikyti mokykloms su skirtingomis vidaus taisyklėmis. Programoje yra įdiegti optimizavimo algoritmai, kurie leis rasti optimalų tvarkaraštį pagal vartotojo pasirinkimus. Tvarkaraščiai bus pateikiami mokyklai mokiniams ir mokytojams atskirai. Visi jie pateikiami tiek ekrane (programos darbo metu), tiek “zip” archyve (jį galima parsisiųsti programai baigus optimizavimo procesą). Kuriant programą buvo naudojamos naujausios JAVA technologijos. Programa lengvai valdoma. Joje pateikiama pagalba kiekviename žingsnyje.