INFORMATICA, 2002, Vol. 13, No. 3, 311–332
© 2002 *Institute of Mathematics and Informatics, Vilnius*

# Bayesian Heuristic Approach to Scheduling

Jonas MOCKUS
*Institute of Mathematics and Informatics, Kaunas Technological University*
*Akademijos 4, Vilnius LT-2021, Lithuania*
*e-mail: jonas2@optimum2.mii.lt*

Received: May 2002

**Abstract.** Real life scheduling problems are solved by heuristics with parameters defined by experts, as usual. In this paper a new approach is proposed where the parameters of various heuristics and their random mixtures are optimized to reduce the average deviations from the global optimum.

In many cases the average deviation is a stochastic and multi-modal function of heuristic parameters. Thus a stochastic global optimization is needed. The Bayesian heuristic approach is developed and applied for this optimization. That is main distinctive feature of this work. The approach is illustrated by flow-shop and school scheduling examples. Two versions of school scheduling models are developed for both traditional and profiled schools. The models are tested while designing schedules for some Lithuanian schools. Quality of traditional schedules is defined by the number of teacher "windows". Schedules of profiled schools are evaluated by user defined penalty functions. That separates clearly subjective and objective data. This is the second specific feature of the proposed approach.

The software is developed for the Internet environment and is used as a tool for research collaboration and distance graduate studies. The software is available at web-sites and can be ran by standard net browsers supporting Java language. The care is taken that interested persons could easily test the results and apply the algorithms and software for their own problems.

**Key words:** scheduling, heuristics, Bayesian approach, combinatorial optimization, global optimization.

## 1. Introduction

Industrial, financial, commercial or any kinds of activity have at least one common feature: the better organized they are, the higher the profit, the better the quality or the lower the cost. Everyone makes a schedule to organize his or her own life. 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 e.t.c. Of course, making a schedule for organization is far more difficult than making a schedule for our everyday life. A failure to make a schedule or devising a wrong schedule can result in delay of a deadline and can cost much of the money (Ahuja *et al.*, 1994; Hajdu, 1997).

First we consider a simple flow-shop scheduling problem. Here the sequence of tools is fixed by technology. One minimizes the make-span, which is the time from the beginning of the first task until the end of the last one. That is a well known simple model just to illustrate how the Bayesian Heuristic Approach (BHA) (Kuryla and Mockus, 1995;

Mockus, 2000) could be applied in scheduling problems. General purpose of BHA is to improve heuristics. That is important because for many discrete optimization problems there are no algorithms to obtain the optimal solution in polynomial time (Traub *et al.*, 1988; Traub and Wizniakowski, 1992; Mockus, 1995). Therefore heuristicas are applied (Pardalos *et al.*, 1993).

The second example is scheduling of traditional school. This is closer to real life tasks. Here the sequence of teaching subjects, regarded as tools, can be changed. One needs to reduce the sum of gaps ("empty" lessons) 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.

The difficult example is scheduling of profiled school. Here eleventh and twelfth grade students are choosing , for example, 14 subjects from 64. This means that each student works by his own schedule. We search for the most convenient feasible schedule. The inconveniences are evaluated by penalty points.

All the software is available at web-sites `http://mockus.org/optimum` (international) and `http://soften.ktu.lt/~mockus` (local). run by a standard browser with Java support.

## 2. Flow-Shop Problem

The flow-shop problem is a simple case of the large and important family of scheduling problems We denote the sets of jobs and machines by $J$ and $S$. Denote by $\tau_{j,s}$ the duration of operation $(j, s)$. Here $j \in J$ denotes a job and $s \in S$ denotes a machine.

Suppose that the sequence of machines $s$ is fixed for each job $j$. One machine can do only one job at a time. Several machines cannot do the same job at the same moment. The objective function $v$ is the make-span $T$, the time to complete all the jobs.

### 2.1. *Permutation Schedule*

The number of feasible decisions for the flow-shop can be very large. One reduces this number, if one considers only the so-called permutation schedules that are subsets of all feasible schedules. The permutation schedule is a schedule with the same job order on all machines. Here one defines a permutation schedule by fixing a sequence of job indices, for example, $1, 2, ..., n$. The schedule is transformed by a single permutation of job indices. As usual, permutation schedules describe the optimal decision well and are easier to implement (Baker, 1974)

### 2.2. *Heuristics*

A simple priority rule is defined by the Longer–Job heuristics

$$h_i(j) = \frac{\tau_j - A_i}{A^i - A_i} + a, \tag{1}$$

where

$$A_i = \min_j \tau_j,\ A^i = \max_j \tau_j,\ a > 0. \tag{2}$$

Here

$$\tau_j = \sum_s \tau_{j,s},$$

where $\tau_j$ is the length of the job $j$.

## 2.3. *Results*

Table 1 illustrates results of Bayesian Heuristic Approach (BHA) (Mockus, 2000) after 100 iterations using the Longer–Job heuristic (1) and different randomization procedures. The objective is a stochastic function. This function defines the shortest make-span found after $K$ repetitions.

In the example, $J^* = S^* = O^* = 10$, where $J^*,\ S^*,\ O^*$ are numbers of jobs, machines, and operations, respectively. Lengths and sequences of operations are generated as random numbers uniformly distributed from zero to ninety-nine. The expectations and standard deviations are estimated by repeating forty times the optimization of a randomly generated problem.

In Table 1 the symbol $f_B$ denotes a mean, and $d_B$ denotes a standard deviation of the make-span.

"BHA" denotes the Bayesian Heuristic Approach optimizing a "mixture" of heuristic algorithms.

$x_0$ defines the optimal ratio of Monte Carlo heuristics (Algorithm 0) meaning equal probabilities,

$x_1$ shows the optimal proportion of linear randomized heuristics (Algorithm 1) where probabilities are proportional to heuristics $h_i$,

$x_2$ is the optimal ratio of greedy heuristics (Algorithm 2) when one selects the longest available job.

Note that randomization is applied in two stages. First time the randomization is to select the algorithm $i = 0, 1, 2$ with probabilities $x_i$. Second time, the randomization is to chose the next job using the algorithm $i$ selected in the first stage.

Table 1

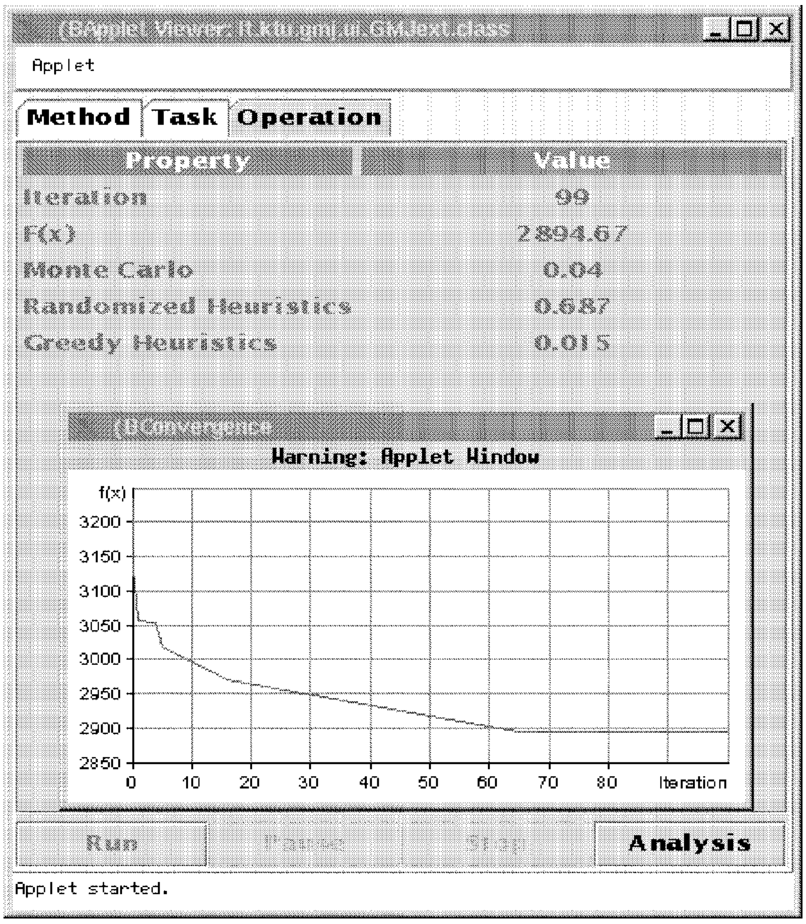
Results of BHA using Longer–Job heuristics

| $R = 100$, $K = 1$, $J^* = 10$, $S^* = 10$, and $O^* = 10$ | | | | | |
|---|---|---|---|---|---|
| Algorithm | $f_B$ | $d_B$ | $x_0$ | $x_1$ | $x_2$ |
| BHA | 6.183 | 0.133 | 0.283 | 0.451 | 0.266 |
| CPLEX | 12.234 | 0.00 | — | — | — |

The objective is to minimize the average make-span that is a stochastic multi-modal function of variables $x_i,\ i = 0, 1, 2$ therefore, the Bayesian search is applied (Mockus, 2000). This explains the term Bayesian Heuristic Approach (BHA).

"CPLEX" denotes the the well known general discrete optimization software after twenty hundred iterations (one CPLEX iteration is comparable to a Bayesian observation). Weak results of CPLEX show that the standard MILP technique is not efficient in solving this specific problem of discrete optimization. It is not yet clear how much one improves the results using specifically tailored B&B. To optimize heuristic parameters (proportions of three different heuristics) for a given Flow-Shop problem, one applies the BHA using optimization system GMJ2 implemented in Java 2 (Greicius and Luksys, 1999).

The optimal proportions of three heuristics: the Monte Carlo, the linear randomization and the "Select-the-Best" are on the output page (see Fig. 1). The convergence line is there, too. These are results of one hundred iterations by the 'Bayes' method.



| Property | Value |
|---|---|
| Iteration | 99 |
| F(x) | 2894.67 |
| Monte Carlo | 0.04 |
| Randomized Heuristics | 0.687 |
| Greedy Heuristics | 0.015 |

Fig. 1. Numerical results and convergence line, the 'FlowShop' task.

## 3. Traditional School Scheduling

School scheduling is an example of the general scheduling problem. First we consider the case where the objective is the number of "empty" lessons when the teachers wait for the next scheduled lectures. One calls these hours as "teacher gaps, " or just "gaps" for short. We search for such schedules that reduce the sum of teacher gaps considering the schedules of fifth to twelfth class of the public high school. Other factors are school-specific and should be included adapting the software to specific schools.

### 3.1. *Constraints*

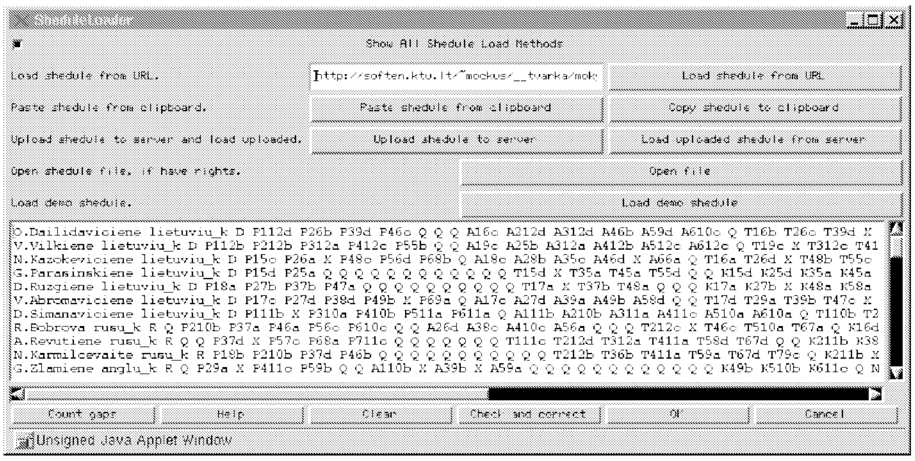
There are constraints:

- 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 "student 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.

### 3.2. *Data Structure*

Fig. 2 shows a fragment of the existing weakly schedule of some Lithuanian school.

- the schedule is defined as a string array $Mokytojai[64][38]$,
- the string $Mokytojai[i][1]$ defines the name of the $i$-th teacher, for example, $O.Dilidaviciene$, $V.Vilkiene$, e.t.c.


Show All Shedule Load Methods
Load shedule from URL.
http://soften.ktu.lt/~mockus/__tvarka/mok
Load shedule from URL
Paste shedule from clipboard.
Paste shedule from clipboard
Copy shedule to clipboard
Upload shedule to server and load uploaded.
Upload shedule to server
Load uploaded shedule from server
Open shedule file, if have rights.
Open file
Load demo shedule.
Load demo shedule
Count gaps
Help
Clear
Check and correct
OK
Cancel
Unsigned Java Applet Window

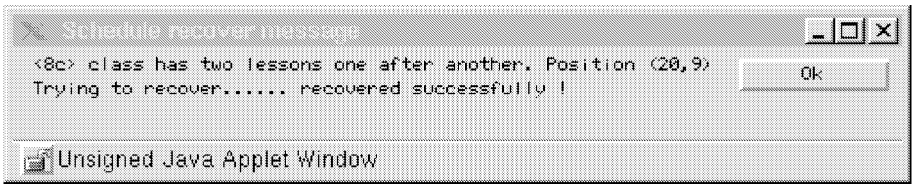
```
O.Dailidaviciene lietuviu_k D P112d P26b P39d P46c Q Q Q A16c A212d A312d A46b A59d A610c Q T16b T26c T39d X
V.Vilkiene lietuviu_k D P112b P212b P312a P412c P55b Q Q A19c A25b A312a A412b A512c A612c Q T19c X T312c T41
N.Kazokeviciene lietuviu_k D P15c P26a X P48c P56d P68b Q A18c A28b A35c A46d X A66a Q T16a T26d X T48b T55c
G.Parasinskiene lietuviu_k D P15d P25a Q Q Q Q Q Q Q Q Q Q Q Q T15d X T35a T45a T55d Q Q K15d K25d K35a K45a
D.Ruzgiene lietuviu_k D P18a P27b P37b P47a Q Q Q Q Q Q Q Q Q Q T17a X T37b T48a Q Q Q K17a K27b X K48a K58a
V.Abromaviciene lietuviu_k D P17c P27d P38d P49b X P69a Q A17c A27d A39a A49b A58d Q Q T17d T29a T39b T47c X
D.Simanaviciene lietuviu_k D P111b X P310a P410b P511a P611a Q A111b A210b A311a A411c A510a A610a Q T110b T2
R.Bobrova rusu_k R Q P210b P37a P46a P56c P610c Q Q A26d A38c A410c A56a Q Q Q T212c X T46c T510a T67a Q K16d
A.Revutiene rusu_k R Q Q P37d X P57c P68a P711c Q Q Q Q Q Q Q T111c T212d T312a T411a T58d T67d Q Q K211b K38
N.Karmilcevaite rusu_k R P18b P210b P37d P46b Q Q Q Q Q Q Q Q Q Q Q T212b T36b T411a T59a T67d T79c Q K211b X
G.Zlamiene anglu_k R Q P29a X P411c P59b Q Q A110b X A39b X A59a Q Q Q Q Q Q Q Q Q Q Q Q K49b K510b K611c Q N
```

Fig. 2. Uploaded schedule-row data.

- the string $Mokytojai[i][2]$ defines the subject of the $i$-th teacher, for example, $lietuviu\_k$ means Lithuanian, $rusu\_k$ means Russian, $anglu\_k$ means English, $matematika$ means Mathematics, $informatika$ means Informatics, e.t.c.
- the string $Mokytojai[i][3]$ defines the code of the classroom of the $i$-th teacher, for example, 0 means no special study, $A$ means a study for Informatics, e.t.c.
- the sequence of strings $Mokytojai[i][j],\ j = 4, ..., 38$ define the class codes of the $i$-th teacher for all weak, for example, in the class code $P19c$, the first symbol $P$ means Monday[1], the second symbol 1 defines the first lesson, and the third symbol $9c$ means the class $9c$, teacher gaps are denoted by symbols $X$, symbols $Q$ denote "open" lessons, a lesson is called "open", if a teacher is free before or after this lesson.

## 3.3. *Permutation and Evaluation Algorithm*

The process of checking and correcting illustrate Fig. 3. The corrected schedule is optimized using the permutation algorithm. The algorithm follows a general pattern of permutation algorithms related to the Bayesian Heuristic Approach (Mockus, 2000)



```
<8c> class has two lessons one after another. Position (20,9)
Trying to recover...... recovered successfully !
```

Fig. 3. Correcting "Double Lecture"

```
 1. record the current schedule $Mokytojai0$ by reading the data file
    $Mokytojai[64][38]$,
 2. record the current number of teacher gaps[2],
 3. set the initial iteration number $it = 0$,
 4. set the current iteration number $it := it + 1,\ it < K$,
 5. if $it = K$, stop and print the current schedule,
 6. set the initial teacher number $i = -1$,
 7. set the current teacher number $i := i + 1$,
 8. if $i > 63$ go to the Step 4,
 9. generate uniformly distributed random number $\xi \in [0, 1]$,
10. go to the Step 7, if $\xi < x$,
11. select a gap of the teacher $i$,
    go to Step 7, if there are no gaps,
12. select an open lesson of the $i$-th teacher,
    go to Step 7, if there are no open lessons,
```

[1]The complete list is $P$ for Monday, $A$ for Tuesday, $T$ for Wednesday, $K$ for Thursday, and $N$ for Friday.
[2]A number of gaps in the current schedule $Mokytojai0$.

13. make a permutation by exchanging the "gap class"[3] with the open one[4],
14. test the feasibility conditions listed in Section 3.1 ,
    go to the Step 16, if the permutation is feasible,
15. restore the feasibility by restoring the teacher gap and go to the Step 7,
16. count the teacher gaps in the permuted schedule,
17. compare the number of the teacher gaps in the current schedule with the permuted one,
18. replace the current schedule by the permuted one,
    if the current number of teacher gaps is less then that in the permuted schedule,
    go to Step 7

The algorithm is very simple and natural. It mimics, in a sense, the usual ways to improve existing schedules. However, replacing a "gap" class by the open one, defined in the Step thirteen, a new gap for another teacher is opened, as usual. The algorithm does not satisfy the convergence conditions (Mockus *et al.*, 1997). To satisfy these conditions the probability to reach any schedule should be positive.

### 3.4. *Running Software*

The GMJ1 and GMJ2 systems described in (Mockus, 2000) are used in (Perlibakas, 1999) for optimization of BHA parameters following these steps:

- select the method and its parameters,
- select the task 'Tvarka' and its parameters including the iteration number $K$, the lower and upper bounds for the randomization parameter $x$, the default value of $x$.

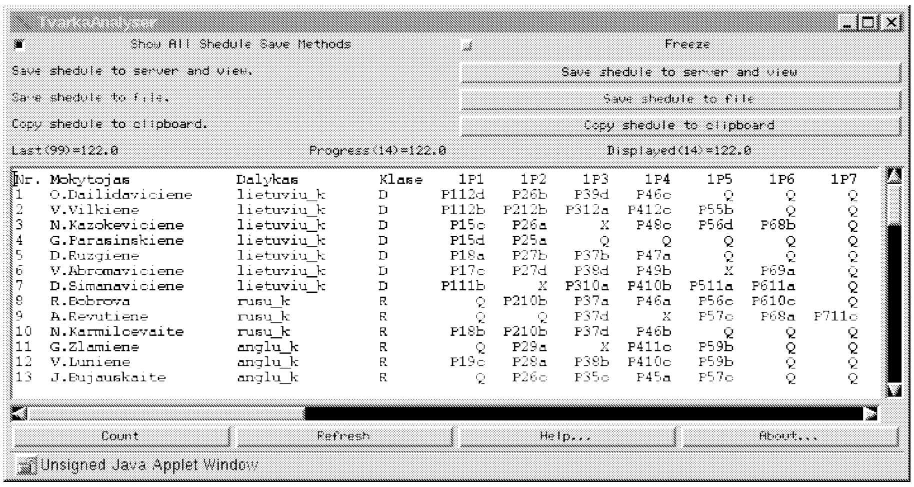
To see the optimal schedule click the button $TvarkaAnalyser$ (see Fig. 5). The software is available at the web-sites `http://mockus.org/optimum` and `http://soften.ktu.lt/~mockus`, the section "Discrete Optimization", examples "School-COMPLETE" and can be run by a browser with Java support.

## 4. Profiled Schools

### 4.1. *Introduction*

The algorithm described in the previous section is for scheduling of traditional schools. In these schools the study schedules are fixed for each grade. Application of this algorithm is more difficult, if classes are divided depending on subjects of choice. For example, some students may choose religion, some others ethics, e.t.c. Here the number of "classes" is greater because the schedules of divided classes of the same grade are different. In addition, classes are divided for some subjects and united for others, as usual.

[3]A gap class is the class in the teacher gap.
[4]An open class is the first or the last class for this teacher.



Save shedule to server and view. — Save shedule to server and view

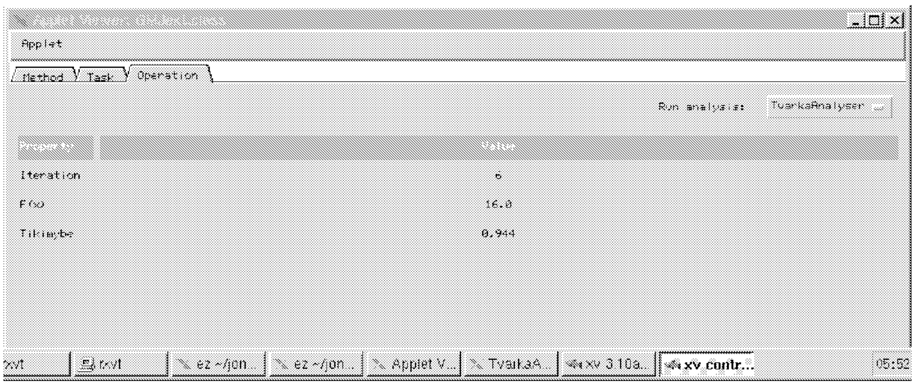
Save shedule to file. — Save shedule to file

Copy shedule to clipboard. — Copy shedule to clipboard

Last(99)=122.0 Progress(14)=122.0 Displayed(14)=122.0

| Nr. | Mokytojas | Dalykas | Klase | 1P1 | 1P2 | 1P3 | 1P4 | 1P5 | 1P6 | 1P7 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | O.Dailidaviciene | lietuviu_k | D | P112d | P26b | P39d | P46c | Q | Q | Q |
| 2 | V.Vilkiene | lietuviu_k | D | P112b | P212b | P312a | P412c | P55b | Q | Q |
| 3 | N.Kazokeviciene | lietuviu_k | D | P15c | P26a | X | P48c | P56d | P68b | Q |
| 4 | G.Parasinskiene | lietuviu_k | D | P15d | P25a | Q | Q | Q | Q | Q |
| 5 | D.Ruzgiene | lietuviu_k | D | P18a | P27b | P37b | P47a | Q | Q | Q |
| 6 | V.Abromaviciene | lietuviu_k | D | P17c | P27d | P38d | P49b | X | P69a | Q |
| 7 | D.Simanaviciene | lietuviu_k | D | P111b | X | P310a | P410b | P511a | P611a | Q |
| 8 | R.Bobrova | rusu_k | R | Q | P210b | P37a | P46a | P56c | P610c | Q |
| 9 | A.Revutiene | rusu_k | R | Q | Q | P37d | X | P57c | P68a | P711c |
| 10 | N.Karmilcevaite | rusu_k | R | P18b | P210b | P37d | P46b | Q | Q | Q |
| 11 | G.Zlamiene | anglu_k | R | Q | P29a | X | P411c | P59b | Q | Q |
| 12 | V.Luniene | anglu_k | R | P19c | P28a | P38b | P410c | P59b | Q | Q |
| 13 | J.Bujauskaite | anglu_k | R | Q | P26c | P35c | P45a | P57c | Q | Q |

Fig. 4. Fragment of the optimized schedule.



Run analysis: TvarkaAnalyser

| Property | Value |
|---|---|
| Iteration | 6 |
| F(x) | 16.0 |
| Tikimybe | 0.944 |

Fig. 5. Results of GMJ1.

In so-called "Profiled Schools" the students of eleventh and twelfth year select, for example, fourteen subjects from sixty available. That means different individual schedules for most of the students. There are no stable student classes anymore. They are replaced by changing "interest groups". The changes happen each hour.

Personal choices of eleventh and twelfth grade students are defined as sets of subjects. Sequences of these subjects and the corresponding class-rooms are defined by the general school schedule.

In this case a new approach is needed for optimal scheduling. An example is the commercial system $MIMOSA$. The system helps to produce feasible schedules by performing corrections, for example, by closing some gaps, and informing about violated constraints and inconveniences. However schedules produced by $MIMOSA$ depend on the human scheduler operating the system. Therefore, comparison with other systems is very difficult. In this sense, the $MIMOSA$ and other similar systems can be regarded mainly as "Support Systems."

The goal of this research is to create different type of optimization system for profiled school scheduling. The main idea is that some responsible person or committee specifies just general objectives and constraints. The schedules, both general and personal, are produced automatically by an optimization system. These schedules may be corrected by an operator considering some additional factors not included in the general objectives and constraints. The human operator can influence the outcome of optimization by choosing an initial schedules, too [5].

## 4.2. *Schedule Representation*

A way to define the complete schedule of profiled school is to represent it as a four-dimensional binary array

$$schedc[M][V][G][K]. \tag{3}$$

Here

$M$ is a number of teachers[6],

$V$ denotes the total number of weekly working hours,

$G$ is a number of different interest groups,

$K$ is a number of school rooms.

$$schedc[i][j][k][l] = 1 \tag{4}$$

means that the teacher $i$ at the hour $j$ teaches the interest group $k$ in the class room $l$,

$$schedc[i][j][k][l] = 0, \tag{5}$$

means no lesson.

The general schedule is represented as an three-dimensional binary array

$$schedg[M][V][K]. \tag{6}$$

To represent the general schedule in two dimensions the two-dimensional array is used

$$schedg2[M][VK], \tag{7}$$

where $VK$ is a string defining both the hour and the class room (see for example 2).

[5]The optimization results depend on the initial schedules because there is no possibility to obtain exact solutions in a limited time.

[6]If a teacher covers several subjects then he/she is represented as several different "virtual" teachers, feasibility conditions prevents cases with two or more lessons by the same "physical" teachers at the same time.

Personal schedules are defined for each interest group $k$ as three-dimensional binary arrays

$$sched_k[M][V][K]. \tag{8}$$

To represent this schedule in two dimensions the two-dimensional array is used

$$sched2_k[M][VK]. \tag{9}$$

The binary representation is simple and convenient for understanding and defining constraints but requires large memory and is not convenient for schedules of real profiled schools.

## 4.3. *Objective Function*

There are no schedules that satisfy all restrictions and personal preferences because they contradict each other as usual. For example, the ministry of education defines a number of rules to be observed, both school teachers and students prefer schedules without gaps, etc. Thus, the realistic objective is to find the best compromise of conflicting interests.

A compromise solution is reached by defining penalties for violation of constraints and disregarding inconveniences. We search for such schedule that minimizes the total penalty function. Only the "physical" constraint may not be violated: a person cannot be in two places at the same time.

We assume, that all other constraints may be violated, at a price. Both normative and convenience constraints are considered in this research.

## 4.4. *Normative Constraints*

The standard restrictions are

1. for each group $k$ the total number of weekly hours $v_{sk} \leqslant V_{sk} < 168$,
2. for each group $k$ the total number of daily hours $v_{dk} \leqslant V_{dk} < 24$,
   for eleventh and twelfth grades $V_{sk} < 5V_{sd}$,
   for the rest $V_{sk} = 5V_{sd}$
3. for each teacher $i$ the total number of weekly hours is $V_i$,
4. for each group $k$ the total number of weekly hours for a subject $n$[7] is $V_{kn}$,
5. not more than one lesson at a time in a class [8],
6. for some subjects and groups "double" lessons [9] are needed,
7. for some subjects and groups "double" lessons are forbidden,
8. for some subjects and groups "double" lessons are allowed,
9. no gaps for students from the first to tenth grade.

[7] If the subject $n$ is covered by just one teacher $i$, then $n = 1$.

[8] This is a normative constraint, because several lessons may be performed in one class room, physically.

[9] This means two lessons in a raw.

## 4.5. *Convenience Constraints*

The usual factors of inconvenience are

1. teacher gaps,
2. student gaps (for eleventh and twelfth grades),
3. inconvenient hours,
4. inconvenient days,
5. inconvenient sequences of lessons.

## 4.6. *Penalty Function*

### 4.6.1. *Normative Penalties*

Formally, the normative constraints cannot be violated. Practically, some minor violations could be tolerated, if this improves other parameters considerably. Thus the normative penalties $c_r$ must be high.

$$C_n = \sum_r c_r N_r, \tag{10}$$

where

$c_r$ is the penalty for violation of the normative $r$,

$N_r$ is the number of corresponding violations.

In this case $r = 1, .., 9$.

The actual numbers $c_r$ are expert estimates and depends on general situation of the school. Therefore, they are parameters defined by users using graphical interfaces, as usual.

### 4.6.2. *Penalties of Inconvenience*

Penalties of inconvenience are defined by users for each inconvenience factor:

1. $c_i$ is the penalty for the teacher $i$ gap,
2. $c_k$ is the penalty for the group $k$ gap,
3. $c_{ji}$ is the penalty of the hour $j$ that is inconvenient for the teacher $i$,
4. $c_{li}$ is the penalty of the day $l$ that is inconvenient for the teacher $i$,
5. $c_{jk}$ is the penalty of the hour $j$ that is inconvenient for the group $k$,
6. $c_s$ is the penalty for inconvenient sequence $s$ of lessons.

The general inconvenience penalty

$$\begin{aligned} C_c = & \sum_i c_i L_i + \sum_k c_k L_k + \sum_i \sum_j c_{ji} L_i^j \\ & + \sum_i \sum_l c_{li} L_i^l + \sum_k \sum_j c_{jk} L_k^j + \sum_s c_s L_s, \end{aligned} \tag{11}$$

where

$L_i$ is the number of gaps of teacher $i$,

$L_k$ is the number of gaps of group $k$,

$L_i^j$ is the number of hours $j$ that are inconvenient for the teacher $i$,

$L_i^l$ is the number of days $l$ that are inconvenient for the teacher $i$,

$L_k^j$ is the number of hours that are inconvenient for the group $k$,

$L_s$ is the number of inconvenient sequences of lessons. The objective function is the total penalty

$$C = C_n + C_c. \tag{12}$$

Then the optimization problem is

$$\min_{\tau \in \Theta} C(\tau), \tag{13}$$

where $C(\tau)$ is the total penalty of some schedule $\tau$,

$\Theta$ is the set of schedules satisfying the physical constraints. The penalties $C(\tau)$ depend on expert evaluations, therefore we regard them as heuristics.

## 4.7. *Defining the Initial Schedule*

In some cases the initial schedule is defined by users. Otherwise it is build by some greedy heuristics. Using the Greedy Heuristics techniques one builds the schedule from scratch. Here the results depend on the sequence of "building" operations. This way, one obtains a schedule that is convenient for the first teacher but may be very inconvenient for the last teacher, e.t.c.

As usual, greedy heuristics define priorities. That means that subjects, classes and persons that are higher on the priority list obtain better schedules. The schedule build by pure greedy heuristics often does not satisfy some important restrictions. Therefore randomization is introduced. It helps to build better initial schedule by random mixing of schedule lines and so extending the range of search.

## 4.8. *Optimization by Permutations*

The initial schedule is improved by permutations. The best obtained schedule is recorded after each iteration. To escape from local minima changes to worse schedules are performed with some decreasing probabilities.

A natural first step while trying to provide convergence is the Simulated Annealing (SA):

One moves from the current schedule $i$ to the permuted schedule $i+1$ with probability

$$r_{i+1} = \begin{cases} e^{\frac{-h_{i+1}}{x/\ln(1+N)}}, & \text{if } h_{i+1} > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{14}$$

Here $N$ is the iteration number and $x$ is the "initial temperature." The logarithmic "cooling schedule" $\ln(1+N)$ follows from convergence conditions (Cohn and Fielding, 1999).

The difference from the traditional SA is that here 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 (14)

$$r_{i+1} = \begin{cases} e^{\frac{-h_{i+1}}{x_1/\ln(1+x_2 N)}}, & \text{if } h_{i+1} > 0, \\ 1, & \text{otherwise,} \end{cases} \tag{15}$$

where $x_1 \geqslant 0$ defines an "initial temperature" of SA and $x_s \geqslant 0$ describes its "cooling rate". SA and other discrete optimization techniques are explained in (Mockus, 2000) by the Knapsack example .

## 4.9. *Software Example*

The profiled school scheduling software implements permutations similar to these of the traditional school considered in (Mockus, 2000). That is only similarity. Important new ideas are developed and implemented while considering profiled school scheduling.

The fist one is that the schedule should be evaluated including both objective and subjective factors in a balanced way. The theoretical framework of this is the Pareto optimum. Simplest way to obtain Pareto optimal schedule is by assigning some penalty points for deviations from the desirable or feasible conditions. The total penalty is minimized. In such a case the penalty points represent subjective opinion of people responsible for schedules. The conditions reflects objective legal and physical factors.

The second new idea is optimization of heuristics by selecting their best parameters. That is achieved in three stages. In the first stage the same trivial permutation heuristics is used as in the traditional schedule. That works if one starts from nearly optimal schedule what is true in traditional schools as usual. In the profiled schools some "globalization" of permutation heuristics is needed.

That is achieved in the second stage by Simulated Annealing (SA) applied with same fixed parameters such as initial "temperature" and the "cooling" rate. The third fixed parameter is the probability to "miss" some teacher waiting in the "improvement" queue. Applying SA one accepts with some small probability schedules with higher penalties so escaping from local minima and improving convergence to the optimal schedule. However the results of SA depend on three arbitrarily fixed parameters.

In the final stage these parameters are optimized using methods of stochastic global optimization developed in the framework of the Bayesian Heuristic Approach (BHA) (Mockus, 2000). Each stage can be involved separately, that makes comparisons of their results much simpler.

There are two software versions: applet and servlet. The graphics of the applet version look better as usual, because graphical programs run at the user site. The servlet version runs using server resources. That is an advantage if user computing resources are limited.

### 4.9.1. *Applet Version*

This version implements only the first stage where the total penalty function is minimized locally using the simplest permutation heuristics That improves the initial schedules as usual. However convergence is not provided. The advanced SA and BHA methods are implemented in the servlet version because servlet mode is more convenient providing schools with servers computing resources and better facilities for data exchange.

To run the applet one visits the web-site `http://mockus.org/optimum` or `http://soften.ktu.lt/~mockus`, enters the problem named "School-PROFILED" in the section "Discrete Optimization", and starts *applet.html* by a browser that supports Java.

### 4.9.2. *User Data*

The data file defines the list of subjects and the students choices. File format is *txt*, one record isl for one subject.

Structure of the record is as follows:

1. Number of grade.
2. Subject code.
3. Teacher code.
4. Number of classrooms.
5. Number of classes per week.
6. List of attending students.

```
Records are  not longer then 11 symbols.
Fields are separated by the symbol |.
The end of the line is marked by the symbol |^
 For example,
11AngLg | ZenMar | 30 | 3 | 11LeonUla | 11KuprMin |^
12MatLg | BirCer | 21 | 4 | 12GerdAbis | 12GirdSaru |
12GedLag |^
```

Here the number 11 denotes eleventh grade, the code $AngLg$ means the $Lg$ version of English, $ZenMar$ is the abbreviation of teacher name, 30 is the classroom number, 3 is the number of weekly lessons, $11LeonUla$ defines a name of student of eleventh grade, e.t.c.

A fragment of user data file is in Fig. 6.

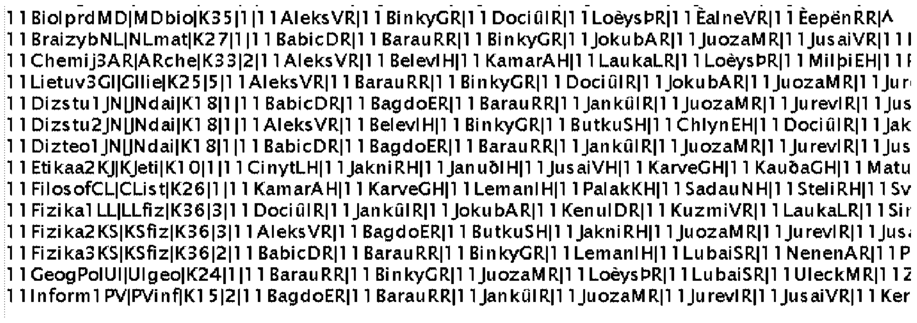
Fig. 7 shows the opening window.

User data is uploaded by CGI interface clicking the $Browse$ button. The scheduling starts by the $Read\ data$ button.

User preferences are expressed defining restrictions and preferences and assigning penalty points for violating these restrictions and preferences.

### 4.9.3. *Initial Schedule*

The initial schedule is built by priority rules and some randomization. For example, continuous lessions are included first, starting from Monday. Then all the lessions are listed that can be performed at the same time without violation of physical restrictions and day-off requests. One lession per day is scheduled. Both student and teacher gaps are ignored



```
1 1 BiolprdMD|MDbio|K35|1 |1 1 AleksVR|1 1 BinkyGR|1 1 DociûlR|1 1 LoèysÞR|1 1 ÈalneVR|1 1 ÈepënRR|A
1 1 BraizybNL|NLmat|K27|1 |1 1 BabicDR|1 1 BarauRR|1 1 BinkyGR|1 1 JokubAR|1 1 JuozaMR|1 1 JusaiVR|1 1 I
1 1 Chemij3AR|ARche|K33|2|1 1 AleksVR|1 1 BelevlH|1 1 KamarAH|1 1 LaukaLR|1 1 LoèysÞR|1 1 MilþiEH|1 1 F
1 1 Lietuv3GI|GIlie|K25|5|1 1 AleksVR|1 1 BarauRR|1 1 BinkyGR|1 1 DociûlR|1 1 JokubAR|1 1 JuozaMR|1 1 Jur
1 1 Dizstu1JN|JNdai|K1 8|1 |1 1 BabicDR|1 1 BagdoER|1 1 BarauRR|1 1 JankûlR|1 1 JuozaMR|1 1 JurevlR|1 1 Jus
1 1 Dizstu2JN|JNdai|K1 8|1 |1 1 AleksVR|1 1 BelevlH|1 1 BinkyGR|1 1 ButkuSH|1 1 ChlynEH|1 1 DociûlR|1 1 Jak
1 1 Dizteo1JN|JNdai|K1 8|1 |1 1 BabicDR|1 1 BagdoER|1 1 BarauRR|1 1 JankûlR|1 1 JuozaMR|1 1 JurevlR|1 1 Jus
1 1 Etikaa2KJ|KJeti|K1 0|1 |1 1 CinytLH|1 1 JakniRH|1 1 JanuðlH|1 1 JusaiVH|1 1 KarveGH|1 1 KauðaGH|1 1 Matu
1 1 FilosofCL|CList|K26|1 |1 1 KamarAH|1 1 KarveGH|1 1 LemanlH|1 1 PalakKH|1 1 SadauNH|1 1 SteliRH|1 1 Sv
1 1 Fizika1LL|LLfiz|K36|3|1 1 DociûlR|1 1 JankûlR|1 1 JokubAR|1 1 KenulDR|1 1 KuzmiVR|1 1 LaukaLR|1 1 Sir
1 1 Fizika2KS|KSfiz|K36|3|1 1 AleksVR|1 1 BagdoER|1 1 ButkuSH|1 1 JakniRH|1 1 JuozaMR|1 1 JurevlR|1 1 Jus
1 1 Fizika3KS|KSfiz|K36|2|1 1 BabicDR|1 1 BarauRR|1 1 BinkyGR|1 1 LemanlH|1 1 LubaiSR|1 1 NenenAR|1 1 P
1 1 GeogPolUI|UIgeo|K24|1 |1 1 BarauRR|1 1 BinkyGR|1 1 JuozaMR|1 1 LoèysÞR|1 1 LubaiSR|1 1 UleckMR|1 1 Z
1 1 Inform1PV|PVinf|K1 5|2|1 1 BagdoER|1 1 BarauRR|1 1 JankûlR|1 1 JuozaMR|1 1 JurevlR|1 1 JusaiVR|1 1 Ker
```

Fig. 6. User data.

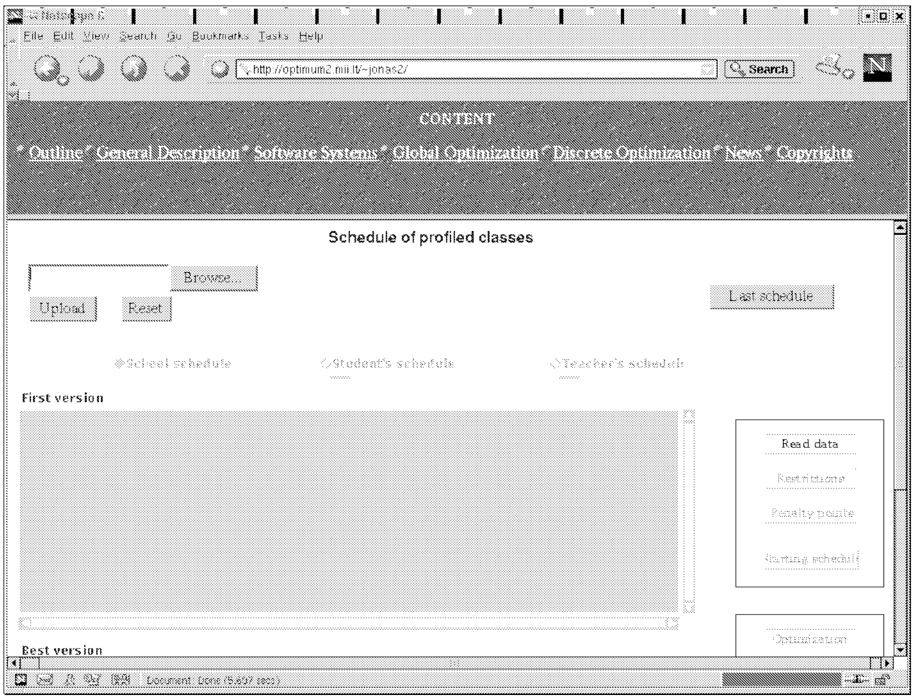


Fig. 7. Opening window.

while building the initial schedule. The main difficulty is to list all the subjects without exceeding the default limit of 9 daily hours. To achieve this the random mixing of schedule lines is used up to 100 times. After that the limit is increased by 1 (first time from the default 9 to 10 hours per day). The initial schedule is completed by moving the largest classes to early hours.

### 4.9.4. *Optimization by Permutations*

First a window opens for starting optimization and defining randomization parameter $x$. The permutations are similar to those in traditional school. The difference is that both the teacher and the student gaps are considered and the probability $x$ to pass a person $i$ is fixed. This reduces optimization possibilities as compared with the traditional school scheduling software where parameter $x$ is optimized. Note that the advanced servlet version includes optimization of all the heuristic parameters.

### 4.9.5. *Servlet Versions*

There are three servlet versions. The simplest one is direct implementation of the applet version. The updated version adds optimization of heuristics by Simulated Annealing (SA) with fixed parameters. The advanced servlet version optimizes parameters of SA completing software implementation of the main theoretical ideas of the Bayesian Heuristic Approach (BHA) (see Fig. 8). Fig. 9 shows how data is uploaded, preferences and restrictions defined. In the upper-left corner is a list of topics. Marking some of them introduces "double time" preferences. Important restriction is maximal number of hours per day. The default value $9$ is shown. Clicking "teacher list" one introduces teacher "days-off" requests.

In the upper-right corner penalty points are defined. That is the subjective part of scheduling defined by persons responsible for scheduling reflecting informal local conditions and preferences.

The upper window in Fig. 10 is for defining optimization parameters. Here the number of iterations is $10$, the initial probability to "miss" a teacher is $0.5$, both the initial "temperature" $X1$ and the "cooling" rate $X2$ of SA are open.

The middle window shows the difference between the initial schedule ($2532$ penalty points) and the optimized one ($1743$ penalty points). To see the complete schedules one clicks on corresponding words shown in the lower window of Fig. 10. The initial school schedule is in the lower part of Fig. 9. The optimized school schedule is in Fig. 11.

Most of notations are similar to those in the user data. There are some Lithuanian terms: $Mokyklos\ tvarkarastis$ means school schedule, $Pirmadienis$ is Monday, $Antradienis$ is Tuesday, $klaida$ means that user data is not correct here.

The optimized schedule of teacher Jancius is in Fig. 12.

The optimized schedule of student Butkus is in Fig. 12.

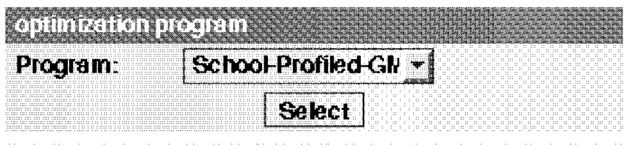
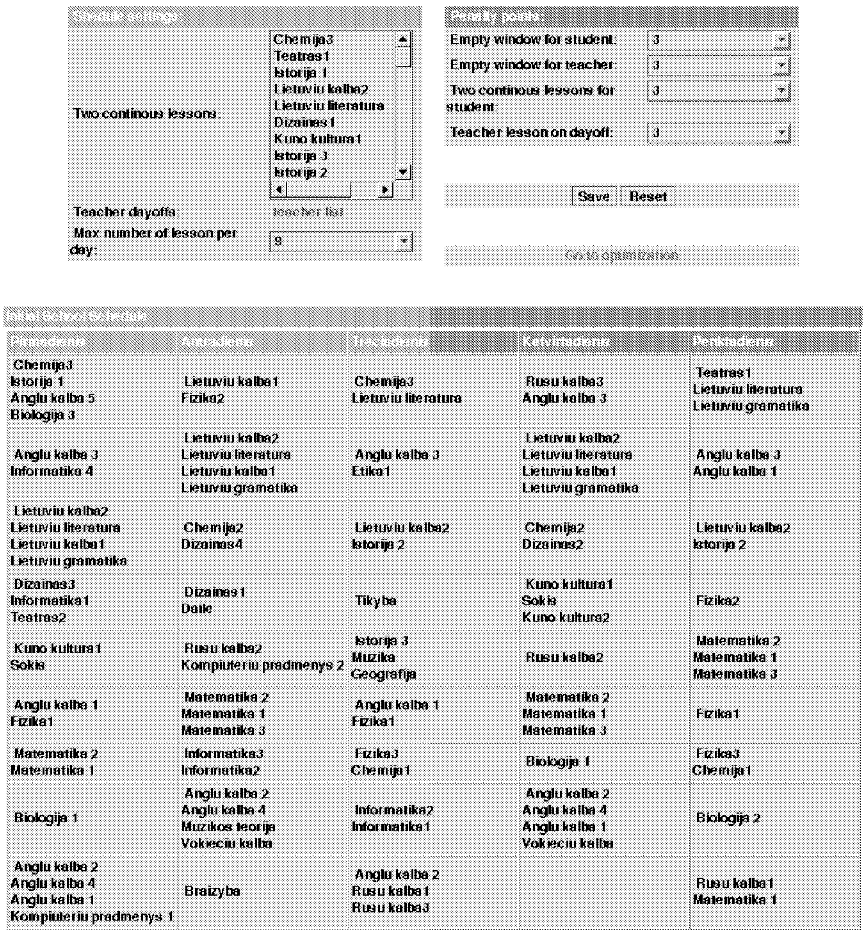


Fig. 8. Opening the program.



Schedule settings:

| | |
|---|---|
| Two continous lessons: | Chemija3<br>Teatras1<br>Istorija 1<br>Lietuviu kalba2<br>Lietuviu literatura<br>Dizainas1<br>Kuno kultura1<br>Istorija 3<br>Istorija 2 |
| Teacher dayoffs: | teacher list |
| Max number of lesson per day: | 9 |

Penalty points:

| | |
|---|---|
| Empty window for student: | 3 |
| Empty window for teacher: | 3 |
| Two continous lessons for student: | 3 |
| Teacher lesson on dayoff: | 3 |

Initial School Schedule

| Pirmadienis | Antradienis | Treciadienis | Ketvirtadienis | Penktadienis |
|---|---|---|---|---|
| Chemija3<br>Istorija 1<br>Anglu kalba 5<br>Biologija 3 | Lietuviu kalba1<br>Fizika2 | Chemija3<br>Lietuviu literatura | Rusu kalba3<br>Anglu kalba 3 | Teatras1<br>Lietuviu literatura<br>Lietuviu gramatika |
| Anglu kalba 3<br>Informatika 4 | Lietuviu kalba2<br>Lietuviu literatura<br>Lietuviu kalba1<br>Lietuviu gramatika | Anglu kalba 3<br>Etika1 | Lietuviu kalba2<br>Lietuviu literatura<br>Lietuviu kalba1<br>Lietuviu gramatika | Anglu kalba 3<br>Anglu kalba 1 |
| Lietuviu kalba2<br>Lietuviu literatura<br>Lietuviu kalba1<br>Lietuviu gramatika | Chemija2<br>Dizainas4 | Lietuviu kalba2<br>Istorija 2 | Chemija2<br>Dizainas2 | Lietuviu kalba2<br>Istorija 2 |
| Dizainas3<br>Informatika1<br>Teatras2 | Dizainas1<br>Daile | Tikyba | Kuno kultura1<br>Sokis<br>Kuno kultura2 | Fizika2 |
| Kuno kultura1<br>Sokis | Rusu kalba2<br>Kompiuteriu pradmenys 2 | Istorija 3<br>Muzika<br>Geografija | Rusu kalba2 | Matematika 2<br>Matematika 1<br>Matematika 3 |
| Anglu kalba 1<br>Fizika1 | Matematika 2<br>Matematika 1<br>Matematika 3 | Anglu kalba 1<br>Fizika1 | Matematika 2<br>Matematika 1<br>Matematika 3 | Fizika1 |
| Matematika 2<br>Matematika 1 | Informatika3<br>Informatika2 | Fizika3<br>Chemija1 | Biologija 1 | Fizika3<br>Chemija1 |
| Biologija 1 | Anglu kalba 2<br>Anglu kalba 4<br>Muzikos teorija<br>Vokieciu kalba | Informatika2<br>Informatika1 | Anglu kalba 2<br>Anglu kalba 4<br>Anglu kalba 1<br>Vokieciu kalba | Biologija 2 |
| Anglu kalba 2<br>Anglu kalba 4<br>Anglu kalba 1<br>Kompiuteriu pradmenys 1 | Braizyba | Anglu kalba 2<br>Rusu kalba1<br>Rusu kalba3 | | Rusu kalba1<br>Matematika 1 |

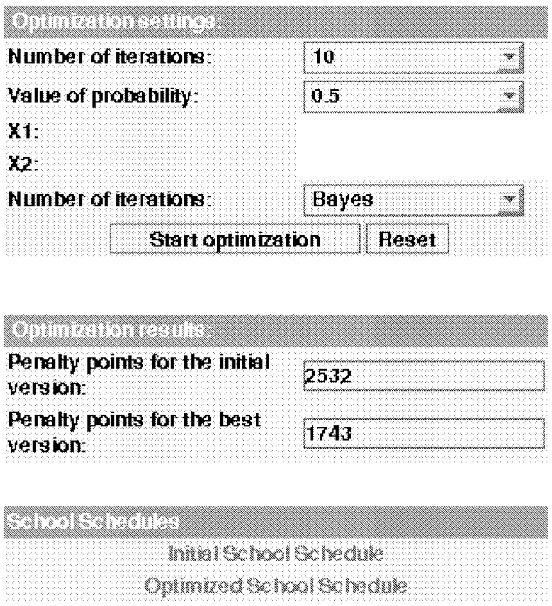
Fig. 9. Uploading data.

## 4.10. *Conclusions*

The profiled school scheduling algorithms and software implements permutations similar to these of the traditional school considered in (Mockus, 2000). That is only similarity. Important new ideas are developed and implemented considering profiled school scheduling.

The fist one is that the schedules should be evaluated including both objective and subjective factors in a balanced way. The theoretical framework of this is the Pareto optimum. Simlest way to obtain Pareto optimal schedule is by assigning some penalty points for deviations from the desirable or feasible conditions. The total penalty is minimized. In such a case the penalty points represent subjective opinion of people responsible for scheduless. Desirable and feasible conditions reflects objective legal and physical factors.

The second new idea is optimization of heuristics by selecting their best parameters.



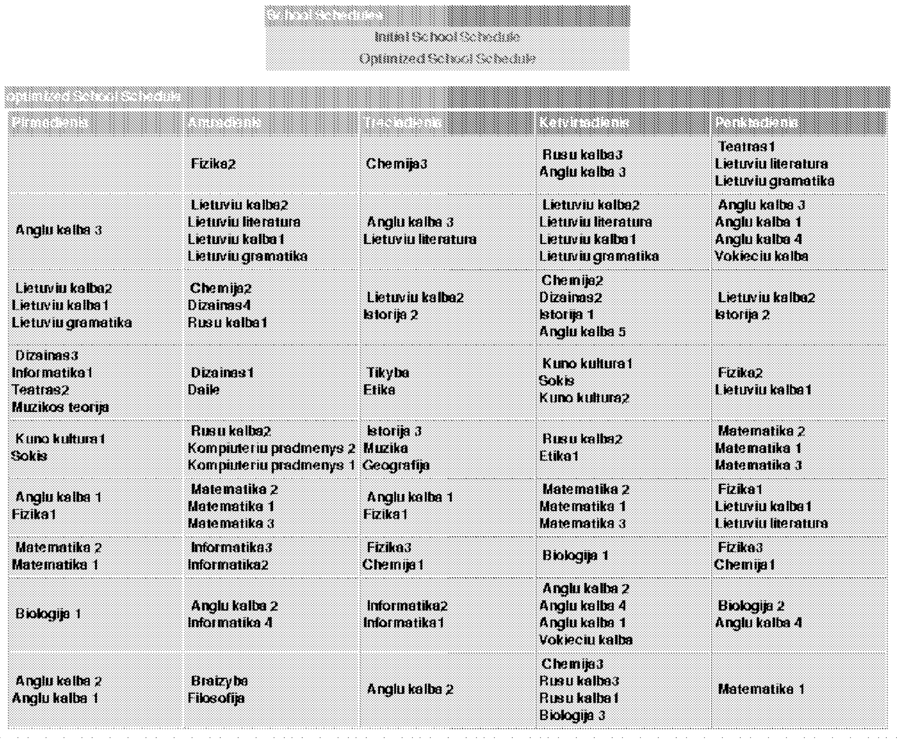
**Optimization settings:**

| | |
|---|---|
| Number of iterations: | 10 |
| Value of probability: | 0.5 |
| X1: | |
| X2: | |
| Number of iterations: | Bayes |

**Optimization results:**

| | |
|---|---|
| Penalty points for the initial version: | 2532 |
| Penalty points for the best version: | 1743 |

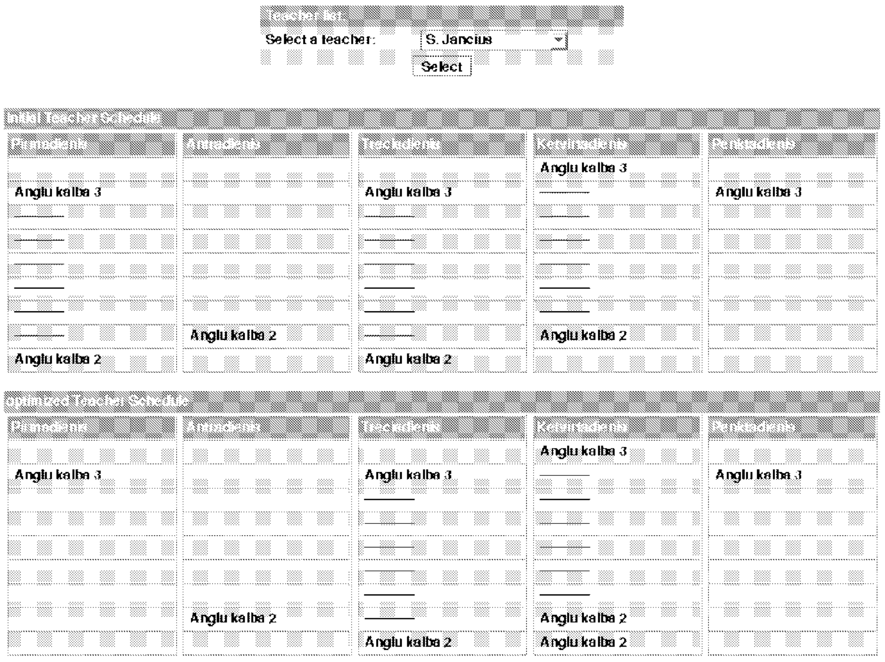
Fig. 10. Optimization window.



Optimized School Schedule

| Pirmadienis | Antradienis | Treciadienis | Ketvirtadienis | Penktadienis |
|---|---|---|---|---|
| | Fizika2 | Chemija3 | Rusu kalba3<br>Anglu kalba 3 | Teatras1<br>Lietuviu literatura<br>Lietuviu gramatika |
| Anglu kalba 3 | Lietuviu kalba2<br>Lietuviu literatura<br>Lietuviu kalba1<br>Lietuviu gramatika | Anglu kalba 3<br>Lietuviu literatura | Lietuviu kalba2<br>Lietuviu literatura<br>Lietuviu kalba1<br>Lietuviu gramatika | Anglu kalba 3<br>Anglu kalba 1<br>Anglu kalba 4<br>Vokieciu kalba |
| Lietuviu kalba2<br>Lietuviu kalba1<br>Lietuviu gramatika | Chemija2<br>Dizainas4<br>Rusu kalba1 | Lietuviu kalba2<br>Istorija 2 | Chemija2<br>Dizainas2<br>Istorija 1<br>Anglu kalba 5 | Lietuviu kalba2<br>Istorija 2 |
| Dizainas3<br>Informatika1<br>Teatras2<br>Muzikos teorija | Dizainas1<br>Daile | Tikyba<br>Etika | Kuno kultura1<br>Sokis<br>Kuno kultura2 | Fizika2<br>Lietuviu kalba1 |
| Kuno kultura1<br>Sokis | Rusu kalba2<br>Kompiuteriu pradmenys 2<br>Kompiuteriu pradmenys 1 | Istorija 3<br>Muzika<br>Geografija | Rusu kalba2<br>Etika1 | Matematika 2<br>Matematika 1<br>Matematika 3 |
| Anglu kalba 1<br>Fizika1 | Matematika 2<br>Matematika 1<br>Matematika 3 | Anglu kalba 1<br>Fizika1 | Matematika 2<br>Matematika 1<br>Matematika 3 | Fizika1<br>Lietuviu kalba1<br>Lietuviu literatura |
| Matematika 2<br>Matematika 1 | Informatika3<br>Informatika2 | Fizika3<br>Chemija1 | Biologija 1 | Fizika3<br>Chemija1 |
| Biologija 1 | Anglu kalba 2<br>Informatika 4 | Informatika2<br>Informatika1 | Anglu kalba 2<br>Anglu kalba 4<br>Anglu kalba 1<br>Vokieciu kalba | Biologija 2<br>Anglu kalba 4 |
| Anglu kalba 2<br>Anglu kalba 1 | Braizyba<br>Filosofija | Anglu kalba 2 | Chemija3<br>Rusu kalba3<br>Rusu kalba1<br>Biologija 3 | Matematika 1 |

Fig. 11. Optimized school schedule.



Teacher list:

Select a teacher: S. Jancius

Select

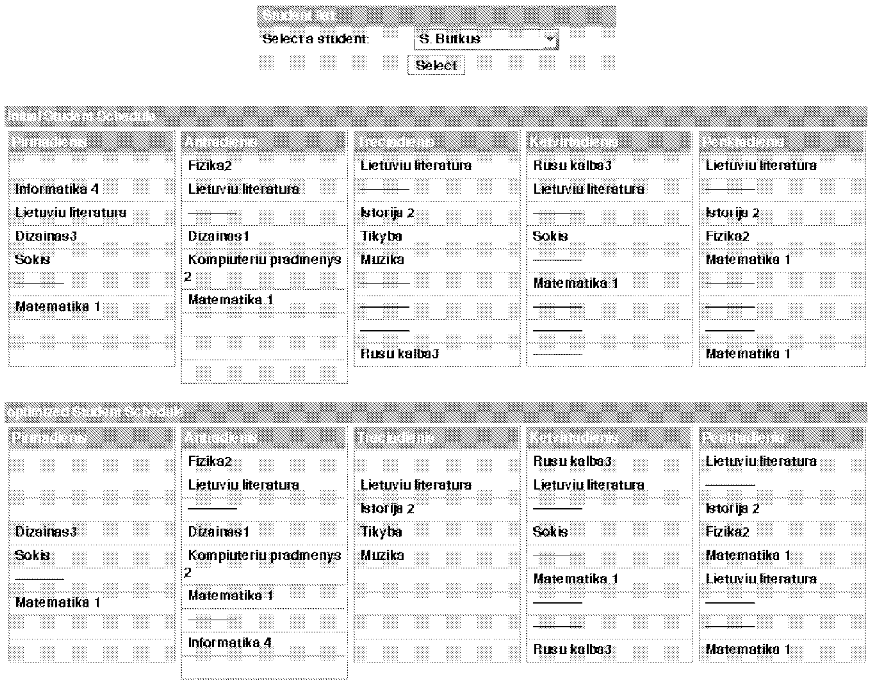
Initial Teacher Schedule

| Pirmadienis | Antradienis | Treciadienis | Ketvirtadienis | Penktadienis |
|---|---|---|---|---|
| | | | Anglu kalba 3 | |
| Anglu kalba 3 | | Anglu kalba 3 | ——— | Anglu kalba 3 |
| ——— | | ——— | ——— | |
| ——— | | ——— | ——— | |
| ——— | | ——— | ——— | |
| ——— | | ——— | ——— | |
| ——— | | ——— | ——— | |
| ——— | Anglu kalba 2 | ——— | Anglu kalba 2 | |
| Anglu kalba 2 | | Anglu kalba 2 | | |

optimized Teacher Schedule

| Pirmadienis | Antradienis | Treciadienis | Ketvirtadienis | Penktadienis |
|---|---|---|---|---|
| | | | Anglu kalba 3 | |
| Anglu kalba 3 | | Anglu kalba 3 | ——— | Anglu kalba 3 |
| | | ——— | ——— | |
| | | ——— | ——— | |
| | | ——— | ——— | |
| | | ——— | ——— | |
| | | ——— | ——— | |
| | Anglu kalba 2 | ——— | Anglu kalba 2 | |
| | | Anglu kalba 2 | Anglu kalba 2 | |

Fig. 12. Optimized schedule of teacher Jancius.



Student list:

Select a student: S. Butkus

Select

Initial Student Schedule

| Pirmadienis | Antradienis | Treciadienis | Ketvirtadienis | Penktadienis |
|---|---|---|---|---|
| | Fizika2 | Lietuviu literatura | Rusu kalba3 | Lietuviu literatura |
| Informatika 4 | Lietuviu literatura | ——— | Lietuviu literatura | ——— |
| Lietuviu literatura | ——— | Istorija 2 | ——— | Istorija 2 |
| Dizainas3 | Dizainas1 | Tikyba | Sokis | Fizika2 |
| Sokis | Kompiuteriu pradmenys 2 | Muzika | ——— | Matematika 1 |
| ——— | Matematika 1 | ——— | Matematika 1 | ——— |
| Matematika 1 | | ——— | ——— | ——— |
| | | ——— | ——— | ——— |
| | | Rusu kalba3 | ——— | Matematika 1 |

optimized Student Schedule

| Pirmadienis | Antradienis | Treciadienis | Ketvirtadienis | Penktadienis |
|---|---|---|---|---|
| | Fizika2 | | Rusu kalba3 | Lietuviu literatura |
| | Lietuviu literatura | Lietuviu literatura | Lietuviu literatura | ——— |
| | ——— | Istorija 2 | ——— | Istorija 2 |
| Dizainas3 | Dizainas1 | Tikyba | Sokis | Fizika2 |
| Sokis | Kompiuteriu pradmenys 2 | Muzika | ——— | Matematika 1 |
| ——— | Matematika 1 | | Matematika 1 | Lietuviu literatura |
| Matematika 1 | ——— | | ——— | ——— |
| | Informatika 4 | | ——— | ——— |
| | | | Rusu kalba3 | Matematika 1 |

Fig. 13. Optimized schedule of student Butkus.

That is achieved in three stages. In the first stage the same trivial permutation heuristics is used as in the traditional schedule. That works if one starts from nearly optimal schedule what is true in traditional schools as usual. In the profiled schools some "globalization" of permutation heuristics is needed.

That is achieved in the second stage by the Simulated Annealing (SA) applied with same fixed parameters such as initial "temperature" and the "cooling" rate The third fixed parameter is the probability to delete the name of some teacher waiting in the "improvement" queue. In this case one accepts with some small probability shedules with higher penalties, too, so improving convergence.

Results of SA depend on all of three arbitrarily fixed parameters. Therefore, in the final stage these parameters are optimized using methods of stochastic global optimization developed in the framework of the Bayesian Heuristic Approach (BHA) (Mockus, 2000).

Each stage can be involved separately, that makes comparison of their results much simpler. Java implementation presents an easy way to compare different schedulinggg versions for any interested person. Some simple examples of software applications presented in this paper makes easy the task of testing and applying the presented scheduling models.

The first few trials did show the better results as compared with the commercial software "MIMOSA" that now is used while sheduling Lithuanian profiled schools. However the statistical analysis of different school sheduling methods, algorithms and software systems is a separate important problem to be considered in future.

## References

Ahuja, N., S. Dozzi, S. Abourizk (1994). *Project Management Techniques in Planning and Controlling Construction Projects*. John Wiley, New York.

Baker, Kenneth R. (1974). *Introduction to Sequencing and Scheduling*. John Wiley & Sons, New York.

Cohn, H., M. Fielding. (1999). Simulated annealing: Searching for an optimal temperature schedule. *SIAM J. Optim.*, **9**, 779–802.

Greicius, A., R. Luksys (1999). *Conversion from C++ to Java of the Flow-Shop Model*. Technical report, Vytautas Magnus University, Faculty of Informatics, Kaunas, Lithuania, 1999.

Hajdu, M. (1997). *Network Scheduling Techniques for Construction Project Management*. Kluwer Academic Publishers.

Kuryla, H., J. Mockus (1995). Bayesian heuristics "learning" in a flow-shop problem. *Informatica*, **6**, 289–298.

Mockus, J. (1995). Average complexity and Bayesian heuristics approach to discrete optimization. *Informatica*, **6**, 193–224.

Mockus, J. (2000). *A Set of Examples of Global and Discrete Optimization: Application of Bayesian Heuristic Approach*. Kluwer Academic Publishers. ISBN 0-7923-6359-0.

Mockus, J., W. Eddy, A. Mockus, L. Mockus, G. Reklaitis (1997). *Bayesian Heuristic Approach to Discrete and Global Optimization*. Kluwer Academic Publishers, ISBN 0-7923-4327-1, Dordrecht–London–Boston.

Pardalos, P.M., K.A. Murthy, T.P. Harrison (1993). A computational comparison of local search heuristics for solving quadratic assignment problems. *Informatica*, **4**, 172–187.

Perlibakas, V. (1999). *Software for the Walras Problem*. Technical report, Kaunas Technological University, Faculty of Informatics, Kaunas, Lithuania (in Lithuanian).

Traub, J.F., G.W. Wasilkowski, H. Wozniakowski (1988). *Information-Based Complexity*. Academic Press, New York.

Traub, J.F., H. Wozniakowski (1992). Perspectives on information-based complexity theory. *Bulletin of the AMS*, **26**, 29–52.

**J. Mockus** graduated Kaunas University of Technology, Lithuania, in 1952. He got his Doctor habilitus degree in the Institute of Computers and Automation, Latvia, in 1967. He is a head of Optimal Decision Theory Department, Institute of Mathematics and Informatics, Vilnius, and professor of Kaunas University of Technology. His research interests include global and discrete optimization.

## Bayes'o heuristiniai metodai optimizuojant tvarkaraščius

Jonas MOCKUS

Realūs tvarkaraščių optimizavimo uždaviniai paprastai sprendžiami naudojant heuristikas kurių parametrus nustato ekspertai. Straipsnyje siūlomas kitas metodas, kai heuristikų parametrai optimizuojami tikslu sumažinti vidutinius nukrypimus nuo optimalių sprendimų.

Neretai vidutiniai nukrypimai išreiškiami dagiaekstramaliomis funkcijomis. Tokiais atvejais reikalingas globalus optimizavimas. Šiuo tikslu buvo sudaryti ir pritaikyti Bayes'o heuristiniai metodai. Tai skiriamoji šio darbo savybė.

Metodai iliustruojami dirbtuvių bei mokyklų tvarkaraščių pavyzdžiais. Nagrinėjamos dvi mokyklų tvarkaraščių optimizavimo versijos; viena skirta tradicinėms, kita profiliuotoms mokykloms. Jų modeliai išbandyti sudarant tvarkaraščius keliose Lietuvos mokyklose. Tradicinių mokyklų tvarkaraščiai vertinami pagal mokytojų "langų" skaičių. Profiliuotų mokyklų tvarkaraščiai vertinami naudojant baudų funkcijas. Tai skiria šį algoritmą nuo daugelio kitų. Programos sudarytos darbui Interneto aplinkoje ir naudojamos atliekant bendrus tyrimus bei distancinėse aukštosiose studijose. Visa programinė įranga laisvai prieinama tinklapiuose naudojant Java kalbą. Daug dėmesio buvo skirta tam, kad suinteresuoti asmenys galėtų lengvai patikrinti rezultatus bei pritaikyti siūlomus algoritmus ir programas savo uždavinių sprendimui.