INFORMATICA, 2008, Vol. 19, No. 1, 45–62
© 2008 *Institute of Mathematics and Informatics, Vilnius*

Investigation of Examples of E-education Environment for Scientific Collaboration and Distance Graduate Studies. Part 2

Jonas MOCKUS
Institute of Mathematics and Informatics
Akademijos 4, 08663 Vilnius, Lithuania
e-mail: jmockus@gmail.com

Received: December 2007

Abstract. The aim is to investigate two emerging information technologies in graduate studies and scientific cooperation. Internet is the first technology. The open source is the second. They help each other in many ways. The joint influence of both is regarded in this paper.

Results of complexity theory show the limitations of exact analysis. That explains popularity of heuristic algorithms. It is well known that efficiency of heuristics depends on the parameters. Therefore automatic procedures for tuning the heuristics help to compare results of different heuristics and enhance their efficiency.

The theory and some applications of Bayesian Approach were discussed in (Mockus, 2006a). In this paper examples of Bayesian Approach to automated tuning of heuristics are investigated. This is the Bayesian Heuristic Approach, in short. The examples of traditional methods of optimization, including applications of linear and dynamic programming, will be investigated in the next paper. These three papers represents three parts of the same work. However each part can be read independently.

All the algorithms are implemented as platform independent Java applets or servlets. Readers can easily verify and apply the results for studies and for real life optimization problems.

The theoretical result is application of unified Bayesian Heuristic Approach for different discrete optimization models. The practical result is adaptation of these models for graduate distance studies and scientific collaboration by a common java global optimization framework.

The software is regularly updated and corrected responding to new programming tools and users reports. However the general structure of web sites remains. The information is on the web site: `http://pilis.if.ktu.lt/~mockus` and four mirror sites.

Key words: Bayesian approach, tuning heuristics, global optimization, distance studies.

Introduction

The traditional way of learning is based on textbooks and formal lectures. This is natural for theoretical studies of mathematics and physics. However for new technologies of informatics and computer sciences a different approach is needed.

For scientific collaboration the possibility to run software developed by colleagues by Internet is essential. One can test directly results of other researchers by running their software with different data. Therefore algorithms, software and results published in the scientific papers can be investigate independently. This possibility is not widely used yet. But the potential appears great. We can test theorems by reading proofs. We can test the software by running Java applets. The snapshots of graphical user interfaces are useful for testing the results, too. They help to do calculations exactly as authors intended.

In open source software no clear boundaries between users and developers are defined. The software developed by a user often is applied by many others. Therefore open source software is important tool for academic studies and scientific collaboration. For example, the Linux operating system is the result of Linus Torvalds graduate studies at the University of Helsinki (Torvalds and Diamond, 2001)[1].

In the Internet environment a platform independent language running software on remote computers is needed. For example, Java, perl, python, php. Java is more efficient for scientific calculations.

The well known results of algorithm complexity show the limitations of exact solutions. That explains popularity of heuristic algorithms.

Investigating heuristic algorithms subjective factors are important. It is well known that efficiency of heuristics depends on some parameters. Researchers often work hard to define the best parameters for the proposed heuristic algorithm before submitting a paper. Therefore the published results reflects not just the quality of the heuristic algorithm but authors personal experience, too. Thus we need some automatic procedure for tuning heuristic parameters. This helps to compare different heuristics and enhance their efficiency.

A set of relevant examples are investigated. Regarding this set as a part of more general E-education environment the examples should be united by some general concepts. We need a theoretical background and some basic software tools for that.

Various examples are regarded as optimization models. That is the general idea. Comparing various heuristics and improving the efficiency we need specific optimization methods. A convenient theoretical concept is the Bayesian approach. This approach is applied for automatic tuning of heuristic parameters and for search of optimal mixtures of heuristics. That is called the Bayesian Heuristic Approach (BHA).

To implement BHA in the Internet environment an open software framework is developed using Java. Students and researchers can conveniently include their own examples as separate tasks of this unified framework. Representing the optimization results specific graphical analysis objects are added to some general display tools. New methods developed by users can be easily included, too.

The paper shows how optimization models can be implemented and updated by graduate students themselves. That reflects the usual procedures of the open source development. This way students not just learn the underlining model but obtain the experience in the development of open source software. The step-by-step improvement of the model and software is at least as important as the final result.

[1]Torvalds attended the University of Helsinki from 1988 to 1996, graduating with a master's degree in computer science. His M.Sc. thesis was titled Linux: A Portable Operating System.

That is a way to accumulate some experience in the completely new field of education when all the information can be easily obtained by internet. The internet users are filtering and transforming the information to meet their own objectives, to build their own models. Here creative approach is needed. No well defined patterns and no well tested models yet. The natural approach to research is by computer experimentation. This approach is convenient for scientific collaboration, too. We see similar patterns in collective development of scientific projects. Here researchers need fast ways to test and to apply results obtained by the colleagues.

Algorithms and models are described with references to web sites of on-line models. Examples of economic, social, and engineering models are regarded as optimization problems. For better understanding models are presented in simplified forms. Therefore computing times are reasonable, as usual.

No "perfect" examples in these web sites. All examples has some advantages and some disadvantages. Improvement of "non-perfect" models is useful both for students and for colleagues. The main objective of this paper is to help establish scientific collaboration in the Internet environment with distant colleagues and students by creating an environment of E-education and scientific collaboration in the fields related to optimization.

Bayesian Heuristic Approach (BHA)

A heuristic is some decision rule defining how the next point of observation depends on observed values. Observation means calculation of the objective function $\phi(\omega)$ at some fixed point $\omega = (\omega_1, \dots, \omega_l)$ or $\omega \in R^l$, in short. Simplest are passive heuristics (Traub *et al.*, 1988; Packel and Wozniakowski, 1987; Sukharev, 1971) when all the calculation points $(\omega^1, \dots, \omega^k)$, $\omega^i = (\omega_1^i, \dots, \omega_l^i)$, $i = 1, \dots, k$ are fixed. Well known examples are 'grids', both deterministic and Monte Carlo.

In this paper sequential heuristics are investigated where next observation ω^{i+1} depends on observed results $\omega^{i+1} = h(x, \omega^1, \dots, \omega^i)$ (Wald, 1947; Wald, 1950; Bellman, 1957). So, the search heuristic $h(x) = h(x, \omega^1, \dots, \omega^i)$ is a function of the past results $(\omega^1, \dots, \omega^i)$ and the heuristic parameters $x = (x_1, \dots, x_m)$.

In BHA tuning of heuristics is regarded as an optimization problem. We search for such parameters $x \in R^m$ of heuristics $h = h(x)$ that provide best results.

However solutions of recurrent equations of sequential statistical analysis are defined just in some special problems (Wald, 1947). No general solution is known.

We by-pass this difficulty by reducing multistage decision function to simple 'two-step' rule. Here defining coordinates of the next observation we assume that next observation will be the last. And so on until the end.

Denote the original function to be optimized as $\phi(\omega),\ \omega \in R^l$. We know just that this function belongs to some family Φ of functions. Thus we cannot define the optimization quality by just a single sample $\phi(\omega) \in \Phi$. All the family Φ have to be regarded. Assume that search time is limited and defined as the number k of search repetitions at fixed x.

Denote by (ϕ^k, ω^k) the results obtained applying k times a heuristic h to a function $\phi(\omega) \in \Phi$. Here $\phi^k = \phi(\omega^k)$, $\omega^k = \omega^k(h)$, and $\omega^k(h)$ is the final decision of heuristics h after k iterations. Formally heuristic h after k iteration transforms the original function $\phi(\omega) \in \Phi$ into another function $f(x) \in F$. Here $f(x) = \phi(\omega^k(h(x)))$ belongs to a family F of functions ϕ transformed repeating k times the heuristic h. Transformed function $f(x)$ shows how the value of the original function $\phi(\omega)$ obtained applying k times the heuristic h depends on the heuristic parameters x.

Denote results of n-th step of search for the best heuristic parameters as $(z^n, x(n))$. Here $z^n = (z_1, \ldots, z_n)$, $x(n) = (x^1, \ldots, x^n)$, $z_i = f(x^i)$, and $x^i = (x^i_1, \ldots, x^i_m)$. Using the same heuristics parameter x different results $f(x)$ are obtained depending on which sample function $\phi(\omega)$ is optimized.

The Bayesian Heuristic Approach minimizes the risk function R(x) (DeGroot, 1970; Mockus, 1989; Diaconis, 1988; Berger, 1985; Kadane and Wasilkowski, 1985) at fixed prior distribution P on a set F of functions $f(x)$. The risk function shows how the expected search results depend on parameters x. The distribution P is regarded as a stochastic model of $f(x)$, $x \in R^m$, where $f(x)$ may be a deterministic or a stochastic function. This is possible because using Bayesian approach uncertain deterministic functions can be regarded as some stochastic functions (Lindley, 1971; DeGroot, 1970; Savage, 1954; Fine, 1973; Zilinskas, 1986). That is essential feature of the Bayesian approach in this setup. For example, if several values of some deterministic function $z_i = f(x_i)$, $i = 1, \ldots, n$ are known then the level of uncertainty can be represented as the conditional standard deviation $s_n(x)$ of the corresponding stochastic function $f(x) = f(x, \nu)$ where ν is a stochastic variable. The aim of Bayesian approach is to optimize average results after fixed number of observations.

The Wiener process (Kushner, 1964a; Kushner, 1964b; Saltenis, 1971; Torn and Zilinskas, 1989) is a convenient stochastic model in the one-dimensional case $m = 1$.

The Wiener model implies continuity of almost all sample functions $f(x)$.

The Wiener model can be extended to many dimensions, too. However, the Markovian property disappears. Therefore approximate model is designed by replacing the traditional Kolmogorov consistency conditions (Mockus, 1989).

The Bayesian statistical methods are designed to optimize decisions when not many observations are available. The good asymptotic behavior is not the aim. Nevertheless some positive asymptotic results, were shown, too (Mockus, 1989; Mockus *et al.*, 1997).

Minimization of $R(x)$ is a complicated auxiliary optimization problem. That means that Bayesian methods are efficient just for complicated functions of a few ($m < 20$) continuous variables.

Optimization of heuristics is important part of Bayesian approach to optimization. This approach filters the stochastic component and optimize globally the multi-modal deterministic component. We call that Bayesian Heuristic Approach (BHA)

Software Framework for Global Optimization (GMJ)

In optimization problems theory and software are interconnected. The final results depend on the mathematical theory of optimization and the software implementation. Thus we have to regard them both.

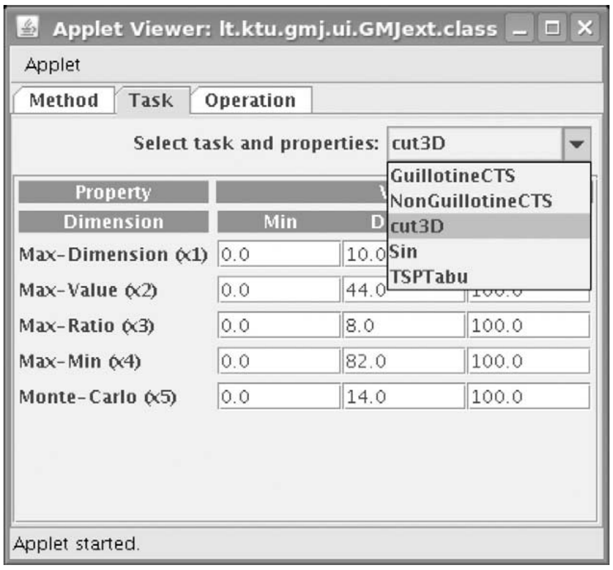
Representing a set of examples as a part of E-education environment some basic software tools are needed. The examples should be united by some common framework. We call that Global Minimizer by Java (GMJ). We regard this as a software implementation of Bayesian Heuristic Approach. We apply this approach for automatic tuning of heuristic parameters and for search of optimal mixtures of heuristics.

GMJ is open for development by users. Users contribute their own optimization methods in addition to the Bayesian ones. User optimization models are included as GMJ tasks.

The results of optimization are represented by GMJ analysis objects. A minimal set of methods, tasks, and analysis objects is implemented by default. The rest depends on users.

The examples illustrate several software implementations of heuristic optimization. The main is the open framework of Java applets GMJ. Fig. 1 shows menu of several user defined tasks. The task "GuillotineCTS" means 2D guillotine cutting (Mockus, 2006a), "NonGuillotineCTS" denotes free 2D cutting, "cut3D" defines 3D packing, both guillotine and free. In the card packing example the Baysian algorithms optimize the initial temperature x_1 and the cooling rate x_2 of Simulated Annealing (4). In the Flow-Shop human operator performs optimization of heuristic parameters. There is no heuristic parameters to optimize in the disk packing example. This way the selected examples represent different ways of heuristic tuning, from automatic by BHA to fixed non parametric heuristic in the Disk example. All the examples will be described in detail later.

The advantage of Java applets is easy access by any browser with Java support. The disadvantage is security restrictions that make exchanges of large files difficult. The signed applets allows some flexibility. However more convenient way to work with large



Select task and properties: cut3D

Property	V		
Dimension	Min	D	
Max-Dimension (x1)	0.0	10.0	
Max-Value (x2)	0.0	44.0	
Max-Ratio (x3)	0.0	8.0	100.0
Max-Min (x4)	0.0	82.0	100.0
Monte-Carlo (x5)	0.0	14.0	100.0

Applet started.

Fig. 1. Menu of GMJ tasks.

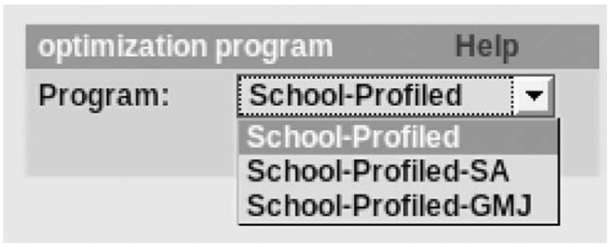


Fig. 2. Menu of optimization algorithms.

files is Java servlet where calculations are performed by server and communications are made by designated server ports, for example `http://soften.ktu.lt:8080/` and `http://pilis.if.ktu.lt:8090/`.

The servlets are used for scheduling of profiled schools where 11th and 12th grade students can choose their own set of subjects. Examples of schedule optimization of single class of 11th grade illustrates the implementation of the Bayesian Heuristic Approach as Java servlet. A reference is “school-auris”, detail description is in a separate section. Fig. 2 shows the servlet menu of different algorithms of shedule optimization. The simplest is “School-Profiled” algorithm that optimizes by closing “gaps” for teachers and moving to better schedules. The “School-Profiled-SA” algorithm implements the Simulated Annealing algorithm (4) with fixed the initial temperature x_1 and the cooling rate x_2. In the “School-Profiled-GMJ” these parameters are optimized by the Bayesian approach.

An important contribution in the development and testing of this software was made by graduate students of the Kaunas Technological University including V. Bivainis, L. Čeponis, V. Kazanavičius, M. Kvedaras, A. Pranckevičius and L. Pupeikienė.

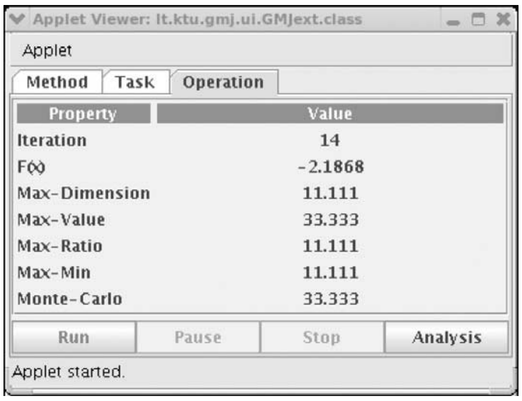
Examples of Tasks

Cutting Stock Problem (CSP)

A set of cut optimization models are good illustrations of the BHA. These models are described in the sections “GUILLOTINE-GMJ2” and “Container Packer-2” of the “Discrete Optimization” part the web site. The set includes three 2D models and a 3D model. The 2D model was briefly mentioned in (Mockus, 2006b). In (Bortfeldt and Gehring, 1997) and (Juraitis *et al.*, 2003) two different heuristics were investigated. Here we describe 3D model using BHA.

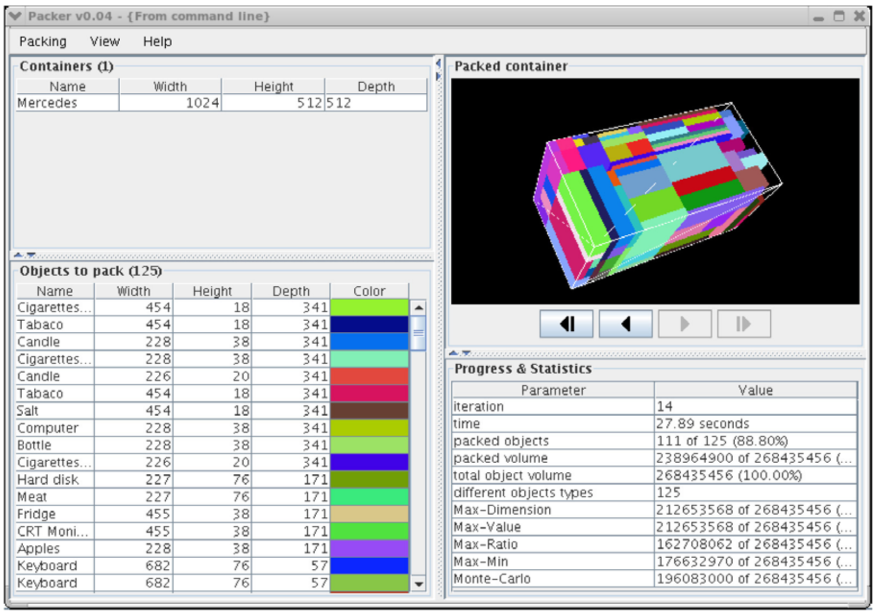
The mixture of five heuristics is minimized. The results shows Fig. 3.

There $F(x)$ is the best result, other lines define optimal percentages of different greedy heuristics:
“Max-Dimension” means “largest-first”, “Max-Value” is “most-valuables-first”, “Max-Ratio” denotes “Nearest-to-square-first”, “MaxMin” is “Maximal-of-shortest-dimension-first”, and “Monte-Carlo” means uniform probability distribution.



Property	Value
Iteration	14
F(x)	-2.1868
Max-Dimension	11.111
Max-Value	33.333
Max-Ratio	11.111
Max-Min	11.111
Monte-Carlo	33.333

Fig. 3. Optimal mixture of packer heuristics.



Containers (1)

Name	Width	Height	Depth
Mercedes	1024	512	512

Objects to pack (125)

Name	Width	Height	Depth	Color
Cigarettes...	454	18	341	🟩
Tabaco	454	18	341	🟦
Candle	228	38	341	🟦
Cigarettes...	228	38	341	🟩
Candle	226	20	341	🟥
Tabaco	454	18	341	🟥
Salt	454	18	341	🟫
Computer	228	38	341	🟨
Bottle	228	38	341	🟩
Cigarettes...	226	20	341	🟪
Hard disk	227	76	171	🟩
Meat	227	76	171	🟩
Fridge	455	38	171	🟨
CRT Moni...	455	38	171	🟩
Apples	228	38	171	🟪
Keyboard	682	76	57	🟦
Keyboard	682	76	57	🟩

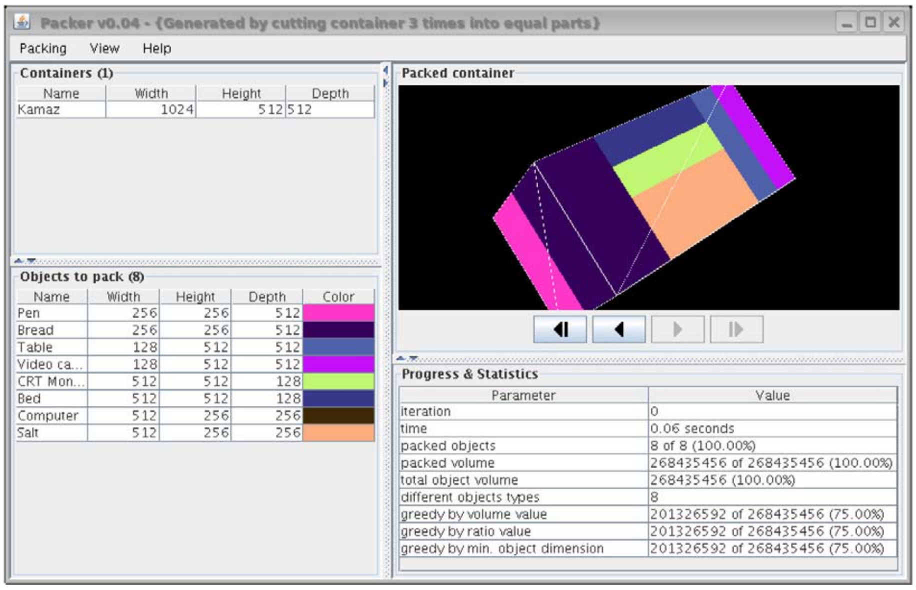
Progress & Statistics

Parameter	Value
iteration	14
time	27.89 seconds
packed objects	111 of 125 (88.80%)
packed volume	238964900 of 268435456 (...
total object volume	268435456 (100.00%)
different objects types	125
Max-Dimension	212653568 of 268435456 (...
Max-Value	212653568 of 268435456 (...
Max-Ratio	162708062 of 268435456 (...
Max-Min	176632970 of 268435456 (...
Monte-Carlo	196083000 of 268435456 (...

Fig. 4. Visualization of non-guillotine packing.

Fig. 4 is visualization of optimization results. The upper line shows the container type and dimensions. The left table defines contents and dimensions of different packages denoted by different colors. In this figure a part of 125 packages are shown. That is input data.

The upper-right image shows optimized packing of the container. The right table shows the output data. That include iteration, computing time, number of packed objects, total volume and the volume packed by optimized mixture of heuristics. Results obtained by all five ’pure’ greedy heuristics: by dimension, by volume, by ratio, by minimal dimension, and by Monte Carlo are shown too, for comparison. We see that in the packed volume of optimized mixture was 2389 and the packed volume of the best greedy heuristic was 2126. The difference is about 12%. This is typical for non-guillotine packing of different objects. The non-guillotine packing is efficient but complicated way. Alternative is guillotine packing where the objects are put into container by layers. For guillotine packing of equal objects the difference between the pure heuristics and the mixture is less, as usual. However that can be large in some instances. Fig. 5 illustrates this. Figs. 6,


Packer v0.04 - (Generated by cutting container 3 times into equal parts)
Packing
View
Help
Packed container

Containers (1)

Name	Width	Height	Depth
Kamaz	1024	512	512

Objects to pack (8)

Name	Width	Height	Depth	Color
Pen	256	256	512	🟪
Bread	256	256	512	🟪
Table	128	512	512	🟦
Video ca...	128	512	512	🟪
CRT Mon...	512	512	128	🟩
Bed	512	512	128	🟦
Computer	512	256	256	🟫
Salt	512	256	256	🟧

Progress & Statistics

Parameter	Value
iteration	0
time	0.06 seconds
packed objects	8 of 8 (100.00%)
packed volume	268435456 of 268435456 (100.00%)
total object volume	268435456 (100.00%)
different objects types	8
greedy by volume value	201326592 of 268435456 (75.00%)
greedy by ratio value	201326592 of 268435456 (75.00%)
greedy by min. object dimension	201326592 of 268435456 (75.00%)

Fig. 5. Visualization of guillotine packing of equal objects.

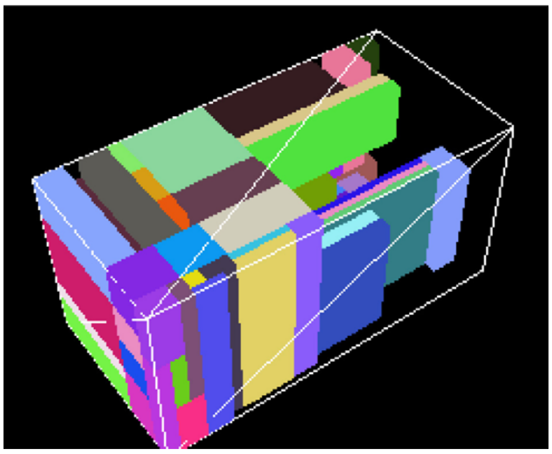


Fig. 6. Container image, unfinished packing.

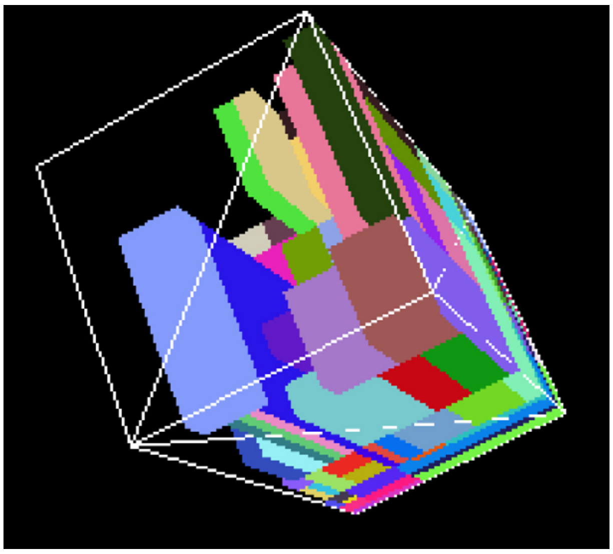


Fig. 7. Container image after rotation.

and 7 illustrate the visualization possibilities of the software. Fig. 6 shows the optimization progress, here we see just 80 objects from total packed number of 111. Fig. 7 shows the fully packed container from the opposite side.

Cards Problem (CP)

Fig. 8 shows optimization of SA parameters x_1 and x_2 (4) for the Cards Problem. The task is to select the optimal set of cards for a server and to define their optimal location. The objective function is the total resource defined as the sum of resources of the cards included in the server.

Assume N empty sites $i = 1, \dots, N$ for M cards $j = 1, \dots, M$. Denote by M_i a subset of cards fitting into space i. Denote by r_j the resource of card j. The objective is maximization of the total server resource

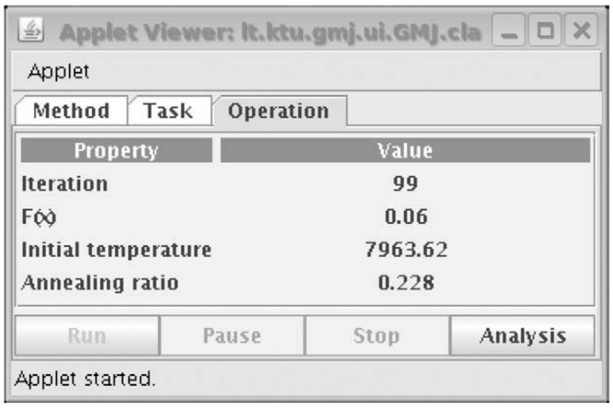
$$\max_y R(x), \tag{1}$$

$$R(x) = \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} r_j, \tag{2}$$

$$\sum_{j=1}^{M} x_{ij} r_j \leqslant 1. \tag{3}$$

Here $x = (x_{ij}, i = 1, \dots, n, j = 1, \dots, M)$ and $x_{ij} = 1$ if the card j is in the site i, otherwise $x_{ij} = 0$.

The initial layout of cards $x = x^0$ is improved by permutations. The best obtained layout is recorded after each iteration. Changes to worse layouts are done with some



Property	Value
Iteration	99
F(x)	0.06
Initial temperature	7963.62
Annealing ratio	0.228

Fig. 8. Optimization of SA parameters for Cards problem.

probabilities defined by the Simulated Annealing algorithm: move from the current layout k to the permuted layout $k+1$ with probability

$$r_{k+1} = \begin{cases} \mathrm{e}^{\frac{-h_{k+1}}{x_1/\ln(1+x_2N)}}, & \text{if } h_{k+1} > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{4}$$

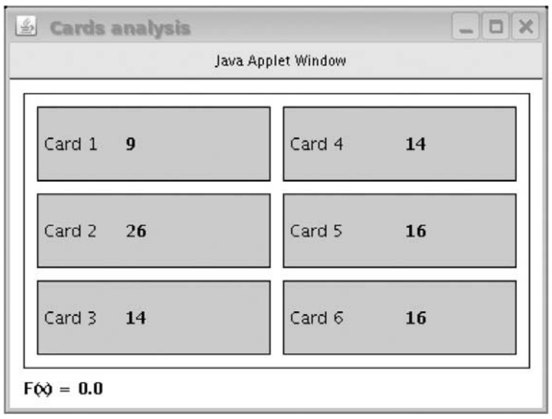
Here N is the iteration number, x_1 is the "initial temperature", and x_2 is the "annealing rate". The logarithmic "cooling schedule" $\ln(1+x_2N)$ follows from SA convergence conditions (Cohn and Fielding, 1999). The difference from the traditional SA is that we optimize parameters x_1 and x_2 for some fixed number of iterations $k = K$. $h_{k+1} = R^k - R^{k+1}$, where R^k is the current value of the server resource R, and R^{k+1} is the predicted value of R.

Fig. 9 shows the optimal location of cards. In this example the optimal set of cards $x = 9, 26, 14, 14, 16, 16$ and the optimal location of the 9th card is in the upper-left place $x_{1,9} = 1$, etc.

Flow-Shop Problem (FSP)

In flow-shop problems the sequence of machines is fixed by technology. The make-span is the objective function to be minimized. This is the time from the beginning of the first job until the end of the last one.

Fig. 10 shows the input window of the flow-shop problem. The upper-left part of the initial window defines shop options: a number of machines, a number of jobs, and a number of iterations. In the upper-right part the data source is selected. In the applet mode "Read from the internet host" is selected. The lower-right part is to select a greedy heuristic. "Longest job" means that a job with longest make-span should be the first one. "Longer remaining" time means a job with longest remaining time. "Gupta" means a priority rule proposed by Gupta (Gupta, 1971). The lower-left part defines randomization. That means a mixture of these three heuristics. In the example the mixture is $0.2, 0.2, 0.6$.



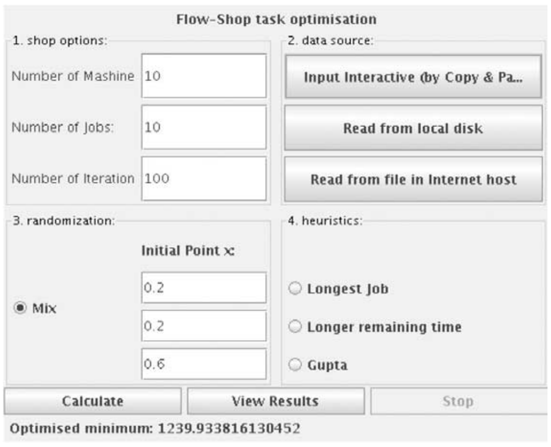
Card 1 9
Card 4 14
Card 2 26
Card 5 16
Card 3 14
Card 6 16
F(x) = 0.0

Fig. 9. Output of the Cards problem.

Fig. 11 shows the output of the flow-shop problem using this mixture. The optimal sequence of jobs is defined by the Gantt diagram (in the upper half of the picture). The Gantt diagram shows the length of jobs as beams on the vertical axis of a time line. Different shades separate different jobs. The lines in the lower part show how the make-span depends on iteration numbers. The optimal make-span is 1239 and is shown in the upper-right corner. Using the Gupta heuristic the make span is 1144. The difference from the mixture results 1131 is just about 1%. That can be explained by the well-known efficiency of the Gupta heuristic (Mockus, 2000).

School Scheduling Problem (SSP)

The objective of this example is to investigate a workable optimization system for profiled school scheduling. In user friendly optimization systems a human operator specifies



Number of Mashine 10
Number of Jobs: 10
Number of Iteration 100
Initial Point x:
[✓ Mix] 0.2 0.2 0.6
Optimised minimum: 1239.933816130452

Fig. 10. Input of the flow-shop problem.

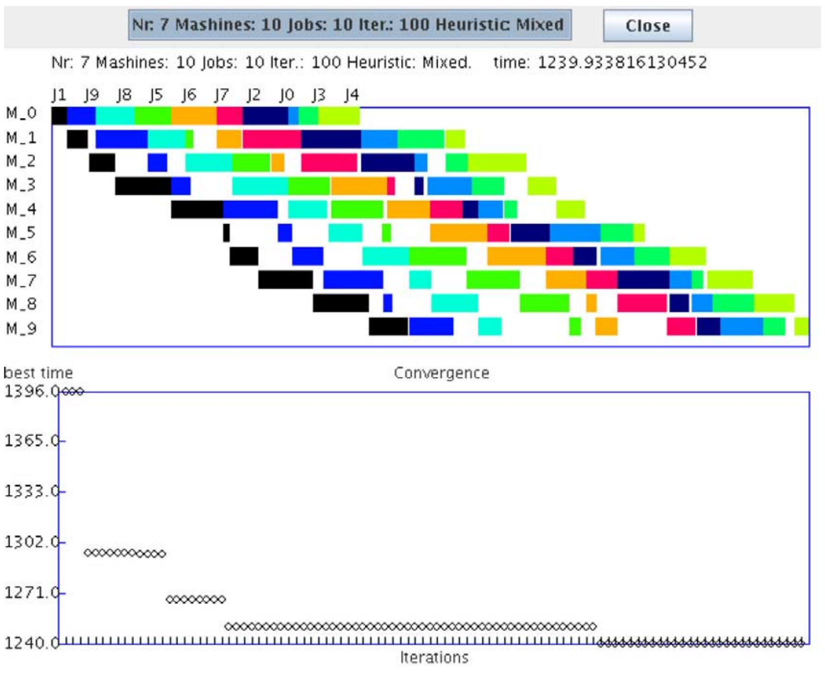

Nr: 7 Mashines: 10 Jobs: 10 Iter.: 100 Heuristic Mixed
Close
Nr: 7 Mashines: 10 Jobs: 10 Iter.: 100 Heuristic: Mixed. time: 1239.933816130452
J1 J9 J8 J5 J6 J7 J2 J0 J3 J4
M_0
M_1
M_2
M_3
M_4
M_5
M_6
M_7
M_8
M_9
best time
Convergence
1396.0
1365.0
1333.0
1302.0
1271.0
1240.0
Iterations

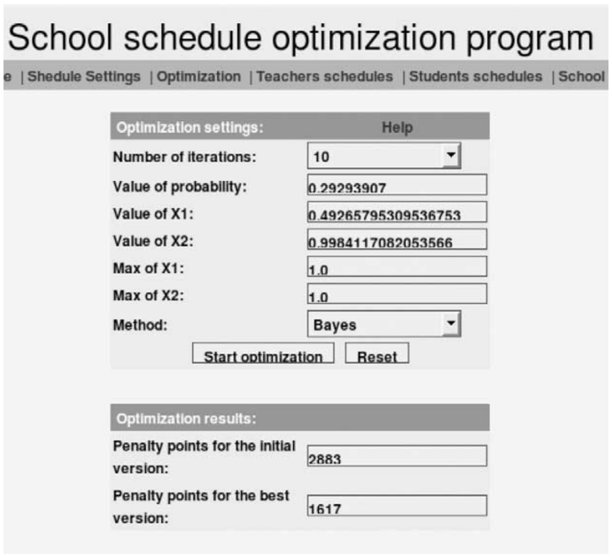
Fig. 11. Output using mixture of heuristics.

just general objectives and constraints. The schedules, both general and personal, are produced automatically. 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.

One cannot define such schedules that satisfy all restrictions and personal preferences because they contradict each other, as usual. For example, the ministry of education defines rules to be observed, the school teachers and students both prefer schedules without gaps. Thus, the objective is to find the best compromise of conflicting interests. We search for such schedule that minimizes the total penalty function.

The "physical" constraints may not be violated, for example, a person cannot be in two places at the same time. We assume, that other conditions may be violated, at a price. This is a simple way to provide Pareto-optimal solution (Pareto, 1906)

The initial schedule is improved by permutations. The best obtained schedule is recorded after each iteration. The permutations are implemented by closing teacher's gaps in a random way. A teacher is selected with probability x_0. In the simplest "close-gap" algorithm just improving permutations are regarded. This way some locally optimal schedules are obtained. Using BHA an initial schedule is improved by SA algorithm (4) with initial temperature x_1 and cooling rate x_2 optimized by the Bayesian method. SA helps to escape from local minimum. BHA provides most efficient way to do that under given conditions. The statistical tests (Mockus, 2006b) show considerably better results for BHA. Illustration is Fig. 12 that shows the output of the school scheduling problem


School schedule optimization program
e | Shedule Settings | Optimization | Teachers schedules | Students schedules | School
Help
Start optimization
Reset

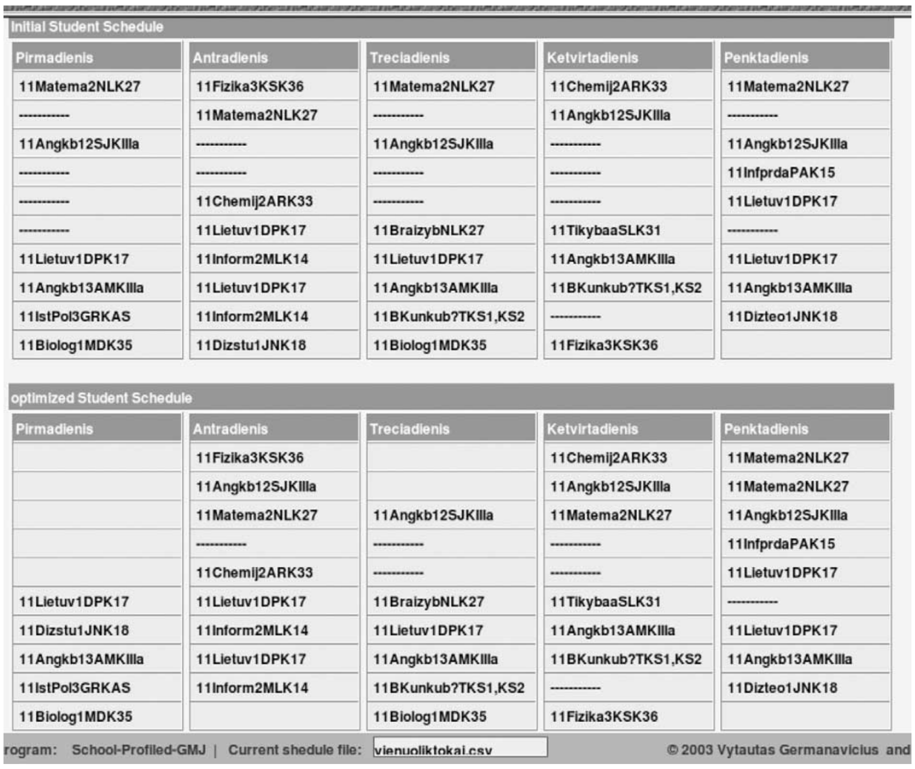
Optimization settings:

Number of iterations:	10
Value of probability:	0.29293907
Value of X1:	0.49265795309536753
Value of X2:	0.9984117082053566
Max of X1:	1.0
Max of X2:	1.0
Method:	Bayes

Optimization results:

Penalty points for the initial version:	2883
Penalty points for the best version:	1617

Fig. 12. Output using optimized SA.



Initial Student Schedule

Pirmadienis	Antradienis	Treciadienis	Ketvirtadienis	Penktadienis
11Matema2NLK27	11Fizika3KSK36	11Matema2NLK27	11Chemij2ARK33	11Matema2NLK27
-----------	11Matema2NLK27	-----------	11Angkb12SJKIIIa	-----------
11Angkb12SJKIIIa	-----------	11Angkb12SJKIIIa	-----------	11Angkb12SJKIIIa
-----------	-----------	-----------	-----------	11InfprdaPAK15
-----------	11Chemij2ARK33	-----------	-----------	11Lietuv1DPK17
-----------	11Lietuv1DPK17	11BraizybNLK27	11TikybaaSLK31	-----------
11Lietuv1DPK17	11Inform2MLK14	11Lietuv1DPK17	11Angkb13AMKIIIa	11Lietuv1DPK17
11Angkb13AMKIIIa	11Lietuv1DPK17	11Angkb13AMKIIIa	11BKunkub?TKS1,KS2	11Angkb13AMKIIIa
11IstPol3GRKAS	11Inform2MLK14	11BKunkub?TKS1,KS2	-----------	11Dizteo1JNK18
11Biolog1MDK35	11Dizstu1JNK18	11Biolog1MDK35	11Fizika3KSK36	

optimized Student Schedule

Pirmadienis	Antradienis	Treciadienis	Ketvirtadienis	Penktadienis
	11Fizika3KSK36		11Chemij2ARK33	11Matema2NLK27
	11Angkb12SJKIIIa		11Angkb12SJKIIIa	11Matema2NLK27
	11Matema2NLK27	11Angkb12SJKIIIa	11Matema2NLK27	11Angkb12SJKIIIa
	-----------	-----------	-----------	11InfprdaPAK15
	11Chemij2ARK33	-----------	-----------	11Lietuv1DPK17
11Lietuv1DPK17	11Lietuv1DPK17	11BraizybNLK27	11TikybaaSLK31	-----------
11Dizstu1JNK18	11Inform2MLK14	11Lietuv1DPK17	11Angkb13AMKIIIa	11Lietuv1DPK17
11Angkb13AMKIIIa	11Lietuv1DPK17	11Angkb13AMKIIIa	11BKunkub?TKS1,KS2	11Angkb13AMKIIIa
11IstPol3GRKAS	11Inform2MLK14	11BKunkub?TKS1,KS2	-----------	11Dizteo1JNK18
11Biolog1MDK35		11Biolog1MDK35	11Fizika3KSK36	

rogram: School-Profiled-GMJ | Current shedule file: vienuoliktokai.csv © 2003 Vytautas Germanavicius and

Fig. 13. Initial and optimized schedules for a single student.

using BHA.

Fig. 13 shows both the initial and the optimized schedule for a single student.

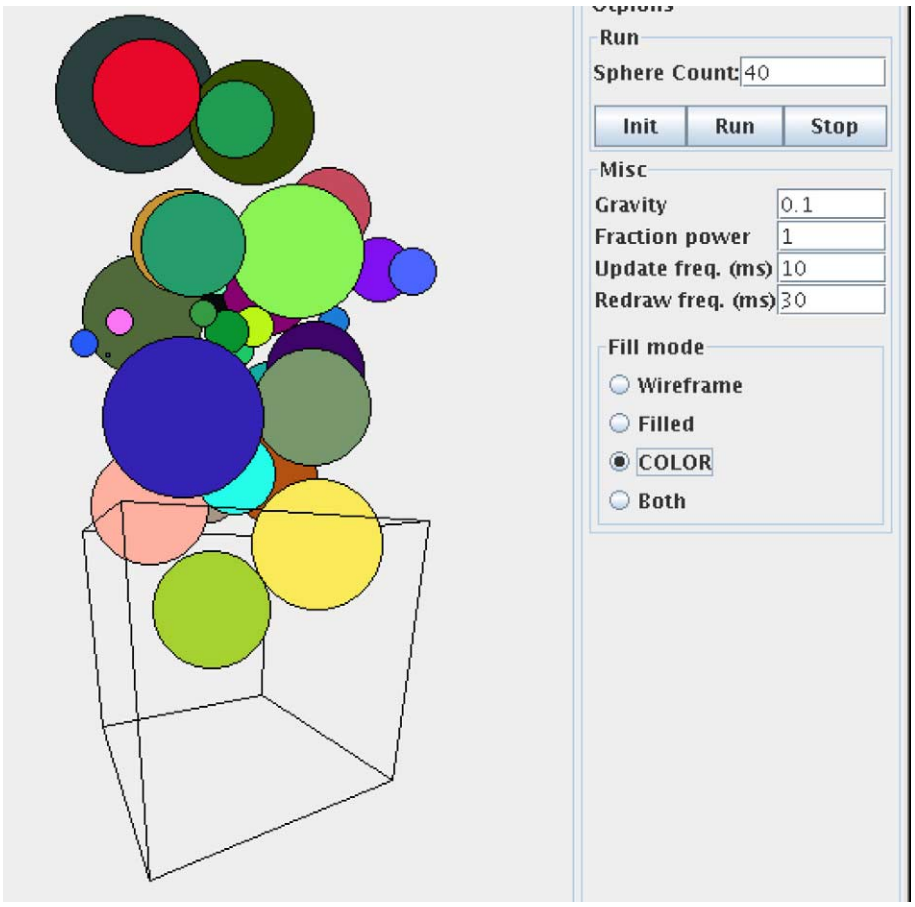
Balls Problem (BP)

Fig. 14 shows the initial state in the form of different balls (represented as disks) floating freely over the container. Optimization is by 'shaking' heuristic.

The shaking heuristic is approximately modelling the physical process of shaking balls. The general objective is maximization of total volume of balls in the rectangular container.

Users can control three visualization parameters: "Gravity" simulates the gravitational force. Small values slow-down the process, large values can be unstable,

"Fraction power" defines distribution of dimensions, larger parameter generates more small spheres,



Sphere Count: 40

Gravity 0.1
Fraction power 1
Update freq. (ms) 10
Redraw freq. (ms) 30

Fill mode: [✓ COLOR]

Fig. 14. Initial state of BP.

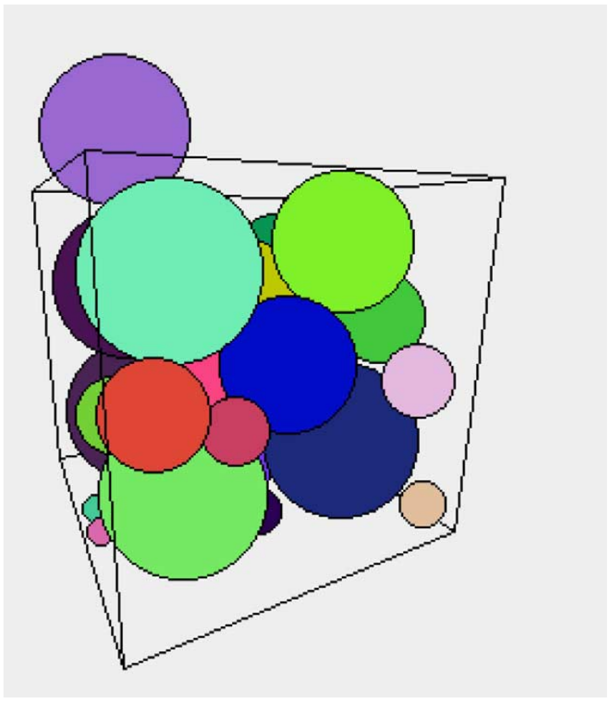


Fig. 15. Ending optimization.

“Update frequency” defines length of an iteration, recommended values are 2–10 ms, “Redraw frequency” is the length of screen redraw, recommended values are 6–30 ms.

There are no optimization parameters in this heuristic.

The graphics is controlled by “Fill Mode”. Three modes can be set:

“Wireframe” means just disk contours,

“Filled” means filled disks,

“Both” means both, this looks best.

The purpose of this example is to illustrate a different approach to the problem of optimal cutting by “physical” models. Fig. 15 shows the results of optimization by shaking heuristic.

Application in Distance Graduate Studies

The models are simple. However, they are based on fundamental results of games theory and global optimization. That makes the models useful for studies of these topics. The software is designed as an open-ended tool of research. In addition using and developing the software students better understand theory and applications. That is useful for graduate studies where research skills are important. New features can be included and new situations investigated. Implementation of the models as Java applet makes the distances almost irrelevant.

Fig. 16. Finnish-Lithuanian session, Finnish site

All the models are included into a general web based system of distance graduate studies and scientific collaboration: `http://pilis.if.ktu.lt/~jmockus`, and some mirror sites.

Now the system is used regularly for distance graduate studies in two Lithuanian universities: Kaunas Technological University and Vilnius Gedimino Technical University. The system was used for international graduate studies, too, including Lappeenranta University of Technology, SF-53851, Lappeenranta, Finland (Heilo and Mockus, 2008). Fig. 16 shows the joint Finnish-Lithuanian videoconferencing.

In the upper-right corner is a picture of Lithuanian instructor. The other three pictures show different Finnish sites.

Conclusions

1. The growing power of internet presents new problems and opens new possibilities for distant scientific collaboration and graduate studies. Therefore some nontraditional ways for presentation of scientific results should be defined.
2. The optimization models show the possibilities of some nontraditional ways of graduate studies and scientific collaboration by creating and using a specific environment for E-education.
3. Examples of applications of the BHA show the efficiency of automated tuning of heuristics.
4. The unified Bayesian Heuristic Approach (BHA) can be efficiently applied for optimization of different heuristics of discrete optimization.
5. The implementation of BHA by a common global optimization framework GMJ presents a useful software environment for graduate distance studies and scientific collaboration.

References

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, New Jersey.

Berger, J. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, New York.

Bortfeldt, A., and H. Gehring (1997). Applying tabu search to container loading problems. In *Operations Research Proceedings*. pp. 533–538.

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

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

Diaconis, P. (1988). Bayesian numerical analysis. In *Statistical Decision Theory and Related Topics*. Springer Verlag. pp. 163–175.

Fine, T.L. (1973). *Theories of Probability*. Academic Press, New York.

Gupta, J.N.D. (1971). A functional heuristic algorithm for the flowshop scheduling problem. *Operational Research Quarterly* (1970–1977), **22**, 39–47.

Heilo, M., and J. Mockus (2008). Web based system for graduate studies: Optimization, games and markets. In *Progress in Industrial Mathematics at ECMI 2006*. Springer, Berlin. pp. 741–746.

Juraitis, M., A. Riskus, T. Stonys (2003). A randomized heuristics for the container loading problem. *Information Technology and Control*, **26**, 23–31.

Kadane, J., and G. Wasilkowski (1985). Average case ϵ-complexity in computer science – A Bayesian view. In *Bayesian Statistics* 2. Elsevier Science Publishers. pp. 361–374.

Kushner, H. (1964a). A new method of locating the maximum point of an arbitrary multi-peak curve in the presence of noise. *J. of Basic Engineering*, **86**, 97–100.

Kushner, H. (1964b). A versatile stochastic model of a function of unknown and varying form. *J. of Mathematical Analysis and Applications*, **5**, 150–167.

Lindley, D. (1971). *Bayesian Statistics*: *A Review*. SIAM, Philadelphia.

Mockus, J. (1989). *Bayesian Approach to Global Optimization*. Kluwer Academic Publishers, Dordrecht-London-Boston.

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

Mockus, J. (2006a). Investigation of examples of e-education environment for scientific collaboration and distance graduate studies, part 1. *Informatica*, **17**, 259–278.

Mockus, J. (2006b). A system for distance studies and applications of metaheuristics. *Journal of Global Optimization*, **35**, 637–651.

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

Packel, E., and H. Wozniakowski (1987). Recent developments in information-based complexity. *Bulletin of the AMS*, **17**, 9–35.

Pareto, V. (1906). *Manuale di Economia Politica*. Societa Editrice Libraria, Milano.

Saltenis, V. (1971). *On a Method of Multi-Extremal Optimization*. Avtomatika i Vychislitelnayya Tekchnika (in Russian).

Savage, L. (1954). *Foundations of Statistics*. Wiley, New York.

Sukharev, A. (1971). On optimal strategies of search of extremum. In *Computational Mathematics and Mathematical Physics* (in Russian).

Torn, A., and A. Zilinskas (1989). *Global Optimization*. Springer-Verlag, Berlin.

Torvalds, L., and D. Diamond (2001). *Just for Fun*: *The Story of an Accidental Revolutionary*. HarperCollins.

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

Wald, A. (1947). *Sequential Analysis*. J. Wiley, New York.

Wald, A. (1950). *Statistical Decision Functions*. J. Wiley, New York.

Zilinskas, A. (1986). *Global Optimization*: *Axiomatic of Statistical Models, Algorithms and their Applications*. Mokslas, Vilnius (in Russian).

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 principal researcher of the Systems Analysis Department, Institute of Mathematics and Informatics, Vilnius, and professor of Kaunas University of Technology. His research interests include global and discrete optimization.

Eletroninės mokymo aplinkos, skirtos moksliniam benradarbiavimui ir aukštosioms nuotolinėms studijoms, pavyzdžių tyrimas. 2-ji dalis

Jonas MOCKUS

Tikslas – ištirti naujų informacinių technologijų įtaką aukštosioms studijoms ir moksliniam bendradarbiavimui.

Sudėtingumo teorijos rezultatai rodo, kad tiksliųjų metodų galimybės yra ribotos. Tai paaiškina heuristinių metodų paplitimą. Heuristikų efektyvumas priklauso nuo parametrų. Todėl reikalingos automatinės procedūros heuristikų optimizavimui.

Preliminarūs nuotolinių studijų rezultatai pateikti pirmajame straipsnyje (Mockus, 2006a). Šiame straipsnyje nagrinėjami pavyzdžiai skirti šių metodų taikymui optimizuojant heuristikų parametrus. Tradicinių optimizavimo metodų atitinkami pavyzdžiai bus pateikti kitame straipsnyje. Visi šie straipsniai yra trys vieno bendro darbo dalys. Tačiau juos galima nagrinėti nepriklausomai.

Visi algoritmai realizuoti nepriklausančios nuo platformos Java kalbos appletų bei servletų formoje. Todėl skaitytojai gali legvai panaudoti visus rezultatus savųjų optimizacinių modelių studijoms, tyrimui ir tobulinimui. Pilna informacija tinklapyje: `http://pilis.if.ktu.lt/~mockus` bei keturiuose jo atspindžiuose.