INFORMATICA, 1995, Vol. 6, No. 3, 289–298

"LEARNING" BAYESIAN HEURISTICS IN FLOW-SHOP PROBLEM

Jonas MOCKUS

Vytautas Magnus University
Vileikos St. 8, 2042 Kaunas, Lithuania

Henrikas KURYLA

Institute of Mathematics and Informatics
Akademijos St. 4, 2600 Vilnius, Lithuania

Abstract. We compare two alternative ways to use the Bayesian approach in heuristic optimization. The "no-learning" way means that we optimize the randomization parameters for each problem separately. The "learning" way means that we optimize the randomization parameters for some "learning" set of problems. We use those parameters later on for a family of related problems.

We define the learning efficiency as a non-uniformity of optimal parameters while solving a set of randomly generated problems. We show that for flow-shop problems the non-uniformity of optimal parameters is significant. It means that the Bayesian learning is efficient in those problems.

Key words: learning, optimization, discrete, global, Bayesian, flow-shop.

1. Introduction. We consider the average deviation as a criterion when designing numerical techniques and algorithms (Mockus, 1989). We call that a Bayesian approach (J. Mockus and L. Mockus, 1991).

We apply the Bayesian approach in a non-traditional way. Namely we define an a priori distribution on a set of randomized heuristic decision rules (A. Mockus, J. Mockus and L. Mockus, 1995).

We optimize the randomization parameters by multiple application of randomized heuristics. We may optimize those parameters for each problem separately. Alternatively we may optimize the randomization parameters only for some "learning" set of problems. We may use the "learned" parameters later on without any additional optimization.

The question is will the learning help? We try to get the answer by Monte Carlo simulation of a set of flow-shop problems using randomized heuristics.

We generate 100 flow-shop problems with 10 tasks and 10 tools each (Biegler *et al.*, 1988). We choose the operation durations uniformly on a set of integers from 0 to 99. We optimize the parameters for each problem.

We regard learning as irrelevant, if the density of optimal values of parameters is uniform. Therefore we define the "non-uniformity" of optimal parameters as a measure of learning efficiency. We show that in the flow-shop problems the non-uniformities are significant.

2. Flow-shop problem

2.1. Definition. Denote by $\tau_{j,s}$ the duration of operation (j, s), where $j \in J$ denotes a job and $s \in S$ denotes a tool. We denote by J and S the set of jobs and tools, respectively. Assumption $\tau_{j,s} = 0$ means that operation (j, s) is irrelevant.

Suppose that the sequence of tools s is fixed for each job j. One tool can do only one task at a time. Several tools cannot do the same task at the same moment.

The decision $d_i(j) \in D_i$ means to start a $j \in J_i$ at the stage i. We define the set of feasible decisions D_i as the set J_i of tasks available at the stage i conforming to the flow-shop rules.

The objective function is the make-span v. Denote by $T_j(d)$ the time when we complete the task j (including the gaps between operations) using the decision sequence d. Then the make-span for d is $v(d) = \max_{j \in J} T_j(d)$.

2.2. Algorithm

2.2.1. Permutation schedule. We can see that the number of feasible decisions is very large. We can reduce this number considering only a small subset of schedules, so-called permutation schedules.

The permutation schedule is a schedule with the same task order on all tools – a schedule that is completely characterized by a single permutation of task indices $1, 2, \ldots, n$. We assume the first operation to be on the first tool, the second – on the second, and so on. The expert opinion is that using permutation schedules we can approach the optimal decision well enough (Baker, 1974).

Denote

$$\tau_j = \sum_{s=1}^{|S|} \tau_{j,s},$$

where $|S|$ stands for the number of tools.

We call τ_j the length of the task j.

2.2.2. Heuristics. Define the heuristics by expression

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

Here

$$A_i = \min_{j \in J_i} \tau_j, \quad A^i = \max_{j \in J_i} \tau_j. \tag{2}$$

Thus heuristics prefer longer tasks at each stage i.

In both cases we define the randomized decision function r_i by expression (3). We optimize it by solving the stochastic optimization problem (2.2.4).

2.2.3. Randomization. An ordinary polynomial may be a good representation if we prefer the probabilities r_i to be expressed as some monotonous functions of the heuristics h_i. Here we use zero-one condition:

$$r_i^0 = r^0(h_i, x) = \sum_{n=0}^{N-1} a_{in} x_n h_i^n(m), \tag{3}$$

where

$$\sum_{n=0}^{N-1} x_n = 1, \quad x_n \geqslant 0, \ n = 0, \ldots, N-1, \tag{4}$$

and

$$a_{in} = \frac{1}{\sum_{m=1}^{M_i} h_i^n(m)}.$$

The number n in expression (3) may be regarded as the "degree of greed". The number $n = 0$ means no greed, because all feasible decisions are equally desirable. If the number of greed n is large, then we prefer the decisions with the best heuristics. Optimizing x we define degrees of greed such that provide the most efficient randomized decision procedure.

2.2.4. Parameter optimization. We denote by $f_K(x)$ the best value of the objective function obtained after K repetitions of the randomized decision procedure $r(x)$ for a fixed x. We defined the randomized decision procedure so that

$$\lim_{K\to\infty} f_K(x) = v, \quad \text{for any fixed } x. \tag{5}$$

It means that all the values of parameters x are "good" asymptotically. However the right choice of x could be very important if the number of repetitions K is not large.

Thus we get the following continuous problem of stochastic programming

$$\min_x f_K(x),$$

where

$$\begin{aligned} &x = (x_0, \ldots, x_{N-1}), \\ &\sum_{n=0}^{N-1} x_n = 1, \\ &x_i \geqslant 0, \quad n = 0, \ldots, N-1. \end{aligned} \tag{6}$$

This way we reduced, in the described sense, the flow-shop problem to the problem of continuous stochastic optimization (2.2.4).

2.3. Results

2.3.1. No-learning mode. Table 1 shows the results of the Bayesian method after 100 iterations using heuristic (1) and different randomization procedures. The number of tasks J, of tools S, and of operations O each of them is equal to 10. We define the lengths and the sequences of operations generating random numbers uniformly from 0 to 99. We estimate the expectations and the standard deviations for each type of randomization using the results of 40 optimization of the same randomly generated problem.

Table 1. The results of Bayesian methods using heuristics (1)

$R = 100$, $K = 1$, $J = 10$, $S = 10$, and $O = 10$					
Randomization	f_B	d_B	x_0	x_1	x_2
Taylor 3	6.173	0.083	0.304	0.276	0.420
CPLEX	12.234	0.00	–	–	–

In Table 1 the symbol f_B denotes the mean, and d_B denotes the standard deviation of span. "Taylor 3" denotes randomization (3) with the number of terms $N = 3$, and "CPLEX" denotes the results of the well known general discrete optimization software after 2000 iterations (one CPLEX iteration is comparable with one Bayesian observation). The bad results of CPLEX show that the standard MILP technique is not efficient solving a specific problem of discrete optimization. It is not yet clear how much one may improve the results using specifically tailored branch-and-bounds techniques. We have some doubts about that.

2.3.2. Learning mode. We define the lengths and the sequences of operations generating random numbers uniformly from 0 to 99. We generate 100 different problems. We optimize randomization parameters of each problem separately by Bayesian techniques. The first three figures show the density of optimal parameters.

Fig. 1 shows the density of the first parameter pair (x_1, x_2).

Fig. 2 shows the density of the second pair (x_1, x_3), and Fig. 3 shows the density of the third pair (x_2, x_3).

The last three figures show the objective as a function of the first pair of parameters (x_1, x_2) for three different samples of the flow-shop problem.

We smoothed all the density functions and objectives to improve the visualization. For the densities (see the first three figures) we used the following smoothed function:

$$T(x) = 1/N \sum_{i=1,N} e^{-C_k|x_i - x|}.$$

We smoothed the objectives (see the last three figures) this way:

$$F(x) = \frac{1}{\sum_{i=1,N} f(x_i)\, e^{-C_k|x_i - x|}} \sum_{i=1,N} e^{-C_k|x_i - x|}.$$

Here $T(x)$ denotes the smoothed optimal parameter density, $F(x)$ denotes the smoothed objective, N is number of points, and $C_1 = C_2 = C_3 = 15$, $C_4 = 22$, $C_5 = 18$, $C_6 = 22$ are smoothing parameters.

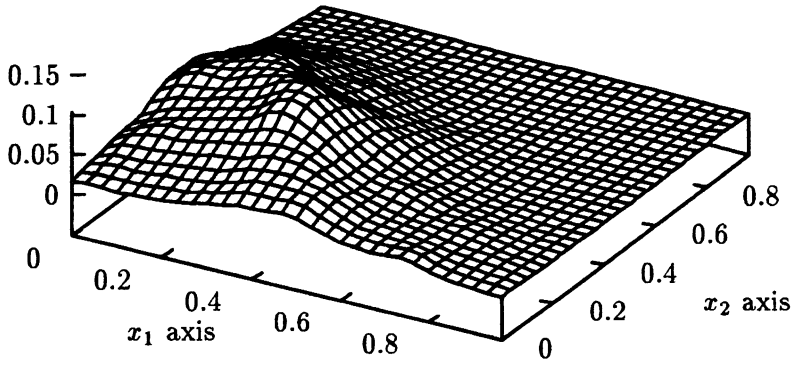

Fig. 1. Density of optimal parameters x_1, x_2.

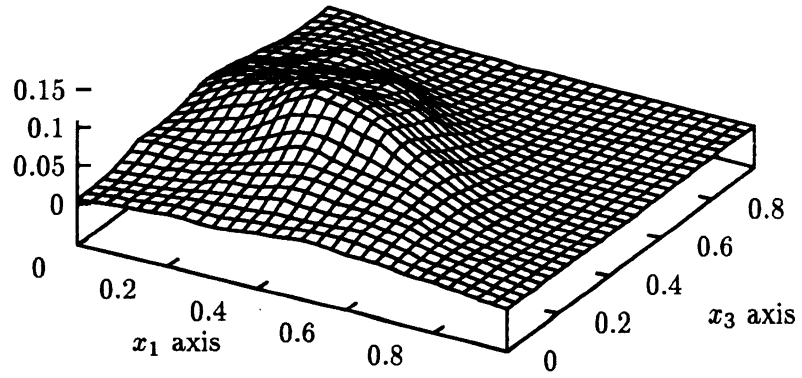

Fig. 2. Density of optimal parameters x_1, x_3.

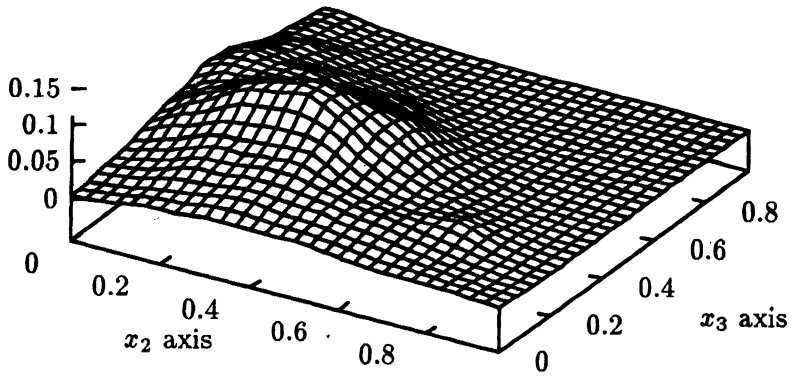

Fig. 3. Density of optimal parameters x_2, x_3.

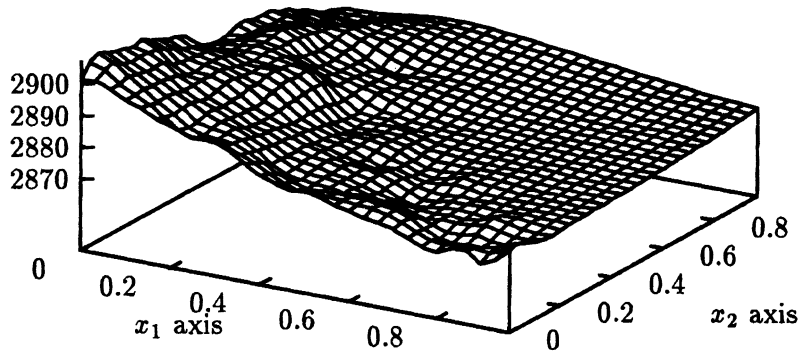

Fig. 4. The objective of the 1-st sample problem as a function of parameters x_1, x_2.

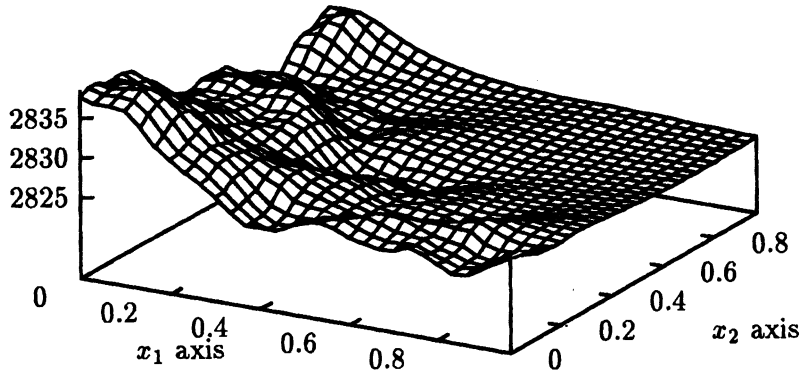

Fig. 5. The objective of the 2-nd sample problem as a function of parameters x_1, x_2.

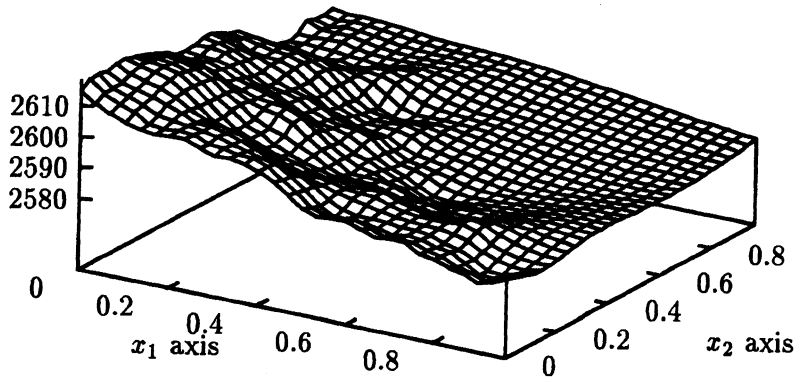

Fig. 6. The objective of the 3-d sample problem as a function of parameters x_1, x_2.

REFERENCES

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

Biegler L.T., I.E. Grossmann and G.V. Reklaitis (1988). Applications of operations research methods in chemical engineering. In R.R. Levary (Ed.), *Engineering Design: Better Results Through OR Methods.* North-Holland, Amsterdam.

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

Mockus, J., and L. Mockus (1991). Bayesian approach to global optimization and applications to multi-objective and constrained optimization. *J. of Optimization Theory and Applications,* **70**(1),

Mockus A., J. Mockus and L. Mockus (1995). Bayesian approach to discrete optimization. *Journal of Global Optimization* (submitted).

Received September 1995

J. Mockus graduated Kaunas Technological University, Lithuania, in 1952. He got his Doctor habilitus degree in the Institute of Computers and Automation, Latvia, in 1967. He is a Professor of Vytautas Magnus University, Kaunas, Lithuania.

H. Kuryla was born in 1969. He graduated the Faculty of Informatics of the Kaunas Technological University in 1992. He is a junior research fellow at the Department of Optimal Decision Theory of the Institute of Mathematics and Informatics. His research interests are applications of global optimization.

BAJESO HEURISTIKŲ „APSIMOKYMAS" SPRENDŽIANT TVARKARAŠČIO PROBLEMĄ

Jonas MOCKUS, Henrikas KURYLA

Straipsnyje sulyginami du būdai naudojant Bajeso metodus heuristiniame optimizavime. Būdas „be apsimokymo" reiškia, kad heuristikų raddomizavimo parametrai optimizuojami kiekvienai problemai atskirai. Būdas „su apsimokymu" reiškia, kad šie parametrai optimizuojami visai problemų šeimai. Atlikus seriją skaičiavimų, sprendžiant eilę tvarkaraščio optimizavimo problemų, parodoma, kad Bajeso heuristiniai metodai gerai tinka naudojant „apsimokymo" būdą.