SYSTEMS ANALYSIS

PROBLEMS OF DISCRETE OPTIMIZATION: CHALLENGES AND MAIN APPROACHES TO SOLVE THEM¹

I. V. Sergienko and V. P. Shylo

UDC 519.854

This paper briefly reviews the current state of the art in the field of discrete optimization problems. Emphasis is on the generalization of the experience gained at the V. M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine in research and development of solution methods and software for various classes of complicated discrete programming problems.

Keywords: *discrete optimization, decrease-vector method, method of global equilibrium search, metaheuristics, computational experiment, algorithm efficiency.*

Formation of the modern information society requires new information technologies. The theory and practice of their development are based on optimization methods, in particular, discrete optimization, and the theory of systems analysis.

Most discrete optimization problems are usually universal (*NP-hard*), which makes it almost impossible to create such efficient methods to solve them as methods of linear and convex optimization. Moreover, with expansion of the range of applications (see, for example, [8, 10, 12–15, 22, 26, 28, 29]), discrete programming problems become more complicated. The computational difficulties faced in solving such problems are frequently due to large dimension, multiextremeness, and inaccurate values of coefficients. This obviously makes it impossible to develop acceptable exact methods for most classes of problems. Moreover, mathematical models (including discrete ones) of applied problems of choosing the best solutions do not usually represent adequately real situations, and are just approximate descriptions. Therefore, it is expedient to solve discrete optimization problems with a definite degree of approximation to the optimum, and with account for a possible inaccuracy and incorrectness in the representation of initial data. In this connection, of special importance are development and study of various approximate methods of solving discrete optimization problems, which do not usually guarantee that the optimal solution will be found. We believe that it is this way that may lead to a significant effect of applying mathematical methods and computers to solve complicated applied problems of discrete optimization.

In the present paper, we briefly review (without claiming completeness) the modern state of the art in the field of discrete optimization problems. Emphasis is on the generalization of the experience gained at the V. M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine, under the guidance and with immediate participation of the authors, of development and study of solution methods and software for different classes of complicated problems in discrete programming.

By a general discrete optimization problem, we will mean the problem of minimizing or maximizing a function defined on a discrete set. Hereafter, we consider a problem

$$\min_{x \in G} f(x), \quad (1)$$

where G is a finite or an enumerable set of admissible solutions (the domain of definition) and $f(x)$ is the objective function defined on this set.

¹The study was partially sponsored by the grant UKM2–2812–KV–06 (SRDF Cooperative Grants Program).

V. M. Glushkov Institute of Cybernetics, National Academy of Sciences of Ukraine, Kiev, Ukraine. aik@public.icyb.kiev.ua; shylo@public.icyb.kiev.ua. Translated from *Kibernetika i Sistemnyi Analiz*, No. 4, pp. 3–25, July–August 2006. Original article submitted July 4, 2006.

It is possible to divide conventionally discrete optimization methods into two groups: exact and approximate. It is well known that many exact methods may be used as approximate ones, and approximate methods may be used, in certain situations, as exact ones or as their component. This is the conditionality of the division of methods.

The main exact methods are branch-and-bound, branch-and-cut, successive analysis of alternatives, etc. [11–14, 17, 20, 26, 31]. Popular ways of solving discrete optimization problems are branch-and-bound methods described in many publications, for example [11, 14, 17, 20]. These methods are distinguished by the way they calculate estimates and branches. The lower-bound estimates of the functional are derived by relaxing the original problem by linearizing its objective function or by reducing the original problem to an integer programming problem, which is then replaced by a linear programming problem. Choosing a branching strategy and a technique to find estimates is a rather complicated problem, which has not been solved adequately yet.

Much attention has been paid recently to the development of solution algorithms for discrete optimization problems that could be implemented on multiprocessor computers. The branch-and-bound algorithm has real capabilities for parallelizing, which greatly expands the range of application of this approach to solve high-dimensional discrete programming problems. Studies in this area are conducted at the Institute of Cybernetics (to be mentioned below) and are far-reaching.

To exactly solve completely or partially integer linear programming problems, branch-and-cut methods are now successfully applied, which are based on the branch-and-bound algorithm and the cut method. Cut methods allow more accurate approximation of integer programming problems due to relaxing integer variables, and specially designed branch-and-bound algorithms efficiently use the obtained estimates to decrease the number of alternatives to be considered. Development of the theory of polyhedrons and creation of powerful problem-oriented cut methods and their combination with branch-and-bound algorithms have revived branch-and-cut methods.

Nowadays, branch-and-cut methods are successfully applied to solve a wide range of complicated discrete optimization problems. Today, the branch-and-cut method is considered the best exact method to solve the traveling salesman problem [43, 74, 100]. These methods are also successfully used to solve problems in scheduling theory, network design, packaging, maximum feasibility, in medical and biological applications [53, 80]. Branch-and-cut methods also find application in the analysis of general discrete programming problems [57], which cannot always be solved successfully by constructing cutting planes. The combination of the branch-and-bound algorithm with the cut method allows us to speed up problem solution. Efficient sets of cuts are presented in [44].

Branch-and-cut methods combined with approximate methods allow us, in many cases, to find stiffly accurate upper and lower bounds for the best value of the objective function. We successfully used such an approach to develop an approximate branch-and-cut algorithm for the multidimensional knapsack problem with Boolean variables and an exact algorithm for finding a maximum-size error-correcting code [36]. The latter problem can be reduced to finding the maximum independent set of nodes of a graph. For this problem, a technique of estimating the potency of independent sets using the specificity of graphs is proposed. This makes it possible to reduce drastically the amount of calculation in the branch-and-cut algorithm.

Combinatorial methods are an important class of exact discrete optimization methods. They include the method of successive analysis and elimination of alternatives, proposed by Mikhalevich and Shor [12–14]. Its scheme is theoretically most general and generates a set of other schemes. For example, dynamic programming methods, branch-and-bound algorithms can be described within its framework.

As indicated above, in solving many discrete optimization problems, in particular, high-dimensional problems, capabilities of exact methods are restricted. This is confirmed by the results of a numerical experiment, solution of applied problems, and a series of theoretical studies [16, 17]. In this connection, development and study of various approximate methods for solving such problems are important.

Many publications, for example [17, 20, 26], are devoted to approximate discrete optimization methods. These methods can be divided conventionally into methods employing exact methods, greedy algorithms, methods of random search, local optimization methods, and metaheuristic methods.

To derive approximate solutions to discrete optimization problems, it is possible to use well-known exact methods: of cuts (direct algorithms), successive analysis of alternatives, branch-and-bound, etc. As a rule, branch-and-bound algorithms generate the best approximate methods. In solving a problem, not only the sequence of approximate solutions is derived but also their deviation from the optimum is estimated.

A rather general approach to approximate solution of discrete optimization problems is implemented in greedy algorithms. They are used to find approximate solutions by sequential determination of variables based on some information on the problem [9, 67]. Papadimitriou and Steiglitz showed in [16] that greedy algorithms exactly solve combinatory

optimization problems for matroids. But for the majority of important discrete programming problems, they find solutions that are not sufficiently close to the optimal ones in the objective function. Therefore, greedy algorithms are usually applied to derive solutions that are then used as starting algorithms in local search.

Random search methods are often used in solving discrete optimization problems. Each of them is determined by the form, the type of representation of the probabilistic distribution and generation of admissible solutions. There exists an uncontrollable (“blind”) random search, where the law of distribution is specified irrespective of the problem statement and does not vary during the solution. These methods are simple to implement but are hardly promising. If the law of probabilistic distribution depends on the problem being solved and varies during the search, then such methods are called methods of directed random search. The methods that combine random search with local optimization produce good results [26, 52, 67], since they make it possible, in certain situation, to combine the advantages of both approaches.

Local optimization methods have intensively been developed for a long time. As far back as at the end of the 50s of the 20th century, local search algorithms were used [59] to solve the traveling salesman problem. Later, various neighborhoods for this problem were analyzed [86] and the strategy of multiple start of local search was introduced, which became the first step to create metaheuristics. Further, local search was applied to solve other problems. In particular, the variable-depth method [16] was successfully used for the graph partition problem [84] and the traveling salesman problem [87]. Nowadays, algorithms based on this method are successfully used in solving unconstrained quadratic programming problems with Boolean variables, the weighted problem on maximum feasibility, etc. See [7, 17, 20, 26, 28] for a detailed review of local optimization methods.

A correct combination of expansion and contraction of the search area to find the global optimum is the general problem for local optimization methods. On the one hand, an output from the domain of the local optimum obtained is necessary, i.e., expansion of the search area (a so-called search diversification). On the other hand, to derive an improved solution, it is expedient to analyze points close to the local optimum found, i.e., to narrow down the search area (search intensification). Finding the necessary balance between the expansion and contraction of the solution search area is the most important problem for local methods.

Discrete programming methods that combine various ideas associated with local optimization are efficient [26]. Among the approaches that employ local search, noteworthy are simulated annealing method [85], tabu search [69, 70], decrease-vector method [17, 27], global equilibrium search (GES) [26, 32], and memetic algorithms [91].

In the beginning of the 60s of the last century, Sergienko proposed in [27] a new approach to develop algorithms for approximate solution of discrete optimization problems (1). A new algorithmic scheme, the so-called scheme of the decrease vector method, is created, which made it possible to design a set of new algorithms of local search. Further it was essentially developed [1–4, 17, 18, 20, 23, 26, 37] and used to create various computer technologies to solve many types of discrete programming problems. The convergence of computational processes and the stability of multicriterion discrete optimization problems were analyzed. The results were presented in many publications, in particular, in [17–20, 26, 28].

The further development of local search methods are metaheuristic methods [41, 65, 68, 82, 95, 96, 98, 99, 105], which appeared in the 80s to solve complex discrete programming problems. Algorithms were developed that admit transitions to solutions with a greater value of the objective function as compared with the current value and transitions to inadmissible solutions, which do not belong to the domain of definition of the problem. These transitions make it possible in some cases to avoid stopping at local minima of poor quality. These approaches include the simulated annealing method [52, 55, 85] and tabu search [69, 70, 72]. There are also methods called metaheuristics, where local search is combined with other heuristic algorithms, for example, with genetic algorithm [78] and ant colony optimization [60], and is their basic component.

Genetic algorithms were proposed in the middle of the 70s [78] and came to be applied to discrete optimization problems in the 80s. These algorithms are based on the idea of improving the set (populations) of found solutions with the use of the approaches based on genetic variability and natural selection. Genetic algorithms were used to solve many discrete optimization problems [66, 105]. They are as yet inferior in speed to simulated annealing and tabu search but continue to be actively developed.

Smoothing algorithms can be applied in solving a wide range of discrete optimization problems. A smoothing algorithm was presented in [75] to solve a traveling salesman problem; eight variants of such an algorithm were used in [58] to solve 70 traveling salesman problems (the number of cities varied from 100 to 2392). The solutions found were compared with the solutions obtained by local search with neighborhoods “2, 3 — substitution” [26]. The smoothing algorithm with a smoothing function known as sequential smoothing appeared to be the best.

Voudouris and Tsang [115] have proposed guided local search as a general method for solving discrete optimization problems. In this method, the objective function is modified by adding some set of penalties, and local search is performed in

bounded perspective domains. Voudouris and Tsang mentioned in [114] that guided local search is conceptually related to tabu search. It was used to solve many problems [109, 111, 114]. Supplemented with the algorithm of fast local search, this method produced the results comparable to those of the best algorithms for the traveling salesman problem [114].

The GRASP algorithm proposed in [61] was initially used in solving a complex problem on a graph vertex cover and then in many other problems [106]. It consists of two stages, which usually iterate several times. At the first stage, a feasible solution is found using greedy algorithms, and at the second stage, attempts are made to improve it (or its random perturbation) using local search. The GRASP algorithm is quite easy to use since it has a few controlled parameters. A similar scheme (called locally stochastic algorithms) is proposed in [67]; it falls into the general pattern of the decrease-vector method [17].

Scatter search and its applications to assignment problems and integer programming problems with mixed variables are described in [71, 72, 89]. This method, as an evolutionary procedure (similar to genetic algorithms) combines available solutions to derive new, better solutions. Such algorithms were used in the 60th to solve scheduling problems. The concept of union (aggregation) of solutions (or constraints) was also successfully used in integer and nonlinear programming.

To solve discrete optimization problems, Dorigo and Gambardella [60] have proposed the ant colony algorithm, which employs analogies to the behavior of an ant colony in delivering foodstuffs to the anthill. This algorithm was applied to solve the traveling salesman problem, the quadratic assignment problem, problems of routing freight traffic, and in communication networks [51, 60, 63].

The idea of systematic variation of local search neighborhood is used in [77, 92]. It resulted in an efficient local-type variable neighborhood algorithm. Note that such a possibility is provided and efficiently used in the decrease-vector method [17]. The variable neighborhood search [92] allows us to explore more and more remote neighborhoods of the current solution and to pass to an improved new solution if it is found. This algorithm is easy to apply and can easily be combined with other metaheuristic algorithms. Different modifications of the variable neighborhood search were also successfully used to solve median problems [76] and finding an extremal graph [54].

Since the 80s, many metaheuristic algorithms have been developed to solve complex applied problems described in terms of discrete programming. Some of them have certain shortcomings — many parameters which are difficult to adjust and much time needed to find feasible solutions. But the main shortcoming of these algorithms is inefficient use of information obtained during the solution, especially in high-dimensional problems. However, the prospects for continuing studies and developing metaheuristic algorithms are promising.

Recently, the Institute of Cybernetics have conducted intensive research to compare the existing and newly developed (more perfect) discrete optimization methods based on local search and used to solve complicated problems. The method of global equilibrium search [32] is among new methods; it is a development of the decrease-vector method [17] and is conceptually close to the simulated annealing method [85]. Despite the successful application of the simulated annealing method to solve many complicated optimization problems, its asymptotic efficiency is even less [62] than that of the trivial method of iterated random local search. The method of global equilibrium search (GES) possesses all advantages of the simulated annealing method and does not have its shortcomings. The results of numerous computational experiments for a set of various classes of discrete programming optimization problems conducted using this method [26, 32–34, 38, 39] allow us to conclude that this research area is promising.

To describe the GES method, we will use the following formulation of the discrete optimization problem:

$$\min \{f(x) | x \in G \cap B^n\}, \quad (2)$$

where $G \subset R^n$, R^n is the set of n -dimensional real vectors and B^n is the set of n -dimensional vectors $x = (x_1, \dots, x_n)$ whose coordinates take the values 0 or 1.

Denote $S = G \cap B^n$. For any $\mu \geq 0$ put $Z(\mu) = \sum_{x \in S} \exp(-\mu f(x))$. Let us define a random vector $\xi(\mu, \omega)$:

$$\pi(x, \mu) = P\{\xi(\mu, \omega) = x\} = \exp(-\mu f(x)) / Z(\mu), \quad x \in S. \quad (3)$$

Distribution (3) is known in statistical physics as the Boltzmann distribution. The simulated annealing method is also related to it: the stationary probability that the Markovian chain generated by this method is at a point x is defined by (3).

Let $S_j^1 = \{x | x \in S, x_j = 1\}$ and $S_j^0 = \{x | x \in S, x_j = 0\}$ be the sets of feasible solutions of problem (2) at which the j th, $j=1, \dots, n$, coordinate is equal to 1 and 0, respectively.

Let us define the following quantities for $j=1, \dots, n$:

$$Z_j^1(\mu) = \sum_{x \in S_j^1} \exp(-\mu f(x)), \quad Z_j^0(\mu) = \sum_{x \in S_j^0} \exp(-\mu f(x)),$$

$$p_j(\mu) = P\{\xi_j(\mu, \omega) = 1\} = \frac{Z_j^1(\mu)}{Z(\mu)}.$$

Assume that the set $\tilde{S}$ is a subset of feasible solutions of problem (2) derived by the GES and

$$\tilde{S}_j^1 = \{x | x \in \tilde{S}, x_j = 1\}, \quad \tilde{S}_j^0 = \{x | x \in \tilde{S}, x_j = 0\}, \quad j=1, \dots, n.$$

Let us present the general scheme of the GES.

```

procedure GES
1  initialize_algorithm_parameters(maxiter, ngen, maxnfail, mu, K)
2  while (stoppage_criterion = FALSE) do
3    if ( $\tilde{S} = \emptyset$ ) then  $\{\tilde{S}$  — set of already known solutions $\}$ 
4       $x \leftarrow \text{construct\_solution}$  {construct random feasible solution  $x$ }
5       $g(X) \leftarrow \text{calculate\_gains}(x)$ 
6       $x_{\max} = x$ ,  $x_{\text{best}} = x$ ,  $g_{\max} = g(x)$ ;
7       $\tilde{S} \leftarrow x_{\max}$ 
8      Elite  $\leftarrow x_{\max}$  {Elite — set of elite solutions}
9    end if
10   nfail  $\leftarrow$  0 {number of full cycles without improvement}
11   while (nfail < maxnfail) do
12      $k \leftarrow$  0 { $k$  — number of current temperature}
13     old_ $x_{\max} \leftarrow x_{\max}$ 
14     while ( $k < K$ ) do
15       calculate_generation_probabilities( $pr^k$ ,  $\tilde{S}$ )
16        $t \leftarrow$  0
17       while ( $t < ngen$ ) do
18          $x \leftarrow \text{generate\_solution}(x_{\max}, pr^k)$ 
19          $R \leftarrow \text{search\_method}(x)$  { $R$  — set of found feasible solutions}
20          $R \leftarrow R \setminus P$  { $P$  — set of forbidden solutions}
21          $\tilde{S} \leftarrow \tilde{S} \cup R$ 
22          $x_{\max} \leftarrow \arg \min_{x \in \tilde{S}} f(X)$ 
23          $g(x_{\max}) \leftarrow \text{calculate\_gains}(x_{\max})$ 
24         if ( $f(x_{\max}) < f(x_{\text{best}})$ ) then
25            $x_{\text{best}} \leftarrow x_{\max}$ 
26         end if
27         reset_elite(Elite,  $R$ )
28          $t \leftarrow t + 1$ 
29       end while
30        $k \leftarrow k + 1$ 
31     end while
32     if ( $f(\text{old\_}x_{\max}) = f(x_{\max})$ ) then nfail  $\leftarrow$  nfail + 1
33     else nfail  $\leftarrow$  0
34      $\tilde{S} \leftarrow$  Elite
35   end while
36    $P = P \cup N(x_{\text{best}}, d_p)$ 
37   Elite = Elite  $\setminus P$ 
38   if (RESTART criterion = TRUE) then Elite  $\leftarrow$   $\emptyset$ 
39    $\tilde{S} \leftarrow$  Elite
40    $x_{\max} = \arg \min_{x \in \tilde{S}} f(x)$ 
41    $g(x_{\max}) \leftarrow \text{calculate\_gains}(x_{\max})$ 
42 end while
end GES

```

Creating GES-based algorithms involves rules that provide the following:

- (i) using information obtained in solving the problem;
- (ii) possibility of intensification and diversification of the optimum search;
- (iii) using the principles of the RESTART technology [24];
- (iv) union of different metaheuristics, creation of a set of algorithms, which allows the best use of the advantages of the algorithms appearing in it and the specificity of the problem.

Let us consider in more detail constructing GES algorithms based on these rules. The outer cycle of iterations (rows 2–42), provides recurrence of the search for the optimal solution. The operators in rows 37–39 allow using the RESTART technology. This cycle runs either a prescribed number of times or until some stopping criterion is met. Such a criterion may be, for example, a value of the objective function that is better than some known record. The temperature cycle (rows 14–31) is a key point in the structure of the GES algorithm. It requires the values of K and temperature μ_k , $k=0, \dots, K$. A series of starts runs in this cycle to find the optimal solution for increasing values of temperature. The cycle and its repetitions allow us to alternate flexibly the modes of diversification and intensification of the search area, which makes the GES method highly efficient.

If the set $\tilde{S}$ of solutions known at the current moment is empty, then the operators in rows 4–9 allow finding the first solution and initializing necessary data. Note that the solutions found by the GES algorithm are not stored (except for the elite solutions appearing in the Elite set) but used to calculate the following quantities:

$$\tilde{Z}_k = \sum_{s \in \tilde{S}} \exp \{-\mu_k f(x)\}, \quad \tilde{F}_k = \sum_{s \in \tilde{S}} f(x) \exp \{-\mu_k f(x)\}, \quad \tilde{E}_k = \frac{\tilde{F}_k}{\tilde{Z}_k}, \quad (4)$$

$$\tilde{Z}_{kj}^l = \sum_{s \in \tilde{S}_j^1} \exp \{-\mu_k f(x)\}, \quad \tilde{F}_{kj}^l = \sum_{s \in \tilde{S}_j^1} f(x) \exp \{-\mu_k f(x)\}, \quad \tilde{E}_{kj}^l = \frac{\tilde{F}_{kj}^l}{\tilde{Z}_{kj}^l},$$

$$k=0, \dots, K, \quad l=0, 1, \quad j=1, \dots, n.$$

They allow finding the probabilities $\tilde{p}^k = (\tilde{p}_1^k, \dots, \tilde{p}_n^k)$ according to one of the following formulas (see, for example, [32]):

$$\tilde{p}_j^k = \tilde{p}_0^k \exp \left\{ \frac{1}{2} \sum_{i=0}^{k-1} \left(\tilde{E}_i^0 + \tilde{E}_{i+1}^0 - \tilde{E}_{ij}^1 - \tilde{E}_{i+1j}^1 \right) (\mu_{i+1} - \mu_i) \right\}, \quad (5)$$

$$\tilde{p}_j^k = \frac{1}{1 + \frac{1 - \tilde{p}_0^k}{\tilde{p}_0^k} \exp \left\{ -0.5 \sum_{i=0}^{k-1} \left(\tilde{E}_{ij}^0 + \tilde{E}_{i+1j}^0 - \tilde{E}_{ij}^1 - \tilde{E}_{i+1j}^1 \right) (\mu_{i+1} - \mu_i) \right\}}, \quad (6)$$

$$\tilde{p}_j^k = \tilde{Z}_{kj}^1 / \tilde{Z}_K, \quad j=1, \dots, n. \quad (7)$$

As we see, the probabilities $\tilde{p}^k$ of generating the initial solutions (row 18) for the *search_method*(x) searching procedure (row 19) are calculated based on the concepts borrowed from the simulated annealing method and depend on the current temperature μ_k and the set $\tilde{S}$ of feasible solutions found earlier.

The values of μ_k , $k=0, \dots, K$, determine the annealing curve and are usually calculated by the formulas $\mu_0=0$, $\mu_{k+1}=\alpha\mu_k$, $k=1, \dots, K-1$. The values of μ_1 and $\alpha>1$ are selected so that the probability $\tilde{p}^K$ would be approximately equal to $x_{\max}$: $\|x_{\max} - \tilde{p}^K\| \cong 0$. Note that the annealing curve is universal and is not adjusted to an isolated problem but is applied in solving all problems. Only the coefficients of the objective function are scaled so that the objective function of the best known solution would be equal to a prescribed value. The initial probability $\tilde{p}^0$ can be specified in different ways: all coordinates of this vector are identical (for example, 0.5); it can be a solution of problem (2) with relaxation of Booleanness conditions or can be obtained by using statistical simulation of the Markovian chain that corresponds to the simulated annealing method for problem (2) at infinite temperature (all feasible solutions are equiprobable).

The improving cycle (rows 11–34) runs until the solution $x_{\max}$ is improved *maxnfail* times. The *calculate_generation_probabilities* procedure (row 15) is used to calculate the probabilities of a random perturbation of the solution $x_{\max}$ by one of the formulas (5)–(7). The cycle of finding new solutions (rows 17–29) is repeated a prescribed number *ngen* of times. Let us present the *generate_solution* procedure.

```

procedure generate_solution ( $x_{\max}, pr^k$ )
1    $x \leftarrow x_{\max}$ ;  $dist = 0$ ;  $j \leftarrow 1$ 
2   while ( $j \leq n$ )
3       if  $x_j = 1$  then
4           if ( $\tilde{p}_j^k < \text{random}[0, 1]$ ) then
5                $x_j \leftarrow 0$ ;  $dist \leftarrow dist + 1$ 
6                $g(X) \leftarrow \text{recalculate\_gains}(x)$ 
7           end if
8       end if
9       else
10          if ( $\tilde{p}_j^k \geq \text{random}[0, 1]$ ) then
11               $x_j \leftarrow 1$ ;  $dist \leftarrow dist + 1$ 
12               $g(x) \leftarrow \text{recalculate\_gains}(x)$ 
13          end if
14      end else
15       $j \leftarrow j + 1$ 
16      if ( $dist = \text{max\_dist}$ ) then break
17  end while

```

This procedure randomly generates the solution x , which is initial for the *search_method* procedure (row 19). For small temperatures μ_k , perturbation of the solution $x_{\max}$ is purely random and equiprobable; for high temperatures, the values of coordinates, identical for the best found solutions, vary a little. The temperature cycle allows diversifying the search (for low temperatures) and intensifying it (for high temperatures). Thus, as the temperature increases, the solutions generated acquire the attributes intrinsic to the best found solutions and finally converge to the solution $x_{\max}$.

In row 1, initial values are assigned to the vector x and variables $dist$ and j . The vector $x_{\max}$ is perturbed in the cycle of rows 2–17. A random quantity $\text{random}[0, 1]$ uniformly distributed over the interval $[0, 1]$ (rows 4 and 10) is used to simulate random events resulting in a change of the solution $x_{\max}$. Note that the decrease vector $g(x)$ is recalculated after any perturbation of $x_{\max}$. The former is an n -dimensional vector defined on a unit neighborhood of the point x :

$$g_j(x) = f(x_1, \dots, 1 - x_j, \dots, x_n) - f(x_1, \dots, x_j, \dots, x_n).$$

Recalculation of the vector $g(x)$ corresponds to the decrease-vector method and allows saving much computational resources for many problems. For example, for an unconstrained quadratic programming problem with Boolean variables, calculation of the objective function requires $O(n^2)$ arithmetic operations, and recalculation of the vector $g(x)$ requires only $O(n)$ such operations.

The perturbation cycle (rows 2–17) ends if the Hamming distance between the $x_{\max}$ and the perturbed solution x becomes equal to the prescribed parameter max_dist (row 16). The parameter max_dist can be specified dynamically and be dependent on the temperature index k . Usually, max_dist is selected on the interval $\left[\frac{n}{10}, \frac{n}{2}\right]$.

Randomized algorithms of local search, decrease vector, tabu search, and simulated annealing [26] were used as an algorithm in the *search_method* procedure in solving various problems. When selecting an algorithm for the procedure, one should take into account characteristics of the problem being solved. Such an algorithm is selected so as to intensify the search and provide efficient analysis of neighborhoods of the initial solution in order to improve it. The choice of algorithm can be dynamic: some algorithms are used for low temperatures, and other for elevated ones.

The *search_method* procedure can find a set of local (in the sense of a neighborhood of certain radius) solutions that form the set R . Rows 20–21 formally point out that the set R has no solutions prohibited at the moment, and that quantities (4) are recalculated. In a real algorithm, search of prohibited solutions is blocked, and recalculation is carried out as new solutions $x \in R$ are found.

In row 22, the solution $x_{\max}$ is found; in row 23, the decrease vector at this point is determined. The *reset_elite* procedure specifies the set *Elite* of elite solutions, which consists of solutions with the best values of the objective function:

$$\min_{x \in \text{Elite}} f(x) \geq \max_{x \in \tilde{S} \setminus \text{Elite}} f(x). \text{ The maximum potency of the set } \text{Elite} \text{ is specified by the parameter } \text{max_elite_size} \text{ and is}$$

selected usually from the interval $[1, n]$.

Upon completion of the improving cycle (rows 11–35), the set P of prohibited points is supplemented with points from some neighborhood with the center at $x_{\max}$, specified by the parameter d_p (row 36), and prohibited points are deleted from the set *Elite* (row 37).

If the analysis of the optimization process reveals that there is the RESTART distribution of the time of solution, then the set *Elite* is made empty, which makes it possible to begin a new and independent restart of the solution.

TABLE 1

m	α	Average values of the objective function				
		AG	Fix + Cuts	CPLEX	TS ^{MPK}	GES
5	0.25	120616	120610	120619	120623	120625.1
	0.5	219503	219504	219506	219507	219509.7
	0.75	302355	302361	302358	302360	302361.8
10	0.25	118566	118584	118597	118600	118614.9
	0.5	217275	217297	217290	217298	217315.2
	0.75	302556	302562	302573	302575	302594.4
30	0.25	115470	115520	115497	115547	115566.5
	0.5	216187	216180	216151	216211	216230.9
	0.75	302353	302373	302366	302404	302427.1

Battiti and Protasi [46] reported the results of experimental calculations as applied to a multidimensional knapsack problem with Boolean variables

$$\max \left\{ cx \mid Ax \leq b, x \in B^n \right\}, \quad (8)$$

where $c, x \in R^n$, $b \in R^m$, and $A \in R^{m \times n}$, coordinates of the vectors c and b and elements of the matrix A being nonnegative. Computational efficiencies of the reactive tabu search [46, 47], genetic algorithm, simulated annealing [85], optimization on neural networks [79, 94], and iterated random local search [17, 48] are compared for test problems (8). To compare the GES method with these methods, a computational experiment has been carried out, with the use of benchmark problems (8) from [47], which can be divided into four types. Problems of the fourth type are most difficult to solve exactly and are distinguished from the problems of the first three types by the representation of coordinates of the vector c . The benchmark problems differ also in dimension: “small” problems (with the number of variables $n = m = 30$) and “large” problems ($n = m = 500$). A hundred of small problems of each of the four types were solved. Among them, the number of problems exactly solved by the GES method constitutes 100%, those solved by other methods are from 17 to 68%. For the first and fourth types, eight large problems were additionally solved.

The greatest difficulty in solving the fourth type of problems by the compared algorithms is due to the strong correlation between the coefficients of the objective function and the corresponding vectors of the constraint matrix. Such a correlation is also usually present in real knapsack problems, which makes the successful solution of such problems by the GES method especially important.

An analysis of the considered methods (specificity of problem (8) was not used therein) and calculation results have shown that the GES method has significant advantages over other methods for all the problems. Other methods show high performance for one problems and very low for others.

Recently, we performed comparative analysis of the algorithms AG [56], Fix + Cuts [97], CPLEX [97], TS^{MPK} [113], GES [26, 32] for solving problems (8). Three groups (30 in each group) benchmark problems from [56] were solved for $n = 500$ and $m = 5, 10, 30$. Each set of the 30 problems consists of three subsets corresponding to $\alpha = 0.25, 0.5, 0.75$ ($\alpha = b_i / \sum_{j=1}^n a_{ij}$, $i = 1, \dots, m$). Table 1 presents average (for 10 problems) best values of the objective function for each of

the considered algorithms. As is seen, the GES algorithm has shown the best results on all the subsets of problems.

A problem on the maximum slit of a digraph frequently arises in practical applications. It can be formulated as follows. Let a digraph $G = G(V, E)$ be given, where V and E are the sets of its nodes and ribs, respectively. It is required to find a subset $S \subset V$ of nodes such that the potency of the slit $C(S) = \{(i, j) \in E \mid i \in S, j \notin S\}$ determined by it would be maximum.

Each oriented rib $e_i \in E$ can be presented as (k_i, l_i) , $i = 1, \dots, |E|$, $|E|$ is the potency of the set E , where k_i and l_i are the initial and final nodes of the rib e_i , respectively. Let us associate each node $v_j \in V$ with a variable $x_j \in \{0, 1\}$, $j = 1, \dots, |V|$, and an arbitrary vector $x = (x_1, \dots, x_{|V|})$ with the set $S(x) = \{v_j \in V \mid x_j = 1\}$. Then we can write the mathematical model of the problem as follows:

$$\max_{x \in B^{|V|}} \left\{ f(x) = \sum_{i=1}^{|E|} x_{k_i} (1 - x_{l_i}) \right\}.$$

To analyze the efficiency of the GES method for solving the problem on the maximum slit of a digraph and compare it with the tabu search (in the tabu_RP modification [47]) and with four modifications of iterated random local search, a computational experiment [26] has been carried out. The problems generated by the technique proposed in [47] were used as the benchmark problems. The experiment consists in the following. Let n be the number of nodes of the graph. For all possible pairs (i, j) , $i, j = 1, \dots, n$, $i \neq j$, a directed arc (i, j) was generated with the probability *density*, and 50 problems were generated for each value of n and each value of *density* (*density* = 0.1; 0.3; 0.5; 0.7; 0.9). All the obtained problems were solved 10 times for different seed states of the randomizer, which changed the course of the problem solution by an isolated algorithm. Thus, 500 test calculations were run for each pair $(n, \text{density})$.

To intensify the search in the GES program implementation presented in computational experiments, determination of a local optimum was followed by several steps of the simulated annealing method, which made it possible to analyze points neighboring the local solution. All the algorithms used in the test runs employed *fevals* equal to the number of calculations of the objective function. It seems more representative to compare the algorithms with respect to this quantity rather than to the number of iterations. The reason is that the objective function is calculated different number of times to go from point to point (iteration) in algorithms. For example, in the modification of the tabu_RP method [47] the objective function is calculated at all points of the neighborhood of the current point, i.e., it is calculated n times at one iteration. In the simulated annealing method, for elevated temperatures, the objective function is calculated only at one point of iteration. Note that the number of iterations hardly reflects the necessary computational expenditures, whereas the number of calculations of the objective function in local-type methods is directly related to the radius of the neighborhood. Thus, it seems more informative to compare the algorithms in the number of calculations of the objective function rather than in the number of iterations; moreover, the former does not depend on the type of the computer. Moreover, the statement “to solve a problem, some method needed to find the objective function at *fevals* points” contains several information levels.

An analysis of the results of the computational experiments indicates about the advantages of the GES over other methods used on the described problems. The GES method exceeds them both in the speed of deriving feasible solutions and in the capability of obtaining very good, almost optimal solutions.

The GES method can also be used to solve a maximum feasibility problem [26], which has numerous applications. The problem can be formulated as follows. Let m disjuncts be given. It is necessary to specify logical variables x_i , $i = 1, \dots, n$, so that the number of true disjuncts composed using these variables would be maximum.

The mathematical model of this problem can be described as follows:

$$\max \left\{ \sum_{i=1}^m z_i \mid \begin{array}{l} \sum_{j \in U_i^+} x_j + \sum_{j \in U_i^-} (1 - x_j) \geq z_i, i = 1, \dots, m, \\ 0 \leq x_j \leq 1, 0 \leq z_i \leq 1, x_j, z_i \text{ are integer, } j = 1, \dots, n, i = 1, \dots, m \end{array} \right\},$$

where U_i^+ is the set of variables appearing in the i th disjunct without negation, and U_i^- are all the remaining variables.

For comparative analysis of the GES method with the known algorithms for the solution of this problem, we used *walksat* programs and a generator of test examples kindly submitted by Dr. Selman. Moreover, we have coded the tabu algorithm following [46]. The results obtained indicate that the GES method is competitive with the considered methods of solution of the problem.

Authors have also successfully solved the weighted maximum feasibility problem using an algorithm based on the GES method. For example, the problem jnh305 was solved 200 times by the GRASP and GRASP + PR algorithms on an SGI Altix 3700 Supercluster computer (with a single processor) and 5000 times by the GES algorithm on a PC Pentium 4, 3.00 GHz. The problem was considered solved when the solution was obtained with the objective function exceeding a prescribed value of $\tilde{f} = 443815$. The relative error of such a solution is no greater than 0.067%. The calculation results are presented in Figs. 1 and 2. The ordinate of an arbitrary point of these curves is a so-called frequency h (the number of problems divided by the general number of solutions) of the problems solved by the time t , and the abscissa is the time t of the solution. The results obtained confirm the high performance of the GES method in solving problems of this class.

Shylo [38] applied the GES method [32] to solve the job-shop scheduling problem, which can be formulated as follows. Consider a set M of machines and a set J of jobs. For each job, an ordered set consisting of $|M|$ operations is given as follows: $\sigma_1^j \prec \sigma_2^j \prec \dots \prec \sigma_{|M|}^j$, where σ_k^j is the k th operation of the j th job, $\sigma_k^j \prec \sigma_{k+1}^j$ means that the operation σ_{k+1}^j may be started only after completion of the operation σ_k^j . Denote by O the set of all operations of jobs from J . Each

Fig. 1. Results of solving problem jnh305 by GRASP and GRASP+PR algorithms.

Fig. 2. Results of solving problem jnh305 by GRP algorithm.

operation σ_k^j is specified by two properties: M_k^j is the number of the machine that performs this operation and $\tau(\sigma_k^j)$ is the time it takes to perform the operation. Denote the time to start performing the operation $\sigma_k^j \in O$ by $t(\sigma_k^j)$. It is necessary to minimize the time $C_{\max}$ it takes to fulfill all jobs in view of sequence and resource requirements. The job-shop scheduling problem is formulated as follows:

minimize

$$C_{\max} \quad (9)$$

provided that

$$C_{\max} \geq t(\sigma_k^j) + \tau(\sigma_k^j) \quad \forall \sigma_k^j \in O, \quad (10)$$

$$t(\sigma_k^j) \geq t(\sigma_m^j) + \tau(\sigma_m^j) \quad \forall \sigma_m^j \prec \sigma_k^j, \quad (11)$$

$$t(\sigma_k^j) \geq t(\sigma_m^j) + \tau(\sigma_m^j) \vee t(\sigma_m^j) \geq t(\sigma_k^j) + \tau(\sigma_k^j) \\ \forall \sigma_k^j, \sigma_m^j \in O \text{ such that } M_k^j = M_m^j, \quad (12)$$

$$t(\sigma_k^j) \geq 0 \quad \forall \sigma_k^j \in O. \quad (13)$$

This problem is *NP*-hard. The problem of dimension 15×15 (15 machines and 15 jobs) has not been solved yet by exact methods; therefore, good approximate algorithms are necessary. Such problems were successfully solved by the methods employing local search: tabu search, simulated annealing, GRASP procedure, etc. [38, 50, 73, 93, 104, 112].

For the considered problem, Shylo proposed in [38] a local-type algorithm based on the GES method. The general scheme of the GES method includes two basic stages: generation of a solution and local search near this solution. To carry out local search for the solution of problem (9)–(13), three types of neighborhoods proposed in [73, 93] are used.

To test the algorithm, we took problems from <http://mscmga.ms.ic.ac.uk/info.html>. The following classes of problems known from publications were solved:

- (i) 5 ORB1-ORB5 problems;
- (ii) 40 LA1-LA40 problems;
- (iii) 80 TA1-TA80 problems; and
- (iv) 80 DMU1-DMU80 problems.

The algorithm proposed for solving the job-shop scheduling problem based on the GES method has shown excellent results for all the classes of the benchmark problems considered. The results obtained were compared to the data from [45, 73, 93, 104] and have confirmed the advantage of the GES algorithm proposed.

In all the problems of the first two classes, optimal solutions were found. TA problems are most difficult. The GES algorithm have found exact solutions to TA51-TA80 problems. Despite the variety of computational experiments and methods used to solve problems of this class, it became possible to improve the records for four problems (TA15, TA19, TA20, and TA32) due to the GES algorithm. In solving problems of the fourth class, upper-bound estimates for 39 problems have been improved.

Note that the algorithm parameters were constant for all the problems, i.e., the algorithms were not adapted.

The GES method was also used to solve the p -median problem [39] formulated as follows. Let sets F and U be given, which consist of m resources and n users, respectively, and the distance function $d: U \times F \rightarrow R^1$ and a constant $p \leq m$ be defined. It is necessary to find a subset $S \subset F$ satisfying the conditions $S \subset F$, $|S| = p$, $\sum_{u \in U} \min_{f \in S} d(u, f) \leq$

$\sum_{u \in U} \min_{f \in Q} d(u, f)$ for all the subsets $Q \subset F$ such that $|Q| = p$. In other words, it is necessary to specify (to open) p resources of F to minimize the sum of distances from each user from U to the nearest open resource.

Since this problem is *NP*-hard [64], its solution involves different approximate methods. The most popular and efficient approaches are the methods involving local search: tabu [108], variable neighborhood search [76], and the GRASP procedure [107].

Shylo proposed in [39] an approximate algorithm based on the GES method [32] to solve the p -median problem. Using the solutions already obtained, the GES method generates random solutions with probabilities adapted during the search. The temperature cycle underlies the method: absolutely random solutions are generated for the initial temperatures, and the solutions close to the best ones found during the whole cycle, at the end of the temperature cycle.

To test the proposed algorithm, a problem from the library located at <http://www.iwr.uni-eidelberg.de/groups/comopt/software/TSPLIB95/> was used. We considered the problem *f11400* usually used in studies on a p -median. It is obtained from the coordinates of 1400 points on a plane, which were initially used to analyze the traveling salesman problem. In the case of the p -median problem, each point is simultaneously a resource and a user, Euclidean distances being used. The problems were solved for p from 10 to 500. For comparison, we implemented three versions of the algorithm: with absolutely random generation of solutions, generation of solutions using the GRASP algorithm [107], and using the GES algorithm. A series of 5000 generations was performed for each version.

It is obvious from the results obtained [39] that the algorithm proposed based on the GES method has made it possible to find high-quality solutions for all the benchmark problems. In some cases, the solutions found using variable neighborhood search [76] appeared worse than those obtained using ordinary local search. For all the problems, except for the problem with $p = 100$, the GES algorithm has produced the results exceeding those obtained for absolutely random generation and for generation using the GRASP procedure.

An analysis of the results of the numerical experiment shows the high performance of local search for the p -median problem. A version of the algorithm even with absolutely random generation of solutions allows deriving qualitative solutions. The version of the GRASP procedure studied enables improving the quality of the solutions being generated; however, for some problems, the best found solutions appeared worse than in the case of absolutely random generation. On the contrary, the GES algorithm has shown stable behavior for all the benchmark problems.

Fig. 3. Results of solving the problem pal6000_2.

The authors successfully used the GES method to solve also the quadratic optimization problem

$$\max \left\{ f(x) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} x_i x_j \mid x \in B^n \right\}, \quad (14)$$

where $A = [a_{ij}]$ is a real symmetric matrix of the order n . Since the conditions $x_i \in \{0, 1\}$ are equivalent to the quadratic equalities $x_i^2 - x_i = 0$, $i = 1, \dots, n$, it is possible to reduce, to the form (14), the unconstrained quadratic programming

problem with Boolean variables and with the objective function $\sum_{j=1}^n c_j x_j + \sum_{i=1}^n \sum_{j=1}^n q_{ij} x_i x_j$ assuming that

$a_{jj} = c_{jj} + q_{jj}$, $j = 1, \dots, n$, $a_{ij} = (q_{ij} + q_{ji}) / 2$, $i, j = 1, \dots, n$, $i \neq j$. To (14), we can also reduce the quadratic optimization problem with Boolean variables $\max \{xAx \mid Dx = b, x \in B^n\}$, where A is a symmetric matrix of the order n , D is an $m \times n$ matrix, and b is a fixed vector in R^m .

Quadratic optimization problems with Boolean variables (14) have a wide range of applications. Among them noteworthy are problems in economy, scheduling theory, solid-state physics, molecular structure, allocation of frequencies of cellular communication, etc. Moreover, important extremum graph problems can be formulated in the form (14): on the maximum slit of a graph, finding the maximum independent set, maximum clique, coloring and covering of a graph [102, 103]. Many methods are proposed to solve such problems (see, for example, [42, 49, 83, 88, 90]).

We have prepared a paper with a detailed presentation of results concerning the application of the GES method to problems (14). Note that this method was used to solve high-dimensional ($n = 5000, 6000$) benchmark problems of the form (14) from [101]. We are grateful to G. Palubeckis who has submitted benchmark problems and corresponding programs. We compared the results obtained on PC Pentium 4, 3.00 Ghz when solving the considered problems in [101] and by the GES method. Figure 3 shows the results of solving the problem pal6000_2 by the MST2 [101] and the GES algorithms. The problem was solved 10 times by each algorithm. It is considered solved when the objective function exceeds a specified value of $\tilde{f} = 14,328,117$. The curves in Fig. 3 have the same structure as curves in Figs. 1 and 2. An analysis of the results obtained shows that, in most cases, the GES method produced better solutions in a shorter time.

As follows from the results, the GES method performed excellently in solving different complicated high-dimensional discrete programming problems and successfully competed with the metaheuristic algorithms, which are the best today.

Note that the main advantage of the GES method is not even its computational efficiency but its synergetic qualities — ability to efficiently use the solutions produced by other algorithms. For example, a synergetic index in the classical method of local search is zero. Synergetic properties allow the GES method to be very efficiently applied in parallel calculations, which is especially valuable for optimization in computer networks, in the Internet. In our opinion, any

optimization problem that meets definite requirements to the structure of optima (for example, to the SO-principle) can be efficiently solved within the framework of the approach we have proposed.

An important research area is the stability and parametric analysis of discrete optimization problems [19, 26]. It is well known that initial data, as a link between reality and a model, cannot be defined uniquely in most cases but are specified with an error, usually depend on many parameters, and need to be improved during solution. Practice suggests that any problem cannot be correctly formulated and solved without using (at least implicitly) the results of stability theory and adjacent problems of parametric analysis [5, 6, 21, 40, 110].

The theoretical research trend in stability, parametric, and postoptimal analysis of optimization problems is based on the use of the properties of point-set mappings. There exists also a constructive approach to the stability of optimization problems, and the radius of stability is one of its key concepts. The studies performed at the Institute of Cybernetics and concerned stability, parametric, and postoptimal analysis of discrete optimization problems [19, 26] fall within both trends. They are oriented toward further development of the theory of solving problems with fuzzy nondeterministic data and sets of problems.

Nowadays, due to development of various information technologies, the Internet, modern communication facilities, interest has increased to reliable codes, which allow correcting data transfer errors. Construction of a maximum-size error-correcting code whose mathematical model is formulated in terms of discrete programming holds an important place among these problems. The authors studied reducing this problem to the problem of finding the maximum independent set of nodes of a graph, its solution, and estimating the size of error-correcting codes for various data conversion models [26, 36].

Two other optimization problems on graphs, search for the maximum clique and search for the minimum vertex coating, are closely related to finding the maximum independent set. The algorithm for solution of one of these problems can be used to solve the two others. Practical applications of these three problems are very extensive. They are concerned with economic studies, problems of scheduling theory, artificial intelligence, information retrieval, signal transmission analysis, classification, and many others. These problems are *NP*-hard; therefore, development of efficient polynomial algorithms for their approximate solution is an urgent problem.

The problem of searching for the maximum independent set of nodes of a graph is considered as a basic one. To solve it, a local-type probabilistic algorithm based on the GES method is proposed and analyzed in [26]. According to this algorithm, local search is carried out not in the space B^n but in a set of independent sets, maximum in inclusion, with a neighborhood selected in a definite way. To analyze the proposed algorithm for efficiency, a computational experiment was performed using graphs from the sets originating during code construction, which compensate for one bit removal — ldc, two bits removal — 2dc, one transposition — ltc, one cyclic transposition — let, one error in the Z channel — lzc. An analysis of the results of the computational experiment has revealed high efficiency of the approximate algorithm developed. The best known solutions have been found for all the examples.

To solve a problem of vertex cover with independent sets, a probabilistic algorithm is developed [37], whose efficiency is analyzed for problems of constructing error-correcting codes. New partitions are obtained to be used for constructing and studying error-correcting codes.

As indicated above, Shylo proposed in [36] an exact algorithm, based on the branch-and-cut method, to search for a maximum-size error-correcting code. The algorithm makes it possible to solve exactly, for the first time, the urgent problem of finding a maximum-size error-correcting code for graph lzc1024.

The theoretical studies and operation parallelizing have resulted in an information technology for constructing maximum-size error-correcting codes. The necessity of parallelizing the optimization process is due [30] to (i) high dimension of problems, which results in significant amount of calculation and time of solution; (ii) a possibility of parallelizing the process over independent branches of the exact branch-and-bound algorithm; and (iii) a possibility of information exchange between approximate algorithms run by different processors, which makes it possible to construct radically new optimization schemes. The information technology designed has allowed solving, for the first time, some important problems or substantially reduce the time it takes to solve other problems of constructing maximum-size error-correcting codes on a SKIT cluster computer.

The theoretical results obtained by the authors have made it possible to solve important practical problems of constructing maximum-size error-correcting codes. For the world level of achievements in this area, see, for example, <http://www.research.att.com/~njas/doc/graphs.html> (the AT&T Labs Research and Development Center, New Jersey, the USA).

We have obtained, for the first time, maximum-size error-correcting codes for graphs 1tc512 (volume 110), 1tc1024 (volume 196), 1tc2048 (volume 352), 1et1024 (volume 171), 1et2048 (volume 316), 1zc4096 (volume 379), 1zc8192 (volume 701), and for graphs lzc with the number of nodes 2^n , $n = 18 - 22, 24$. For the first time, we have exactly solved the

problems of finding maximum-size error-correcting codes for graphs 1dc512, 1tc512, 1et512, 1et1024, 1tc1024, and 1zc1024. These graphs, taken from the above Internet site, are the initial data for urgent unsolved problems. We have passed the obtained maximum-size error-correcting codes to the AT&T Labs Research and Development Center.

To accelerate the solution of complex discrete optimization problems, theoretical fundamentals have recently been created. In particular, a RESTART technology [24] is developed, which is based on the new concepts of RESTART distribution and RESTART stopping criterion of the algorithm. The fact that the distribution of some characteristics of optimization algorithm is a RESTART distribution can be efficiently used in organizing an optimization process. To apply the RESTART technology, it is expedient that (i) the optimization algorithm include random steps, which makes efficiency indices random (and, most likely, having a RESTART distribution); (ii) a RESTART test run for the algorithm and the problem, which should clarify the possibility (and expediency) of applying the RESTART stopping criterion. The RESTART technology allows, in many cases, substantially decreasing the average time of problem solving. Procedures of choosing optimization algorithms are developed and analyzed, which are based on the method of probabilistic decomposition and the Wald criteria of successive analysis.

A promising approach was proposed in [25] to parallelize the optimization process, where instead of the operations fulfilled by algorithm, its copies are parallelized. Let a stochastic algorithm be available to solve a discrete optimization problem, and its random behavior be defined by a randomizer. By its copy, we will call a version of such an algorithm obtained for one seed state of the randomizer. Obviously, different copies of the initial algorithm are created for different seed states of the randomizer; they allow searching for different solutions.

According to the approach proposed in [25], copies of the initial algorithm are created and used to solve the discrete optimization problem involving all the processors available. In some cases, such an approach solves the problem faster than operation parallelizing does. Calculations can be accelerated, for example, by using the properties of the RESTART distribution. The results of the computational experiment on the maximum slit of a digraph by the tabu algorithm and the multidimensional knapsack problem with Boolean variables by the GES algorithm with parallelizing copies of these algorithms show that the optimization can be accelerated both with and without information exchange between the copies of the algorithm.

A possible approach to creating optimization methods of new generation is one that would allow us to purposefully collect and use information about the problem being solved and to adapt the algorithm to a specific problem. Due to the development of new information technologies, the Internet, computer networks, we should speak not about an optimization method in conventional sense but about a union or a team of algorithms that run in parallel, accumulate information, and exchange it.

REFERENCES

1. V. I. Artemenko and I. V. Sergienko, "A P -method of solving integer linear programming problems with Boolean variables," Dokl. AN USSR, Ser. A, No. 4, 72–75 (1980).
2. O. V. Volkovich, V. A. Roschin, and I. V. Sergienko, "On models and methods of solving integer quadratic programming problems," Kibernetika, No. 3, 1–15 (1987).
3. V. V. Glushkova and V. P. Shylo, "Problems of optimal arrangement of switching equipment," in: Mathematical Methods of Decision-Making under Uncertainty [in Russian], V. M. Glushkov Inst. Cybernetics, AS USSR, Kiev (1990), pp. 69–71.
4. L. F. Gulyanitskii, I. V. Sergienko, and A. N. Khodzinskii, "Problems of development and analysis of parallel discrete optimization methods," in: Programming Languages and Parallel Computers: Algorithms and Algorithmic Languages [in Russian], Nauka, Moscow (1990), pp. 62–78.
5. V. A. Emelichev and V. A. Perepelitsa, "Complexity of discrete multicriterion problems," Diskret. Matem., Issue 1, 6, 3–33 (1994).
6. V. A. Emelichev and D. P. Podkopaev, "Stability and regularization of vector problems of integer linear programming," Diskret. Analiz Issled. Oper., Ser. 2, 8, No. 1, 47–69 (2001).
7. Yu. I. Zhuravlyov, Selected Scientific Works [in Russian], Magistr, Moscow (1998).
8. N. Z. Shor, I. V. Sergienko, V. P. Shylo, et al., Problems of Optimal Design of Reliable Networks [in Ukrainian], Naukova Dumka, Kiev (2005).
9. M. M. Kovalev and V. M. Kotov, "Suboptimal algorithms in integer programming," Dokl. AN BSSR, 26, 969–972 (1982).

10. N. K. Maksishko and V. A. Perepelitsa, Analysis and Prediction of the Evolution of Economic Systems [in Russian], Poligraph, Zaporozhye (2006).
11. M. Minou, Mathematical Programming: Theory and Algorithms, J. Wiley and Sons, New York (1986).
12. V. S. Mikhalevich and V. L. Volkovich, Computational Methods of the Analysis and Design of Complex Systems [in Russian], Nauka, Moscow (1982).
13. V. S. Mikhalevich and A. I. Kuksa, Methods of Sequential Optimization in Discrete Network Problems of Optimal Resource Allocation [in Russian], Nauka, Moscow (1983).
14. V. S. Mikhalevich, V. A. Trubin, and N. Z. Shor, Optimization Problems of Production and Transportation Scheduling [in Russian], Nauka, Moscow (1986).
15. M. V. Mikhalevich and I. V. Sergienko, Simulation of Transition Economy: Models, Methods, and Information Technologies [in Russian], Naukova Dumka, Kiev (2005).
16. C. H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice Hall (1987).
17. I. V. Sergienko, Mathematical Models and Methods for Solving Discrete Optimization Problems [in Russian], Naukova Dumka, Kiev (1988).
18. I. V. Sergienko and N. F. Kaspshitskaya, Models and Methods of Solving Combinatory Optimization Problems on a Computer [in Russian], Naukova Dumka, Kiev (1981).
19. I. V. Sergienko, L. N. Kozerskaya, and T. T. Lebedeva, Stability and Parametric Analyses of Discrete Optimization Problems [in Russian], Naukova Dumka, Kiev (1995).
20. I. V. Sergienko, T. T. Lebedeva, and V. A. Roshchin, Approximate Methods of Solving Discrete Optimization Problems [in Russian], Naukova Dumka, Kiev (1980).
21. I. V. Sergienko, T. T. Lebedeva, and N. V. Semenova, "Existence of solutions in vector optimization problems," *Cybern. Syst. Analysis*, Vol. 36, No. 6, 823–828 (2000).
22. I. V. Sergienko, I. V. Parasyuk, and N. I. Tukalevskaya, Computer-Aided Data Processing Systems [in Russian], Naukova Dumka, Kiev (1976).
23. I. V. Sergienko, and N. V. Filonenko, "Solution of some stability problems in integer linear programming," *Dokl. AN USSR, Ser. A*, No. 6, 79–82 (1982).
24. I. V. Sergienko, V. P. Shilo, and V. A. Roshchin, "RESTART technology for solving discrete optimization problems," *Cybern. Syst. Analysis*, Vol. 36, No. 5, 659–666 (2000).
25. I. V. Sergienko, V. P. Shilo, and V. A. Roshchin, "Optimization parallelizing for discrete programming problems," *Cybern. Syst. Analysis*, Vol. 40, No. 2, 184–189 (2004).
26. I. V. Sergienko, and V. P. Shylo, Discrete Optimization Problems: Challenges, Solution Techniques, and Analysis [in Russian], Naukova Dumka, Kiev (2003).
27. I. V. Sergienko, "A method of solving extremum problems," *Avtomatika*, No. 5, 15–21 (1964).
28. I. V. Sergienko, Computer Science in Ukraine: Development and Problems [in Ukrainian], Naukova Dumka, Kiev (1999).
29. I. V. Sergienko, Computer Science and Computer Technologies [in Ukrainian], Naukova Dumka, Kiev (2004).
30. I. Sergienko and V. Koval', "SKIT: A Ukrainian supercomputer project," *Visn. NAN Ukrainy*, No. 8, 3–13 (2005).
31. A. Schrijver, Theory of Linear and Integer Programming, John Wiley and Sons (1986).
32. V. P. Shilo, "The method of global equilibrium search," *Cybern. Syst. Analysis*, Vol. 35, No. 1, 68–74 (1999).
33. V. P. Shilo, "Results of the experimental investigation of the efficiency of the global-equilibrium-search method," *Cybern. Syst. Analysis*, Vol. 35, No. 2, 253–261 (1999).
34. V. P. Shylo, "Solution of multidimensional knapsack problems by global equilibrium search," in: *Theory of Optimal Solutions* [in Russian], V. M. Glushkov Inst. Cybern., NAS Ukraine, Kiev (2000), pp. 10–13.
35. V. P. Shilo, "New lower bounds of the size of error-correcting codes for the Z -channel," *Cybern. Syst. Analysis*, Vol. 38, No. 1, 13–16 (2002).
36. V. P. Shylo, "Exact solution of the problem of creating code maximum-size error-correcting," in: *Computer Mathematics* [in Russian], V. M. Glushkov Inst. Cybern., NAS Ukraine, No. 2, 145–152, Kiev (2000).
37. V. P. Shylo and D. A. Boyarchuk, "Algorithm of creating a covering with independent sets," in: *Computer Mathematics* [in Russian], V. M. Glushkov Inst. Cybern., NAS Ukraine, No. 2, 151–157, Kiev (2001).
38. O. V. Shylo, "Algorithm of global equilibrium search to solve the scheduling problem," in: *Computer Mathematics* [in Russian], V. M. Glushkov Inst. Cybern., NAS Ukraine, No. 2, 150–160, Kiev (2004).

39. O. V. Shylo, "Algorithm of global equilibrium search to solve the p -median problem," in: Theory of Optimal Solutions [in Russian], No. 3, 150–158, V. M. Glushkov Inst. Cybern., NAS Ukraine, Kiev (2004).
40. Yu. Yu. Chervak, Optimization. Unimprovable Choice [in Ukrainian], Uzhgorod National University, Uzhgorod (2002).
41. E. H. L. Aarts and J. K. Lenstra, Local Search in Combinatorial Optimization, J. Wiley and Sons, Chichester (1997).
42. M. M. Amini, B. Alidaee, and G. A. Kochenberger, "A scatter search approach to unconstrained binary quadratic programs," in: D. Corne, M. Dorigo, and F. Glover (eds.), New Ideas in Optimization, McGraw-Hill, London (1999), pp. 317–329.
43. D. Applegate, R. Bixby, V. Chvatal, et al., The Traveling Salesman Problem, Techn. Rep. DIMACS, Rutgers Univ., New Brunswick, NJ (1994).
44. E. Balas, S. Ceria, and G. Cornuejols, "Mixed 0-1 programming by lift-and-project in a branch-and-cut framework," *Manag. Sci.*, **42**, 1229–1246 (1996).
45. E. Balas and A. Vazacopoulus, "Guided local search with shifting bottleneck for job-shop scheduling," *Manag. Sci.*, **44**(2), 262–275 (1998).
46. R. Battiti and M. Protasi, "Reactive search, a history sensitive heuristic for MAX-SAT," *ACM J. Exper. Algorithms*, No. 3, 23–39 (1997).
47. R. Battiti and G. Tecchiolli, "Local search with memory: benchmarking RTS," *Prepr., Univ. di Trento* (1994).
48. R. Battiti and G. Tecchiolli, "The reactive tabu search," *ORSA J. Comput.*, **6**, No. 2, 126–140 (1996).
49. J. E. Beasley, "Heuristic algorithms for the unconstrained binary quadratic programming problem," Working Paper, The Management School, Imperial College, London, England (1998).
50. S. Binato, W. J. Hery, D. Loewenstern, et al., "GRASP for job shop scheduling," in: Essays and Surveys on Metaheuristics, Kluwer, Acad. Publ., Boston (2001), pp. 59–79.
51. E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm Intelligence: from Natural to Artificial Systems, Oxford Univ. Press, New York (1999).
52. E. Bonomi and J. L. Lutton, "The n -city traveling salesman problem: statistical mechanics and the Metropolis algorithm," *SIAM Rev.*, **26**, No. 4, 551–568 (1984).
53. A. Caprara and A. Fischetti, "Branch and cut algorithms," in: M. Dell'Amico, F. Maffioli, S. Martellochapter, et al. (eds.), *Annot. Bibliogr. in Combin. Optim.* (1997).
54. G. Caprossi and P. Hansen, "Variable neighborhood search for extremal graphs," *The AutoGraphix System, Discrete Math.*, **212**, 29–44 (2000).
55. V. Cerny, "Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm," *J. Optim. Theory Appl.*, **45**, 41–51 (1985).
56. P. C. Chu and J. E. Beasley, "A genetic algorithm for the multidimensional knapsack problem," *J. Heuristic*, **4**, 63–86 (1998).
57. C. Cordier, H. Marchand, R. Laundy, et al., "Bc-opt: a branch-and-cut code for mixed integer programs," *Math. Program.*, **86**(2), 335–353 (1999).
58. S. Coy, B. Golden, and E. Wasil, "A computational study of smoothing heuristics for the traveling salesman problem," *Eur. J. Oper. Res.*, **124** (1), 15–27 (2000).
59. G. A. Croes, "A method for solving traveling salesman problems," *Oper. Res.*, **6**, 791–812 (1958).
60. M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evolut. Comput.*, **1** (1), 53–66 (<http://iridia.ulb.ac.be/mdorigo/ACO/ACO.html>) (1997).
61. T. A. Feo and M. G. C. Resende, "Greedy randomised adaptive search procedures," *J. Global Optimiz.*, **6**, 109–133 (1989).
62. A. G. Ferreira and J. Zerovnik, "Bounding the probability of success of stochastic methods for global optimization," *Comput. Math. Appl.*, **25**, 1–8 (1993).
63. L. Gambardella, E. Taillard, and M. Dorigo, "Ant colonies for the quadratic assignment problem," *Oper. Res. Soc.*, **50**, 167–176 (www.idsia.ch/~luca) (1999).
64. M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, New York (1979).
65. M. Gendreau, A. Hertz, and G. Laporte, "New insertion and postoptimization procedures for the traveling salesman problem," *Oper. Res.*, **40**, 1086–1094 (1992).

66. N. L. J. Ulder, E. H. L. Aarts, H. J. Bandelt et al., "Genetic local search algorithms for the traveling salesman problem," Proc. 1st Intern. Workshop on Parallel Problem Solving from Nature (1990), pp. 109–116.
67. E. Girlich and M. Kowaljow, Nichtlineare discrete optimizierung, Acad.-Verlag, Berlin (1981).
68. F. Glover, "A template for scatter search and path relinking," in: J. K. Hao, E. Lutton, E. Ronald, et al. (eds.), Artificial Evolution, Lect. Notes Comp. Sci., **1363**, Springer-Verlag, Berlin (1998), pp. 13–54.
69. F. Glover, "Tabu search. I," ORSA J. Comput., No. 1, 190–206 (1989).
70. F. Glover, "Tabu search. II," ORSA J. Comput., No. 2, 4–32 (1989).
71. F. Glover, A. Lokketangen, and D. Woodruff, "Scatter search to generate diverse MIP solutions," in: M. Laguna and J. L. Gonzalez-Velarde (eds.), Computing Tools for Modeling, Optim. and Simul.: Interfaces in Comp. Sci. and Oper. Res., Kluwer Acad. Publ., Boston (2000), pp. 299–317.
72. F. Glover and M. Laguna, Tabu Search, Kluwer Acad. Publ., Boston (1997).
73. J. Grabowski, E. Nowicki, and S. Zdralka, "A block approach for single machine scheduling with release dates and due dates," Eur. J. Oper. Res., **26**(2), 278–285 (1986).
74. M. Grotschel and O. Holland, "Solution of large-scale traveling salesman problems," Math. Program., **51** (2), 141–202 (1991).
75. J. Gu and X. Huang, "Efficient local search with search space smoothing: a case study of the traveling salesman problem," IEEE Trans. Syst., Man., Cybernetics, **24** (5), 728–735 (1994).
76. P. Hansen and N. Mladenovic, "Variable neighborhood search for the p -median," Location Sci., **5** (4), 207–226 (1997).
77. P. Hansen, N. Mladenovic, and D. Perez-Brito, "Variable neighborhood decomposition search," J. Heuristics, **7**(3), 335–350 (2001).
78. J. Holland, Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor (1975).
79. J. J. Hopfield and D. W. Tank, " "Neural" computation of decisions in optimization problems," Biol. Cybernetics, **52**, 141–152 (1985).
80. M. Junger, G. Reinelt, and S. Thienel, "Practical problem solving with cutting plane algorithms in combinatorial optimization," Combinatorial Optimization: DIMACS Ser. in Discrete Math. and Theor. Comput. Sci., **20**, 111–152, AMS (1995).
81. O. Kariv and S. L. Hakimi, "An algorithmic approach to network location problems; Part 2. The p -medians," SIAM J. Appl. Math., **37**, 539–560 (1969).
82. R. M. Karp, "The probabilistic analysis of some combinatorial search algorithms," in: J. F. Trumb (ed.), Algorithms and Complexity, Acad. Press, New York (1976), pp. 1–16.
83. K. Katayama and H. Narihisha, "Performance of simulated annealing-based heuristic for unconstrained binary quadratic programming problem," Eur. J. Oper. Res., **134**, No. 1, 103–119 (2001).
84. B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," Bell Syst. Tech., **49**, 291–307 (1970).
85. S. Kirkpatrick, C. D. Gelatti, and M. P. Vecchi, "Optimization by simulated annealing," Science, **220**, 671–680 (1983).
86. S. Lin, "Computer solutions of the traveling salesman problem," Bell Syst. Tech. J., **44**, 2245–2269 (1965).
87. S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," Oper. Res., **21**, 498–516 (1973).
88. A. Lodi, K. Allemand, and T. M. Leibling, "An evolutionary heuristics for quadratic 0-1 programming," Eur. J. Oper. Res., **119**, 662–670 (1999).
89. R. Marti, H. Lourenco, and M. Laguna, "Assigning proctors to exams with scatter search," in: M. Laguna and J. L. Gonzalez-Velarde (eds.), Computing Tools for Modeling, Optim. and Simul.: Interfaces in Comput. Sci. and Oper. Res., Kluwer Acad. Publ., Boston (2000), pp. 215–227.
90. P. Merz and B. Freisleben, "Greedy and local search heuristics for unconstrained binary quadratic programming," J. Heuristics, **8**, 197–213 (2002).
91. P. Merz and B. Freisleben, "Fitness landscapes, memetic algorithms and greedy operators for graph bipartitioning," Evolut. Comput., **8**, No. 1, 61–91 (2000).
92. N. Mladenovic and P. Hansen, "Variable neighborhood search," Comput. Oper. Res., **24** (11), 1097–1100 (1997).
93. E. Nowicki and C. Smutnicki, "A fast taboo search algorithm for the job shop problem," Manag. Sci., **42** (6), 797–813 (1996).

94. M. Ohlsson, C. Peterson, and B. Soderberg, "Neural networks for optimization problems with inequality constraints: the knapsack problem," *Neural Comput.*, No. 5, 331–339 (1993).
95. D. S. Johnson, C. R. Aragon, L. A. McGeoch, et al. (eds.), "Optimization by simulated annealing: an experimental evaluation. Part I," in: *Graph Partitioning, Oper. Res.*, **37**, 865–892 (1989).
96. D. S. Johnson, C. R. Aragon, L. A. McGeoch, et al. (eds.), "Optimization by simulated annealing: an experimental evaluation. Part II," in: *Graph Coloring and Number Partitioning, Oper. Res.*, **39**, 378–406 (1991).
97. M. A. Osorio, F. Glover, and P. Hammer, "Cutting and surrogate constraint analysis for improved multidimensional knapsack solutions," *Techn. Rep.*, Hearin Center for Enterprise Science, HCES-08-00 (2000).
98. I. Osman and J. Kelly, "Meta-heuristics: an overview," in: I. Osman and J. Kelly (eds.), *Meta-Heuristics: Theory and Applications*, Kluwer Acad. Publ., Boston (1996).
99. I. Osman and G. Laporte (eds.), "Metaheuristics in combinatorial optimization," in: *Ann. Oper. Res.*, Baltzer, Amsterdam (1996).
100. M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Rev.*, **33** (1), 60–100 (1991).
101. G. Palubeckis, "Multistart tabu search strategies for the unconstrained binary quadratic programming problem," *Ann. Oper. Res.*, **131**, 259–282 (2004).
102. P. M. Pardalos and G. A. Rodgers, "Branch and bound algorithm for the maximum clique problem," *Comput. Oper. Res.*, **19**, 363–375 (1992).
103. P. M. Pardalos and J. Xue, "The maximum clique problem," *J. Global Optimiz.*, **4**, 301–328 (1994).
104. F. Pezzella and E. Merelli, "A tabu search method guided by shifting bottleneck for the job shop scheduling problem," *Eur. J. Oper. Res.*, **120**, 297–310 (2000).
105. C. Reeves (ed.), *Modern Heuristic Techniques for Combinatorial Optimization*, Halsted Press, New York (1993).
106. M. Resende, "Greedy randomized adaptive search procedures (GRASP)," *Techn. Rep.*, AT&T Labs. Res., 98.41.1, Florham Park, NJ (www.research.att.com) (1998).
107. M. G. C. Resende and R. A. Werneck, "GRASP with path-relinking for the p -median problem," *AT&T Labs. Res.*, *Techn. Rep.*, TD-5E53XL, Sept. 18 (2002).
108. E. Rolland, D. A. Schilling, and J. R. Current, "An efficient tabu search procedure for the p -median problem," *Eur. J. Oper. Res.*, **96**, 329–342 (1996).
109. B. DeBacker, V. Fumon, P. Shaw et al. (eds.), "Solving vehicle routing problems using constraint programming and metaheuristics," *Heuristics*, **6** (4), 501–523 (2000).
110. V.A. Emelichev, E. Girlich, Yu. V. Nikulin et al. (eds.), "Stability and regularization of vector problems of integer linear programming," *Optimization*, **51** (4), 645–676 (2002).
111. E. Tsang and C. Voudouris, "Fast local search and guided local search and their application to British Telecom's workforce scheduling problem," *Oper. Res. Lett.*, **20**, 119–127 (1997).
112. P. J. M. Van Laarhoven, E. H. L. Aarts, and J. K. Lenstra, "Job shop scheduling by simulated annealing," *Oper. Res.*, **40** (1), 113–125 (1992).
113. M. Vasquez and Hao Jin-Kao, *A Hybrid Approach for the 0-1 Multidimensional Knapsack Problem*, Washington, Seattle (2001).
114. C. Voudouris and E. Tsang, "Guided local search and its application to the traveling salesman problem," *Eur. J. Oper. Res.*, **113**, 469–499 (1999).
115. C. Voudouris and E. Tsang, "Partial constraint satisfaction problems and guided local search," in: *Proc. 2nd Intern. Conf. Prac. Appl. Constr. Technol.* (1996), pp. 337–356.