International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 17 (2015) pp 38127-38132

A Resource-And-Time Method To Optimize The Performance Of Several Interrelated Operations

Vladimir Anisimov, Evgeniy Anisimov, Mikhail Sonkin

Russian Customs Academy (Moscow), National Research Tomsk Polytechnic University (Tomsk, 30 Str. Lenina)

Abstract
Rapid development and automation of production steadily increase the scope required for the development of scheduling algorithms, which consider various constraints. A special but wide class of scheduling problems encountered in the practical control of complex systems forms the problems of minimizing the time taken to perform several operations, with constraints on the number and interchangeability of operators and total value of operations. In particular, the problems of working out and implementing innovative development projects refer to this class. A method is proposed to solve the scheduling problem of minimizing the time taken in performing several operations, with constraints on the number and interchangeability of operators and total value of operations. The proposed algorithm obtains both exact and approximate solutions of a problem in scheduling theory, which is involved in minimizing the time taken in performing a set of interrelated operations with constraints on the number of interchangeable operators and the total cost of the operation performance.

Keywords: operations research, queuing theory, method, algorithm, scheduling problem, control of projects, resources, optimization.

1. STATEMENT OF THE PROBLEM

Meeting the challenges of planning and management has become particularly relevant in the 20th century. At that time, a new branch of mathematics, namely, operations research, and related disciplines, namely, queuing theory, scheduling theory, theory of automatic control, optimal control theory, and theory of multicriteria decision making, have emerged.

Scheduling theory is one of the most popular disciplines because of its theoretical and practical point of view in the field of operations research. Scheduling problems are related to construction schedules, namely, in ordering certain operations (operations) in terms of time and/or performers (devices). For example, scheduling problems arise in the following situations:

- work, that is, when a worker needs to arrange separate operations implementing management tools and time;
- transport, that is, in scheduling the operations of trains, airplanes, and public urban transport;
- planning activities in educational institutions;
- planning the employment of personnel, such as doctors on duty;
- performing complex construction projects, such as buildings and ships;
- planning of sports events; and
- operating computer networks with packet-scheduling priority information.

At the same time, scheduling should consider restrictions on the sequence of works and limitations associated with the performers. The purpose of solving such problems includes constructing feasible schedules in which all limits are met or determining the optimal feasible schedule for a particular optimality criterion. For example, an optimal schedule for speed (i.e., minimizing the total execution time of all work) schedules with minimal cost. The solution for scheduling problems is complicated because most of them are NP-hard (i.e., the algorithms for the solutions, which are implemented on a computer, may require an unacceptably large amount of time to solve practical problems «large dimension»).

Informally, many scheduling problems are optimizing, that is, they choose (finding) among the set of feasible schedules (schedules permit the conditions of the problem) to obtain decisions, which reach an “optimal” value of the objective function. Usually, *optimal* means a minimum or a maximum value of the objective function. Feasible schedule is understood in terms of its practicability and optimality [1, 5].

One of the main problems is classifying the scheduling problems and establishing their complexity. The most settled classification in the present-day scheduling problem classification was proposed by Graham [7].

The vast majority of surveyed scheduling problems are NP-hard. However, the practice requires the solution of such problems. Several approaches exist [4, 11]. The first approach is to develop polynomial heuristic algorithms. Some heuristic algorithms that detect errors estimate the solution. Such algorithms are called approximation. Algorithms that guarantee both relative error and absolute error are also available. Some NP-hard problems admit the existence of the so-called approximation scheme. In this scheme, you can determine an approximate solution to the relative error. This error does not exceed any given value of $\varepsilon > 0$ for the time polynomial in $1 / \varepsilon$ and the size of the input data of the problem, which is a polynomial approximation scheme (FPT AS). Problems with no approximation scheme are highly important to establish the limit ε, for which the possibility of finding the ε-approximate solution in polynomial time, PT AS, exists. Metaheuristic algorithms that are «good» solutions close to the optimum in a reasonable time are common at present. The disadvantage of these algorithms is the lack of assessment for solution quality. How the optimum solution differs from the worst case is not known. The exact methods for solving NP-hard problems are also given considerable attention in studies on scheduling theory. The most widely used method of search reduction is called branch-and-bound method. Calculated lower bounds of the objective function (in the case of minimization) and combinatorial properties use tasks to perform search reduction. Solving the problems of scheduling theory is also a widely applied method of dynamic programming. Most scheduling problems can be formulated as integer linear programming problems. The method of constraint programming has recently become widespread (defense, in English literature-Constraint Programming). One of the areas of its successful application is scheduling theory. Some complex scheduling problems can be optimally solved by algorithms that use elements of several methods, which is one of the most promising approaches [8, 9].

A special but wide class of scheduling problems encountered in the practical control of complex systems forms the problems of minimizing the time taken to perform several operations, with constraints on the number and interchangeability of operators and the total value of operations. In particular, the problems of working out and implementation of innovative development projects and others refer to this class.

To formalize these problems, the structure and interrelation of operations are conveniently represented in form of a network as follows:

$$G = \{(i, j)\}, \quad i, j = 0,1,\ldots,m, \quad i < j,$$

where *i, j* are node numbers and *(m + 1)* is the number of nodes.

With each operation in this network, we associate an arc (i, j) that connects the *i*th and *j*th nodes. The node $i = 0$ represents a start point in the system of analyzed operations. The nodes $i = 1, 2,..., m$ represent the last points that have already been performed, which correspond to arcs entering each node. *N* represents the total number of operations [2].

A sequence of operations is subject to the following condition: the operation corresponding to an arc coming from any node can be commenced only when all operations that enter that node have been completed.

Each operation (i, j) is characterized by its length $\tau(i, j)$ and the required number of operators $n(i, j)$.

To perform the operations, the set $R = \{1, 2,..., K\}$ of operators is employed (where k is the operator's number and K is the number of operators).

The interchangeability of operators is described by the matrix

$$\Delta = \|\delta_k(i, j)\|, \; k = 1,2,\ldots,K, \quad (i, j) \in G,$$

where

$$\delta_k(i, j) = \begin{cases} 1, \text{ if the } k-\text{th operator can be used to perform the } (i, j)-\text{th operation,} \\ 0 \quad -\text{other wise.} \end{cases}$$

The cost of calling on the operators to perform the operation is described by the matrix

$$C = \|c_k(i, j)\|, \quad k = 1,2,\ldots, \quad K(i, j) \in G,$$

where $c_k(i, j)$ is the cost of unit time for the *k*th operator to perform the *(i, j-)*th operation.

The schedule of performing the whole set of operations is defined by the set

$$Y = \{X_Y(i, j), r_Y(i, j) \mid (i, j) \in G, r_Y(i, j) \subseteq R\}, \qquad (1.1)$$

where $X_Y(i, j)$ is the time at which operation (*i,j*) begins for implementation of schedule *Y;* and $r_Y(i, j)$ is the set of operators called on to perform the (*i,j*)th operation according to schedule *Y*.

We assume that the interruption of an operation that has been started $(i, j) \in G$ is not allowed, and the complexity of the selected operators $r_Y(i, j)$ during the performance does not change.

The cost $\Omega_Y(i, j)$ of the operations (*i,j*) to implement schedule *Y* is defined by the proportion

$$\Omega_Y(i, j) = \sum_{k \in r_Y(i, j)} c_k(i, j)\tau(i, j). \qquad (1.2)$$

The set of the paths from the initial node to the terminal node is determined as *G*. The time taken to perform the whole set of operations is equal to the maximum length T_L of the path $L \in G_L$ from the initial node $i = 0$ of graph *G* to the terminal $j = m$ in the implementation of schedule *Y.*

This problem of minimizing the time taken uses the conventional notation to perform a set of operations with constraints on the number of operators. Their interchangeability and cost can be represented formally as the following mathematical programming problem:

A schedule

$$Y^* = \{X_Y^*(i, j), r_Y^*(i, j) \mid (i, j) \in G, r_Y(i, j) \subseteq R\} \qquad (1.3)$$

of the performance of a set *G* of interrelated operations, must be determined. This schedule is subject to the condition that

$$T^* = T(Y^*) = \min_Y \max_{L \in G_L} T_L(Y) \qquad (1.4)$$

with constraints on

$$X_Y(i, j) \ge \max_{l,i}\{X_Y(l, i) + \tau(l, i)\}, \quad (i, j) \in G; \qquad (1.5)$$

$$\sum_{(i, j) \in r_Y(i, j)} \delta_k(i, j) = n(i, j); \qquad (1.6)$$

$$\sum_{(i, j) \in F_Y(t)} (n(i, j) \le K; \qquad (1.7)$$

$$\sum_{k=1}^{K} \delta_k(i, j) \ge n(i, j); \qquad (1.8)$$

$$\sum_{(i, j) \in G} \Omega_Y(i, j) \le \Omega^*, \qquad (1.9)$$

where $F_Y(t)$ is a set of operations that are being performed at the current time *t* implementing the schedule *Y*; and Ω^* is the maximum cost of the set *G*.

Condition (1.4) shows the tendency of minimizing the time taken to perform the whole set of operations in problems (1.3) to (1.9).

Condition (1.5) establishes that the operations issued from any node of the graph can only be started after all the operations that have entered that node are completed.

Condition (1.6) requires that an established number of operators be allocated for each operation.

Condition (1.7) means that the number of operators used at any one time cannot be more than the total number of operators.

Condition (1.8) requires that the number of operators with the respective qualifications is large enough for each operation to be performed.

Condition (1.9) establishes that the total value of the set of operations cannot be over an acceptable level.

Problems (1.3) to (1.9) formally represent a nonlinear problem of discrete heterogeneous resources in the entire network. These problems can be classified as *NP*-hard, as described in the schedule theory. The exact methods of solving these problems are presented in [1, 2]. However, only renewable recourses, namely, operators and their productivity, are considered in this kind of model. The calendar plans for realization of different projects are formed. Not only the renewable resources but also the existing unrenewable ones must be considered. In problems (1.3) to (1.9), they are defined as the cost $\Omega_Y(i,j)$ of each completed operation $(i,j) \in G$ for the implementation of schedule *Y*, with the constraint (1.9). Resulting from the introduction of this ratio model, problems (1.3) to (1.9) are further generalizations of [1] model of schedule theory. Exact algorithms for solving problems (1.3) to (1.9) are not currently available. However, the management practice necessity of implementing complex projects inspires the development of such algorithms. The purpose of this study is to build one of them.

2. DESCRIPTION OF DEVELOPED METHOD

The existence of a solution for problems (1.3) to (1.9) is necessary and sufficient for the following reasons:

a) the cast and interchangeability of operators provide the ability to perform the whole complexity of *G* operations, and

b) the cost of performing complex *G* operations does not exceed a prescribed level Ω^*.

Formally, the fulfillment of the first of these requirements is the term (1.8). This term means that professionals can be selected from the whole group of operators to perform any work of complex *G* operations. To check the fulfillment of the second requirement, we consider the set

$$R^* = \{r^q(i,j)\},\ (i,j) \in G,\ q = 1,2,\ldots \qquad (2.1)$$

of all possible options of resource allocation for the corresponding operations. This set definitely involves another one, namely,

$$\Omega = \{\Omega^q(i,j), q = 1,2,\ldots, (i,j) \in G\} \qquad (2.2)$$

of the costs $\Omega^q(i,j)$ of the performance of corresponding operations in selected options of resource allocation. Set (2.2) elements are defined by the proportion

$$\Omega^q(i,j) = \tau(i,j) \sum_{k \in r^q(i,j)} C_k(i,j). \qquad (2.3)$$

On the basis of (2.3), the second requirement, which ensures the existence of the solution for problems (1.3) to (1.9), can be presented in the following form:

$$\sum_{(i,j)\in G} \min_q \Omega^q(i,j) \le \Omega^*,\ q = 1,2,\ldots. \qquad (2.4)$$

Checking the feasibility of the requirements (1.8) and (2.4) before the start of the search procedure for solving problems (1.3) to (1.9) can be useful. If these requirements are feasible, then the solution of problems (1.3) to (1.9) exists. The algorithm of this search is based on the branch-and-bound procedure for solving the problems of schedule theory proposed in [2].

The algorithm consists of a finite number of steps and rests on the following constructions:

a) Set $V = \{S\}$ of fragment *S* of schedule *Y* is represented, which is feasible in terms of constraints (1.5) to (1.7) and (1.9) in the form of a subset tree (branching).

b) The lower bound of the objective function (1.2) is calculated for these subsets.

c) Feasible schedules are determined.

d) The optimality of these schedules is checked.

The proposed procedure [2] permits defining the calendar plan (1.3) of performance of complex *G* operations, thereby fulfilling conditions (1.5) to (1.8).

The special point in this algorithm considers branching and condition (1.9) continually. If it is violated, then the solution obtained by continuing the branches of the tree of options is not feasible, and a new branch should be made.

In this algorithm, branching is constructed through a dichotomous scheme. In such a scheme, each vertex v_s (*S*th branch of the tree) is a feasible element. If the (i,j)th operation corresponding to the element is started at the time $x_S(i,j)$ with the $r_S(i,j)$th alternative allocation of operators, then

$$v_S = \{x_S(i,j), r_S(i,j)\}. \qquad (2.5a)$$

If the (i,j)th operation for the $r_S(i,j)$th alternative allocation of operators is not begun at time $x_S(i,j)$, then

$$v_S = \varnothing. \qquad (2.5б)$$

For each branch $S \in V$, the values $x_S(i,j),\ (i,j) \in G$ (the times at which the corresponding operations are started) have to be chosen from the increasing sequence

$$t_S = \{t_S^n\},\ n = 1,2,\ldots,$$

where $t_S^1 = 0$ and subsequent times $t_S^n,\ n = 2,3,\ldots$ are found with the recurrence formula

$$t_S^n = \min_{(i,j) \in F(t_S^{n-1})} \{x_S(i,j) + \tau(i,j)\}, \qquad (2.6)$$

where $F_S(t_S^{n-1})$ is the set of operations (i,j) previously included in the *S*th branch and uncompleted by the time t_S^{n-1}, that is,

$$F_S(t_S^{n-1}) = \{(i,j) \mid (i,j) \in G, x_S(i,j) \le t_S^{n-1} \le x_S(i,j) + \tau(i,j)\}. \qquad (2.7)$$

Thus, t_S is the sequence of times at which operations included in a given branch of the tree are completed, and the

corresponding operators become free. The condition $t_S^1 = 0$ reflects the fact that all alternative schedules start at time 0.
Setting the starting times of operations not in accordance with the previous sequences does not permit reduction of the time to perform the set of operations. The reason is that for any schedule *Y* that contains the fragment *S*, an earlier start of an operation $(i, j) \in S$ belongs to the sequence t_S by definition. Thus, later starts of operations lying on critical paths for schedule *Y* also belong to that sequence. The starting times of operations, which do not belong to critical paths, can vary within the corresponding time reserves. At the same time, the boundaries of these reserves also belong to this sequence, and variation within boundaries does not help to reduce the time taken to perform a set of operations as a whole. Thus, the optimal times for the start of operations with regard to criterion (1.4) in schedule (1.3) must belong to respective sequence t_S.
To realize the dichotomous scheme, we consider the following set related to (2.1):

$$D = \{d_q \mid q = 1, 2, \dots\}, \qquad (2.8)$$

where $d_q = 1$ if allocation $r^q(i, j)$ is made for the (i, j)th operation; otherwise, $d_q = 0$.

Then, the serial number of q element $d_q = 1$ of set D characterizes the operation that has been performed, the particular allocation of operators for it, and its cost.
With regard to (2.8), branching to form an optimal schedule (1.3) involves choosing for each time t_S^n feasible variables $d_q \in D$ and establishing their values, which takes the form $\nu_S = \{t_S^n, d_q = 1\}$, if the $(i, j) \in G$ operation corresponding to d_q is started at time $x_S(i, j) = t_S^n$ with allocation $r^q(i, j)$, and $\nu_S = \{t_S^n, d_q = 0\}$, if this operation is not started with allocation $r^q(i, j)$ at time t_S^n.

The set P_S^n of variables $d_q \in D$, which may be included in the Sth fragment of the schedule at time t_S^n contains the values $d_q \in D$ corresponding to operations $(i, j) \in G$, which are not previously in S for which

$$x(h, i) + \tau(h, i) \le t_S^n, \quad (h, i) \in G; \qquad (2.9\text{a})$$

$$r^q(i, j) \subseteq R_S^n, \qquad (2.9\text{б})$$

$$\Omega^* - \sum_{m \in D_S(t_S^{n-1})} d_m \Omega_m + \Omega_q \ge 0, \qquad q = 1, 2, \dots, \qquad (2.9\text{в})$$

where R_S^n is the set of unoccupied operators for the *S*th fragment of the schedule at time t_S^n; and $D_S(t_S^{n-1})$ is the set of variables d_q included in the *S*th fragment of the schedule at time t_S^{n-1}.
The first of these conditions determines the operations by time t_S^n. All proceedings have been performed, the second identifies those for which unoccupied suitably qualified operators exist, and the third marks those for which the cost of operator performance is suitable [3].
The estimate W_S of the lower boundary of the objective function (1.4) can be taken as the maximum length of the path from the initial *G* to the last node of *G* found, without allowing for resource constraints (1.7) and (1.9), for operations not included in *S*. Then, if corresponding time t_S^n stops on the next branching, then $d_q = 1$, $W_S(d_q = 1)$ is determined as follows:

a) operations $(h, i) \in G$, which were already in the *S*th schedule fragment, that is, operations for which $x_S(h, i) < t_S n$ are started at the respective times $x_S(h, i)$ and are finished at $x_S(h, i) + \tau(h, i)$;

b) for an operation $(i, j) \in G$ corresponding to the $d_q = P_S^n$, which is placed into the branch of the tree of alternatives $x_S(i, j) = t_S^n$ in the given step;

c) for operations $(i, j) \in G$ corresponding to variables $d_u \in P_S^n, \quad u \ne q$, which cannot be placed in the schedule at time t_S^n simultaneously with t_S^{n+1} because of the resource constraint (2.9), the length is defined by the proportion $t_S^{n+1} + \tau(e, h)$.

However, if $d_q = 0$ on the given branching step, then $W_S(d_q = 0)$ is as follows:

a) operations $(h, i) \in G$, which were already in the *S*th scheduling fragment (operations, which $x_S(h, i) < t_S^n$, are started at time $x_S(h, i)$ and finished at time $x_S(h, i) + \tau(h, i)$);

b) for the operation (i, j) corresponding to the variables $d_q \in P_S^n$ started at time t_S^{n+1} and finished at $t_S^{n+1} + \tau(h, i)$;

c) for other operations $(e, h) \in G$ corresponding to the variables $d_u \in P_S^n, \quad u \ne q$ started at time t_S^n and finished at time $t_S^n + \tau(e, h)$.

An important element of this method of solving problems (1.1) to (1.9), which has a considerable influence on its convergence, is the technique used to select the next operation and the allocation of operators for it, that is, the choice of variables $d_q \in P_s^n$ for inclusion in the *S*th branch method at time t_S^n. In this method, the choice of d_q at the next branching step is made in two stages: first, the operation is selected, and then, operator allocation is selected from the alternatives. The next operation is chosen in accordance with the following preference sequence:

$$\min T_j^{(n)} \to \max \tau(i, j) \to \min i \to \min j,$$

that is, the first operation to be included is the one that corresponds to the shortest last completion time $T_j^{(n)}$. If several completion times exist, then the largest operation is chosen. Moreover, if several such operations exist, then the operation with smallest numbers i, j is chosen. The last completion periods must be found on the basis of the given fragment *S*.

The allocation of operators $r(i, j)$ for the chosen operation (i, j) is determined from the condition that the value

$$\sum_{k \in r(i,j)} \sum_{(l,h) \notin S} \delta_k(l,h),$$

is a minimum, that is, the operators that are least able to perform any of the other remaining operations $(l,h) \notin S$ are allocated.

The operation (i, j) and the allocation $r(i, j)$ chosen in this way uniquely define the next variable $d_q \in P_S^n$, which is included in the *S*th branch of the tree of alternatives at time.

The route through the tree is in accordance with the rule "go to the right." Then, the current schedule fragment, the smallest previously obtained value of the objective function, and the corresponding feasible schedule must be stored when solving the problem.

This rule, allied with the method of choosing the operations and the operators examined earlier, constitutes the approximate algorithm for solving problems (1.3) to (1.9) and allows the first feasible solution to be obtained in a finite number of steps, which is equal to the number *N* of operations in the network *G*.

Each *S*th branch ends if all *N* operations are included, that is, a feasible schedule *Y* has been obtained, or if

$$W_s \geq T^0(1-\mu), \quad 0 \leq \mu \leq 1, \tag{2.10}$$

where T^0 is the smallest (record) value or the objective function for previous feasible solutions; and μ is the given permissible deviation of the objective function from the optimal solution (optimization accuracy). If condition (2.10) holds, then a previous record in the current branch is impossible to improve and continuing it is pointless.

The search for the solution ends if condition (2.10) holds for all remaining branches. The way of routing through the tree corresponds to the second return to root vertex. The last record is the required optimal value of the objective function (1.4), and the corresponding feasible schedule is the optimal schedule *Y*.

3. CONCLUSION

The application of the described procedure provides an analysis of all possible schedules and eliminates duplicates when revising them. For its implementation, storing in computer memory only the current fragment of the plan, the smallest of the previously obtained values of the objective function (1.4), and the corresponding feasible schedule to perform the set of operations is sufficient. This rule of routing through the tree in combination with the method of choosing operations and resources at each step of branching is the approximate algorithm for solving problems (1.3) to (1.9), thereby allowing the creation of the first valid spot calendar plan for a finite number of steps equal to the number of *N* operations in the network *G*. However, the proposed algorithm does not correspond to the idea of building spot plans. In accordance with the scheme of tree options at each step of branching, a decision to include or not to include a certain operation in the plan should be considered. The options for resource reservation are similarly considered. Therefore, the proposed scheme of branching with the rule of obtaining sequences of t_s moments by choosing the set of operations and resource allocation provides a revision of all possible options of the calendar plan.

Overall, the proposed algorithm leads to both exact and approximate solutions of a problem in scheduling theory, which is involved in minimizing the time taken to perform a set of interrelated operations with constraints on the number of interchangeable operators and the total cost of the operation performance.

REFERENCES

[1] Anichkin A.S., Semenov V.A. Current models and methods scheduling theory. Proceedings of the Institute for System Programming of Russian Academy of Sciences. 2014. T. 26. №-3. P. 5-50.

[2] Anisimov V.G., Anisimov Ye.G. A branch-and-bound algorithm for one class of scheduling problem. The magazine of computational mathematics and mathematical physics, 1992, T.32, № 2, P.2000-2005.

[3] Anisimov V.G., Anisimov Ye.G., The algorithm for the optimal distribution of discrete heterogeneous resources on the net. The magazine of computational mathematics and mathematical physics, 1997, T.37, № 1, P.54-60.

[4] Brooks G.N., White C.R. An algorithm for finding optimal or near– optimal solutions to the production scheduling problem // J. Ind. Eng.– 1965.– V. 16, N 1.– P. 34 – 40.

[5] Conway R.W., Maxwell W.L., Miller L.W. Theory of Scheduling. Addison-Wesley, Reading, MA. 1967.
[6] Gawiejnowicz S., Pankowska L. Scheduling jobs with varying processing times // Information Processing Letters V. 54. 1995. P. 175-178.
[7] Graham R.L., Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G Optimization and approximation in deterministic sequencing and scheduling: a survey // Ann. Discrete Optimization. 1979. V. 2. P. 287– 325.
[8] Lazarev A.A., Gafarov E.R. Scheduling theory. Problems and algorithms. M.: 2011, 222.
[9] Tanayev V.S., Shkurba V.V. Introduction to the theory of schedules. M.: Science, 1975.
[10] Kobak V. Red D.G., Neudorf R.A. Advanced distribution algorithm for solving inhomogeneous scheduling problems. Southern Federal University. Engineering. 2008. № 9 (86). Pp 152-156.
[11] Korbut A.A., Segal J.C., Finkelstein J.J. Branch and bound method. Review of theories, algorithms, programs and applications // Operations Forsch. Statist., Ser. Optimiz. 1977. V. 8. № 2. P. 253-280.