Computational Technologies, 2022, том 27, № 4, с. 118–126. © ФИЦ ИВТ, 2022
Computational Technologies, 2022, vol. 27, no. 4, pp. 118–126. © FRC ICT, 2022

ISSN 1560-7534
eISSN 2313-691X

DOI:10.25743/ICT.2022.27.4.010

Schedule formation in multi-project developments management systems

M. V. Purgina[1], A. S. Dobrynin[2,*], R. S. Koynov[2]
[1]Novosibirsk State University of Economics and Management, 630099, Novosibirsk, Russia
[2]Siberian State Industrial University, 654007, Novokuznetsk, Russia
[*]Corresponding author: Dobrynin Alexey S., e-mail: serpentfly@mail.ru
Received March 18, 2022, revised April 12, 2022, accepted April 20, 2022

Modern mathematical apparatus, as well as achievements in the field of system analysis, game theory, graph theory, make it possible to algorithmize the process of solving the schedulling problem for various areas of human activity. To date, the areas of effective application of many well-known, including modern heuristic and metaheuristic, algorithms that have shown good results in practice have been analyzed and identified. However, despite the achievements in the field of discrete optimization, schedulling and network planning, new tasks of drawing up the so-called coordinated schedules in the field of multi-project planning, which take into account the preferences (requests) of specific schedulling executors, are still of practical interest. Approaches and main stages of solving the problems of building coordinated schedules in multi-project planning are considered, which is relevant for the development of new generation software and tools.

Keywords: schedulling, multiproject planning, digraph, multigraph, domain, clustering.

Citation: Purgina M.V., Dobrynin A.S., Koynov R.S. Schedule formation in multi-project developments management systems. Computational Technologies. 2022; 27(4):118–126. DOI:10.25743/ICT.2022.27.4.010.

Introduction

The article considers theoretical and applied aspects of managing multi-project developments. The proposed approaches can be used to create a new generation of instrumental information systems, project management systems, which are distinguished by more complex management approaches compared to the well-known and widely used WBS (Work Breakdown Structure) models. The approaches considered in the article are partially based on some methods of calendar, network planning [1, 2], some of them are well studied, some algorithms used in practice have proven theoretical efficiency, which allows solving real practical problems in automatic mode. Of particular interest is the creation of application tools for different or similar project teams working simultaneously (in parallel) with several projects. Currently, there are no developed schedulling complexes for multi-project planning on the market, which is connected, firstly, with the complexity of such systems, and secondly, with the lack of a holistic view of their work. Justification of the application of the multigraph model in the problem of multiproject schedulling. The modern project support tools presented on the market are largely analogues of the well-known MS

Project system developed by Microsoft and implement models and methods similar in nature and algorithms. Analyzing such systems, a number of interesting conclusions can be drawn. For example, some products use hierarchical models of enlarged tasks, project stages, without taking into account the fact that projects can be executed simultaneously or in parallel. Also, the preferences (requests) of performers are not taken into account, which does not allow building coordinated schedules in automatic mode. The vast majority of well-known modern schedulling tools for project activities use fairly common decomposition approaches, very often the WBS hierarchical model. Such tools are perfectly compatible with SADT and IDEF0, however, do not take into account a number of important features. Let us hypothetically imagine that the processes or real applied work of the project activity are described in the SADT language, which is a common practice. Then the applied implementation of the schedulling tool in a high-level programming language can be based on the ideas of recursive hierarchical decomposition by John McCarthy, Richard Bellman [3, 4], Jacon [5], Lester Ford, Floyd and other researchers. But the reality is almost always much more complicated.

We cannot guarantee that the actual project activity will be the same as in the SADT model. We must state (with regret) the obvious fact: any hierarchical models are too limited for use in network and schedulling problems. They are not adequate for solving real problems.

Consider a network diagram $\overline{G} = (\overline{V}; \overline{E})$, in which a set of design works (processes) $\{jb_i\}$ are associated with arcs (connections) of the digraph $jb_i \equiv e_i \in \overline{E}$ and corresponding events $\nu_x \in \overline{V}$, each of those describes the end of a certain enlarged stage of the project x. The obvious advantage of such a network model over the above hierarchical one is the additional flexibility, since each next level in the tree hierarchy always has a single root. Thus, some event ν_x can aggregate specific achieved results obtained on sets of previously performed parallel works $\{\nu_{x-1}, \nu_{x-2}, \ldots, \nu_0\}$, which in principle cannot be achieved using a hierarchical model.

Suppose the digraph is presented and formed in a functional or object-oriented programming language (a simple abstraction), which does not create any particular difficulties in the process of applied implementation of this model, but the limitations that arise at this stage are also very significant. In fact, the model of a set of works in the form of a single digraph is not optimal and suitable for practice — it is not much better than a hierarchical model. Such is the reality. But there is a way out. It consists in increasing the level of abstraction and unifying the overall scheme with getting rid of unnecessary assumptions and problems. The more abstract and high-level a model is, the better it is for practice, almost always. Existing solutions are too specific, which does not allow them to be practically suitable. A work package model in the form of a network graph is great, but what should be done if you need to have one model of several work packages, including those performed in parallel in time? A higher level model is needed.

Multiproject planning actually implies that the main model of activity of the joint team of project developers is a directed multigraph $\overline{GM} = (\overline{V_{dm}}; \overline{E_{dm}})$, such that $\overline{GM} = \{\overline{G_1}, \overline{G_2}, \ldots, \overline{G_{Dm}}\}$ on a set of connectivity domains $\overline{Dm}$. Using the multigraph model $\overline{GM}$, in comparison with the digraph model $\overline{G}$, allows you to go up a notch and take into account some additional elements in network planning, for example, the parallel execution of several projects by the same organization. This requires applied tools for solving multi-project problems, which are not currently available.

Implementation of the problem of multiproject planning for managing schedules on graphs

Consider a specific example of the representation of the problem of multiproject planning in the form of a complex of typical problems of schedulling on graphs. Let's take a typical scenario, when several parallel projects (let's say four) are executed with different start dates and contain unique, different jobs with different durations, each graph node has a unique identifier (string, number or Guid). Simplified abstract examples of projects are presented in Fig. 1.

As can be seen from the models shown in Fig. 1, many parallel projects A, B, C, D are independent or may have common work used (testing, writing project documentation, instructions for users). There is a need for generalized, unified methods for solving such problems in practice, which are the basis for creating powerful tools for multi-project planning using a deterministic Turing machine [6].

A. Stages of solving the problem of multiproject planning. Let's briefly consider the formulation of the problem of forming schedules agreed with the performers in multiproject planning.

Given:

1. A set of works or processes, for which properties are set: associations, when the enlarged work consists of simpler works; connectivity; lack of closed loops:

$$\gamma_{JB_i,JB_j} = \begin{cases} 1, & \text{if } JB_i \to JB_j, \\ -1, & \text{if } JB_j \to JB_i, \\ 0, & \text{if } JB_i \to JB_j \text{ independent.} \end{cases}$$

2. A given multigraph $\overline{GM} = (\overline{V_{dm}}; \overline{E_{dm}})$, such as $\overline{GM} = \{\overline{G_1}, \overline{G_2}, \ldots, \overline{G_{Dm}}\}$, on the set of connectivity domains. Describes the work planning sequence $jb_i \equiv e_i \in \overline{E_{dm}} \in \overline{E}$.
3. Segments of planning $t^{st}_{dm} \in \overline{T}$, or initial (starting) $pt^{st}_{dm}[t, d, m, y]$ points for the formation of schedules at a certain time.
4. The normative duration T^H_{jb} of the planned work, which is approved by the performers and agreed with the Center. To determine it, the methods of passive and active examination can be used, as well as a survey of performers with the definition of "pessimistic"

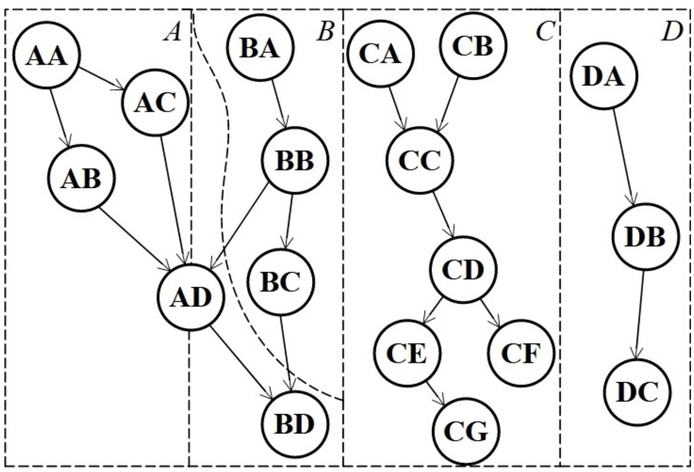

Fig. 1. A simplified example of a multigraph representation for projects A, B, C, D. AD – BD, BC – BD — common works in use; AA, AC, BA — designation of stages or milestones of the project

and "optimistic" estimate $T_{jb}^{\min}$, $T_{jb}^{\max}$. Other ways to determine the standards can be obtained using the techniques of restorative-predictive normalization [7, 8] or the case approach [9].

5. Basic time interval ΔT (window, slot), which is an indivisible segment of continuous time $[t, t + \Delta T]$.

List of main restrictions.

6. A set of hard constraints, formalized in the form of corresponding tuples of working and non-working time:

$$H_p[d, (t + \Delta T)] = \begin{cases} 1, & \text{if the day } d \text{ and time work } [d, (t + \Delta T)] \text{ is allowed;} \\ 0, & \text{if work is not allowed.} \end{cases}$$

7. Many soft restrictions. They are formed during the implementation of the procedure for agreeing on the preferences of performers and the Center $\{Ag_i\}$ (Ag — an agent):

$$S_{pi}[d, (t + \Delta T)] = \begin{cases} 1, & \text{preferred accommodation } (a_i, jb_k) \text{ for } Ag_i; \\ 0, & \text{accommodation is not desirable.} \end{cases}$$

8. Coordinated planning $\pi(S)_p$ and incentive $\sigma_i(R_p^*, \tilde{Y}_p)$, mechanisms, where $\pi(S)_p$ is a plan based on information S, in hierarchical games with information exchange [10] (games by B.Yu. Germeier). The incentive function $\sigma_i(R_p^*, \tilde{Y}_p)$ depends on the equilibrium, according to Stackelberg, schedules R_p^* and actions of agents $\tilde{Y}_p$ aimed at its implementation.
9. The domain model of requests and wishes of the agent S_{pi} and the method of its identification (remap) with "soft" restrictions actually describes a set of directories of classifiers of roles, operations, resources: $\overline{CL} = \{\overline{CLR}, \overline{CLO}, \overline{CLRS}\}$, a set of elements of classifiers CLR_{jb}, CLO_{jb}, and a classifier of resources CLR_{jb} for all works $lb \in \overline{JB}$ that are planned by the Center for based on common elements $\overline{CLR}$, $\overline{CLO}$, $\overline{CLRS}$ individual for differing specific portfolio p projects; a set of assets $\overline{A}$, each of which has a corresponding use value Z_a, including all performers (agents) $\{Ag_i\}$. Assets can belong to a certain class and (or) group a subset of roles (own a set of competencies) $Ag_i \in \{CLR_{ag} \in CLR\}$, $i = 1 \dots N$.

Criterion.

The problem under consideration uses a three-component optimality criterion $\overline{Q} = \{Q_1, Q_2, Q_3\}$, in which components can be distinguished: time, costs and the number of requests taken into account and wishes of schedulling executors.

Elements of the criterion: time $Q_1 = \sum\limits_{i=1}^{L} t_l^{cr} \to \inf$, where t_l^{cr} is the duration l-th (l — the serial number) of work on the critical paths of the multigraph $\overline{GM} = (\overline{V_{dm}}; \overline{E_{dm}})$; costs, $Q_2 = \sum\limits_{j=1}^{M} z_j^A \to \inf$, where z_j^A is the cost of using the j-th asset, which is schedulled for the execution of schedule activities; the number of unfulfilled soft constraints (Ag_i — requests and wishes of the agent), $Q_3 = \sum\limits_{\gamma=1}^{V} \mu_{i\gamma} \to \inf$, $\mu_{i\gamma} = f[S_{pi}[d, (t + \Delta t)]]$.

Required.

To form an optimal Q, according to the criterion, coordinated work schedule with assets (resources) $\overline{A}$ tied to them, which satisfies all regulatory requirements and restrictions. Formation of a detailed solution of the problem shown above is beyond the scope of this article. A detailed description of the mechanisms for constructing consistent schedules in hierarchical systems is a separate time-consuming and interesting task, which, due to a limited volume, cannot be considered here. Let us present in an enlarged way the main basic stages of solving the problem of multiproject planning, which can be performed on a computer, they are partially described in [11, 12]. Modern high-level programming languages such as C++, C#, Java, Python, Javascript are suitable for solving the formulated non-trivial task. The main parts of this procedure are presented below.

B. Stage of formation of project connectivity domains. The connected domain $dm_j \in \overline{DM}$ is actually a space that contains a collection (set) of interconnected nodes of the digraph $\nu_j \in \overline{V}$. To determine this domain, some starting point (base neighbourhood) is required, from which the execution of the search procedure on the graph is initiated. The ideal choice here is the set of initial nodes of the multigraph $\{\nu^{st}_{j=1}, \nu^{st}_{j=2}, \ldots, \nu^{st}_{j=n}\} \in \overline{V}$ for which it is characteristic that the vector of parent nodes $\overline{V_{prm}}$ in relation to them does not contain elements (empty set) $\nu^{st}_j \to \overline{V_{prm}} \in \overline{V}^{def} = \varnothing$. As a search procedure, it is permissible to use "deep" (deep search) or "wide" (wide search) search on a multigraph $\overline{GM} = (\overline{V_{dm}}; \overline{E_{dm}})$. The process of formation of connectivity domains in a multigraph (see Fig. 1) is shown in Fig. 2.

C. The stage of semantic clustering of problem connectivity domains. As you can see, the solution of the problem of the previous stage forms domains of interconnected nodes $\nu_j \in \overline{V}$ of the multigraph $\overline{GM}$, which contain common elements (vertices and arcs). The solution of the problem of clustering (unification of domains) according to a certain semantic attribute, condition will allow us to select both independent digraphs and form graphs that have common jobs. Clustering can be practiced according to the statements presented below. If the domains partially (weakly) intersect, while non-intersecting nodes have or do not have parents (children), then the multigraph contains common jobs. If domains partially (strongly) intersect, while non-intersecting nodes do not have parents (children), then there is a separate digraph that is not reduced to a network structure. Otherwise, we are dealing with a separate digraph, which is reduced to a network structure. An example of solving the problem of semantic clustering is shown in Fig. 3.

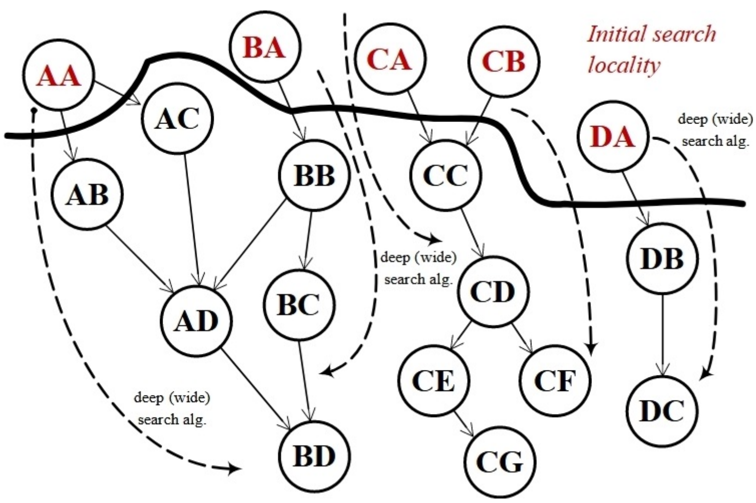

Fig. 2. Construction of connection domain of a digraph. Task solution: AA $\to$ $\{$AA, AB, AC, AD, BD$\}$; BA $\to$ $\{$BA, BB, AD, BC, BD$\}$; CA $\to$ $\{$CA, CC, CD, CE, CF, CG$\}$; CB $\to$ $\{$CB, CC, CD, CE, CF, CG$\}$; DA $\to$ $\{$DA, DB, DC$\}$

Further, in the course of solving the problem of multiproject planning, the digraph is reduced to a network form, when the structure does not contain cycles (the cycles are opened with the addition of fictitious jobs), and the set of end nodes of the digraph $\overline{G} = (\overline{V}; \overline{E})$ is supplemented by initial and final nodes with fictitious connections.

D. The stage of bringing connectivity domains to a network structure. Determination of node vectors of terminal neighbourhoods $dm_j = \overline{DM}$ in each connectivity $\overline{V_{okr}} = \overline{V}$ domain and supplementing them with fictitious jobs to form a network representation (Fig. 4).

Let's consider more briefly the remaining stages.

E. Topological sorting of a multigraph nodes. Topological sorting of nodes of a multigraph $\overline{GM} = (\overline{V_{dm}}; \overline{E_{dm}})$, which results in a vector of nodes being ordered based on the position of the node in the overall topology of the graph. The sorting of the nodes of the digraph is performed according to the initial order given by the nodes and edges on the subset of vertices. The mechanism of topological sorting of a directed graph uses

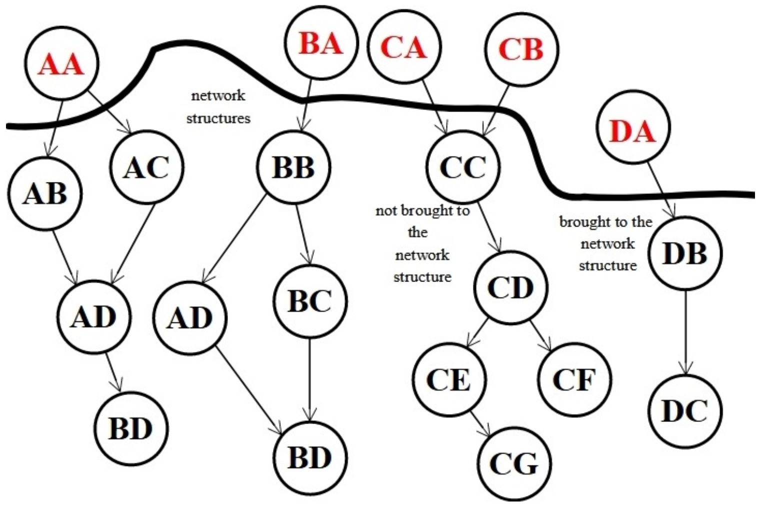

Fig. 3. Solving the subproblem of semantic clustering. Task solution: AA $\rightarrow$ $\{AA, AB, AC, AD, BD\}$; BA $\rightarrow$ $\{BA, BB, AD, BC, BD\}$; CA $\rightarrow$ $\{CA, CB, CC, CD, CE, CF, CG\}$ — "non-network" view; DA $\rightarrow$ $\{DA, DB, DC\}$

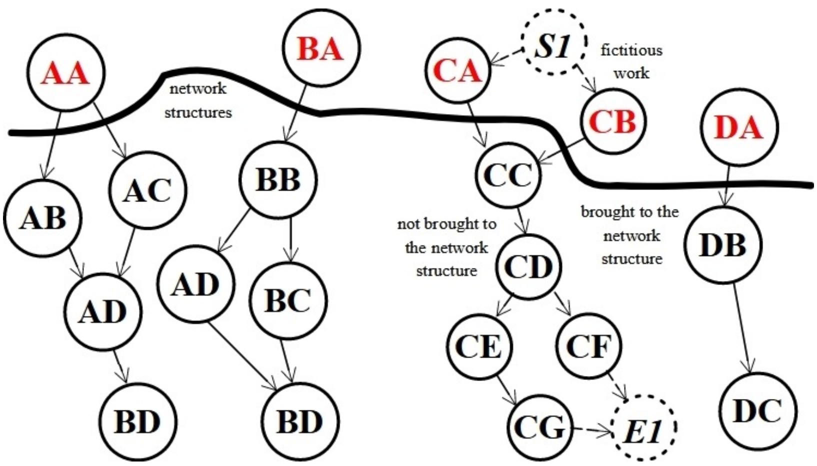

Fig. 4. An example of reduction to a network structure. Task solution: AA $\rightarrow$ $\{AA, AB, AC, AD, BD\}$; BA $\rightarrow$ $\{BA, BB, AD, BC, BD\}$; CA $\rightarrow$ $\{CA, CB, CC, CD, CE, CF, CG\}$ — "non-network" view; DA $\rightarrow$ $\{DA, DB, DC\}$

the gradual removal of nodes that do not have parent elements, followed by the restoration of the graph. For this, the historical vector of previously removed elements (arcs, nodes) is used. Consequently, for some typical algorithm (synthesis, visualization), a normalized data structure is subsequently formed, which is subsequently used to unify the solution of particular problems.

F. Determining the terminal vertices of the multigraph. Determining the terminal vertices of the multigraph $\overline{GM}$, as well as the critical paths $\overline{G} = (\overline{V}; \overline{E})$ for all digraphs included in the composition $\overline{GM}$.

The vertices correspond to the start (initial) and end nodes $\{\nu_{dm}^{st}, \nu_{dm}^{nd}\} \in \overline{V}$ in the topologically sorted graph vector. Calculation of the time duration of critical paths for all digraphs T_{dm}^{cp}. The critical path is calculated using a time difference and a real number, which allows you to work in terms of calendar time and relative weights at the same time. This method also allows solving various logistical problems of laying paths, routes and building transport schedules.

G. Determination of an early date $t_p(e_j)$ for the completion of an event e_j. The early time required to complete all the work preceding this event e_j (direct iteration) is determined in accordance with the expression:

$$t_p(e_j) = \max(t_p(e_i) + \tau(e_i, e_j)); \quad (e_i, e_j) \in U_{e_j}^+; \quad (e_i, e_j) \in \overline{E},$$

where $t_p(e_j)$ — the early date of the event i; $\tau(e_i, e_j)$ — duration (weight) of work between events (e_i, e_j); $U_{e_j}^+$ — the set of jobs (directed arcs) related to the event e_j. Early dates can be determined using the ant colony, Edgar Dijkstra or Bellman – Ford algorithms [3, 4].

H. Determination the late date $t_\Pi(e_j)$ for the completion of an event e_j (reverse iteration). It is defined as the point in time after which there is as much time left before the critical deadline as is necessary to complete all the work following this event:

$$t_p(e_j) = \min(t_p(e_i) - \tau(e_i, e_j)); \quad (e_i, e_j) \in U_{e_j}^-; \quad (e_i, e_j) \in \overline{E},$$

where $t_p(e_j)$ — the early date of the event e_i; $\tau(e_i, e_j)$ — duration of work between events (e_i, e_j); $U_{e_j}^-$ — the set of jobs emerging from the event e_i.

The delay for an event is determined according to the expression:

$$RS(e_i) = t_\Pi(e_i) - t_p(e_i).$$

I. Determination of the total delay for work. Determination of the total delay for work — in fact, the maximum duration by which work can be delayed or increased in its duration, without changing the critical period:

$$RS_\Pi(e_i, e_j) = t_\Pi(e_j) - t_p(e_i) - \tau(e_i, e_j).$$

J. Determining the delay for work. Determining the delay for work — the maximum amount of time by which the duration of this work can be increased without changing the initial dates of subsequent work, provided that the previous event occurred at its early date:

$$RS_C(e_i, e_j) = t_p(e_j) - t_p(e_i) - \tau(e_i, e_j).$$

Consistent implementation of all the stages considered above makes it possible in practice to generate schedules for organizations engaged in multi-project activities in automatic or automated mode. That will allow you to create a new generation of project management tools.

Conclusion

The article discusses the main stages of automatic schedulling in the problem of multiproject planning. The implementation of these stages using high-level planning languages (C++, C#, Java, Python, etc.) will allow developing tool-software tools for planning the activities of enterprises, organizations of a fundamentally new level, the distinguishing feature of which is the focus on complex design and process activities, support for mixed and parallel planning. Separate elements of the presented enlarged schedulling procedure are implemented in the software developed by the authors, including the TmBuilder coordinated schedulling system [13].

References

[1] **Golenko D.I.** Statisticheskie metody setevogo planirovaniya i upravleniya [Statistical methods of network planning and control]. Moscow: Izdatel'stvo Nauka; 1968: 400. (In Russ.)

[2] **Adel'son-Vel'skiy G.M.** O nekotorykh voprosakh setevogo planirovaniya. Issledovaniya po diskretnoy matematike [On some questions of network planning. Research in Descrete Mathematics]. Moscow: Nauka; 1973: 251. (In Russ.)

[3] **Bellman R., Gross O.** Some combinatorial problems arising in the theory of multistage processes. Journal of the Society for Industrial and Applied Mathematics. 1945; 2(3):124–136.

[4] **Bellman R.** Mathematical aspects of scheduling theory. Journal of the Society for Industrial and Applied Mathematics. 1956; 4(3):168–205.

[5] **Jackson J.R.** Ocheredi s dinamicheskim pravilom prioriteta. Kalendarnoe planirovanie [Queues with dynamic priority rule]. Moscow: Izdatel'stvo Progress; 1966: 392. (In Russ.)

[6] **Turing A.M.** Computer machinery and intelligence. Mind. 1950; (49):495.

[7] **Avdeev V.P., Kartashov V.Ja., Myshlyaev L.P., Ershov A.A.** Vosstanovitel'no-prognoziruyushchie sistemy upravleniya [Recovery-predictive control systems]. Kemerovo: Izdatel'stvo Kemerovskogo Gosudarstvennogo Universiteta; 1984: 89. (In Russ.)

[8] **Avdeev V.P., Myshljaev L.P., Solov'ev V.N.** On the recovery-predictive control of technological processes. Izvestiya Vuzov. Chernaya Metallurgiya. 1978; (10):165–168. (In Russ.)

[9] **Varshavskiy P.R., Eremeev A.P.** Realization of decision search methods on the base of analogies and precedents in support systems for decision making. Vestnik Moskovskogo Energeticheskogo Instituta. 2006; (2):77–87. (In Russ.)

[10] **Germeyer Yu.B.** Igry s neprotivopolozhnymi interesami [Games with non-opposing interests]. Moscow: Nauka; 1976: 326. (In Russ.)

[11] **Dobrynin A.S., Kulakov S.M., Zimin V.V., Bondar' N.F.** On the formation of a complex of IT-provider's tools for creating the schedules of service introduction process. Nauchnoe Obozrenie. 2013; (8):92–99. (In Russ.)

[12] **Dobrynin A.S., Kulakov S.M., Koynov R.S., Grachev A.V.** Setting up of schedules in the time planning tasks. Vestnik Astrakhanskogo Gosudarstvennogo Tehnicheskogo Universiteta. Seriya: Upravlenie, Vychislitel'naya Tekhnika i Informatika. 2014; (4):103–111. Available at: `https://cyberleninka.ru/article/n/formirovanie-raspisaniy-v-zadachah-vremennogo-planirovaniya`. (In Russ.)

[13] **Dobrynin A.S., Koynov R.S., Kulakov S.M., Zimin V.V.** Svidetel'stvo o gosudarstvennoy registratsii programmy dlya EVM № 2014613280 Rossiyskaya Federatsiya. Programma postroeniya raspisaniy v proektno-protsessnoy deyatel'nosti i servisnom upravlenii [Certificate of state registration of the computer program No. 2014613280 Russian Federation. Scheduling program in design and process activities and service management; copyright holder Dobrynin A.S.] 2014; registration date 21.03.2014: 1.

Вычислительные технологии, 2022, том 27, № 4, с. 118–126. © ФИЦ ИВТ, 2022 ISSN 1560-7534
Computational Technologies, 2022, vol. 27, no. 4, pp. 118–126. © FRC ICT, 2022 eISSN 2313-691X

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

DOI:10.25743/ICT.2022.27.4.010

Составление расписаний в мультипроектных системах управления

М. В. Пургина[1], А. С. Добрынин[2,*], Р. С. Койнов[2]

[1]Новосибирский государственный университет экономики и управления, 630099, Новосибирск, Россия

[2]Сибирский государственный индустриальный университет, 654007, Новокузнецк, Россия

[*]Контактный автор: Добрынин Алексей Сергеевич, e-mail: serpentfly@mail.ru

Поступила 18 марта 2022 г., доработана 12 апреля 2022 г., принята в печать 20 апреля 2022 г.

Аннотация

Важный фактор повышения качества планирования — построение расписаний в проектной, производственной и экономической деятельности. Целесообразно использовать автоматизированные процедуры построения расписаний. Следует отметить, что некоторые теоретические и прикладные аспекты их формирования хорошо проработаны и описаны в литературе. Тем не менее в реальной практике деятельности человеческих коллективов очень распространен случай, когда проектные или производственные группы одновременно работают с несколькими проектами. Поэтому задача автоматизированного построения расписаний для мультипроектной деятельности является актуальной. В статье рассмотрены новые подходы и механизмы разработки программно-инструментальных средств автоматизированного составления расписаний для мультипроектной деятельности, которые были опробованы авторами при создании соответствующего программного обеспечения.

Ключевые слова: планирование, многопроектное планирование, орграф, мультиграф, домен, кластеризация.

Цитирование: Пургина М.В., Добрынин А.С., Койнов Р.С. Составление расписаний в мультипроектных системах управления. Вычислительные технологии. 2022; 27(4):118–126. DOI:10.25743/ICT.2022.27.4.010. (на английском)