Journal of Computational Science 75 (2024) 102203



Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs





Project indicators and flexible project structure generators

Zsolt T. Kosztyán [a,*], Gergely L. Novák [a,b]

[a] Department of Quantitative Methods, University of Pannonia, Egyetem str. 10, Veszprém, H-8200, Hungary
[b] Continental Automotive Hungary Ltd., Házgyári str. 6-8., Veszprém, H-8200, Hungary

ARTICLE INFO

Keywords:
Project generator
Structural flexibility
Project indicators
Matrix-based project structures

ABSTRACT

Project indicators are essential in characterizing project structures, structural complexity, duration, slacks, and resource demands. This information can fundamentally influence how the scheduling or resource allocation algorithm performs. In addition, most of these indicators assume a fixed project structure, while recent project management approaches, such as agile, hybrid, and extreme approaches, consider the priority of task completion and flexible precedence between tasks. By specifying the desired level of flexibility, the flexible structure generator (FSG) can transform conventional project database instances into flexible instances. This transformation allows scholars to analyze the various implemented project indicators in a flexible project environment. The generation of the specified minimal and maximal structure variants serves to bridge the gap between traditional and emerging flexible project scheduling algorithms.

1. Motivation and significance

Despite surveys showing that a significant proportion of projects can no longer be considered traditional projects with fixed structure [1], scholars have paid little attention to the study of flexible projects [2]. Project-related indicators are conventionally employed for fixed project structures and are associated with traditional project management approaches. In our study, we expand these indicators to consider flexible project plans and use the Flexible Structure Generator (FSG) to enable the exploration of flexible structures related to agile, hybrid, and extreme project management, where tasks and their dependencies may change in response to evolving requirements and uncertainties.

Regarding indicator values, Vanhoucke [3] has previously shown that structural parameters can vary across different types of real-life projects. For example, a software project enables more parallel tasks than does a construction project. In addition, [4] has also shown that (1) databases containing different simulated projects and databases containing real projects differ not only structurally, e.g., in terms of complexity and parallelity but also in their use of time and resources, and parameters have the most important role with respect to scheduling and resource allocation problems [5]; (2) including flexible structures extends the range of possible project indicators and brings the parameters of simulated projects closer to the employed range of the indicators in real projects.

The proposed software was originally implemented to support the recently published paper [4], where the main focus was on the comparison between real-life and simulated project structures. As a result, primarily single-mode and single-project structures were examined. The application calculates the most important structural, time- and resource-related indicators adapted for the matrix-based representation and is now ready to calculate indicators for multimodal completion structures where more than one alternative time/cost/resource demand is attached to each task. In addition, the application calculates more project indicators and supports multiple parallel-running projects. The software is also verified [6].

With the flexible structures generated by FSG, one can discover solutions for highlighted cases as well. The opportunities for the application of the introduced software are twofold. (1) Extending calculations of project indicators to consider flexible projects allows scholars to analyze, to classify, and to characterize fixed and flexible projects. (2) The effect of the rate of set flexibility parameters for project structures, indicators, and the applied approaches for solving scheduling problems can be examined.

The opportunities of the FSG and the calculation of project indicators enable us to specify the following new research questions:

- **RQ$_1$** What is the role of structural flexibility for project indicators?
- **RQ$_2$** How can flexible projects be classified by the employed project indicators?

The present paper provides further insights regarding the implementation and utilization of the FSG, as well as the expanded set of indicators incorporated into the tool for both research and real-world applications.

* Corresponding author.
E-mail addresses: kosztyan.zsolt@gtk.uni-pannon.hu (Z.T. Kosztyán), gergely.novak@continental.com (G.L. Novák).

https://doi.org/10.1016/j.jocs.2023.102203
Received 20 June 2023; Received in revised form 29 November 2023; Accepted 15 December 2023
Available online 18 December 2023
1877-7503/© 2023 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

This paper is outlined as follows. The current Section 1 gives the motivational background and significance of our study with a review of the relevant literature. In Section 2, we describe the software and its functionalities, including the architecture. Examples of using the software are provided in 3. In Section 4, we discuss the implications of our findings for project management. Finally, we conclude with a summary of our key contributions and avenues for future research in Section 5.

Traditional project management approaches try to predict and to minimize change [7], and they assume that the activities have a fixed order of execution [8] in the project plans. Agile and hybrid methods [9] are designed to adapt to uncertainty, e.g., with continuous reprioritization of tasks and their dependencies. Extreme project management thrives in an environment with vague goals and solutions, typically found in research and development projects [1].

Wysocki [1] showed that these flexible approaches are applied in 80% of all IT projects, and [10–15] also showed that these methods have already increased in almost all kinds of projects. The project scheduling problems in flexible projects are still little studied.

In its basic form, the extensively researched resource-constrained project scheduling problem (RCPSP) involves scheduling fixed activities considering precedence and resource constraints to optimize objectives. Several extensions were established (for an overview, we refer to Hartmann and Briskorn [16] and Sanchez et al. [17]). We briefly review the literature related to alternative or flexible structures. Capek et al. [18] extend the basic RCPSP with alternative process plans that may vary in terms of resource requirements, processing times, the number and types of activities, and the precedence relations between them. Kellenbrink and Helber [19] discuss mandatory, optional, and caused activities in the selection problem of flexible project structures (RCPSP-PS). Cajzek and Klanšek [20] investigated the cost optimization of project schedules under constrained resources and alternative production processes. Cajzek and Klanšek [20] investigated the cost optimization of project schedules under constrained resources and alternative production processes. Tao and Dong [21] deal with AND/OR networks as alternative activity chains (RCPSP-AC) and proposes the multiproject scheduling variant of the problem (RCMPSP-AC). This extension was also combined with time-related flexibility (ACTF) by Hauder et al. [22]. Servranckx and Vanhoucke [23] define different types of relations between alternative subgraphs based on the concepts of nested and linked alternatives in RCPSP-AS. Servranckx et al. [24] further studied RCPSP extensions for more complex variants of alternative subgraphs.

Numerous indicators are introduced in the literature with an ongoing effort to develop new indicators. A general overview is given by Vanhoucke in [25]. For multiple projects, Browning and Yassine [26] and Van Eynde and Vanhoucke [27,28] provide an overview. Recently, Vanhoucke and Coelho [29] and Van Eynde et al. [30] extended their work including indicators for both generation and classification of projects.

Flexible project management approaches require flexible project plans, allowing the possibility of project restructuring, task reprioritization, or both. Nevertheless, most of these methods assume a fixed [31] logic plan or deal with only a limited number of scheduling alternatives [18,19,21–23,32], leaving a notable gap that requires attention.

The proposed software is directly used in the study of Kosztyán et al. [4], where a set of original and modified flexible project structures are compared by using their calculated project indicators. Both the calculation of project indicators and the proposed FSG are based on a Unified Matrix-based Project-planning model (UMP) model, which specifies two mandatory domains, such as Logic Domain (LD) and Time Domain (TD), and four supplementary domains, such as Cost Domain (CD), Quality Domain (QD), Nonrenewable Resource Domain (ND), and Renewable Resource Domain (RD), or submatrices. When calculating project indicators and generating flexible project structures, we use the LD, TD, and RD domains only.

TD is an n by w matrix domain (submatrix) of positive real values for durations, where n is the number of tasks and w is the number of completion modes. In the case of $w = 1$, the project plan is called single-modal, while in the case of $w > 1$, the project plan is called a multimodal project plan.

RD is an n by $w \cdot \rho$ domain for renewable resources of each task, where ρ is the number of renewable resources.

LD is a mandatory domain that specifies an individual logic structure (i.e., single) or, in a multiproject environment, a set of individual projects. In the case of fixed project structures, LD is binary, where diagonal values are always one, and the binary outdiagonal values specify the adjacency matrix of the given project structure. Nevertheless, the proposed UMP enables (1) $0 < [LD]_{ii} < 1$ relative priority values of the task completion in the diagonal and (2) $0 < [LD]_{ij} < 1$ flexible dependency values in the outdiagonal. Importantly, all existing calculations of project indicators require a fixed project structure. In contrast, flexible projects, such as agile, hybrid, or extreme projects, may contain supplementary tasks and flexible dependencies. In addition, several agile approaches [33] limit the number of mandatory tasks in a project. As these flexible approaches have already taken the lead in many areas, not just in IT, we cannot ignore these priorities and flexible dependencies when modeling a project. One way is to specify all possible project plans, where after the decision, all cells must contain either one or zero.

Kosztyán [34] showed that a flexible diagonal value representing a supplementary task becomes zero, representing the decision to exclude or postpone the given task from the project. In addition, all dependencies and all demands of the given task are excluded (see Fig. 1). Excluding a task can reduce the project duration, project cost, and availability of resources. However, it reduces the project scope and may reduce the quality of the project [2]. Excluding a flexible dependency, or formally, a less than one value in the outdiagonal goes to zero, increases the number of parallel tasks to decrease the project duration. Nevertheless, this approach increases the resource demands (see Fig. 1).

In some cases, it may be necessary to specify all possible project structures [34]. An algorithm to define these structures has been proposed [34], and a software implementation has been developed [35]; however, the number of possible cases is 2^n. For this reason, [4] proposed defining structures in the analysis that assign the minimum and maximum time and resource consumption. [4] called these structures minimal, maximal, minimax, and maximin structures. These four structures can be used to specify the minimum/maximum time and resource consumption for a flexible project. Thus, for flexible projects, it is usually sufficient to add these four structures to the project plans. Fig. 1 shows a flexible project plan where there are two mandatory tasks (task A and task C) and one supplementary task (task B) is specified, while all dependencies are flexible.

Fig. 1 shows that the maximal (minimal) structure stores (neglects) all supplementary tasks and dependencies, giving the project plan the longest (shortest) duration, while the maximin (minimax) structure stores (neglects) all supplementary tasks but neglects (stores) all flexible dependencies, giving the greatest (lowest) resource-demanded project plan.

The FSG enables specifying a flexible project structure from existing traditional project structures. The specified *flexibility parameter,* (fp), determines the mean of the rate of flexible dependencies ($f\%$) and the rate of supplementary tasks (s%) (see Fig. 2). Following the conventional practice in agile projects to maintain 60% of tasks as a minimum usable subset (MUST), the decision was made to constrain the flexibility parameter to the range 0%–40%.

The left side of Fig. 2 shows an original project plan, where five tasks (from task A to task E) are specified. The given flexibility parameter (fp) of FSG may specify different kinds of project structures. FSG can specify project structures, where there are only flexible dependencies with no supplementary tasks, but it can specify project

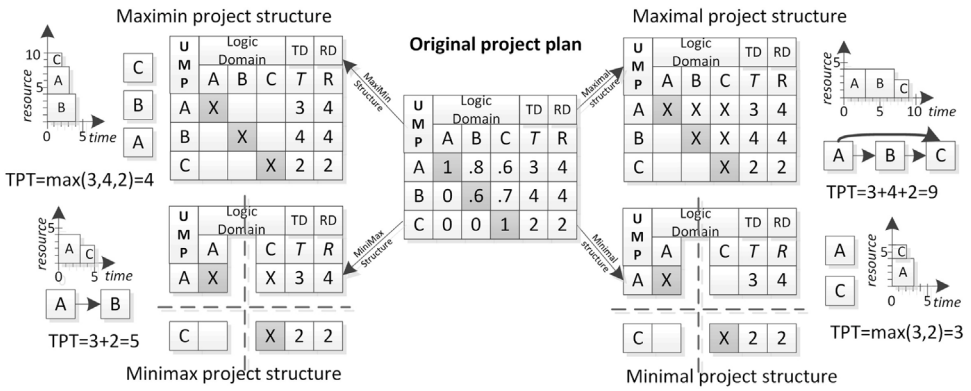


Fig. 1. Minimal, maximal, minimax, and maximin structures of the flexible project plan.
Source: [4]

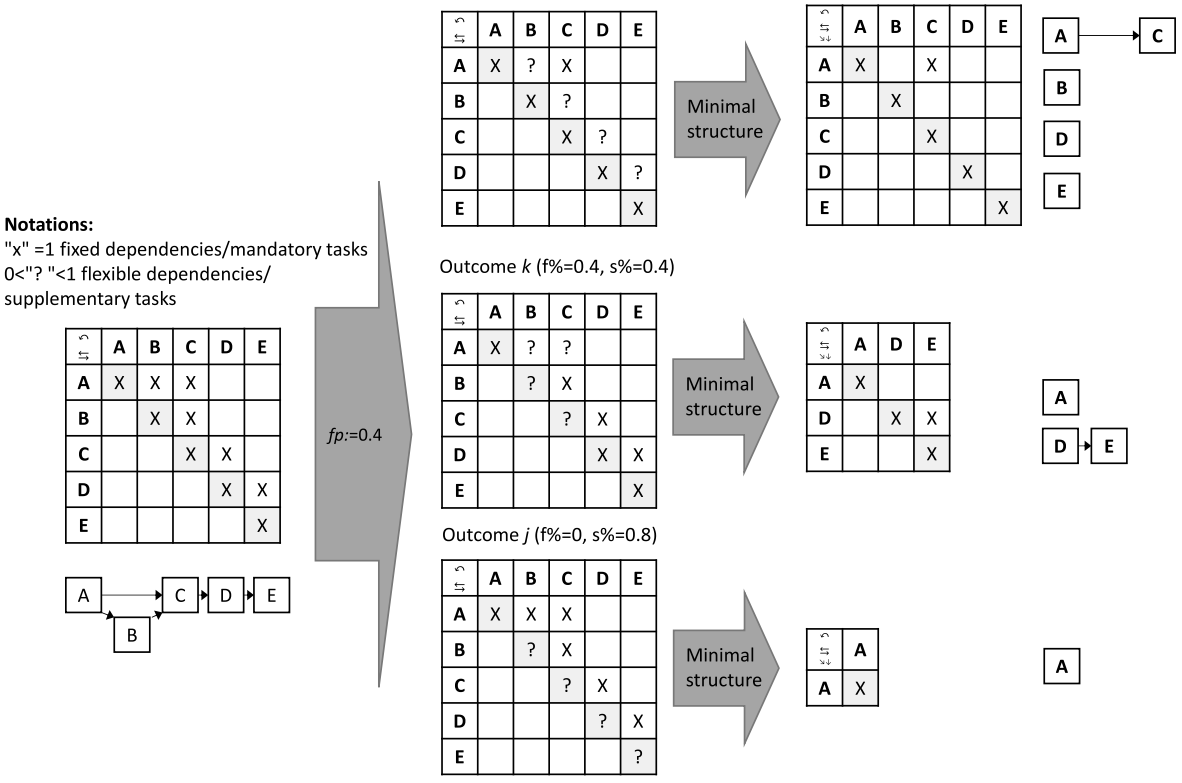


Fig. 2. Example of generating flexibility.

structures with several supplementary tasks but no flexible dependencies. From a flexible project structure, minimal (shortest duration), maximal (longest duration, and in this case the original), minimax (lowest resource-demanded), and maximin (lowest resource-demanded) project structures can be specified.

Fig. 2 shows how to make a flexible project plan from an existing fixed project plan. Kosztyán et al. [4,36] implemented a parser tool to collect 12 existing and most frequently used single and 4 multiproject databases. After using the developed FSG, these project structures can be made flexible. Kosztyán [35] offered a set of algorithms for solving flexible project scheduling problems. However, using the possible structures, in particular, the proposed minimal, maximal minimax, and maximin structures, the newly designed project structures can also be evaluated with traditional solvers. In addition, project indicators, which can be used only on fixed structures, can be calculated for all possible or highlighted flexible project structures.

2. Software description

As Fig. 2 shows, FSG generates a flexible structure from the original, fixed project structure.

An original logic structure of a project yields an activity-on-node network, denoted as $G = (N, \mathcal{A})$ directed graph, where $N = \{A_1, \ldots, A_n\}$ (A_i is often shortened to i) is the set of nodes (i.e., tasks), and $\mathcal{A} \subset N \times N$ is the set of arcs (i.e., dependencies). $n = |N|$ is the number of tasks, and $|\mathcal{A}|$ is the number of dependencies. Furthermore, the matrix representation of the logic plan is the LD of the UMP matrix, where $\mathbf{LD} \in \{0,1\}^{n \times n}$, for each $i \leq n$ $[\mathbf{LD}]_{ii} = 1$, and for each $i \neq j$, we have $(A_i, A_j) \in \mathcal{A}$ if and only if $[\mathbf{LD}]_{i,j} = 1$ (otherwise $[\mathbf{LD}]_{i,j} = 0$).

Since available project databases do not specifically address flexible structures or are limited to the level of alternative activity groups rather than individual activities, our initial step involves generating these flexible project structures. Let $\mathbf{LD} \in \{0,1\}^{n \times n}$ and $\mathbf{LD}' \in [0,1]^{n \times n}$ the

Table 1
Applied indicators.

Name	Short description	Adapted from	Besides single mode single project, applicable for:		
			single mode multiproject	multi-mode single project	multi-mode multiproject
Structural indicators					
I1	number of nodes (i.e., tasks)	[37,38]	X	X	X
I2	serial or parallel structure	[37,38]	X	X	X
I3	task distribution	[37,38]	X	X	X
I4	rate of short arcs	[37,38]	X	X	X
I5	rate of long arcs	[37,38]	X	X	X
I6	topological float	[37,38]	X	X	X
T-DENSITY	total activity density	[39]	X	X	X
XDENSITY	average activity density	[39]	X	X	X
C	network complexity	[40]	X	X	X
CNC	coefficient of network complexity	[41]	X	X	X
OS	order strength	[42]	X	X	X
Time related indicators					
TPT	total project time	[39]	X	X	X
XDUR	average activity duration	[39]	X	X	X
VA-DUR	variance in activity duration	[39]	X	X	X
PCTSLACK	percent of activities possessing positive total slack	[39]	X		
XSLACK	average total slack per activity	[39]	X		
TOTSLACK-R	total slack ratio	[39]	X		
XSLACK-R	average slack ratio	[39]	X		
PCTFREESLACK	percent of activities possessing positive free slack	[39]	X		
XFREESLACK	average free slack per activity	[39]	X		
Renewable resource-related indicators					
RF	resource factor (i.e., density of **RD**)	[43]	X		
$PCTR_j$	percent of activities that require resource type j	[39]	X		
RU	resource use	[44]	X		
$DMND_j$	the average demand resource type j	[39]	X		
RC	resource constraint	[39]	X		
RS	resource strength	[43]	X		
UTIL	utilization of resources	[39]	X		
$TCON_j$	constraints of resource j over time	[39]	X		
$OFACT_j$	obstruction of resource j	[39]	X		
$UFACT_j$	underutilization of resource j	[39]	X		
UTIL	utilization of resources	[39]	X		
NARLF'	resource distribution front/backloaded	[27]	X		
Gini	equality distribution of resource demands	[27]	X		
Distributional indicators					
$\alpha^{dist}(...)$	variation of multiple indicator values	[45]	X		

modified logic domain as follows:

$$l'_{ij} = [\mathbf{LD'}]_{ij} := \begin{cases} u_{ij} \text{ if } l_{ij} = 1 \text{ and } v_{ij} \le fp \\ l_{ij} \text{ otherwise} \end{cases} \quad (1)$$

where $l_{ij} = [\mathbf{LD}]_{ij}$ and $u_{ij}, v_{ij} \sim U[0,1]$ are uniformly distributed random probability variables (r.v.) and $fp \in [0,1]$ is a fixed flexibility parameter set for computer runs. The goal is to have the ratio of the number of (supplementary tasks + flexible dependencies) with respect to the total number of LD elements be approximately fp, which is ensured by "$v_{ij} \le fp$". The *weights* of these flexible objects are set by the r.v. u_{ij}. Any other distribution can be chosen instead of a uniform distribution. We note that $\mathbf{LD'}$ already contains flexible dependencies ($i \ne j$) and supplementary tasks ($i = j$). However, complexity, time-related, and resource-related indicators address only fixed project structures.

The modified logic domain is used to specify only the minimal, maximal, minimax, and maximin structures, as follows:

$$l_{ij}^{\min} = \left\lfloor l'_{ij} \right\rfloor, \quad (2)$$

$$l_{ij}^{\max} = \left\lceil l'_{ij} \right\rceil, \quad (3)$$

$$l_{ij}^{\text{minimax}} = \begin{cases} \left\lfloor l'_{ij} \right\rfloor \text{ if } i = j \\ \left\lceil l'_{ij} \right\rceil \text{ if } i \ne j \text{ and } \left\lfloor l'_{ii} \right\rfloor = \left\lfloor l'_{jj} \right\rfloor = 1 \\ 0 \text{ otherwise,} \end{cases} \quad (4)$$

$$l_{ij}^{\text{maximin}} = \begin{cases} \left\lceil l'_{ij} \right\rceil \text{ if } i = j \\ \left\lfloor l'_{ij} \right\rfloor \text{ if } i \ne j, \end{cases} \quad (5)$$

where $l_{ij}^{\min}, l_{ij}^{\max}, l_{ij}^{\text{minimax}}, l_{ij}^{\text{maximin}}$ are the (i,j) cells of the logic domains of the minimal, maximal, minimax and maximin structures, respectively, with $(i,j) = \{1,2,\dots,n\}$ (see Fig. 1).[1]

Minimal, maximal, minimax and maximin structures can also be involved in the databases. Of course, any other possible implementation structure can be specified by rounding up or down the cell values of the logic domain. However, in the case of single completion modes and the early schedule, the minimal structure provides the minimal task duration and minimal project budget, while a maximal structure provides the highest project score (widest project scope). In addition, the minimax (maximin) structure provides the highest (lowest) renewable resource demands.

For comparability reasons, real and artificial databases can be examined only for single projects in Kosztyán et al. [4]. However, both the FSG and the project indicators calculation work on multiple, parallel-running projects. Individual and multiple projects can be compared by calculating the average indicators per project. In addition, to compare simulated and real-life datasets, Kosztyán et al. [4] considered only

[1] The $\lceil\cdot\rceil$ ($\lfloor\cdot\rfloor$) operators denote the rounding up (rounding down) of real numbers.

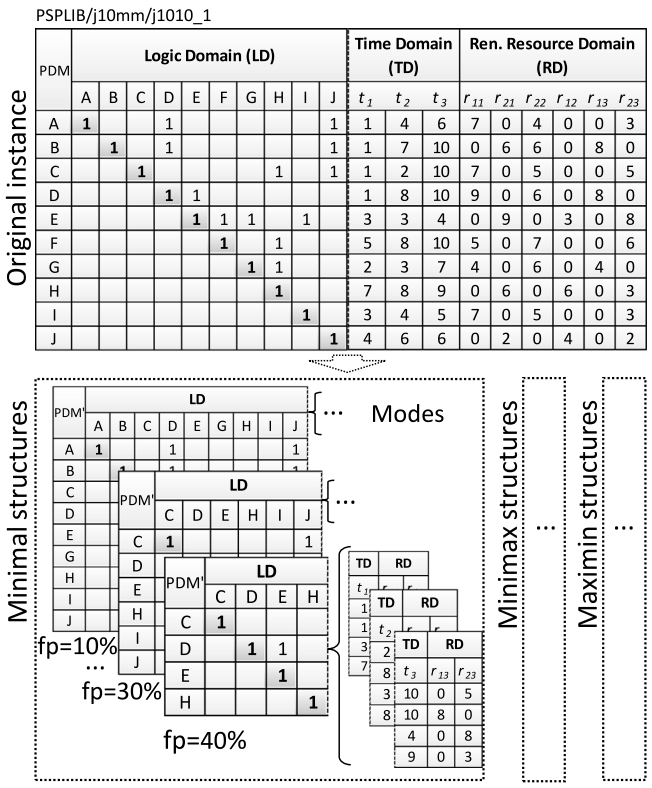


Fig. 3. Minimal, maximal, minimax, and maximin structures generated on different flexibility levels and for each completion mode (real example).

single-mode project structures. However, the implemented calculators are ready to compute the indicators of multimode project structures.

2.1. Software architecture

The developed FSG can set the level of structural flexibility for the project. Structural flexibility describes the rates of the flexible dependencies and supplementary tasks.

First, we set

$$S\text{-SET} := \{l'_{ii} | l'_{ii} \sim P(0,1), 0 < l'_{ii}\} \tag{6}$$

$$F\text{-SET} := \{l'_{ij} | l'_{ij} \sim P(0,1), i \neq j, 0 < l'_{ij}\} \tag{7}$$

where $P(0,1)$ is an arbitrary continuous distribution on the open interval (0,1). Then, let

fp = *flexibility parameter*, which shows the total number of flexible dependencies and supplementary tasks across all tasks and dependencies as follows:

$$fp = \frac{|F\text{-SET} \cup S\text{-SET}|}{n(n+1)/2} \tag{8}$$

(fp is set before the computer runs as the *approximate* ratio of flexible objects in Eq. (1), while Eq. (8) calculates the *exact* value of this ratio. Hereafter, we use this latter value for fp.)

$f\% =$ *rate of flexible dependencies* calculates the sum of flexible dependencies across all dependencies as follows:

$$f\% = \frac{|F\text{-SET}|}{n(n-1)/2} \tag{9}$$

$s\% =$ *rate of supplementary tasks* calculates the sum of supplementary (prioritized) tasks across all tasks as follows:

$$s\% = \frac{|S\text{-SET}|}{n} \tag{10}$$

We observe that $fp = \frac{a+b}{c+d}$ if $f\% = \frac{a}{c}$ and $s\% = \frac{b}{d}$, which has the notation $\frac{a}{c} \boxplus \frac{b}{d} = \frac{a+b}{c+d}$. For a, b, c, d positive (in this case), $\frac{a}{c} \boxplus \frac{b}{d}$ is always between $\frac{a}{c}$ and $\frac{b}{d}$; thus, fp is always between $f\%$ and $s\%$, and all three depend only on Eq. (1).

Table 1 summarizes the applied indicators on the project plans.

Three indicator types are examined. The first group is *structural indicators*, such as complexity and flexibility indicators, which consider only the logic domain of the project domain matrices. The second group of indicators consists of *demand indicators*, which consider other domains, such as time domains (time-related indicators) and renewable resource domains (such as renewable resource-related indicators). The last group is formed by a single indicator, which uses other indicators to show their distributional properties rather than provide any characteristics on its own.

The software architecture consists of the related indicator function groups logically organized within separate files. This structure simplifies the maintenance of the different indicator types described above.

The tool can calculate a single indicator for a selected instance without flexibility by using the *'indicators_<indicator-type>.m'* file. To generate all flexible structure instances and to calculate and store indicator values for all datasets, the *'indicators_all.m'* file is provided.

2.2. Software functionalities

The software has two main functionalities: calculating values for user-selected indicators and for database instances (without generating flexible instances) and generating flexible project structures, including the calculation of their structural, time-related, and resource-related indicators. The tool reads compatible files from the data (input) folder, a place where the already exported matrix-based instances of the project-parsers tool should be copied in advance. Batch processing of all datasets and indicators is also supported.

When flexible structures are also generated, the flexible structure generator also stores the newly created Project Domain Matrix (PDM)s and other necessary variables in the result folder as MAT containers. This feature is added for reproducibility when flexible tasks and dependencies might vary due to random (or arbitrary) selection. Nevertheless, the pseudorandom seed is fixed at the beginning of the generation process by default. In the case of large datasets, the user can optionally abort and resume the processing as the software moves the already processed files to the *data_done* folder.

The results are saved to a comma-separated value (CSV) format in the *'results-<current-date>.csv'* file. The results CSV file has a predefined header format, which contains information in addition to project indicator values, such as filename, database, type (maximal, minimal), flexibility parameter, and mode number. Indicator values for multiple modes are calculated as separate entries in the results file for each mode, which enables their individual analysis. For multiple project data structures, both average indicator values and a list of individual values are given for traceability, verification, and reuse.

The software is designed to support the easy extension of the already-supported indicator list. Unit tests are provided for each indicator with a cross-check of results from the literature or with test and actual instance examples.

3. Illustrative examples

To calculate specific indicator values from existing matrix-based instances, the following example can be used.

Input: *load('test_data/j301_10_NTP.mat', 'PDM', 'num_modes')*, where the first argument is the location of the instance, the second argument is the PDM variable to be loaded, and the third argument is the variable containing the number of modes.

Usage: $[NSLACK, PCTSLACK] = indicator_slack(PDM, num_modes)$, where the slack-related indicators are selected (or ignored by

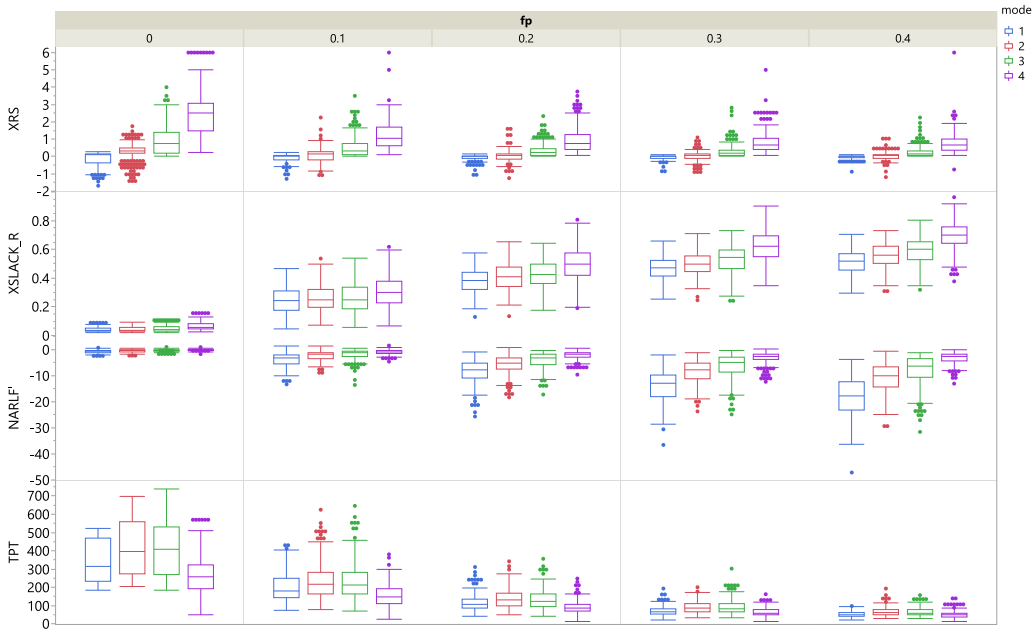


Fig. 4. Demand-related indicators of each completion mode at different flexibility levels (Boctor dataset [46]).

using ~) on the left, and the function is called on the right with the necessary input variables as arguments.

Output: $[NSLACK, PCTSLACK]$ variables (array) contain the calculated indicator values, e.g., NSLACK = 22, PCTSLACK = 0.73.

To calculate indicator instances, including the generation and storage of all flexible structures and their results, the following example can be used.

Input: all subfolders in *../data/* containing MAT files (i.e., *.mat) having the matrix-based instances to process.

Usage: *indicators_all*

Output: The output files are stored in the folder *../results/*, including the MAT files of flexible instances with a filename pattern ''<instance-name>_fp<#>_mode<#>.mat', and 'results- <current-date >.csv' as the results file containing all calculated indicator values. The number after 'fp' shows the flexibility parameter 0%–40% of the instance.

As illustrated in an actual example from the PSPLIB dataset, Fig. 3 depicts a sequence of steps unfolding for every instance of the original dataset: the generation of four specified structures, repeated for each level of flexibility parameter fp, and for each individual completion mode, is generated. The only exception is the maximal structure, where the structure remains the same due to $fp = 0$; here, only the modes are selected and added. This procedure separates the different completion modes for processing. Each of the generated structures is saved individually during the process, and the associated indicator values are appended to a new row in the results file. Subsequent actions such as consolidation, visualization, statistical analysis, and the potential for further processing of the stored instances are possible using external tools.

To run a corresponding unit test suite for an indicator type (group):

Example. *runtests('indicators_resource_test')*

Note: ../test_data/folder contains necessary input test files for the provided unit tests.

Fig. 4 presents an example with various selected indicators to examine demands associated with different modes. The values of these indicators vary as different levels of flexibility are applied. These selected indicators cover each demand category, such as duration, slack, and renewable resources. Notably, the total project time (TPT) decreases with increased flexibility, reaching its minimum in mode four. The normalized average resource load factor (NARLF') indicates a shift towards more front-loaded resources in the project as flexibility increases. Additionally, the average slack ratio (XSLACK_R) increases with both higher mode numbers and flexibility, while the average resource strength (XRS) rises with a higher mode number but decreases with increased flexibility. For a detailed description of each indicator, we refer to Appendix A.

4. Impact

Although both the FSG and the calculators of project indicators can be applied alone, these software applications are members of the series of matrix-based flexible project management software. The recently published Matrix-based Flexible Project Planning (MFPP) [35] MATLAB package contains several project scheduling algorithms for flexible projects. Kosztyán et al. [4] offer parsers to collect the most frequently used simulated and real-life project databases.

Table 1 shows that the characterization of the project structure and demands has several indicators. However, flexibility has no indicators, and quality and cost demands have only a few indicators. None of the indicators are interval indicators. This result demonstrates that the result of each indicator is a scalar or, in the case of multimode completions, a vector. However, in the case of flexible projects, several possible projects have different project demands; therefore, the indicators should be specified as an interval or a distribution.

To ensure the comparability of simulated and real-life projects, $\mathbf{RQ}_1$ investigated only single-mode projects [4]. However, most project scheduling algorithms and most simulated project structures consider multiple completion modes. Nevertheless, the software applications can calculate the multimodal versions of several indicators (see Table 1). This study fills a gap in several respects. Since the role of flexible projects is growing in actual practice [1], it is important to know how the introduction of flexible, such as agile, extreme approaches, may

affect the project characteristics, such as the complexity of the projects, duration slacks, and resource usage. Since the various kinds of projects enable different rates of flexibility, this research should be analyzed in different kinds of settings of the flexibility parameters. Although there is an existing solver for flexible projects [35], the role of project indicators in the efficiency of scheduling flexible projects has still not been studied. We calculate possible or highlighted (minimal, maximal, minimax, and maximin) project structures of a flexible project structure, making the opportunity to employ existing traditional scheduling algorithms.

Vanhoucke [3] showed that different fixed project structures have different project parameters. This classification is still missing for flexible projects, and **RQ**$_2$ is still unanswered. However, our applications provide an opportunity for scholars to study flexible project structures or to convert fixed projects into flexible ones.

The implemented calculators for project indicators and FSG are freely available. They can be obtained from MATLAB Central.[2] The developer version can be downloaded from GitHub.[3] The verified capsule is published in the Code Ocean's site.[4] Solvers for matrix-based project planning can also be downloaded from MATLAB Central.[5]

The software license follows an MIT license. Therefore, all software is freely available for scholars and for commercial use. We encourage the community to introduce new indicators, share insights, and collectively advance the understanding of flexible project structures.

5. Conclusions

In this study, we propose two software packages. The first, Flexible Structure Generator (FSG), makes a link between traditional and flexible, such as agile, hybrid, or extreme project planning. After specifying the rate of flexibility, FSG can convert fixed project structures into flexible ones. In this way, the transition to flexible project management approaches can also be modeled. From a flexible project plan, FSG specifies highlighted (minimal, maximal) structures to enable the use of existing project scheduling and resource allocation algorithms on the specified project structures. The second software package contains project indicators that characterize the complexity and the time-related and resource-related demands of the projects. Connecting this to the FSG, the flexible projects can already be analyzed and classified.

Nomenclature

Latin symbols

$a_i(T)$ scheduled execution time interval of task a_i

a_i task i

D maximal number of short arcs

EF_i, LF_i early and late finish for task i

ES_i, LS_i early and late start to task i

$f\%$ rate of flexible dependencies

fp flexibility parameter, the ratio of flexible dependencies and prioritized tasks to all tasks and dependencies

FS_i free slack of task i

k number of task completion modes

$L(i_1, i_2)$ length of an arc between tasks i_1 and i_2, i.e., the difference between their progressive level numbers

$\overline{L}$ longest (critical) path

$l_{ij} = [\mathbf{LD}]_{ij}$ element of the logic domain, task occurrence if $i = j$, and arc that represents the precedence relation between tasks i and $j \neq i$ (in this case, $l_{ij} = 1$ means task i precedes task j)

n number of tasks

n'_L number of arcs with length L

m maximal number of progressive levels

$\overline{P}$ task path (sequence)

P_i set of immediate predecessors of task i

r_{ij} demand of task i for renewable resource type j

$r_{ij}(\tau)$ demand of task i for renewable resource type j at time τ

$s\%$ rate of supplementary tasks

S_i set of immediate successors of task i

t_i duration of task a_i

TS_i total slack of task i

w_i width of progressive level i, $i = 1, \dots, m$

Greek symbols

α_j availability of renewable resource type j

α_w total absolute deviation from the average width

η number of types of nonrenewable resources

ρ number of types of renewable resources

Calligraphic symbols

$\mathcal{A}$ set of arcs (dependencies)

$|\mathcal{A}|$ number of dependencies in a project structure

S project structure, set of (to-be-) realized tasks

$\overline{S}$ project schedule of project structure S

CRediT authorship contribution statement

Zsolt T. Kosztyán: Conceptualization, Data curation, Methodology, Writing – original draft, Writing – review & editing. **Gergely L. Novák:** Conceptualization, Data curation, Methodology, Resources, Software, Writing – review & editing, Writing – original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The link of shared data and indicators are incorporated in the manuscript.

Acknowledgments

This work has been implemented by the TKP2021-NVA-10 project with support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the 2021 Thematic Excellence Programme funding scheme. Gergely L. Novák's cooperation was supported by the Doctoral Student Scholarship Program of the Co-Operative Doctoral Program of the Ministry of Innovation and Technology financed by the National Research, Development, and Innovation Fund. Zsolt T. Kosztyán's research contribution supported by Project no. K 142395

[2] https://www.mathworks.com/matlabcentral/fileexchange/124045-project-indicators-and-flexible-project-structure-generators

[3] https://github.com/novakge/project-indicators

[4] https://codeocean.com/capsule/9888650/tree/v1

[5] https://www.mathworks.com/matlabcentral/fileexchange/87639-matrix-based-flexible-project-planning

has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the K_22 "OTKA" funding scheme.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jocs.2023.102203.

References

[1] R.K. Wysocki, Effective Project Management, eight ed., John Wiley & Sons, Nashville, TN, 2019.

[2] Z.T. Kosztyán, G. Novák, R. Jakab, C. Hegedűs, Survive IT! survival analysis of IT project planning approaches, Oper. Res. Perspect. 7 (2020) 1–51.

[3] M. Vanhoucke, Measuring the efficiency of project control using fictitious and empirical project data, Int. J. Project Manage. 30 (2) (2012) 252–263, http://dx.doi.org/10.1016/j.ijproman.2011.05.006, URL http://www.sciencedirect.com/science/article/pii/S026378631100069X.

[4] Z.T. Kosztyán, G. Novák, R. Jakab, I. Szalkai, C. Hegedűs, A matrix-based flexible project planning library and indicators, Expert Syst. Appl. 216 (2023) 119472, http://dx.doi.org/10.1016/j.eswa.2022.119472.

[5] P. Brucker, A. Drexl, R. Möhring, K. Neumann, E. Pesch, Resource-constrained project scheduling: Notation, classification, models, and methods, European J. Oper. Res. 112 (1) (1999) 3–41, http://dx.doi.org/10.1016/S0377-2217(98)00204-5.

[6] Z.T. Kosztyán, G.L. Novák, Project indicators and flexibility generator for matrix-based datasets, 2022, http://dx.doi.org/10.24433/CO.5304543.v1, https://www.codeocean.com/.

[7] D. Ciric, B. Lalic, D. Gracanin, N. Tasic, M. Delic, N. Medic, Agile vs. Traditional approach in project management: Strategies, challenges and reasons to introduce agile, Procedia Manuf. 39 (2019) 1407–1414.

[8] R. Pellerin, N. Perrier, A review of methods, techniques and tools for project planning and control, Int. J. Prod. Res. 57 (7) (2019) 2160–2178.

[9] D. Ciric Lalic, B. Lalic, M. Delić, D. Gracanin, D. Stefanovic, How does the project management approach impact project success? From traditional to agile, Int. J. Manag. Projects Business 15 (3) (2022) 494–521.

[10] A. Stare, Agile project management in product development projects, Procedia - Soc. Behav. Sci. 119 (2014) 295–304, http://dx.doi.org/10.1016/j.sbspro.2014.03.034.

[11] D. Ciric, B. Lalic, D. Gracanin, N. Tasic, M. Delic, N. Medic, Agile vs. Traditional approach in project management: Strategies, challenges and reasons to introduce agile, Procedia Manuf. 39 (2019) 1407–1414, http://dx.doi.org/10.1016/j.promfg.2020.01.314.

[12] D. Özkan, A. Mishra, Agile project management tools: A brief comparative view, Cybern. Inf. Technol. 19 (4) (2019) 17–25, http://dx.doi.org/10.2478/cait-2019-0033.

[13] E.S. Hidalgo, Adapting the scrum framework for agile project management in science: case study of a distributed research initiative, Heliyon 5 (3) (2019) e01447, http://dx.doi.org/10.1016/j.heliyon.2019.e01447.

[14] A. Yasaman, A. Nazari, M.R. Hafezi, S.A.H. Maghool, Prioritizing agile project management strategies as a change management tool in construction projects, Int. J. Constru. Manag. 22 (4) (2022) 678–689, http://dx.doi.org/10.1080/15623599.2019.1644757.

[15] T. Dingsøyr, S. Nerur, V. Balijepally, N.B. Moe, A decade of agile methodologies: Toward explaining agile software development, J. Syst. Softw. 85 (6) (2012) 1213–1221, http://dx.doi.org/10.1016/j.jss.2012.02.033.

[16] S. Hartmann, D. Briskorn, An updated survey of variants and extensions of the resource-constrained project scheduling problem, European J. Oper. Res. 297 (1) (2022) 1–14, http://dx.doi.org/10.1016/j.ejor.2021.05.004, URL https://www.sciencedirect.com/science/article/pii/S0377221721003982.

[17] M. Gomez Sanchez, E. Lalla-Ruiz, A. Fernandez Gil, C. Castro, S. Voa, Resource-constrained multiproject scheduling problem: A survey, European J. Oper. Res. 309 (3) (2023) 958–976, http://dx.doi.org/10.1016/j.ejor.2022.09.033, URL https://www.sciencedirect.com/science/article/pii/S0377221722007639.

[18] R. Čapek, P. Šůcha, Z. Hanzálek, Production scheduling with alternative process plans, European J. Oper. Res. 217 (2) (2012) 300–311, http://dx.doi.org/10.1016/j.ejor.2011.09.018.

[19] C. Kellenbrink, S. Helber, Scheduling resource-constrained projects with a flexible project structure, European J. Oper. Res. 246 (2) (2015) 379–391, http://dx.doi.org/10.1016/j.ejor.2015.05.003.

[20] R. Cajzek, U. Klanšek, Cost optimization of project schedules under constrained resources and alternative production processes by mixed-integer nonlinear programming, Eng. Constr. Archit. Manag. 26 (10) (2019) 2474–2508, http://dx.doi.org/10.1108/ECAM-01-2019-0013, URL https://www.ingentaconnect.com/content/mcb/286/2019/00000026/00000010/art00016.

[21] S. Tao, Z.S. Dong, Multimode resource-constrained project scheduling problem with alternative project structures, Comput. Ind. Eng. 125 (2018) 333–347, http://dx.doi.org/10.1016/j.cie.2018.08.027.

[22] V.A. Hauder, A. Beham, S. Raggl, S.N. Parragh, M. Affenzeller, Resource-constrained multiproject scheduling with activity and time flexibility, Comput. Ind. Eng. 150 (2020) 106857, http://dx.doi.org/10.1016/j.cie.2020.106857.

[23] T. Servranckx, M. Vanhoucke, Strategies for project scheduling with alternative subgraphs under uncertainty: similar and dissimilar sets of schedules, European J. Oper. Res. 279 (1) (2019) 38–53, http://dx.doi.org/10.1016/j.ejor.2019.05.023.

[24] J.C. Tom Servranckx, M. Vanhoucke, Various extensions in resource-constrained project scheduling with alternative subgraphs, Int. J. Prod. Res. 60 (11) (2022) 3501–3520, http://dx.doi.org/10.1080/00207543.2021.1924411.

[25] M. Vanhoucke, J. Coelho, J. Batselier, An overview of project data for integrated project management and control, J. Mod. Proj. Manag. 3 (3) (2016) 6–21.

[26] T.R. Browning, A.A. Yassine, Resource-constrained multiproject scheduling: Priority rule performance revisited, Int. J. Prod. Econ. 126 (2) (2010) 212–228, http://dx.doi.org/10.1016/j.ijpe.2010.03.009.

[27] R. Van Eynde, M. Vanhoucke, Resource-constrained multiproject scheduling: benchmark datasets and decoupled scheduling, J. Sched. 23 (3) (2020) 301–325, http://dx.doi.org/10.1007/s10951-020-00651-w.

[28] R. Van Eynde, M. Vanhoucke, New summary measures and datasets for the multiproject scheduling problem, European J. Oper. Res. 299 (3) (2022) 853–868, http://dx.doi.org/10.1016/j.ejor.2021.10.006, URL https://www.sciencedirect.com/science/article/pii/S0377221721008511.

[29] M. Vanhoucke, J. Coelho, An analysis of network and resource indicators for resource-constrained project scheduling problem instances, Comput. Oper. Res. 132 (2021) 105260, http://dx.doi.org/10.1016/j.cor.2021.105260, URL https://www.sciencedirect.com/science/article/pii/S0305054821000526.

[30] R. Van Eynde, M. Vanhoucke, J. Coelho, On summary measures for the resource-constrained project scheduling problem, Ann. Oper. Res. (2023) 1–33.

[31] D.M. Franco-Duran, J.M.d.l. Garza, Review of resource-constrained scheduling algorithms, J. Constr. Eng. Manag. 145 (11) (2019) 03119006, http://dx.doi.org/10.1061/(ASCE)CO.1943-7862.0001698.

[32] S. Creemers, B.D. Reyck, R. Leus, Project planning with alternative technologies in uncertain environments, European J. Oper. Res. 242 (2) (2015) 465–476, http://dx.doi.org/10.1016/j.ejor.2014.11.014.

[33] J. Stapleton, DSDM, Dynamic Systems Development Method: The Method in Practice, Cambridge University Press, 1997.

[34] Z.T. Kosztyán, Exact algorithm for matrix-based project planning problems, Expert Syst. Appl. 42 (9) (2015) 4460–4473, http://dx.doi.org/10.1016/j.eswa.2015.01.066.

[35] Z.T. Kosztyán, MFPP: Matrix-based flexible project planning, SoftwareX 17 (2022) 100973, http://dx.doi.org/10.1016/j.softx.2022.100973.

[36] Z.T. Kosztyán, G.L. Novák, Project dataset parsers to matrix-based formats, 2022, http://dx.doi.org/10.24433/CO.0837444.v1, https://www.codeocean.com/.

[37] L.V. Tavares, Advanced Models for Project Management, Vol. 16, Springer Science & Business Media, 1999, http://dx.doi.org/10.1007/978-1-4419-8626-9.

[38] M. Vanhoucke, J. Coelho, D. Debels, B. Maenhout, L.V. Tavares, An evaluation of the adequacy of project network generators with systematically sampled networks, European J. Oper. Res. 187 (2) (2008) 511–524, http://dx.doi.org/10.1016/j.ejor.2007.03.032.

[39] J.H. Patterson, Project scheduling: The effects of problem structure on heuristic performance, Nav. Res. Logist. Q. 23 (1) (1976) 95–123, http://dx.doi.org/10.1002/nav.3800230110.

[40] A. Sprecher, Generation of instances by ProGen, in: Resource-Constrained Project Scheduling, Springer, 1994, pp. 70–90, http://dx.doi.org/10.1007/978-3-642-48397-4_6.

[41] E.W. Davis, Project network summary measures constrained-resource scheduling, AIIE Trans. 7 (2) (1975) 132–142, http://dx.doi.org/10.1080/05695557508974995.

[42] A.A. Mastor, An experimental investigation and comparative evaluation of production line balancing techniques, Manage. Sci. 16 (11) (1970) 728–746, http://dx.doi.org/10.1287/mnsc.16.11.728.

[43] R. Kolisch, A. Sprecher, A. Drexl, Characterization and generation of a general class of resource-constrained project scheduling problems, Manage. Sci. 41 (10) (1995) 1693–1703, http://dx.doi.org/10.1287/mnsc.41.10.1693.

[44] E.L. Demeulemeester, M. Vanhoucke, W. Herroelen, RanGen: A random network generator for activity-on-the-node networks, J. Sched. 6 (1) (2003) 17–38, http://dx.doi.org/10.1023/A:1022283403119.

[45] E. Labro, M. Vanhoucke, Diversity in resource consumption patterns and robustness of costing systems to errors, Manage. Sci. 54 (10) (2008) 1715–1730.

[46] F.F. Boctor, Heuristics for scheduling projects with resource restrictions and several resource-duration modes, Int. J. Prod. Res. 31 (11) (1993) 2547–2558, http://dx.doi.org/10.1080/00207549308956882.



Zsolt T. Kosztyán is a full professor and head of the Department of Quantitative Methods at the University of Pannonia.

His research interest is the development of methodologies to manage complex management problems and systems relating to mathematical models and algorithms of project management, production, maintenance, and network science. This research area is on the frontier between management science, applied informatics, and applied network science. He worked as a senior research fellow at the Institute of Advanced Studies, KRAFT Social Innovation Lab, and at MTA-PE (Hungarian Academy of Sciences–University of Pannonia) Budapest Ranking Research Group. He was a single or corresponding author of more than 200 publications and three books.



Gergely L. Novák is a lead software project manager at Continental Automotive.

His area of research lies within management science and operations research, having a focus on developing methodologies for project and portfolio management. With an emphasis on scheduling and optimization problems, he has a particular interest in standard project databases and project-related indicators. With over a decade of experience in research and development, he has gained expertise in the automotive sector, contributing to software projects for electronic brake systems.