Computers & Industrial Engineering 190 (2024) 110079



Contents lists available at ScienceDirect

Computers & Industrial Engineering

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





The effect of autonomous team role selection in flexible projects

Zsolt T. Kosztyán [a,*], Péter Harta [a,b], István Szalkai [c]

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

ARTICLE INFO

Keywords:
Software project scheduling
Synergy-based modeling
DISC personalities
Autonomous team role selection

ABSTRACT

Today, most software projects are flexible, such as agile, extreme, or hybrid. In addition, this approach is increasingly being adopted for other types of projects. Most software project scheduling problems (SPSPs), more broadly, project scheduling problems (PSPs) and resource allocation methods, assume a fixed project structure. If we use these traditional methods, we cannot model and therefore understand why certain flexible, e.g., agile, policies are essential and how they help project management. This research shows how a matrix-based project planning approach can model team roles and synergies between employees. Extending the synergy-based software project scheduling problem, this study shows the effects of autonomous team role selection, highlighted in agile projects. The proposed model can manage (1) flexible projects, (2) synergies between employees, and (3) the difference between hard skills and soft skills. The proposed method is evaluated in a real-life case study. The proposed method is compared with existing, nonsynergy-based project scheduling methods. We found that positive synergy between employees can reduce the makespan of the project, and the best solution is given in autonomous team role selection.

1. Introduction

In Industry Revolution 4.0 (I40), the role of Information Technology (IT) is increasing. Industry Revolution 5.0 (I50) may mean that software is no longer made by humans but by machines, but today, software products must be made by the collaboration of closely cooperating developers (Oza et al., 2013; Whitehead, 2007). The importance of software development is also underlined by the fact that a widely studied (Vega-Velázquez et al., 2018), so-called Software Project Scheduling Problem (SPSP), deals specifically with the correct selection of human resources in Software Project Environment (SPE). This problem is very similar to that of the MS-RCPSP problem, which has similar limitations and shortcomings (Arashpour et al., 2018; Wongwai & Malaikrisanachalee, 2011). In both the cases of SPSP and MS-RCPSP, the task duration depends on the skills of employees (Alba & Chicano, 2007; Chang et al., 2008; Myszkowski et al., 2019). They specify fixed (Crawford et al., 2014; Luna et al., 2014) or predefined (Birjandi et al., 2019) precedences between tasks. In addition, cooperation or synergies between employees are neglected. These constraints, however, severely limit the practical applicability of these methods. Software applications, such as agile and hybrid approaches and projects, are now primarily developed flexibly (Wysocki, 2019) where not all precedence of development processes is fixed (Aslam & Ijaz, 2018).

(Structural) flexibility here means that the customer can change his or her requirements and priorities during the life of the project, but the cost and duration of the project must remain fixed (Wysocki, 2019). In the project plan, this flexibility appears in the flexible relationships between the tasks (because the customer can change their dependency relations) and in the minimum number of tasks to be completed (there are minimum requirements that cannot be changed).

Priorities are assigned to tasks established in cooperation with the customer (Paetsch et al., 2003). This priority determines whether the task is ultimately implemented in this or subsequent development (Al-Saqqa et al., 2020). In addition, agile approaches rely heavily on continuous communication and collaboration between developers, the synergistic effect of which can significantly reduce the makespan of the Software Development Projects (SDPs) (Winter, 2015).

The main idea of SPSP is that human resources are assigned skill abilities that enable them to perform a task (Alba & Chicano, 2007). Later, researchers (Vega-Velázquez et al., 2018) investigated how the level of these skills affects the time needed to perform a task. Moreover, skills can vary over time, even when based on learning and forgetting factors (Shen et al., 2018; Vega-Velázquez et al., 2018). In most models,

* Corresponding author.
E-mail addresses: kosztyan.zsolt@gtk.uni-pannon.hu (Z.T. Kosztyán), peter.harta@continental-corporation.com (P. Harta), szalkai.istvan@mik.uni-pannon.hu (I. Szalkai).

https://doi.org/10.1016/j.cie.2024.110079
Received 12 April 2023; Received in revised form 30 January 2024; Accepted 14 March 2024
Available online 18 March 2024
0360-8352/© 2024 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

the level of skills directly affects the task duration. A higher level of skills affects a shorter task duration, but employees' skills cannot be added. Instead of skills, which cannot be summarized, Kosztyán et al. (2022) suggested using *work ability* or *skill performance*. For example, the programming or testing skills of two employees cannot be added. However, encoded and verified programmed rows or evaluated functions can be added. In addition, it is much easier to set minimum constraints for skill performance than for the skills themselves. This model strongly supports agile approaches, where one of the requirements is cross-functionality based on the agile manifesto, i.e., that resources have similar capabilities to replace each other. The customer is interested only in the final product. What is less interesting is who ultimately performs the task and how they share the development of tasks. Nevertheless, it should be noted that such performances, which can eventually be summed, are mainly meaningful only for hard skills. For soft skills, such as communication skills, it is not, or only with great difficulty, possible to specify a skill performance. Thus, in our view, both ideas have a place in modeling. Therefore, in the proposed model, binary and ordinal levels of skills and cardinal skill performances can also be modeled.

The synergy between employers and the relation to performance has been extensively studied (see e.g., Curşeu et al., 2015; Larson, 2007, 2009); however, to our knowledge, Kosztyán et al. (2022) is the only study that considers synergy between employees, and they propose a new class of SPSP, called Synergy-based Software Project Scheduling Problem (SSPSP). Kosztyán et al. (2022) investigated artificial projects and synergy networks to determine the effects of synergies. However, this approach was not evaluated on a real-life example. One of the greatest challenges is to put the theoretical method into practice by identifying synergistic effects between employees. We recognize that SPSP has been supplemented with consideration of different personality types (Stylianou & Andreou, 2012). Numerous studies (LePine et al., 2011; Peslak, 2006; Pieterse et al., 2018) have shown that synergy is strongly influenced by personality type and team role. Belbin (2012) found correspondence between team roles, personality types, and synergies. In addition, some studies (Choi et al., 2008; Sfetsos et al., 2009; Walle & Hannay, 2009) have investigated the relationship between personality type and the success of paired programming (Nosek, 1998). Belbin suggested separating team roles with different personality types into so-called Belbin team roles and grouping them into triads. Forming a team with different Belbin team roles can reduce the opportunity for team conflict and increase the amount of work relations between them. Moreover, Hardiman (1997) and Capretz (2003) studied the most common personality types of software development teams, and Capretz and Ahmed (2010) investigated the impact of personality types on software team performance by using the widely known Myers–Briggs personality library (Myers, 1962).

Before the agility method was found and the importance of small teams was recognized, the Myers–Briggs library, which included 16 different personality types, was more likely to be used because it was better for the selection of larger project teams. Many articles (Flores-Parra et al., 2018; Meslec & Curşeu, 2015; Mostert, 2015) address Belbin team selection as well as the most popular Myers–Briggs (Capretz, 2003; Peslak, 2006) or Big Five factor traits (Peeters et al., 2006). In contrast, the DISC personal assessment tool is more flexible due to the small number of different personality traits (only 4). Because of its simplicity and popularity, DISC personality types have gradually increased in recent years and can be used for the selection of small agile teams (Lykourentzou et al., 2016; Nikolaou & Oostrom, 2015).

The DISC personality model, developed by Marston (1928) in 1928, categorizes individuals into several personality types based on behavioral differences. These types are represented by the acronym DISC, which stands for dimensions of openness and orientation. This model includes the dimensions of dominance (D), influence (I), steadiness (S), and conscientiousness (C). Every personality type corresponds to a distinct color: red for dominance (D), yellow for influence (I), green for steadiness (S), and blue for conscientiousness (C). Although individuals may display many personality types to different extents, they can largely be categorized under the type that best represents them. Table 1 offers a concise summary of these individuals.

Table 1
Brief description of DISC personality types.

Personality type	Characterization	Main properties
Dominance	extrovert, task-oriented, the leader of the team, takes responsibility	active, powerful, confronting, obstinate
Influence	extrovert, people-oriented, the soul of the team, maintains motivation	friendly, optimistic, easy-going, unstable
Steadiness	introvert, people-oriented, the parent of the team, maintains stability	team-player, reliable, loyal, retractive
Conscientiousness	introvert, task-oriented, the brains of the team, precisely promotes solutions	accurate, analytical, perfectionist, mistrustful

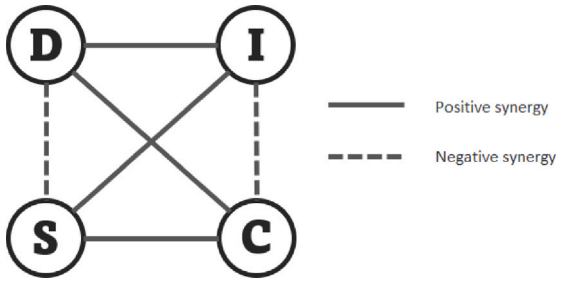


Fig. 1. The assumed synergy network between DISC personality types and our edits.

While the allocation of the different DISC personality types can strongly influence team performance and well-being (Lykourentzou et al., 2016), another study (Antoniou, 2019) found that there is no relation between balanced and unbalanced teams. Since we distinguish between synergy networks (a measure of joint work) and sociograms (a measure of social relations), we can conclude that balanced DISC groups have a strong impact on the synergy network among team members. Based on the literature (Scullard & Baum, 2015), we estimated the positive and negative synergies between DISC personality types in a balanced DISC group (see Fig. 1). We can also conclude that the sociometric analysis of a balanced DISC team shows only positive relations between team members because members fill it with heterogeneous personalities. However, in this article, we focus only on the benefits of the project, which are represented by the synergy network.

The synergy network revealed positive pairwise synergies between D and I, D and C, I and S, and S and C and negative pairwise synergies between D and S and between D and C. The details are shown in Table 2

Using different team roles or personalities to select a team is called personality-based selection, and the team is the so-called synergistic team. If we do so by considering only the skill of the employees, we can call this a skills-based selection, and the formed team is the so-called multiskilled team. In this study, we use the DISC personality types to determine the synergy network because it is simpler to measure four personality types and because of the extensive study of its effects.

Related to the literature, our proposed study makes two contributions.

C_1 The proposed method allows scholars to model soft and hard skills in flexible projects.

Table 2
Synergies of pairs.

(a) Pairs with positive synergy

First type	Second type	Benefit from the first type	Benefit from the second type
D	I	leadership and direction	inspire and motivate others
D	C	*focus on achieving results*	*attention to detail*
S	I	stability and consistency	build relationships
S	C	work collaboratively	attention to detail

(b) Pairs with negative synergy

First type	Second type	Loss from the first type	Loss from the second type
D	S	delegate and make pressure	inflexible
C	I	standoffish	open to discuss

C_2 The proposed model shows how to investigate the role of autonomous or directed team-role selection.

The proposed modification of SSPSP shows how to model an agile project (C_1), where

1. Completion priorities can be assigned to tasks;
2. It can manage flexible dependencies;
3. can manage the synergies and team roles of employees;
4. soft-skill and hard-skill performances are separated.

Managerially speaking, flexible projects encompass project designs that are agile, extreme, or hybrid in nature. From a methodological standpoint, a flexible project prioritizes adaptability in its structure. In a project that allows for adaptability, the arrangement of activities can be organized based on their level of importance. As a result, in projects that follow agile, extreme, or hybrid methodologies, which are used mainly in software project management (see in detail in Wysocki, 2019), lower priority tasks can be omitted or delayed until subsequent (sub)projects, known as sprints, are obtained. Furthermore, interdependence among activities can also be adaptable, enabling sequential or concurrent completion and mirroring the technological adaptability of software projects.

To address these expansions of the original SPSP, we employed a matrix-based representation. Matrix-based approaches offer the advantage of effectively handling projects with a high degree of flexibility (Kosztyán et al., 2023). For instance, when a task of lesser importance is omitted or postponed from the project, the matrix is diminished by removing the corresponding column and row of the omitted or delayed task(s) (Kosztyán, 2015). This reduction in the matrix effectively reduces the problem at hand. (2) Both the management and methodological aspects can be effectively addressed by utilizing domains in a multidomain mapping matrix (Browning, 2014) to plan and coordinate all the requirements, synergies, and task-employee dedications (Kosztyán et al., 2022).

The extended model manages binary, ordinal, nonadditive (soft) skills and additive (hard) skill performances (C_1). The DISC profiles of employees provide expected synergies between developers (Figure 1 and Table 1). After team roles are specified, the effect of team role selection mechanisms can be investigated (C_2), and the following research question (RQ) can be answered.

RQ Which kind of team role selection mechanism provides the shortest project duration considering the constraints?

The remainder of this paper is organized as follows. Section 2 introduces the modified SMM matrix and the extended SSPSP problem. Section 3 describes how to apply this method to the standard multi-skilled project scheduling problem. Section 4 presents the case study, where we can see how to consider advanced features, such as synergies and hard skill performances, in the modified SSPSP problem. Section 5 presents the results and discussion. Finally, Section 6 concludes the paper.

2. Methods

The SSPSP specifies a flexible project plan modeled in a Multidomain Mapping (MDM) matrix. Kosztyán et al. (2022) proposed a six-domain matrix model. The proposed version of the SMM matrix contains six domains and two column vectors. The modified SMM matrix is an $m+n\times m+s+n+1$ matrix, where the number of employees is m, the number of skilled performances is s, and the number of tasks is n.

1. The first domain (**Y**), an m by m submatrix, represents the synergy between employees. In the case of $i \neq j$
   - $[\mathbf{Y}]_{ij} > 1$ represents positive (or favorable) synergy,
   - $[\mathbf{Y}]_{ij} = 1$ represents neutral synergy,
   - $0 < [\mathbf{Y}]_{ij} < 1$ represents negative (or unfavorable) synergy between employees i and j.

   Furthermore, we assume that $[\mathbf{Y}]_{ii} = 1$ and that $[\mathbf{Y}]_{ij} = [\mathbf{Y}]_{ji}$.
2. The second domain is the skill domain (**S**). The skill domain is an m by s submatrix, where every skill or skill performance is a non-negative number ($[\mathbf{S}]_{ij} \in \mathbb{R}_0^+$). $s_j := s_{\cdot j} := \left[[\mathbf{S}]_{1j}, [\mathbf{S}]_{2j}, \dots, [\mathbf{S}]_{mj}\right]$ represent skill vectors. We suppose that there are h **hard** skills, represented by the $[s_1, s_2, \dots, s_h]$ hard skill performance vector, and $s-h$ **soft** skills, represented by the $[s_{h+1}, \dots, s_s]$ skill vector. $[\mathbf{S}]_{ij} = 0$ means that employee i has no work ability for skill j. **S** can store either *binary* or *cardinal* levels of skills. In the case of the binary representation, $[\mathbf{S}]_{ij}$ is either 0 or 1. If the levels of skills (e.g., level of language skill, level of communication skill, etc.) are represented, then $[\mathbf{S}]_{ij} \in \mathbb{N}$. These binary and ordinal skills are referred to as *nonadditive*; in other words, there is no meaning of the sum $\sum_i [\mathbf{S}]_{ij}$. The cardinal skill performance (usually hard skill performance) is *additive*, and $[\mathbf{S}]_{ij} \in \mathbb{R}_0^+$. Synergistic factors may also modify skill performance. Let ε be a subset of employees; then, the *joint skill* of ε is

$$S_j^{\varepsilon} := \overline{Y}_{\varepsilon} \cdot \sum_{i\in\varepsilon} [\mathbf{S}]_{ij} \tag{1}$$

   where $\overline{Y}_{\varepsilon}$ is the *geometric mean* of synergies:

$$\overline{Y}_{\varepsilon} := \begin{cases} 1 & \text{if} \quad |\varepsilon| \le 1 \\ \sqrt[\eta]{\prod_{i,j\in\varepsilon,\ i<j} [\mathbf{Y}]_{i,j}} \quad \text{where } \eta = \frac{|\varepsilon|\cdot(|\varepsilon|-1)}{2} & \text{if} \quad |\varepsilon| > 1 \end{cases} \tag{2}$$

3. The third domain is a matching domain (**M**). **M** is an m by n domain, where $[\mathbf{M}]_{ij} \in [0,1]$ represents the maximal relative amount of assignments of employee i to task j. If $[\mathbf{M}]_{ij} = 0$ ($[\mathbf{M}]_{ij} = 1$), then employee i is not (fully) assigned to task j.
4. The fourth domain is the activity or task domain (**A**). The activity domain is an n by n square matrix ($[\mathbf{A}]_{ij} \in [0,1]$), where the diagonal represents the relative priorities of task (activity) completion. $[\mathbf{A}]_{ii} = 1$ represents the *mandatory* task, which must be performed ultimately and cannot be postponed. $0 < [\mathbf{A}]_{ii} < 1$ represents the *supplementary* task, which, depending on the constraints, can be postponed to a later project (or a later subproject, called a sprint in agile project management). A higher $[\mathbf{A}]_{ii}$ on the diagonal represents a greater priority (greater score). A postponed task's precedence and demands are also neglected. $[\mathbf{A}]_{ij}, i \neq j$ represents the precedence between tasks a_i and a_j ("a_i must end before a_j starts" ($a_i \prec a_j$), or "no precedence between a_i and a_j" ($a_i \sim a_j$)). $[\mathbf{A}]_{ij} = 1$ represents the fixed dependency between task i and task j ($a_i \prec a_j$),

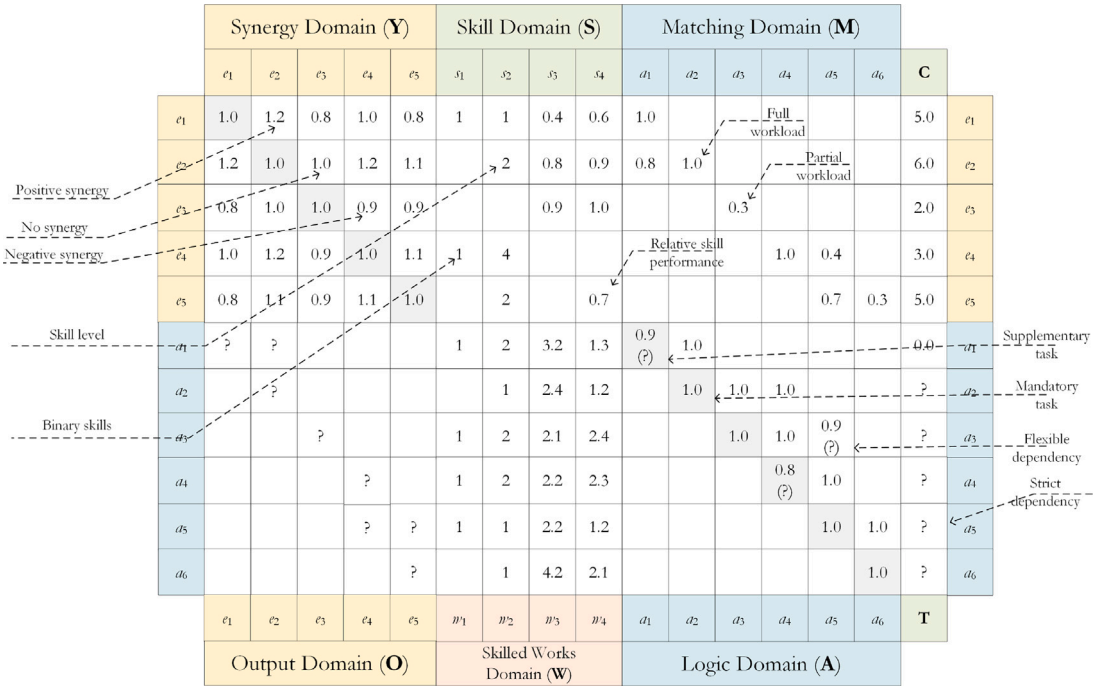


Fig. 2. The proposed, modified SMM matrix.

while $0 < [\mathbf{A}]_{ij} < 1$ represents flexible dependency ("$a_i \bowtie a_j$"), which, depending on the constraints, can be either prescribed or relaxed. Importantly, after the optimization, depending on the constraints, every $0 < [\mathbf{A}]_{ii} < 1$ and $0 < [\mathbf{A}]_{ij} < 1$ value must be either 1 or 0, and $[\mathbf{A}]_{ij} = 1$ must imply $[\mathbf{A}]_{ii} = 1$ and $[\mathbf{A}]_{jj} = 1$. This means that the proposed algorithm has to decide which supplementary tasks have to be completed or postponed and which flexible tasks have to be prescribed or relaxed.

5. The fifth domain is the skilled-word domain (**W**). **W** is the n by s matrix. $[\mathbf{W}]_{ji}$ stores the required skilled work of skill i for task j. In the case of binary and ordinal skills, $[\mathbf{W}]_{ji}$ can represent a minimum requirement of skills for completing tasks j, e.g., minimum level of communication and minimum level of language skills, while in the case of skill performance, $[\mathbf{W}]_{ji}$ represents a minimal amount of skilled work, such as tested functions and documented programming codes.
6. The last domain is the output domain (**O**), which contains the solution of the SSPSP algorithm. **O** is an n by m matrix (of nonnegative real numbers), where the element $[\mathbf{O}]_{j,i} > 0$ represents the (final) allocation of employee i to task j. $[\mathbf{O}]_{j,i}$ is the proposed *ratio* of the working time e_i is allocated to a_j; clearly, $[\mathbf{O}]_{j,i} = 0$ means *no* allocation. $[\mathbf{O}]_{j,i} \leq [\mathbf{M}]_{i,j}$ and $\sum_{j=1}^{n}[\mathbf{O}]_{j,i} \leq 1$ must hold for each $j = 1, 2, \ldots, n$ and $i = 1, \ldots, m$, while $\sum_{j=1}^{n}[\mathbf{M}]_{i,j} \leq 1$ are *not* required for any $i = 1, \ldots, m$.
7-8. The modified SMM matrix contains two extra column vectors. **C**, the first is an m by 1 column vector containing the salary of employees, and the second, **T** is an n by 1 column vector containing the scheduled start time of the tasks.

Fig. 2 shows an example of a filled modified SMM matrix, where the number of employees is five and the number of tasks is six (four mandatory tasks, two supplementary tasks). The example represents one binary, one ordinal skill and two skill performances. The symbol "?" represents the variable cells that must be optimized with the proposed algorithm.

The *duration* of activity a_j is denoted by[1] $a_j^{dur}(\mathbf{O})$. (This depends on resources modified by the synergy factor, as calculated in Eqs. (5) and (6).) The *starting time* of a_j is $a_j^{start}(\mathbf{O})$, and the finishing time is[2] $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + a_j^{dur}(\mathbf{O})$ (see Eq. (7)). The *duration of the project* is denoted by p_{dur} or Total Project Time (TPT), and its cost is p_{cost} or Total Project Cost (TPC). The monthly *salary* of employee e_i is denoted by e_i^{salary} or $[\mathbf{C}]_i$.

Since task a_j *requires* $[\mathbf{W}]_{j,k}$ skilled work, the *required time* (duration) to fulfill the requirement (skill) k of task j without synergies is:

$$a_{j,k}^{dur}(\mathbf{O}) = \frac{[\mathbf{W}]_{j,k}}{\sum_{i=1}^{m}\left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right)}, \tag{3}$$

and the *adjusted required time* (with synergies) is:

$$a_{j,k}^{dur,adj}(\mathbf{O}) = \frac{[\mathbf{W}]_{j,k}}{\overline{Y}_{\varepsilon_j} \cdot \sum_{i=1}^{m}\left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right)} \tag{4}$$

where $\varepsilon_j := \{i : 0 < [\mathbf{O}]_{j,i}\}$ is the set of employees ultimately assigned to task j.

Without considering the synergies, the duration time of task j is:

$$a_j^{dur}(\mathbf{O}) = \max_{0<[\mathbf{W}]_{j,k}} \left\{ a_{j,k}^{dur}(\mathbf{O}) \right\} \tag{5}$$

and (with synergies):

$$\widetilde{a}_j^{dur}(\mathbf{O}) := a_j^{dur,adj}(\mathbf{O}) = \max_{0<[\mathbf{W}]_{j,k}} \left\{ a_{j,k}^{dur,adj}(\mathbf{O}) \right\}. \tag{6}$$

The durations are used to calculate the finish times of the activities $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + \widetilde{a}_j^{dur}(\mathbf{O})$, where

$$a_j^{start}(\mathbf{O}) \geq \begin{cases} 0 & \text{if } \nexists\, a_i \in A,\ a_i \prec a_j \\ \max\{a_i^{end}(\mathbf{O}) : a_i \prec a_j\} & \text{otherwise} \end{cases}. \tag{7}$$

a_i is **critical** if neither its starting nor ending time can be changed when optimizing the project time: $a_{j_1}^{end}(\mathbf{O}) = a_i^{start}(\mathbf{O})$ and $a_i^{end}(\mathbf{O}) =$

[1] From here "(**O**)" refers to the final output **O** of the algorithm.
[2] We recall that $a_i \prec a_j$ implies $a_i^{end}(\mathbf{O}) \leq a_j^{start}(\mathbf{O})$.

$a^{start}_{j_2}(\mathbf{O})$ for some j_1, j_2 such that $a_{j_1} \prec a_i \prec a_{j_2}$. The "critical" status of an activity can be varied during the optimization algorithm[3]

The values calculated above enable us to calculate the duration of the project as follows:

$$\mathrm{TPT}_{nosyn} := \max\{a^{end}_j(\mathbf{O}) : j = 1, \dots, n\}. \tag{8}$$

$$\mathrm{TPT}_{syn} := \max\{\widetilde{a}^{end}_j(\mathbf{O}) : j = 1, \dots, n\}. \tag{9}$$

If all supplementary tasks are postponed and all flexible tasks are excluded (formally, $\mathbf{A}_{\min} = \lfloor \mathbf{A} \rfloor$), in the case of full assignments (formally, $\mathbf{O} = \mathbf{M}^T$), we obtain a minimum TPT ($\mathrm{TPT}_{\min}$). However, in practice, this approach is usually not feasible, given the constraints. The maximal TPT is infinite if there is no assignment.

The cost of the project (TPC) can be calculated as the sum of the salaries of employees that are paid for their dedication to the project. Since positive synergy decreases and negative synergy increases the duration a^{dur}_j to $\widetilde{a}^{dur}_j$, the project cost can be calculated with and without the synergistic effect, obtaining TPC_{syn} and TPC_{nosyn}, respectively. Formally:

$$\mathrm{TPC}_{syn} = \sum_{i=1}^{m}\sum_{j=1}^{n}([\mathbf{C}]_i \times [\mathbf{O}]_{j,i} \times \widetilde{a}^{dur}_j(\mathbf{O})), \tag{10}$$

$$\mathrm{TPC}_{nosyn} := \sum_{i=1}^{m}\sum_{j=1}^{n}([\mathbf{C}]_i \times [\mathbf{O}]_{j,i} \times a^{dur}_j(\mathbf{O})). \tag{11}$$

The maximal amount of costs $\mathrm{TPC}_{\max}$ occurs in the case of full assignment and if all supplementary tasks are decided to be completed. The minimum value is 0 if there is no assignment to any task.

The Total Project Score (TPS) is not influenced by the synergy. It depends only on the set of decided-to-complete tasks, denoted by $\mathcal{A}$.

$$\mathrm{TPS} := \sum_{i \in \mathcal{A}} [\mathbf{A}]_{ii} \tag{12}$$

$\mathrm{TPS}_{\min}$ ($\mathrm{TPS}_{\max}$) occurs if all supplementary tasks are postponed (completed).

2.1. Possible target functions

Now, we declare the possible objective functions that we seek to optimize *simultaneously* (in Eq. (16)) by using the following algorithm:

$$\mathrm{TPT} \to \min, \tag{13}$$

$$\mathrm{TPC} \to \min, \tag{14}$$

and

$$\mathrm{TPS} \to \max. \tag{15}$$

These objective (target) functions can be considered a multiobjective problem or a composite objective (target) function and can be specified as follows.[4] (Here, C_s, C_p, C_c and C_t are given reasonable constants):

$$z := 1 - \sqrt[w]{\left(\frac{C_t - \mathrm{TPT}}{C_t - \mathrm{TPT}_{\min}}\right)^{w_t} * \left(\frac{C_c - \mathrm{TPC}}{\mathrm{TPC}_{\max}}\right)^{w_c} * \left(\frac{\mathrm{TPS} - C_s}{\mathrm{TPS}_{\max} - C_s}\right)^{w_s}} \to \min, \tag{16}$$

The positive numbers w, which represent weights, are derived from the preferences of the decision makers and can be expressed as the sum of w_t, w_c, and w_s positive numbers. The decision makers' preferences can be determined using the analytic hierarchy process (AHP), which relies on pairwise comparisons to assess the importance of target functions. In this investigation, we assigned equal weights to each variable, with $w_t = w_c = w_s = 1$. Pareto optimization is particularly advantageous when distinct outcomes are obtained with each goal function. Multiobjective optimization yields a Pareto front, which allows the decision maker to select the most appropriate solution. Simultaneously, if the outcomes produced by the objective functions exhibit negligible disparities, then multiobjective optimization is not justified.

We assume the constraints $\mathbf{CR}_1 - \mathbf{CR}_9$ below.

2.2. Constraints

Based on the literature (see e.g. Chen et al., 2017; Kia et al., 2016; Kosztyán et al., 2022; Maghsoudlou et al., 2017; Wang et al., 2022), we can state the following constraints:

- CR_1 All required resources for the project are of human force type and are always available. The employment of each employee e_i in the *project* is not allowed to exceed its *maximum* value: $e^w_i := \sum_{j=1}^{n}[\mathbf{O}]_{j,i} \le e^{maxw}_i := \sum_{j=1}^{n}[\mathbf{M}]_{i,j}$. Clearly,[5] $0 \le e^w_i \le 1$ by $\sum_{j=1}^{n}[\mathbf{O}]_{j,i} \le 1$. In addition, each activity must be performed by at least one human resource.
- CR_2: The set of skills that an activity requires must be a subset of the union of skills of the employees who perform this activity (for each activity a we have $\{s : [\mathbf{W}]_{a,s} > 0\} \subseteq \bigcup_{e \in \varepsilon_a}\{s : [\mathbf{S}]_{e,s} > 0\}$, where $\varepsilon_a \subseteq \{e_1, \dots, e_m\}$ is the set of employees assigned to a). Moreover, the "joint" level of skills of employees involved is at least the level required by the activity planned. (More precisely, for any activity a and skill s, such that $[\mathbf{W}]_{a,s} > 0$, we require (i) for *nonadditive* skill s, there are several $e \in \varepsilon_a$ such that $[\mathbf{S}]_{e,s} \ge [\mathbf{W}]_{a,s}$, (ii) for *additive* skill s, we need $S^{\varepsilon}_s \ge [\mathbf{W}]_{a,s}$.) Finally, for each *critical* activity a and *soft* skill s required to a at the minimum level $w_{i,j}$ (i.e., $[\mathbf{W}]_{a,s} \ge w_{a,s}$), at least one employee $e \in \varepsilon_a$ must possess skill s at minimum level $w_{a,s}$ (i.e., $[\mathbf{S}]_{e,s} \ge w_{i,j}$ must hold).

There are further constraints in SSPSP to manage flexible projects and to specify the set of implemented tasks.

- CR_3: The TPS must be greater than a specified (score) constraint C_s, formally: TPS> C_s.

The following six additional constraints are the constraints of the project plan:

- CR_4: Overwork is allowed up to a *certain level* (roughly: $E^w = \sum_{i=1}^{m} e^w_i \le K^w$ for some constant K^w, with minor exceptions).
- CR_5: The development costs include the total salaries of all employees assigned in projects, and TPC must be less than the cost constraint (C_c).
- CR_6: The TPT must be less than the time constraint (C_t).
- CR_7: Each task must be started and executed once without interruption. In addition, the manpower assigned to a skill of each task must perform the assigned skill continuously without interruption.
- CR_8: All the workforces assigned to various skills of each activity should start their work concurrently.
- CR_9: The staff is fixed throughout the entire development process.

Importantly, predefined precedences must be maintained. However, in a flexible project, flexible precedences can change. In addition, tasks can be excluded or postponed to a later subproject (usually called a sprint). The structure of the implementation is the output of the optimization.

[3] and $a^{dur}_j(\mathbf{O})$ and $\widetilde{a}^{dur}_j(\mathbf{O})$ are minimized by the algorithm.

[4] Instead of Pareto sets, we combined the multiobjective functions (13), (14) and (15) into the single function in (16).

[5] see the matching domain ($\mathbf{M}$) in Fig. 2.

2.3. Applied hybrid genetic algorithm

Since SPSP is NP-hard (Xiao et al., 2013), which is a special case of SSPSP. Kosztyán et al. (2022) showed that SSPSP is also NP-hard. Kosztyán et al. (2022) also proposed a hybrid genetic algorithm for solving the SSPSP problem. The suggested Hybrid Genetic Algorithm (HGA) has two phases. In the first phase, a GA is employed to specify the set of completed tasks and the final precendences. In phase 2, a Nelder–Mead Minimization (NMM) method is used to refine Scheduled Start Time (SST) to balance the resource demands. We extended and parallelized this algorithm, which was also implemented in MATLAB. The original settings of HGA are published in Kosztyán et al. (2022); therefore, we focus only on the extensions and modifications.

A chromosome encodes a probable solution of SSPSP. The proposed modification of the original chromosome structure enables the selection of a synergy network from a pool. Synergy networks are influenced by team roles. N different team roles specify N different synergy networks. In addition, the proposed method distinguishes skills into binary and ordinal skills, which are usually soft skills, and additive skill performances, which usually describe the performance of hard skills. Finally, we organized the chromosomes into a multichromosome structure (see Fig. 3 to decrease the computational time and to share the best chromosomes in parallel computations. The proposed chromosome structure has four parts.

The first element of the chromosome is the selected number of the synergy network. It is an ordinal value from 1 to N. The second part is a binary sequence of the decision outcome of completing supplementary tasks and flexible dependencies. The length of this part of the chromosome is $n_F = n_s + n_f$, where n_s is the number of supplementary tasks and n_f is the number of flexible dependencies. The third part of the chromosome encodes the assignment ratios from the output domain. These real values must be in the interval. The number of assignment ratios (n_A) is the number of nonzero elements from the match domain (M). The last part encodes the SST of tasks. The number of elements in this part is n. Therefore, the number of elements of a chromosome vector is $N_c = 1 + n_F + n_A + n$. All four parts of the chromosome are of a different type; therefore, different crossover, mutation, and selection mechanisms must be proposed for these parts. For the first two parts of the chromosome, a uniform crossover mechanism is used. However, the parents may be infeasible; therefore, we assume that the feasible parents' genes are ten times dominant. In other words, a gene is ten times more likely to originate from feasible parents than from infeasible parents. For the third and fourth (continuous) parts of the chromosome, an arithmetic crossover function is used. Since all chromosomes encode a solution to the modified SSPSP, the feasibility of a solution can be checked. Although it may take some time to assess feasibility, our findings indicate that increasing the number of feasible individuals for recombination leads to improved convergence without compromising the quality of the best solution. In the case of Design of Experiments (DoE), this parameter has also been tuned (see Section 2.4).

A two-step mutation process is used, in which the first step is general and is conducted for all parts of the chromosome. In the first step, the algorithm selects a fraction of the vector entries of an individual for mutation, where each entry has a probability of being mutated. According to the results, this rate is specified as 0.05. In the second step, although the same mechanism is used when the mutation operator is implemented, the two parts of the chromosomes must be distinguished. In this case, the adaptive feasible mutation function is used. The mutation operator chooses a direction and step length that satisfy the bounds and linear constraints. After the mutation operator is used, the requirements of the excluded tasks and their task dependencies must be eliminated (set to 0).

To reduce the computation time, we specified a multichromosome structure. Since the length of the chromosomes is equal in all runs, GA can be parallelized. In a multichromosome structure, there are p chromosome vectors. The value of p is usually related to the number of processors (or graphical coprocessors). The pool contains M multichromosomes in a generation. The processors evaluate them in parallel, and selected chromosomes or parts of the chromosomes can be migrated (see Fig. 3).

Table 3
Tuned hyperparameters of GA.

Operators	Parameters
Population size	250
Badges of chromosomes	4
Tournament size	9
Elite count	0.05
Crossover fraction	0.82
Rate of feasible chromosomes in crossover	0.88
Probability of mutation of a gene in a chromosome	0.05
Maximal rate of migrated chromosomes	0.09
Tolerance value	1E−8
Maximal iteration	150

2.4. Hyperparameter tuning of GA

The changed SSPSP problem is more similar to the original SSPSP problem from Kosztyán et al. (2022), except for the formation of multichromosomes. We used the hyperparameters suggested there as a starting point because they were already tuned. We used the DoE method to adjust the hyperparameters of GA following (Reeves & Wright, 1999)'s theoretical background, and the optimization results were analyzed by a highly robust method, the so-called regression tree ensemble model of the MATLAB regression learner app. During the calculation, 10-fold cross-validation was used, and the hyperparameters were tuned via Bayesian optimization. The target function was the convergence speed. After tuning, the following parameters of GA are used (see Table 3).

After the hyperparameters of GA are tuned, the number of populations in a badge of chromosomes is set to 250, while the number of badges is 4. The number of badges depends on the core of the processors. Since the employed computers can run four independent threads in two cores, these values produced the best performance. The best selection mechanism was the tournament, for which the tournament size was 9. The elite count was 0.05. This value specifies how many individuals in the current generation are guaranteed to survive to the next generation. The crossover fraction specifies the fraction of each population (other than elite children) that consists of crossover children. After fine-tuning, 82% of the chromosomes were crossover children. The rate of feasible children was 88% in a crossover function. This value ensures fast convergence, but infeasible children help us not to consider only local minima. The probability of a gene mutation was 5%, while the maximal rate of chromosome migration was 9%. The applied tolerance was 1E-8, while 150 iterations were always sufficient.

3. Experiments

To initially test the modified SSPSP algorithm, we used the Imoprose database (Myszkowski et al., 2019). We used the 15_6_10_9 project from these databases. It contains 15 tasks, 6 employees and 9 skills. However, this database does not use the following:

- *synergies*; therefore, first, we considered the synergy domain as a full one matrix ($\mathbf{Y} := \mathbf{1}, [\mathbf{Y}]_{ij} := 1$);
- *flexible dependencies*; therefore, we used only binary values in **A**.
- *Skill performance*; therefore, we considered skill as an ordinal variable, where the addition operator had no meaning.
- *nonbinary employee-task assignments*; therefore, we allowed only 0 or 1 values in domains **M** and **O**.

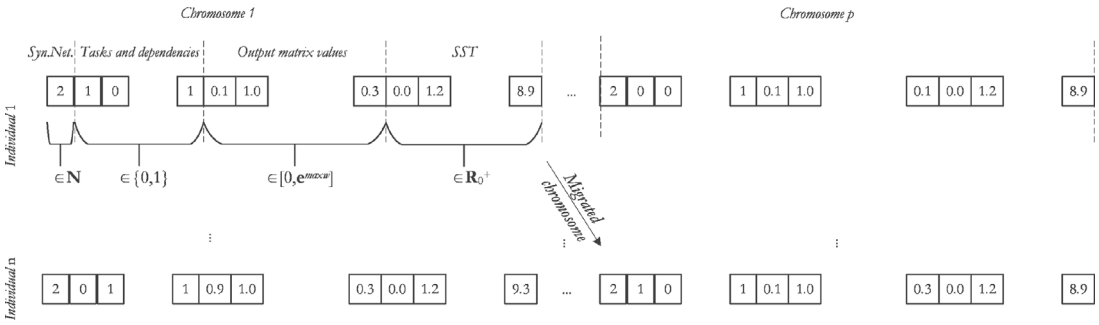


Fig. 3. The proposed multichromosome structure.

Table 4
Skill assignments and the precedence matrix.

(a) Skill assignments

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
e_1					1		2		2
e_2				1	2		1	1	
e_3	2					1	1	1	
e_4	2			2			2		2
e_5			1			2	2		
e_6	1	2							1

(b) Precedence matrix

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_1	1									1					
a_2		1	1	1											
a_3			1												
a_4				1						1					
a_5					1										
a_6						1	1								
a_7							1		1				1		
a_8								1							
a_9									1	1			1		
a_{10}										1				1	
a_{11}											1				
a_{12}												1			
a_{13}													1		
a_{14}														1	
a_{15}															1

(c) Task assignments

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
e_1		1				1						1			
e_2					1			1		1					
e_3	1										1		1		1
e_4			1	1											
e_5									1					1	
e_6							1								

Table 4 shows the skill assignments (see Table 4(a)), the precedence matrix (see Table 4(b)), and the employee-task assignments (see Table 4(c)).

Due to the restriction of one person per task, we restricted our evaluation to a maximum of one participant per task. After the optimization, the **T** vectors for the durations were **T** = [26.0006, 21.0008, 24.0001, 21.0004, 39.0000, 33.0004, 13.0014, 12.0000, 9.0003, 16.0002, 9.0001, 36.0004, 14.0000, 31.0000, 17.0000], where in the case of optimal assignments, the values of the duration times were $\mathbf{T}_{opt}$ = [26, 21, 21, 39, 33, 13, 12, 9, 16, 9, 36, 14, 31, 17]. After running the proposed HGA method, the TPC was 16279.1, while the optimal TPC was 16278.6. Although the results are very close to the validated real values, the real advantage of the proposed method is highlighted when synergies between employees are stated, not only are binary assignments considered, and the addition of hard skill performances is allowed (see Section 4).

Although the original test dataset does not specify synergies between employees, in the second step of the experiment, we considered the following synergy domains ($\mathbf{Y}_I$).

$$[\mathbf{Y}_I]_{ii} = 1,\ [\mathbf{Y}_I]_{ij} = y_I,\ y_I \in \{0.5, 0.6, \dots, 1.5\},\ i \neq j. \tag{17}$$

Fig. 4 shows the dependence of lead time and costs on the change in synergy. Considering Eqs. (8)–(9) and (10)–(11), we calculated TPT and TPC when considering and neglecting synergies. If the rate of TPT_{syn}/TPT_{nosyn} (TPC_{syn}/TPC_{nosyn}) is lower than one, then one can see how the makespan (total cost) is reduced, while a value greater than one indicates the effect of the negative synergies.

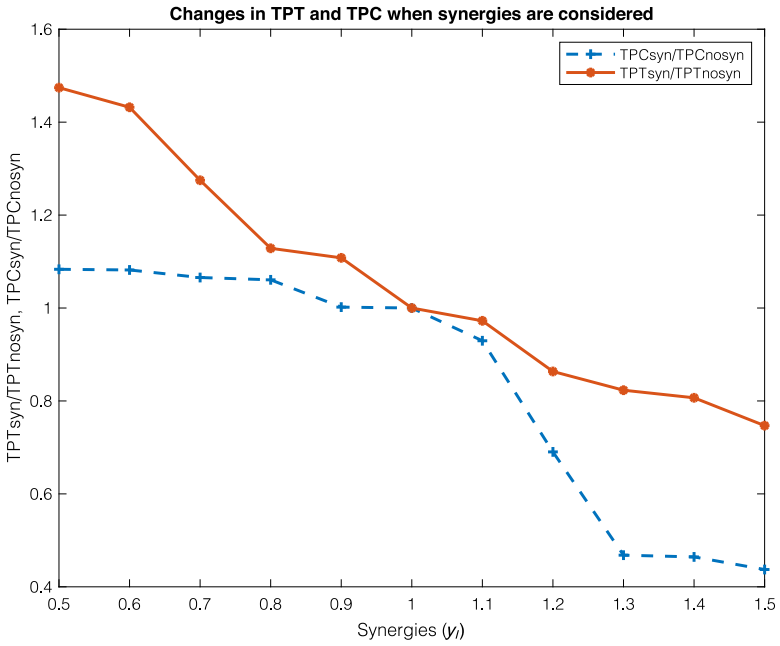
Since the synergy between all employees in this task changes simultaneously in the same direction, the lead times and costs are also sensitive to changes in the synergy. Fig. 4 shows that costs (TPC_{syn}/TPC_{nosyn}) are more sensitive to positive synergies ($y_I > 1$), while lead time (see TPT_{syn}/TPT_{nosyn}) is more sensitive to negative synergies ($y_I < 1$).

Myszkowski, Skowroński, and Sikora (2015) compared different methods for solving MS-RCPSP problems. Table 5 shows a comparison of the results. The first four algorithm results are from Myszkowski, Skowroński, and Sikora (2015)'s runs, and we use these results as a reference. These algorithms include (1) a simple duration-oriented heuristic algorithm, presented by Myszkowski et al. (2013); (2) a duration-oriented greedy algorithm; (3) the well-known metaheuristic ant colony algorithm (ACO); and (4) the modified hybrid ant colony algorithm (HAntCO), which can use priority rules against the simple ACO, where (2-3-4) are presented by Myszkowski, Skowroński, Olech, and Oślizło (2015). The original SSPSP algorithm was implemented by Kosztyán et al. (2022), and the modified, proposed SSPSP algorithm was used as the HGA algorithm.

Table 5 shows that HAntCo outperforms all the other methods and that the SSPSP algorithms have similar performances. The performances of the SSPSP algorithms in minimizing TPT and TPC can be considered average. Moreover, the advantages of these methods unfold when flexible task dependencies, different types of skills, and synergies between employees are accounted for.

4. Case study

The implication of our proposed method was evaluated in the R&D sector of a large high-tech automotive company. In this sector, the main target is to provide software codes for the company's products, which sometimes consist of millions of lines of code. Software codes are developed iteratively and incrementally following agile approaches. The applied project is an iteration of a real-life software development project with nine tasks where all tasks are mandatory except for one comfort feature. Each feature needs to be implemented and evaluated separately but in parallel based on customer requirements. Software integration must be started if all features are evaluated except for one



[

]

Fig. 4. The dependency of total project time and total project cost on synergies.

Table 5
Comparison of existing methods in MS-RCPSP (n is the number of tasks, e is the number of employees, p is the number of precedences, s is the number of skills, TPT is the total project time, TPC is the total project cost. ACO is the Ant Colony Optimization, HAntCO is a modified (heuristic) Ant Colony Optimization, SSPSP is the algorithm for the synergy-based software scheduling problem.

n	e	p	s	Heuristic		Greedy		ACO		HAntCO		original SSPSP		modified SSPSP	
				TPT	TPC	TPT	TPC	TPT	TPC	TPT	TPC	TPT	TPC	TPT	TPC
100	10	26	15	37	126 361	38	119 336	32	124 687	31	126 216	35	124 168	35	124 154
100	10	27	9	38	44 309	38	43 438	34	44 999	33	42 199	37	43 756	36	43 750
100	10	47	9	41	142 759	40	135 161	36	143 100	34	140 865	38	140 483	38	140 489
100	10	48	15	36	135 534	44	120 664	33	133 062	33	133 495	37	130 698	37	130 693
100	10	64	9	39	113 124	43	117 993	35	110 643	33	113 774	38	113 898	38	113 892
100	10	65	15	40	152 955	43	140 782	35	150 294	32	149 185	38	148 305	38	148 313
100	20	22	15	25	117 493	24	112 135	20	120 949	19	123 642	22	118 568	23	118 556
100	20	23	9	32	53 154	32	50 279	32	52 119	23	53 358	30	52 235	31	52 242
100	20	46	15	28	138 270	29	133 739	25	138 565	24	138 568	27	137 286	27	137 294
100	20	47	9	21	129 160	28	140 626	21	124 817	18	134 312	22	132 235	22	132 247
100	20	65	15	32	110 503	34	118 569	27	109 831	27	108 991	30	111 987	31	111 974
100	20	65	9	25	127 149	24	124 291	23	130 934	21	126 659	24	127 267	23	127 261
100	5	20	9	57	40 539	55	40 958	50	41 029	53	40 811	55	40 841	55	40 849
100	5	22	15	63	119 266	77	128 354	60	119 434	60	119 158	65	121 570	65	121 555
100	5	46	15	75	202 238	80	202 607	67	204 110	67	204 730	72	203 422	73	203 437
100	5	48	9	72	193 383	78	196 893	62	191 712	62	191 888	69	193 471	69	193 487
100	5	64	15	71	141 407	66	141 882	62	144 972	61	143 956	65	143 068	65	143 073
100	5	64	9	71	102 439	67	107 014	61	102 777	61	101 297	65	103 385	66	103 399
200	10	128	15	71	180 812	78	198 378	62	178 264	60	178 375	68	183 960	68	183 969
200	10	135	9	216	105 593	216	93 426	216	99 375	186	103 561	209	100 508	209	100 492
200	10	50	15	66	189 660	75	183 673	63	191 856	62	190 956	67	189 042	67	189 045
200	10	50	9	66	251 158	70	250 732	65	250 075	64	250 850	67	250 721	67	250 717
200	10	84	9	70	224 121	66	222 976	69	226 666	66	222 655	69	224 121	68	224 110
200	10	85	15	65	304 277	68	301 357	61	306 949	62	302 064	65	303 677	64	303 682
200	20	145	15	36	275 983	46	277 097	36	278 199	35	272 504	39	275 947	39	275 956
200	20	150	9	183	92 821	183	95 667	186	91 461	177	92 567	183	93 146	183	93 143
200	20	54	15	37	295 786	41	290 656	39	299 993	34	298 822	38	296 334	39	296 330
200	20	55	9	37	230 150	37	232 766	38	231 094	36	223 879	38	229 484	38	229 486
200	20	97	15	49	290 399	69	346 527	42	280 951	42	277 860	51	298 948	51	298 935
200	20	97	9	35	273 378	43	282 379	37	275 819	35	278 797	38	277 608	38	277 596
200	40	130	9	112	101 879	112	90 907	112	94 488	108	104 965	112	98 066	112	98 079
200	40	133	15	24	276 456	23	279 170	27	281 933	24	279 073	25	279 167	25	279 178
200	40	45	15	31	260 738	32	269 623	25	248 717	23	256 687	29	258 946	28	258 942
200	40	45	9	22	270 758	23	276 416	26	273 632	25	270 428	25	272 819	24	272 824
200	40	90	9	24	290 028	20	294 909	26	287 694	24	298 340	24	292 752	24	292 758
200	40	91	15	19	249 909	35	250 843	25	257 927	23	241 492	26	250 059	26	250 049
	Mean:			54.61	176 498.58	57.69	178 117.31	51.94	176 197.97	49.39	176 027.19	53.92	176 719.25	53.83	176 719.44

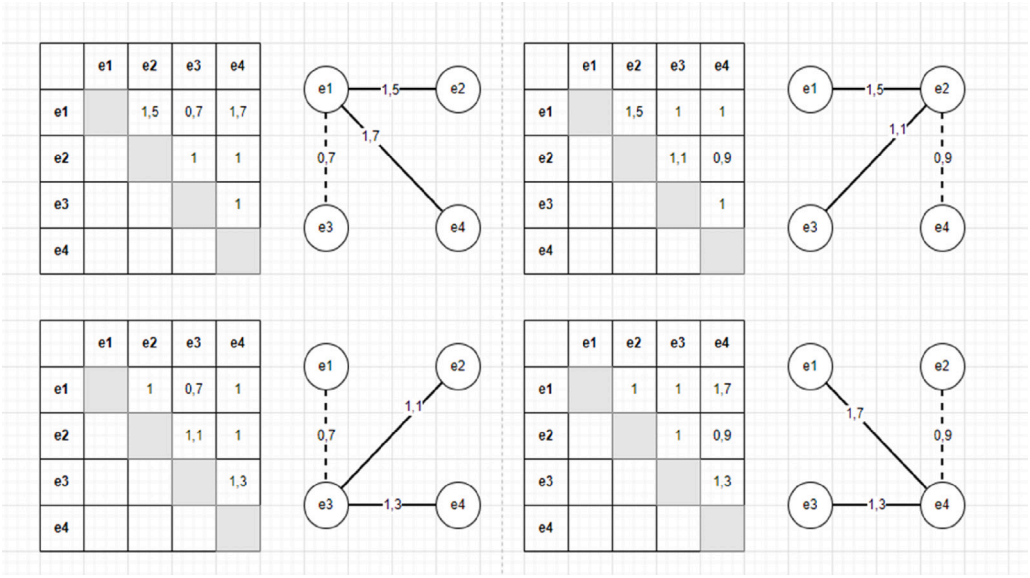


Fig. 5. Possible dominant synergy networks.

Table 6
Skills versus personality types.

(S) Skill	Soft skills						Hard skill performances			
Domain	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
e_1 (D)	1.0	0.8	0.3	0.8	0.8	0.8	4.0	1.5	2.0	1.5
e_2 (I)	0.8	1.0	1.0	0.6	1.0	0.4	0.0	3.0	1.5	1.0
e_3 (S)	0.5	0.5	0.9	0.5	0.3	0.6	2.5	2.0	3.5	3.0
e_4 (C)	0.4	0.5	0.5	1.0	0.4	1.0	5.0	0.0	4.0	1.5

feature that has a special integration test-relevant part. This task set is generally for 4 people and is commonly used in almost all software development projects. This project was selected as an example of our method because the executive team includes four people with different DISC personalities. All of them took part in dedicated DISC training where their personalities were identified by a dedicated trainer.

Based on the applied DISC method, four synergy networks are specified. The personality types of four team members were selected according to the four DISC personality types: dominance (e_1), influence (e_2), steadiness (e_3), and conscientiousness (e_4). The six soft skills assessed were leadership ability (s_1), communication ability (s_2), team player attitude (s_3), problem-solving skills (s_4), interpersonal skills (s_5), and analytical thinking (s_6). The data were measured by a 10-point Likert scale and divided into ten parts to obtain normalized data. According to the literature (Geissler, 2014; Looi et al., 2011; Orense, 2024), employees whose personality type is dominant have the best leadership abilities, and those whose personality type is influential have the best communication skills and team player attitudes (see Table 6). While soft skills usually depend on personality type, hard skill performance usually depends on former experiences. These hard skills were the implemented functions per week (s_7), User Experience/User Interface (UX/UI) designs per week (s_8) UX/UI, documented functions and test results per week (s_9), and writing deploy and testing scripts per week (s_{10}).

Table 7 shows the skill-work requirements for each task.

According to Fig. 1, based on the former project results and DISC assessment questionnaire by the trainer, the following synergy matrix was specified (see Table 8).

Since the synergy matrix is symmetric and the diagonal values are 1.0, it is sufficient to specify the upper triangular part of the synergy matrix. Table 8 shows that there is positive synergy between employees who have dominance (D) and influence (I) or conscientiousness (C)

Table 7
Skilled-work requirements for each task.

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}
a_1	1.00	1.00	1.00	0.35	0.83	0.83	4.00	6.50	9.50	4.00
a_2	0.35	0.38	0.60	0.81	0.63	0.75	10.00	6.00	6.00	4.00
a_3	0.42	0.42	0.80	0.77	0.83	0.83	9.20	3.50	8.00	3.50
a_4	0.38	0.35	0.48	0.96	0.79	0.83	10.50	5.50	5.50	2.50
a_5	0.35	0.38	0.56	0.85	0.63	0.79	9.80	6.00	5.50	3.00
a_6	0.31	0.31	0.40	0.73	0.42	0.88	8.00	6.50	4.00	3.50
a_7	0.81	0.85	0.80	0.38	0.83	1.00	3.50	4.50	2.00	6.00
a_8	0.38	0.77	0.36	1.00	0.42	0.83	10.00	6.00	4.00	4.00
a_9	0.96	0.96	0.40	0.35	1.00	0.17	2.00	1.50	9.50	7.00

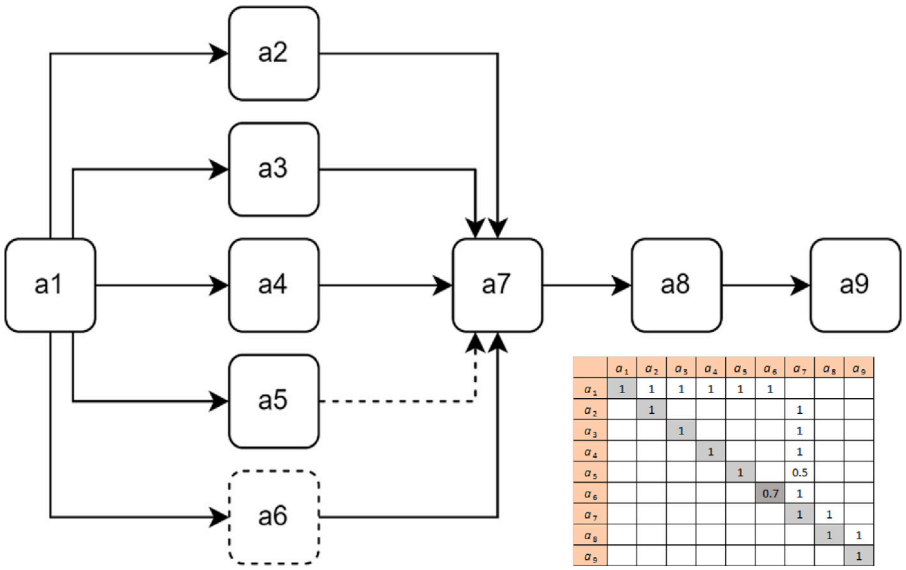
Table 8
Synergy matrix between employees whose personality types are categorized by DISC. (Only upper triangular are specified).

	e_1 (D)	e_2 (I)	e_3 (S)	e_4 (C)
e_1 (D)		1.5	0.7	1.7
e_2 (I)			1.1	0.9
e_3 (S)				1.3
e_4 (C)				

personality types, but there are negative synergies between dominance and steadiness personality types. Importantly, the specified synergy matrix is in line with the suggestion of the literature, as we saw in the introduction; however, to maintain generality, in the simulations, ± 20% are added randomly to every positive and negative synergy value. If we do not select a leader for this group, the team roles are selected autonomously. However, if we select a leader in a small group, only the synergy of the team leader and the other employees can be dominant; therefore, we obtain four different so-called *dominant synergy networks* (see Fig. 5, where only positive or negative synergies are noted).

The company followed several agile project management approaches. The employed SCRUM[6] methodology prescribed that the project be split into smaller subprojects (sprints). The examined sprint has nine tasks derived from a project template: design (a_1), implementation

[6] SCRUM is not an abbreviation.



	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
a_1	1	1	1	1	1	1			
a_2		1					1		
a_3			1				1		
a_4				1			1		
a_5					1		0.5		
a_6						0.7	1		
a_7							1	1	
a_8								1	1
a_9									1

Fig. 6. Logical structure of task precedences.

of function A (a_2), implementation of function B (a_3), implementation of function C (a_4), implementation of function D (a_5), implementation of extra function E (a_6), testing function E (a_7), maintenance/improvements (a_8), and integration (a_9). This template is used in most sprints. The priorities are determined by Dynamic System Development Method (DSDM), where mandatory tasks have relative priority, i.e., a task score of 1. Lower priorities have lower task scores. The flexible dependencies came from the technology. The logic network is specified in Fig. 6.

Fig. 6 shows that there is only one supplementary task (a_6) and one flexible dependency (a_5, a_7). The matching domain, which contains the maximal rates of assignments, was one in each cell.

The simulation specified five team roles, four target functions, and three relative constraints ($C_x\% \in \{C_s\%, C_t\%, C_c\%\}$).

The relative constraints are calculated by the minimal and maximal requirements as follows:

$$C_x\% = \frac{\mathrm{TPX}_{\max} - C_x}{\mathrm{TPX}_{\max} - \mathrm{TPX}_{\min}} \tag{18}$$

where $C_x \in [\mathrm{TPX}_{\max}, \mathrm{TPX}_{\min}]$, TPX ∈ {TPT,TPC,TPS}. If employees are not assigned to any of the tasks, $\mathrm{TPT}_{\max} = \infty$, or $\mathrm{TPC}_{\min} = 0$, the minimal assignment is specified as half of the maximal assignments ($e^{minw} = e^{maxw}/2$). In this way, $\mathrm{TPT}_{\max}$ and $\mathrm{TPC}_{\min}$ can be calculated. In this simulation, $C_x\% \in \{0.5, 0.6, \ldots, 1.0\}$. It provides 6^3 kinds of constraint settings. In every setting, we specified 100 simulations to consider the sensitivity of the estimation of synergy values. Therefore, we obtain $6^3 \cdot 5 \cdot 4 \cdot 100 = 432,000$ SMM matrices.

The goal of the simulations was to answer the research question (RQ). In this way, we calculated the effect of synergies on various kinds of team roles and compared them with those of autonomous selection. According to the case study, the following additional specific subquestions can be stated.

- Sub RQ_1 Is there any significance in considering synergies?
- Sub RQ_2 According to the constraints and target functions, which team role selection provides the best solutions?
- Sub RQ_3 Which team role selection is the least sensitive to changes in constraints?

5. Results and discussion

Related to the main RQ: *"Which kind of team role selection mechanism provides the shortest project duration considering the constraints?"* the first

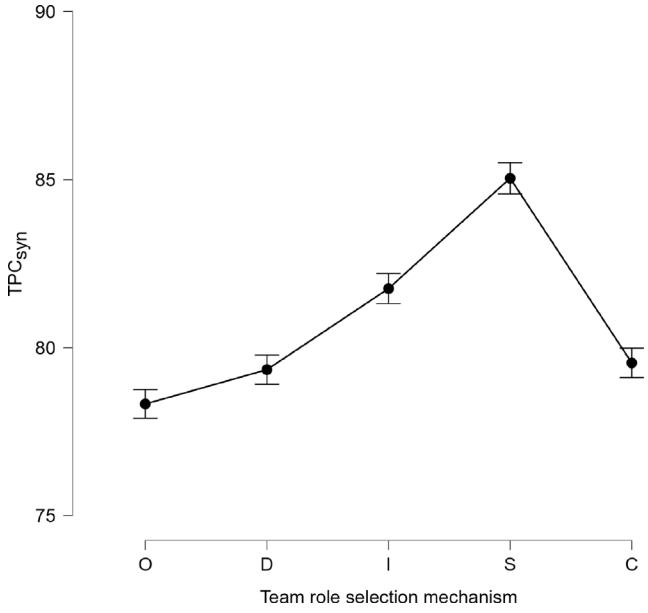


Fig. 7. Project costs for various kinds of team role selection mechanisms where there is no constraint ($C_c\% = C_t\% = 1, C_s\% = 0$).

Sub RQ_1 has to be answered: *"Is there any significance of considering synergies? "*

Table 9(a) shows the results of the descriptive statistics of project cost (TPC, 1000 EUR) and duration (TPT, week), and Table 9(b) shows the results of the pairwise t test between considering (syn) and neglecting (nosyn) synergies.

Table 9(a) shows that considering positive and negative synergies may reduce both the project duration (TPT) and the project cost (TPC). The expected value of TPT is more than two weeks unless synergies are considered. Considering the synergy between employees may reduce the costs to 4146 EUR. Nevertheless, Table 9(b) shows that only the difference in project costs is significant. Therefore, only TPC is examined.

The subquestion Sub RQ_2 *"According to the constraints and target functions, which team role selection provides the best solutions?"* is strongly

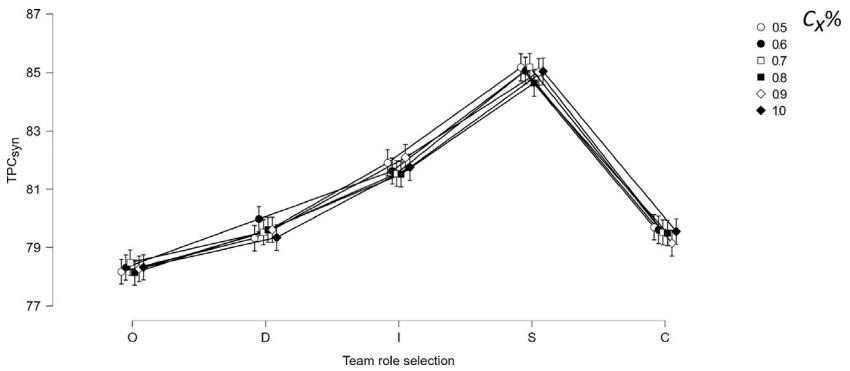


Fig. 8. Project costs for various kinds of team role selection mechanisms, where the constraint ($C_c\% = C_t\% = 1 - C_s\%$) is equal.

Table 9
Comparison of TPT and TPC.

(a) Descriptive statistics

	N	Mean	SD	SE
TPT_{syn}	432,000	1.947	1.280	1.947
TPT_{nosyn}	432,000	2.025	1.331	2.025
TPC_{syn}	432,000	78.932	7.588	1.155
TPC_{nosyn}	432,000	83.078	8.079	1.232

(b) Paired samples T-Test

Measure 1		Measure 2	t	df	*p*
TPT_{syn}	–	TPT_{nosyn}	−1.000	431,999	0.317
TPC_{syn}	–	TPC_{nosyn}	−52.409	431,999	< .001

Note. Student's t-test.

related to the original RQ. Table 10 shows the results of ANAlysis Of VAriance (ANOVA).

Table 10 shows that only the team role selection variable is significant for project cost. Neither the target function nor the constraints are significant.

Fig. 7 shows the TPC_{syn} values if the relative cost ($C_c\%$), time ($C_t\%$), and score/scope ($C_s\%$) constraints are neglected.

The lowest cost occurs if autonomous (O) team role selection is employed, while the greatest cost occurs if such an employee leads a team whose personality type is stable (S).

Although Table 10 shows that the chancing constraints have no significant effect on the project cost, according to subquestion 3 Sub RQ_3, it is important to answer the following question: "Which team role selection is the least sensitive to the changes in the constraints?" The Bartlett test showed that changing constraints have no significant effect on the variance of the project cost.

Fig. 8 also shows that constraints do not influence the confidence interval of the project cost.

6. Summary and conclusion

In this study, we propose a modification of the SSPSP problem to distinguish between soft skills and hard skills. We showed how to examine the team role selection mechanism for project time and project cost. To address this problem, a HGA algorithm with multichromosomes was proposed. The simulations were based on a real-life project, and the case study showed how to fill the modified SMM matrix. Answering RQ, a case study showed that autonomous team role selection provides the lowest project cost. The case study presented in this manuscript demonstrates the effectiveness of the proposed method in the R&D sector of a large high-tech automotive company. The results show that the autonomous team role selection mechanism can significantly reduce project costs while maintaining project quality.

The purpose of this study is for scholars to propose a method that provides a new approach to modeling soft and hard skills in flexible projects, which can help scholars better understand the impact of team role selection on project performance. Additionally, the modified SMM matrix and extended SSPSP can be used as a basis for future research in this area. For managers, the proposed method provides a practical tool for project management. The method showed how, in the case of flexible projects, the autonomous group role can help reduce costs for different personality types. This method can also be used to test other projects; therefore, in later research, we can answer the question of which projects should rely on autonomous selection. By using the autonomous team role selection mechanism, managers can optimize project duration while maintaining project quality. This can lead to significant cost savings and increased efficiency in project management.

CRediT authorship contribution statement

Zsolt T. Kosztyán: Writing – review & editing, Writing – original draft, Software, Methodology, Formal analysis, Conceptualization. **Péter Harta:** Writing – review & editing, Writing – original draft, Visualization, Supervision, Resources, Project administration, Methodology, Investigation, Formal analysis. **István Szalkai:** Writing – review & editing, Writing – original draft, Validation, Formal analysis, Data curation.

Declaration of competing interest

We wish to confirm that there are no known conflicts of interest associated with this publication and that there has been no significant financial support for this work that could have influenced its outcome.

Data availability

Data will be made available on request.

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. Zsolt T. Kosztyán's research contribution supported by Project no. K 142395 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.

Table 10
The effect of team role selection considering the various kinds of target function.

ANOVA - TPC_{syn}	Sum of squares	df	Mean square	F	p
Team role	6,123,182	4	1,534,457	2.655	0.031
Target	1,141,542	3	379,888	0.660	0.577
C_t%	3,254,545	5	649,867	1.129	0.342
C_c%	3,280,458	5	655,003	1.138	0.338
C_s%	2,430,254	5	485,072	0.842	0.519
Residuals	2.49e+11	431,977	575,796		

References

Al-Saqqa, S., Sawalha, S., & AbdelNabi, H. (2020). Agile software development: Methodologies and trends. *International Journal of Interactive Mobile Technologies*, *14*(11), http://dx.doi.org/10.3991/ijim.v14i11.13269.

Alba, E., & Chicano, J. F. (2007). Software project management with {GAs}. *Information Sciences*, *177*(11), 2380–2401. http://dx.doi.org/10.1016/j.ins.2006.12.020.

Antoniou, A. (2019). Compatibility of small team personalities in computer-based tasks. *Challenges*, *10*(1), 29. http://dx.doi.org/10.3390/challe10010029.

Arashpour, M., Kamat, V., Bai, Y., Wakefield, R., & Abbasi, B. (2018). Optimization modeling of multi-skilled resources in prefabrication: Theorizing cost analysis of process integration in off-site construction. *Automation in Construction*, *95*, 1–9. http://dx.doi.org/10.1016/j.autcon.2018.07.027.

Aslam, W., & Ijaz, F. (2018). A quantitative framework for task allocation in distributed agile software development. *IEEE Access*, *6*, 15380–15390. http://dx.doi.org/10.1109/ACCESS.2018.2803685.

Belbin, R. M. (2012). *Team roles at work*. Routledge.

Birjandi, A., Mousavi, S. M., Hajirezaie, M., & Vahdani, B. (2019). Optimizing and solving project scheduling problem for flexible networks with multiple routes in production environments. *Journal of Quality Engineering and Production Optimization*, *4*(1), 175–196. http://dx.doi.org/10.22070/jqepo.2019.3870.1091.

Browning, T. R. (2014). Managing complex project process models with a process architecture framework. *International Journal of Project Management*, *32*(2), 229–241. http://dx.doi.org/10.1016/j.ijproman.2013.05.008.

Capretz, L. F. (2003). Personality types in software engineering. *International Journal of Human-Computer Studies*, *58*(2), 207–214. http://dx.doi.org/10.1016/S1071-5819(02)00137-4.

Capretz, L. F., & Ahmed, F. (2010). Why do we need personality diversity in software engineering? *ACM SIGSOFT Software Engineering Notes*, *35*(2), 1–11. http://dx.doi.org/10.1145/1734103.1734111.

Chang, C. K., Jiang, H.-y., Di, Y., Zhu, D., & Ge, Y. (2008). Time-line based model for software project scheduling with genetic algorithms. *Information and Software Technology*, *50*(11), 1142–1154. http://dx.doi.org/10.1016/j.infsof.2008.03.002.

Chen, R., Liang, C., Gu, D., & Leung, J. Y. (2017). A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution. *International Journal of Production Research*, *55*(21), 6207–6234. http://dx.doi.org/10.1080/00207543.2017.1326641.

Choi, K. S., Deek, F. P., & Im, I. (2008). Exploring the underlying aspects of pair programming: The impact of personality. *Information and Software Technology*, *50*(11), 1114–1126. http://dx.doi.org/10.1016/j.infsof.2007.11.002.

Crawford, B., Soto, R., Johnson, F., Monfroy, E., & Paredes, F. (2014). A max-min ant system algorithm to solve the software project scheduling problem. *Expert Systems with Applications*, *41*(15), 6634–6645. http://dx.doi.org/10.1016/j.eswa.2014.05.003.

Curşeu, P. L., Meslec, N., Pluut, H., & Lucas, G. J. M. (2015). Cognitive synergy in groups and group-to-individual transfer of decision-making competencies. *Frontiers in Psychology*, *6*, 1375. http://dx.doi.org/10.3389/fpsyg.2015.01375.

Flores-Parra, J. M., Castañón-Puga, M., Evans, R. D., Rosales-Cisneros, R., & Gaxiola-Pacheco, C. (2018). Towards team formation using belbin role types and a social networks analysis approach. In *2018 IEEE technology and engineering management conference* (pp. 1–6). IEEE, http://dx.doi.org/10.1109/TEMSCON.2018.8488386.

Geissler, D. L. (2014). *The black and white effect of being labeled: the influence of being labeled by the DiSC personality model from the perspective of the labeled participants*.

Hardiman, L. T. (1997). Personality types and software engineers. *Computer*, *30*(10), 10. http://dx.doi.org/10.1109/MC.1997.625290.

Kia, R., Shahnazari-Shahrezaei, P., & Zabihi, S. (2016). Solving a multi-objective mathematical model for a multi-skilled project scheduling problem by CPLEX solver. In *2016 IEEE international conference on industrial engineering and engineering management* (pp. 1220–1224). IEEE, http://dx.doi.org/10.1109/IEEM.2016.7798072.

Kosztyán, Z. T. (2015). Exact algorithm for matrix-based project planning problems. *Expert Systems with Applications*, *42*(9), 4460–4473. http://dx.doi.org/10.1016/j.eswa.2015.01.066.

Kosztyán, Z. T., Bogdány, E., Szalkai, I., & Kurbucz, M. T. (2022). Impacts of synergies on software project scheduling. *Annals of Operations Research*, *312*, 883–908. http://dx.doi.org/10.1007/s10479-021-04467-5.

Kosztyán, Z. T., Novák, G., Jakab, R., Szalkai, I., & Hegedűs, C. (2023). A matrix-based flexible project-planning library and indicators. *Expert Systems with Applications*, *216*, Article 119472. http://dx.doi.org/10.1016/j.eswa.2022.119472.

Larson, J. R., Jr. (2007). Deep diversity and strong synergy: Modeling the impact of variability in members' problem-solving strategies on group problem-solving performance. *Small Group Research*, *38*(3), 413–436. http://dx.doi.org/10.1177/1046496407301972.

Larson, J. R., Jr. (2009). *In search of synergy in small group performance*. New York, US: Psychology Press, http://dx.doi.org/10.4324/9780203848784.

LePine, J. A., Buckman, B. R., Crawford, E. R., & Methot, J. R. (2011). A review of research on personality in teams: Accounting for pathways spanning levels of theory and analysis. *Human Resource Management Review*, *21*(4), 311–330. http://dx.doi.org/10.1016/j.hrmr.2010.10.004.

Looi, Q. E., See, S. L., Tay, C. S., & Ng, G. K. (2011). Understanding users by their DISC personality through interactive gaming. In *HCI international 2011–posters' extended abstracts: international conference, HCI international 2011, Orlando, FL, USA, July 9-14, 2011, proceedings, part i 14* (pp. 312–316). Springer, http://dx.doi.org/10.1007/978-3-642-22098-2_63.

Luna, F., González-Álvarez, D. L., Chicano, F., & Vega-Rodríguez, M. A. (2014). The software project scheduling problem: A scalability analysis of multi-objective metaheuristics. *Applied Soft Computing*, *15*, 136–148. http://dx.doi.org/10.1016/j.asoc.2013.10.015.

Lykourentzou, I., Antoniou, A., Naudet, Y., & Dow, S. P. (2016). Personality matters: Balancing for personality types leads to better outcomes for crowd teams. In *Proceedings of the 19th ACM conference on computer-supported cooperative work & social computing* (pp. 260–273). http://dx.doi.org/10.1145/2818048.2819979.

Maghsoudlou, H., Afshar-Nadjafi, B., & Niaki, S. T. A. (2017). Multi-skilled project scheduling with level-dependent rework risk; three multi-objective mechanisms based on cuckoo search. *Applied Soft Computing*, *54*, 46–61. http://dx.doi.org/10.1016/j.asoc.2017.01.024.

Marston, W. M. (1928). Review of a short outline of comparative psychology. *Journal of Abnormal and Social Psychology*, *23*(2), 256–257. http://dx.doi.org/10.1037/h0068416.

Meslec, N., & Curşeu, P. L. (2015). Are balanced groups better? Belbin roles in collaborative learning groups. *Learning and Individual Differences*, *39*, 81–88. http://dx.doi.org/10.1016/j.lindif.2015.03.020.

Mostert, N. M. (2015). Belbin–the way forward for innovation teams. *Journal of Creativity and Business Innovation*, *1*, URL https://www.journalcbi.com/belbin-for-innovation-teams.html.

Myers, I. B. (1962). *The Myers-Briggs Type Indicator: Manual (1962)*. Consulting Psychologists Press, http://dx.doi.org/10.1037/14404-000.

Myszkowski, P. B., Laszczyk, M., Nikulin, I., & Skowroński, M. (2019). IMOPSE: a library for bicriteria optimization in multi-skill resource-constrained project scheduling problem. *Soft Computing*, *23*(10), 3397–3410. http://dx.doi.org/10.1007/s00500-017-2997-5.

Myszkowski, P. B., Skowroński, M. E., Olech, Ł. P., & Oślizło, K. (2015). Hybrid ant colony optimization in solving multi-skill resource-constrained project scheduling problem. *Soft Computing*, *19*, 3599–3619. http://dx.doi.org/10.1007/s00500-014-1455-x.

Myszkowski, P. B., Skowroński, M. E., & Podlodowski, Ł. (2013). Novel heuristic solutions for multi-skill resource-constrained project scheduling problem. In *2013 federated conference on computer science and information systems* (pp. 159–166). IEEE.

Myszkowski, P. B., Skowroński, M. E., & Sikora, K. (2015). A new benchmark dataset for multi-skill resource-constrained project scheduling problem. In *2015 federated conference on computer science and information systems* fedCSIS, (pp. 129–138). IEEE, http://dx.doi.org/10.15439/2015F273.

Nikolaou, I., & Oostrom, J. K. (2015). Employee recruitment, selection, and assessment. *Contemporary Issues for Theory and Practice. Hove, East Sussex: Routledge*, http://dx.doi.org/10.4324/9781315742175.

Nosek, J. T. (1998). The case for collaborative programming. *Communications of the ACM*, *41*(3), 105–108. http://dx.doi.org/10.1145/272287.272333.

Orense, R. C. (2024). Correlation on the DISC personality profile and leadership styles of the student leaders of CAS in AY 2014–2015. (p. 90). Created Annually by San Beda College Alabang.

Oza, N., Fagerholm, F., & Münch, J. (2013). How does Kanban impact communication and collaboration in software engineering teams? In *2013 6th international workshop on cooperative and human aspects of software engineering* (pp. 125–128). IEEE, http://dx.doi.org/10.1109/CHASE.2013.6614747.

Paetsch, F., Eberlein, A., & Maurer, F. (2003). Requirements engineering and agile software development. In *WET iCE 2003. proceedings. twelfth IEEE international workshops on enabling technologies: infrastructure for collaborative enterprises, 2003* (pp. 308–313). IEEE, http://dx.doi.org/10.1109/ENABL.2003.1231428.

Peeters, M. A., Van Tuijl, H. F., Rutte, C. G., & Reymen, I. M. (2006). Personality and team performance: a meta-analysis. *European journal of personality*, *20*(5), 377–396. http://dx.doi.org/10.1002/per.588.

Peslak, A. R. (2006). The impact of personality on information technology team projects. In *Proceedings of the 2006 ACM SIGMIS CPR conference on computer personnel research: forty four years of computer personnel research: achievements, challenges & the future* (pp. 273–279). http://dx.doi.org/10.1145/1125170.1125233.

Pieterse, V., Leeu, M., & van Eekelen, M. (2018). How personality diversity influences team performance in student software engineering teams. In *2018 conference on information communications technology and society* (pp. 1–6). IEEE, http://dx.doi.org/10.1109/ICTAS.2018.8368749.

Reeves, C. R., & Wright, C. C. (1999). Genetic algorithms and the design of experiments. In L. D. Davis, K. De Jong, M. D. Vose, & L. D. Whitley (Eds.), *Evolutionary algorithms* (pp. 207–226). New York, NY: Springer New York, http://dx.doi.org/10.1007/978-1-4612-1542-4_12.

Scullard, M., & Baum, D. (2015). *Everything DiSC Manual*. John Wiley & Sons.

Sfetsos, P., Stamelos, I., Angelis, L., & Deligiannis, I. (2009). An experimental investigation of personality types impact on pair effectiveness in pair programming. *Empirical Software Engineering*, *14*, 187–226. http://dx.doi.org/10.1007/s10664-008-9093-5.

Shen, X.-N., Minku, L. L., Marturi, N., Guo, Y.-N., & Han, Y. (2018). A Q-learning-based memetic algorithm for multi-objective dynamic software project scheduling. *Information Sciences*, *428*, 1–29. http://dx.doi.org/10.1016/j.ins.2017.10.041.

Stylianou, C., & Andreou, A. S. (2012). A multi-objective genetic algorithm for software development team staffing based on personality types. In *Artificial intelligence applications and innovations: 8th IFIP wG 12.5 international conference, AIAI 2012, Halkidiki, Greece, September 27-30, 2012, proceedings, part i 8* (pp. 37–47). Springer, http://dx.doi.org/10.1007/978-3-642-33409-2_5.

Vega-Velázquez, M. Á., García-Nájera, A., & Cervantes, H. (2018). A survey on the software project scheduling problem. *International Journal of Production Economics*, *202*, 145–161. http://dx.doi.org/10.1016/j.ijpe.2018.04.020.

Walle, T., & Hannay, J. E. (2009). Personality and the nature of collaboration in pair programming. In *2009 3rd international symposium on empirical software engineering and measurement* (pp. 203–213). IEEE, http://dx.doi.org/10.1109/ESEM.2009.5315996.

Wang, M., Liu, G., & Lin, X. (2022). Dynamic optimization of the multi-skilled resource-constrained project scheduling problem with uncertainty in resource availability. *Mathematics*, *10*(17), 3070. http://dx.doi.org/10.3390/math10173070.

Whitehead, J. (2007). Collaboration in software engineering: A roadmap. In *Future of software engineering* (pp. 214–225). IEEE, http://dx.doi.org/10.1109/FOSE.2007.4.

Winter, B. (2015). *Agile performance improvement: The new synergy of agile and human performance technology*. Springer, http://dx.doi.org/10.1007/978-1-4842-0892-2.

Wongwai, N., & Malaikrisanachalee, S. (2011). Augmented heuristic algorithm for multi-skilled resource scheduling. *Automation in Construction*, *20*(4), 429–445. http://dx.doi.org/10.1016/j.autcon.2010.11.012.

Wysocki, R. K. (2019). *Effective project management: traditional, agile, extreme*. John Wiley & Sons, http://dx.doi.org/10.1002/9781119562757.

Xiao, J., Ao, X. T., & Tang, Y. (2013). Solving software project scheduling problems with ant colony optimization. *Computers & Operations Research*, *40*(1), 33–46. http://dx.doi.org/10.1016/j.cor.2012.05.007.