Hybrid time-quality-cost trade-off problems

Zsolt T. Kosztyán[1], István Szalkai[2]

[1]*Department of Quantitative Methods,* [2]*Department of Mathematics,*
[1−2]*University of Pannonia, Hungary*

Abstract

Agile and hybrid project management has become increasingly popular among practitioners, particularly in the IT sector. In contrast to the theoretically and algorithmically well-established and developed time-cost and time-quality-cost project management methods, agile and hybrid project management lacks a principle foundation and algorithmic treatment. The aim of this paper is to fill this gap. We propose a matrix-based method that provides scores for alternative project plans that host flexible task dependencies and undecided, supplementary task completion while also covering traditional time-quality-cost trade-off problems. The proposed method can bridge the agile and traditional approaches.

Keywords: Time-quality-cost trade-off problems, Hybrid project management approaches, Matrix-based project planning

1. Introduction

The importance of time-cost trade-off problems was recognized over five decades ago, with the nearly simultaneous development of project planning techniques [1]. From the 1960s to the 1980s, continuous time-cost relationship problems were addressed extensively in the literature [see, e.g., 2, 3]. The discrete time-cost trade-off problem (DTCTP), which can be treated as a specific resource-allocation problem [4], is a well-known problem in the project management literature [see, e.g., 5, 6, 7]. At first, Ref. [8] suggested that the quality of a completed project may be affected by project crashing. They developed a solution procedure that considers trade-offs among time, cost and quality in a continuous mode. Since discrete time-cost trade-off problems (DTCTP) are NP-hard problems, discrete time-quality-cost trade-off problems (DTQCTP) are also NP-hard problems and are therefore usually solved using heuristic or meta-heuristic methods. However, continuous versions of these problems can usually be solved within a polynomial computational time (e.g., in the case of linear trade-off functions between time-cost and time-quality) [9]. All of these problems assume a fixed-logic plan, whereas recent project management (e.g., agile and hybrid) approaches allow for the restructuring or reorganization of the project. They approaches apply flexible-logic plans instead of fixed-logic plans. This paper extends the traditional trade-off problem to address flexible project plans.

Continuous and discrete versions of time-cost and time-quality-cost trade-off analyses assume that the time, cost and quality of an option within an activity are deterministic. However, the time, quality and cost may be uncertain. The stochastic versions of time-cost and time-quality-cost trade-off problems [see, e.g., 10, 11] treat time, quality and cost as uncertain parameters. In the proposed method, the task (e.g., time/cost/resource) demands are not uncertain, but the logical structure is. The proposed model can address uncertainty regarding supplementary task completion and/or uncertain or flexible dependencies.

It is interesting to combine uncertain task durations, uncertain cost demands, uncertain quality parameters (undecided), supplementary task completion and uncertain or flexible task dependencies into one stochastic model; however, this paper mainly focuses on how to extend continuous and discrete time-quality-cost trade-off methods to treat flexible dependencies and (undecided or uncertain) supplementary task completion.

Every traditional trade-off method assumes an accepted logic plan by which the tasks and the dependencies between them are determined. However, several project management approaches, e.g., agile and extreme project management (see Ref. [12]), allow for one to restructure or reorganize the project plan in response to changes in the client's demands.

Wysocki found in a 2009 study of the practices of software project managers that only 20% of IT projects were managed using a traditional project management (TPM) methodology. Methods for investment and construction projects usually cannot be directly applied to software development or R&D projects, as these are managed using agile project management (APM) approaches. Currently, hybrid (i.e., combinations of traditional and agile) approaches are becoming increasingly popular [see, e.g., 13, 14]. However, these approaches lack a principled foundation and algorithmic treatment. The aim of this paper is to fill this gap.

Whereas a project manager who follows a TPM approach uses TCTP/TQCTP method(s) to reduce task duration, an agile project manager tries to restructure the project. The project duration can be reduced without increasing the project cost by reducing the number of flexible dependencies. However, in real project situations, most dependencies are fixed; therefore, the TPM and APM approaches should be integrated.

There are different combinations of agile and traditional project management approaches [see, e.g., 15, 16, 13]. However, to the best of our knowledge, there is no exact algorithm that can be used to in hybrid time-quality-cost trade-off problems. Nevertheless, production development and IT projects, such as introducing and setting up new information systems, may require that part of the project be reorganized, particularly in the development phase. However, decreasing the time demands of mandatory tasks may also be an important requirement. Neither the agile nor the traditional approach can address this situation properly. Traditional approaches, or network-based methods, assume static logic plans, but the reorganization of projects may produce insufficient reductions in project duration and/or supplementary tasks, and important tasks may be excluded from the project due to budget constraints and/or project deadlines. A hybrid project management (HPM) approach may combine the traditional and agile approaches; however, HPM approaches are not yet supported by project planning methods. The proposed algorithm combines the agile and traditional approaches. This method extends traditional time-cost and time-quality trade-off methods by allowing for the restructuring and reorganizing of projects.

The proposed hybrid time-cost and hybrid time-quality-cost trade-off models manage flexible project plans and allow us to restructure or reorganize these project plans to satisfy customer and management demands. In contrast to the traditional project scoring and selection methods, there is no need to specify all project alternatives to select the most desirable project scenario or the one with the shortest duration or lowest cost.

To handle flexible project plans, matrix-based techniques will be used instead of traditional network-based project planning techniques.

The basis of the proposed methods is a matrix-based method, the project domain matrix (PDM) [see 17]. The PDM is an n by m matrix, where n is the number of tasks, $m = n+t+c+q+r$, t is the number of possible durations, c is the number of possible (direct) costs, q is the number of possible quality parameters, and r is the number of possible resource demands of tasks.

The PDM has five domains. The first domain is the logic domain (LD), which is described as an n by n project expert matrix (PEM) [see 18] or numerical dependency structure matrix (NDSM)[see 19][1]. Since the PEM

[1]The NDSM does not represent supplementary tasks but can represent flexible depen-

has specified and semi-specified versions, the PDM is *specified* if and only if the LD is specified; otherwise, the PDM is *semi-specified*.

The other domains are the time domain (TD), cost domain (CD), quality domain (QD) and resource domain (RD). If the demands are deterministic, we say that the PDM is *deterministic*; otherwise, the PDM is *non-deterministic*. In this study, the deterministic versions of hybrid time-quality-cost trade-off problems are considered: the TD, CD, QD, and RD contain deterministic values but at least two completion modes. Therefore, this version is a semi-specified, deterministic, multi-modal PDM.

Whereas the basis of the proposed model is the PDM, the basis of the proposed method is the expert project ranking (EPR) algorithm [see 17], which can evaluate specified and semi-specified deterministic PDMs. However, that method cannot address the trade-off problem. Therefore, although EPR can be used to schedule a flexible project plan and can thus be used in agile project management approaches, it cannot address trade-offs between time and cost or between time and quality and therefore cannot be used in hybrid project management directly.

This paper proposes a hybrid time-quality-cost trade-off model to bridge APM and TPM.

The proposed hybrid algorithm combines the features of EPR and time-quality-cost trade-off problems to solve hybrid time-quality-cost trade-off problems.

The proposed algorithm can be used not only for project planning but also for project risk management. Despite risk management and mitigation not being the main focus of this paper, in the section of the simulation beyond traditional risk management, in which project networks are usually assumed to have a fixed logic plan [20] or be a result of a negotiation [21], it was possible to measure the effect of the ratio of flexible dependencies and the ratio of uncertain (supplementary) task completions. The use of flexible dependencies and supplementary tasks enables us to model and compare different project management approaches.

The paper organized as follows: after this section, in Section 2, the mathematical background is described. In Section 3, we present the proposed algorithm, and different types of project management approaches are modeled and compared. In the last section (Section 4), we summarize the conclusions and discuss the limitations of the proposed algorithm and future

dencies; however, the PEM can represent both flexible dependencies and supplementary tasks directions.

2. Solving hybrid time-quality-cost trade-off problems

In this section, a (resource-constrained) hybrid time-quality-cost trade-off problem (RC-HTQCTP) is first specified. Then, a matrix-based model representation is proposed. At the end of this section, an exact algorithm for a hybrid continuous time-quality-cost trade-off problem is proposed. The decisions for finding the optimum will be directed by *score functions* and *matrices* (P, Q) and time-quality-cost functions; thus, we need several definitions and notations before proceeding.

2.1. Definitions and problem statements

In the proposed model, mandatory and supplementary activities are distinguished.

Definition 1. *We call any finite set* $A = \{a_1, ..., a_n\}$ *the set of* ***possible activities*** *or* ***tasks*** *in the project. The subset of* ***supplementary task*** *is* $\widetilde{A} = \{\widetilde{a}_1, ..., \widetilde{a}_\sigma\} \subseteq A$*, where* $\widetilde{A}$ *is any fixed subset of* A*. Then,* $\overline{A} = A \setminus \widetilde{A}$ *is the subset of* ***mandatory tasks****.*

Whereas mandatory (or high-priority) tasks must be realized, supplementary (or lower-priority) tasks can be omitted from the project or postponed to the next or another project. Decisions about *supplementary* task realization always have two options: to include or to exclude.

S denotes the set of tasks that will be fulfilled by the algorithm (furthermore called as *project scenario*). The number of possible project scenarios is 2^σ, where $\sigma = \left|\widetilde{A}\right|$.

Definition 2. *Any function* $P : A \rightarrow [0, 1]$ *is called the* ***score function of*** *task* ***inclusion*** *if* $P(a_i) = 1$ *for* $a_i \in \overline{A}$ *and* $P(a_i) \in [0, 1)$ *for* $a_i \in \widetilde{A}$*.*
The function $Q : A \rightarrow [0, 1]$ *is called the* ***score function of*** *task* ***exclusion*** *if* $Q(a_i) = 0$ *for* $a_i \in \overline{A}$ *and* $Q(a_i) \in (0, 1]$ *for* $a_i \in \widetilde{A}$*.*

The task inclusion and exclusion scores can mean probability, importance or relative priority values.

Example 1. *If every task completion (inclusion) score is a probability value, then* $Q = 1 - P$*.*

Definition 3. *For any associative and* monotone[2] *operation $\otimes$ on $\mathbb{R}^+$, we define the* ***aggregation function*** $\otimes : \Xi(A) \to \mathbb{R}$ *as*

$$\otimes(S) := \bigotimes_{a \in S} P(a) \ \otimes \bigotimes_{a \in A \setminus S} Q(a), \tag{1}$$

where $\otimes$ indicates Σ or Π (addition or multiplication, respectively) or $\vee$ or $\wedge$ (maximum or minimum, respectively).

Example 2. *If scores are probability values, then we have $\otimes = \Pi$.*
If scores indicate the importance of task completion, then $\otimes = \Sigma$.

The PHASE ONE of the proposed algorithm decides which tasks will be included in the project scenario by maximizing $\otimes(S)$, fulfilling certain *time*, *quality* and *cost* requirements. Nevertheless, the proposed model also treats flexible relations between two tasks. Therefore, in PHASE TWO of the proposed algorithm, we decide *the order in which* we will complete these tasks. This order is a *relation* of tasks, and our algorithm receives three types of relations (dependencies) as inputs: **no dependency**, **required** and **flexible**. We have to *resolve* (i.e., decide) all flexible relations.

Definition 4. *The relation triplet $(\prec, \sim, \bowtie)$ is a* ***relation representation*** *of a hybrid project plan if for any $i, j \leq n$ and $i \neq j$, we let*
(i) *$a_i \prec a_j$ represent the* ***strict*** *or* ***required*** *dependencies between tasks a_i and a_j, i.e., a_j may not be started unless activity a_i has been completed;*
(ii) *$a_i \sim a_j$ represents* ***no dependency*** *between tasks a_i and a_j, i.e., the starting time of a_j is not affected by a_i;*
(iii) *$a_i \bowtie a_j$ represents* ***flexible*** *dependencies between tasks a_i and a_j.*

Remark 1. *It is important to note that $\prec$is naturally a transitive and asymmetric relation, which excludes* circles *such as $a_1 \prec a_2 \prec ... \prec a_1$. Therefore, by a standard topological ordering algorithm, we may assume that $a_i \prec a_j \Rightarrow i < j$.*

Whereas *strict* dependency $a_i \prec a_j$ between tasks a_i and a_j must be realized (in a sequential manner) and $a_i \sim a_j$ means indifference (we may choose either sequential or parallel realization), *flexible* dependencies ($\bowtie$) must be resolved. In this case, we can decide whether these tasks will

[2]$\otimes$ is **monotone** if $x \leq y$ and $u \leq v$ implies $x \otimes u \leq y \otimes v$ for $x, y, u, v \in \mathbb{R}^+$.

be completed in a *sequential* manner, i.e., $a_i \prec a_j$ will be declared, or in a *parallel* manner, $a_i \sim a_j$. When we decide that these tasks should be completed in either a sequential or a parallel manner, we say that the flexible dependency is **decided** to realize (include) or to resolve (exclude). If every flexible relation is decided to realize or resolve, we obtain a *project structure.*

Proposition 1. *For any binary relation $\prec, \sim, \bowtie$ on A, the triplet $(\prec, \sim, \bowtie)$ is a relation representation of a hybrid project plan if and only if $\{\prec, \sim, \bowtie\}$ is a partition of $A \times A \setminus \iota$ (ι is the diagonal) and $\prec$ is a strict partial ordering.*

The easiest method to input and modify *all* data is by using a special *$n \times n$ matrix* $\mathbf{M}$, which we call a *hybrid logic plan.* ($[\mathbf{M}]_{i,j}$ denotes the *entry* in row i and column j.)

Definition 5. *The **matrix representation** of an **input** for a hybrid logic plan means that any $n \times n$ matrix with entries $\emptyset, X, ?$, i.e., $\mathbf{M} \in \{X, \emptyset, ?\}^{n \times n}$, assuming the following requirements:*
(i) *for any $i \le n$, $[\mathbf{M}]_{i,i} =$ "X" for $a_i \in \overline{A}$ and $[\mathbf{M}]_{i,i} =$ "?" for $a_i \in \widetilde{A}$;*
(ii) *for any $i, j \le n$, $i \ne j$ $[\mathbf{M}]_{i,j} =$ "X" $\iff a_i \prec a_j$, $[\mathbf{M}]_{i,j} =$ "$\emptyset$" $\iff a_i \sim a_j$ and $[\mathbf{M}]_{i,j} =$ "?" $\iff a_i \bowtie a_j$.*

On the diagonal, $[\mathbf{M}]_{i,i} =$ "X", "$\emptyset$", "?" represent whether a task a_i will be executed, will not be executed, or will be decided later, respectively. Similarly, $[\mathbf{M}]_{i,j} =$ "?" for any $i \ne j$ indicates a flexible task dependency, about which we have to decide in the algorithm[3]. The algorithm will change "?" to either "X" or "$\emptyset$" in $\mathbf{M}$ step-by-step on the diagonal in PHASE ONE and on the off-diagonal in PHASE TWO.

From Remark 1, we know that considering the entries X, $\mathbf{M}$ is an upper triangular matrix both in the input and in all subsequent steps of the algorithm. In what follows, $\mathbf{M}$ is any *such* matrix $\mathbf{M} \in \{X, \emptyset, ?\}^{n \times n}$.

Proposition 2. *For any realized project scenario $S \subseteq A$, the* diagonal *of the matrix representation does not contain "?": $\mathbf{M}_{i,i} =$ "X" for $a_i \in S$ and $\mathbf{M}_{i,i} =$ "$\emptyset$" for $a_i \in A \setminus S$. Furthermore, $\mathbf{M}$ does not contain "?" at all when all flexible dependencies are decided.*

[3]The reader is allowed to replace the symbols "$\emptyset$", "X" and "?" with any numbers. In Table 1, we use 0 for $\emptyset$, 1 for X and numbers from $(0, 1)$ for ?.

Example 3. *Let* $\mathbf{M}$ *be a hybrid project plan.*

$$\mathbf{M} = \begin{pmatrix} X & ? & \emptyset \\ \emptyset & ? & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix} \tag{2}$$

Considering diagonal values, two possible project scenarios ($\mathbf{M}'_1$ *and* $\mathbf{M}'_2$*) can be specified:*

$$\mathbf{M}'_1 = \begin{pmatrix} X & ? & \emptyset \\ \emptyset & X & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix}, \mathbf{M}'_2 = \begin{pmatrix} X & ? & \emptyset \\ \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix} \tag{3}$$

Based on $\mathbf{M}'_1$*, two types of project structures can be specified (see* $\mathbf{M}''_{11}$ *and* $\mathbf{M}''_{12}$*):*

$$\mathbf{M}''_{11} = \begin{pmatrix} X & X & \emptyset \\ \emptyset & X & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix}, \mathbf{M}''_{12} = \begin{pmatrix} X & \emptyset & \emptyset \\ \emptyset & X & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix} \tag{4}$$

Since we plan to omit all tasks $a \in A \setminus S$, we might also omit the rows and columns of $\mathbf{M}$ that do not belong to S, which might be important for computational purposes (saving time and memory) and could be defined axiomatically. We leave these details to the reader.

Example 4.

$$\mathbf{M}'_1 = \begin{pmatrix} X & ? & \emptyset \\ \emptyset & X & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix} := \begin{pmatrix} X & ? \\ \emptyset & ? \end{pmatrix}, \mathbf{M}'_2 = \begin{pmatrix} X & ? & \emptyset \\ \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset \end{pmatrix} := (X) \tag{5}$$

Similarly to Definition 2, we are given *score values* for each dependency.

Definition 6. *Any two* matrices $\mathbf{P}, \mathbf{Q} \in [0,1]^{n \times n}$ *are called* ***score functions*** *of the* input *matrix* $\mathbf{M}$ *if*
(i) *for every* $i \leq n$*, we have* $[\mathbf{P}]_{i,i} = P(a_i)$ *and* $[\mathbf{Q}]_{i,i} = Q(a_i)$*;*[4]
(ii) *for every* $i, j \leq n$*,* $i \neq j$ *we have*
$[\mathbf{M}]_{i,j} = \text{“}X\text{”} \iff [\mathbf{P}]_{i,j} = 1$ *and* $[\mathbf{Q}]_{i,j} = 0$*,*

$[\mathbf{M}]_{i,j} = \text{“}\emptyset\text{”} \iff [\mathbf{P}]_{i,j} = 0$ *and* $[\mathbf{Q}]_{i,j} = 1$*, and*

$[\mathbf{M}]_{i,j} = \text{“?”} \iff 0 < [\mathbf{P}]_{i,j}, [\mathbf{Q}]_{i,j} < 1$.

[4] P and Q were introduced in Definition 2.

Let us emphasize that (i) and (ii) above are requirements for the input matrix $\mathbf{M}$*. Whereas the "?" entries in* $\mathbf{M}$ *change during the algorithm run,* $\mathbf{P}$ *and* $\mathbf{Q}$ *remain unchanged.*

The off-diagonal values of $[\mathbf{P}]_{i,j}$ represent the score of realizing the task dependency between a_i and a_j, whereas $[\mathbf{Q}]_{i,j}$ represents the score of resolving the task dependency between a_i and a_j.

Example 5. *Suppose that diagonal of the matrix* $\mathbf{P}$ *represents the probability values of tasks in matrix* $\mathbf{M}$*. Further suppose that* $[P]_{i,j}, i \neq j$ *elements represent the probability values of the dependency between task* i *and task* j*. In this case, we can assume that* $\mathbf{Q} = \mathbf{1} - \mathbf{P}$*. Eq.(6) shows possible score matrices for* $\mathbf{M}$*.*

$$\mathbf{M} = \begin{pmatrix} X & ? \\ \emptyset & ? \end{pmatrix}, \mathbf{P} = \begin{pmatrix} 1 & 0.7 \\ 0 & 0.6 \end{pmatrix}, \mathbf{Q} = \mathbf{1} - \mathbf{P} = \begin{pmatrix} 1 & 0.3 \\ 1 & 0.4 \end{pmatrix} \tag{6}$$

Corollary 1. *If* $\mathbf{M}$ *contains no "?" in the diagonal, then for the represented scenario* $S \subseteq A$*, we have*

$$\otimes (S) = \otimes_{diag} (\mathbf{M}) . \tag{7}$$

From the diagonal of $\mathbf{M}$, we can provide sharp bounds on $\otimes (S)$ (see Eq. (1)) in each step of the algorithm.

Example 6. *In the case* $\otimes = \prod$*,* $\otimes_{diag}^{\max} (\mathbf{M}) = 1 \cdot \max(0.6, 0.4) = 0.6$*, which is a score value of scenario* $\mathbf{M}'_1$*, whereas* $\otimes_{diag}^{\min} (\mathbf{M}) = 1 \cdot \min(0.6, 0.4) = 0.4$*, which is a score value of* $\mathbf{M}'_2$*.*

Definition 7. *For any* $\mathbf{M}, \mathbf{N} \in \{X, \emptyset, ?\}^{n \times n}$*, we say that*
(i) $\mathbf{N}$ *is an* ***in-/out-****diagonal* ***extension*** *of* $\mathbf{M}$ *if all the symbols "*X*" and "*$\emptyset$*" in* $\mathbf{M}$ *remain unchanged in* $\mathbf{N}$ *and some (possibly none) of the "?"* inside/outside *of the diagonal of* $\mathbf{M}$ *are changed to "*X*" or "*$\emptyset$*". In this case,* $\mathbf{M}$ *is a* ***restriction*** *of* $\mathbf{N}$*.*
(ii) *For* $i, j \leq n$ *and* $[\mathbf{M}]_{i,j} =$ *"?", we denote by* $\mathbf{M}[i, j = X]$ *and* $\mathbf{M}[i, j = \emptyset]$ *the matrices if only* $\mathbf{M}_{i,j}$ *has been changed to either "*X*" or "*$\emptyset$*".*
(iii) $\mathbf{N}$ *is an* ***in-/out-*** *diagonal* ***closure*** *of* $\mathbf{M}$ *if* $\mathbf{N}$ *contains no "?"s in/out of the diagonal and* $\mathbf{N}$ *in/out extends* $\mathbf{M}$*.*

Remark 2. *Note that every possible project scenario and every possible project structure are restrictions of the hybrid project plan.*

When replacing a "?", the algorithm makes newer *extensions* of the recent (or input) matrix, and our goal in PHASE ONE and PHASE TWO are suitable *in-* and *out-closures* of the input matrix, respectively. *In-closures* represent scenarios, whereas *out-closures* decide all flexible dependencies.

Definition 8.

$$\begin{aligned}\otimes_{diag}(\mathbf{M}) &:= \bigotimes_{[\mathbf{M}]_{i,i}=\text{"}X\text{"}} P(a_i) \otimes \bigotimes_{[\mathbf{M}]_{i,i}=\text{"}\emptyset\text{"}} Q(a_i), &(8)\\ \otimes^{\min}_{diag}(\mathbf{M}) &:= \otimes_{diag}(\mathbf{M}) \otimes \bigotimes_{[\mathbf{M}]_{i,i}=\text{"?"}} \min\{P(a_i), Q(a_i)\}, &(9)\\ \otimes^{\max}_{diag}(\mathbf{M}) &:= \otimes_{diag}(\mathbf{M}) \otimes \bigotimes_{[\mathbf{M}]_{i,i}=\text{"?"}} \max\{P(a_i), Q(a_i)\}. &(10)\end{aligned}$$

For some $\otimes$ (e.g., multiplying probabilities), $\otimes^{\max}_{diag}(\mathbf{M})$ may be less than $\otimes_{diag}(\mathbf{M})$ (see also Definition 6). However, we have the following inequalities:

Theorem 1. *For any* $\mathbf{M}$*,*

$$\otimes^{\min}_{diag}(\mathbf{M}) \leq \otimes^{\max}_{diag}(\mathbf{M}). \tag{11}$$

For any in-extension $\mathbf{N}$ *of* $\mathbf{M}$*,*

$$\otimes^{\min}_{diag}(\mathbf{M}) \leq \otimes^{\min}_{diag}(\mathbf{N}) \leq \otimes^{\max}_{diag}(\mathbf{N}) \leq \otimes^{\max}_{diag}(\mathbf{M}), \tag{12}$$

and for any $i \leq n$*,*

$$\begin{aligned}\otimes^{\min}_{diag}(\mathbf{M}) &= \min\left\{\otimes^{\min}_{diag}(\mathbf{M}[i,i=X]) \ , \ \otimes^{\min}_{diag}(\mathbf{M}[i,i=\emptyset])\right\}, &(13)\\ \otimes^{\max}_{diag}(\mathbf{M}) &= \max\left\{\otimes^{\max}_{diag}(\mathbf{M}[i,i=X]) \ , \ \otimes^{\max}_{diag}(\mathbf{M}[i,i=\emptyset])\right\}. &(14)\end{aligned}$$

For any in-closure $\mathbf{N}$ *of* $\mathbf{M}$*,*

$$\otimes^{\min}_{diag}(\mathbf{N}) = \otimes_{diag}(\mathbf{N}) = \otimes^{\max}_{diag}(\mathbf{N}) \tag{15}$$

and

$$\otimes^{\min}_{diag}(\mathbf{M}) \leq \otimes_{diag}(\mathbf{N}) \leq \otimes^{\max}_{diag}(\mathbf{M}). \tag{16}$$

For any $\mathbf{M}$*, there are* $\mathbf{N}_1$ *and* $\mathbf{N}_2$ in-closures *such that*

$$\otimes^{\min}_{diag}(\mathbf{M}) = \otimes_{diag}(\mathbf{N}_1) \quad \textit{and} \quad \otimes_{diag}(\mathbf{N}_2) = \otimes^{\max}_{diag}(\mathbf{M}). \quad \square \tag{17}$$

In PHASE THREE, we must decide *how to treat* the elements of A by choosing elements from the sets of **modes** (or *methods*) called *protocols.*

Definition 9. ***(i)*** *Any finite set of quadruplets* $W = \{(t_i, q_i, c_i, \mathbf{r}_i) : i = 1, ..., k\}, \mathbf{r}_i = \{r_{i,1}, .., r_{i,r}\}$ *of* positive *real numbers is called a* ***discrete*** time-quality-cost trade-off ***protocol (DTQCTp)*** *with resource demands if* $t_1 < \ ... < t_k,\ q_1 < \ ... < q_k$ *and* $c_1 \geq \ ... \geq c_k,\ \mathbf{r}_1 \geq \ ... \geq \mathbf{r}_k$ *and* $k \in \mathbb{N}$ *is arbitrary. We may write* $t_{\min}$, $t_{\max}$, $q_{\min}$, $q_{\max}$, $c_{\min}$, $c_{\max}$, $\mathbf{r}_{\min}$ *and* $\mathbf{r}_{\max}$ *instead of* t_1, t_k, q_1, q_k, $\mathbf{r}_k$, $\mathbf{r}_1$, c_k *and* c_1, *respectively.*
If for each $a \in A$, *we are given a protocol* W_a, *then we call the set* $\mathcal{W} = \{W_a : a \in A\}$ *a discrete time-quality-cost trade-off* ***problem (DTQCTP)*** *on* A.
(ii) *Any* positive, continuous, strictly decreasing function $w : [t_{\min}, t_{\max}] \to [q_{\min}, q_{\max}] \times [c_{\min}, c_{\max}] \times [\mathbf{r}_{\min}, \mathbf{r}_{\max}]$ *is called a* ***continuous*** time-quality-cost trade-off ***protocol (CTCQTp)*** *with resource demands, where* $0 < t_{\min}, t_{\max}$,
$q_{\min}, q_{\max}, c_{\min}, c_{\max}, \mathbf{r}_{\min},$ *and* $\mathbf{r}_{\max}$ *are also assumed.*
If for each $a \in A$, *we are given a protocol* w_a, *then we call the set* $\mathcal{W} = \{w_a : a \in A\}$ *a continuous time-quality-cost trade-off* ***problem (CTQCTP).***
(iii) *Any finite set of two dimensional continuous random variables* $\xi = \{\mu_i : i = 1, ..., k\}$ *on any set* Ω *is called a* ***stochastic*** time-quality-cost trade-off ***protocol (STQCTp)*** *if the set of* expected values $\mathcal{E} := \{(t_i, q_i, c_i, \mathbf{r}_i) : i = 1, ..., k\}$, *i.e.,* $(t_i, q_i, c_i, \mathbf{r}_i) = E(\mu_i)$ *for* $i \leq k$, *forms a DTQCTp.*
If for each $a \in A$, *we are given a protocol* ξ_a, *then we call the set* $\mathcal{S} = \{\xi_a : a \in A\}$ *a stochastic time-quality-cost trade-off* ***problem (STQCTP).***

We interpret $(t, q, c, \mathbf{r}) \in W_a$ or $w_a(t) = (q, c, \mathbf{r})$ as paying **cost** c with **quality** q and resource (vector) $\mathbf{r}$ to solve the element $a \in A$ in **time** t using the **mode** assigned to $(t, q, c, \mathbf{r})$. For a parallel explanation of discrete and continuous problems, we write $(t, q, c, \mathbf{r}) \in w_a$ *in both cases.* The elements $t^a_{\min}$, $t^a_{\max}$, $q^a_{\min}$, $q^a_{\max}$, $c^a_{\min}$, $c^a_{\max}$, $\mathbf{r}^a_{\min}$ and $\mathbf{r}^a_{\max}$ may be different in different protocols w_a (W_a) for each $a \in A$ in general. The cases $t_{\min} = t_{\max}$, $q_{\min} = q_{\max}$, $c_{\min} = c_{\max}$ or $\mathbf{r}_{\min} = \mathbf{r}_{\max}$ are also allowed.
The final goal of our algorithm is an optimal project *schedule.*

Definition 10. *Let* $S \in \Xi(A)$ *be any realized project scenario and* $\mathcal{W}$ *be either a CTQCTP or a DTQCTP. A* ***project schedule*** *is a set*

$$\overrightarrow{w} = \{(t^a, q^a, c^a, \mathbf{r}^a) : a \in S\}, \tag{18}$$

where $(t^a, q^a, c^a, \mathbf{r}^a) \in w_a$ *for* $a \in S$.

We are now ready to provide upper and lower *bounds* for time and cost in each stage (in any PHASE) of the algorithm, that is, for any matrix $\mathbf{M} \in \{X, \emptyset, ?\}^{n \times n}$.

Definition 11. ***(i)*** *For any* $\mathbf{M}$ *and* $\mathcal{W}$ *DTQCTP or CTQCTP, the minimal* ***cost-bound*** *is*

$$C_{\min}(\mathbf{M}, \mathcal{W}) := \sum_{[\mathbf{M}]_{i,i} = "X"} c^a_{\min}. \tag{19}$$

(ii) *For any project schedule* $\overrightarrow{w}$*, the* ***total*** project ***cost*** *of* $\overrightarrow{w}$ *is*

$$\mathbf{c}(\overrightarrow{w}) := \sum_{(t^a, q^a, c^a, \mathbf{r}^a) \in w_a \ , \ a \in S} c^a. \tag{20}$$

To quantify project quality, both quality parameters and the task completion scores ($[\mathbf{P}]_{i,i}$) are considered.

Definition 12. ***(i)*** *Denote* $p := \sum_{[\mathbf{M}]_{i,i} = "X"} [\mathbf{P}]_{i,i}$ *as the sum of task (completion) scores. For any* $\mathbf{M}$ *and* $\mathcal{W}$ *DTQCTP or CTQCTP, the maximal* ***(relative) quality bound*** *is*

$$Q_{\max}(\mathbf{M}, \mathcal{W}) := 1. \tag{21}$$

(ii) *For any project schedule* $\overrightarrow{w}$*, the* ***total*** project ***quality*** *of* $\overrightarrow{w}$ *is*

$$\mathbf{q}(\overrightarrow{w}) := \frac{\sum\limits_{(t^a, q^a, c^a, \mathbf{r}^a) \in w_a \ , \ a \in S} q^a}{\sum\limits_{a \in A} q^a_{\max}}. \tag{22}$$

The quality function ($\mathbf{q}(\overrightarrow{w})$) has two components. The first component of the multiplication is the ratio the sum of scores for the selected tasks (sum of task scores for selected/all tasks), whereas the second component is the weighted geometric mean of the quality components for the selected tasks, where the weights are the task scores.

For time bounds, we must not forget the $\prec$ dependencies.

Definition 13. ***(i)*** *For any* real ***path***

$$\overrightarrow{P} = \text{“}a_{i_1} \prec a_{i_2} \prec \ \dots \prec a_{i_k}\text{”} \tag{23}$$

($\mathbf{M}_{i_j, i_{j+1}} =$ “X” *for* $1 \le i_j < k$*), the minimal* ***time bound*** *of the path is*

$$T_{\min}\left(\overrightarrow{P}, \mathcal{W}\right) := \sum_{a \in \overrightarrow{P}} t^a_{\min}. \tag{24}$$

(ii) $\overrightarrow{P}$ *is the* ***longest min-path*** *of* $\mathbf{M}$ *if* $T_{\min}\left(\overrightarrow{P},\mathcal{W}\right)$ *is* maximal*, assuming that* $\overrightarrow{P}$ *contains mandatory tasks only (i.e., assuming* $\mathbf{M}_{i,i}$ = "X" *whenever* $a_i \in \overrightarrow{P}$*). We denote this maximum by*

$$T_{\min}\left(\mathbf{M},\mathcal{W}\right) := \max_{P}\ T_{\min}\left(\overrightarrow{P},\mathcal{W}\right). \tag{25}$$

Thus, $\overrightarrow{P}$ *is called the* ***critical path****, and the set* $\{a_{i_1}, a_{i_2}, ..., a_{i_k}\}$ *is called the set of* ***critical activities****.*
(iii) *For any project schedule* $\overrightarrow{w}$*, the* ***total*** project ***time*** *of* $\overrightarrow{w}$ *is*

$$\mathbf{t}\left(\overrightarrow{w}\right) := \sum_{(t^a,q^a,c^a,\mathbf{r}^a)\in w_a\ ,\ a\in\overrightarrow{P}} t^a, \tag{26}$$

where $\overrightarrow{P}$ *is any longest min-path.*

The length and definition of the longest min-path do not depend on the project schedule $\overrightarrow{w}$ since t^a_{min} are summed in Eq. 26. In fact, critical paths are longest min-paths. Clearly, $t(\overrightarrow{w}) \geq T_{min}(\mathbf{M},\mathcal{W})$ for any $\mathcal{W}$ and $\overrightarrow{w}$.A *longest min-path* in any $\mathbf{M}$ can be found by a standard algorithm within $O(n+d)$, where n is the number of tasks and d is the number of dependencies.

Definition 14. *Denote* $A(\overrightarrow{w},t) \subseteq A$ *as a set of running activities in time* t *for the schedule* $\overrightarrow{w}$*. The maximal resource demand for resource* k *is*

$$r_k(\overrightarrow{w}) := \max_{t} \sum_{a_i\in A(\overrightarrow{w},t)} r_{i,k}\ ,\ k := 1,..,r. \tag{27}$$

Theorem 2. *For any* $\mathbf{M}$ *and for any* in- *or* out-closure $\mathbf{N}$ *of* $\mathbf{M}$*,*

$$C_{\min}\left(\mathbf{M},\mathcal{W}\right) \leq C_{\min}\left(\mathbf{N},\mathcal{W}\right), \tag{28}$$

$$Q_{\max}\left(\mathbf{M},\mathcal{W}\right) \geq Q_{\max}\left(\mathbf{N},\mathcal{W}\right) \tag{29}$$

and

$$T_{\min}\left(\mathbf{M},\mathcal{W}\right) \leq T_{\min}\left(\mathbf{N},\mathcal{W}\right). \tag{30}$$

Furthermore, for any project schedule $\overrightarrow{w}$ *(for the scenario* S*, determined by the diagonal of* $\mathbf{N}$*),*

$$C_{\min}\left(\mathbf{N},\mathcal{W}\right) \leq \mathbf{c}\left(\overrightarrow{w}\right), \tag{31}$$

$$Q_{\max}\left(\mathbf{N},\mathcal{W}\right)\geq\mathbf{q}\left(\overrightarrow{w}\right) \tag{32}$$

and

$$T_{\min}\left(\mathbf{N},\mathcal{W}\right)\leq\mathbf{t}\left(\overrightarrow{w}\right). \tag{33}$$

For $\mathbf{M}\in\{X,\emptyset\}^{n\times n}$ and $\overrightarrow{w}$, we also use the following notation for **total project quality**, **cost**, **time** and **resource** demand:

$$TPC\left(\mathbf{M},\overrightarrow{w}\right):=\mathbf{c}\left(\overrightarrow{w}\right), \tag{34}$$

$$TPQ\left(\mathbf{M},\overrightarrow{w}\right):=\mathbf{q}\left(\overrightarrow{w}\right) \tag{35}$$

and

$$TPT\left(\mathbf{M},\overrightarrow{w}\right):=\mathbf{t}\left(\overrightarrow{w}\right) \tag{36}$$

and

$$\mathbf{TPR}\left(\mathbf{M},\overrightarrow{w}\right):=\left[r_1(\overrightarrow{w})\ ,..,r_r(\overrightarrow{w})\right]^T. \tag{37}$$

The matrix representation of a project *structure* contains no "?" symbols at all.

In PHASE TWO, we must address the *score values* of the *dependencies* on A using the off-diagonal elements of P and Q (see Definition 6).

Definition 15. *For any associative operation $\otimes$ on $\mathbb{R}$, we define the **aggregation function for project structures** as*

$$\otimes_{nd}\left(\mathbf{M}\right):=\bigotimes_{\mathbf{M}_{i,j}="X"\ ,\ i\neq j}\mathbf{P}_{i,j}\ \otimes\bigotimes_{\mathbf{M}_{i,j}="\emptyset"\ ,\ i\neq j}\mathbf{Q}_{i,j} \tag{38}$$

and its extreme values

$$\otimes_{nd}^{\min}\left(\mathbf{M}\right)\ :\ =\otimes_{nd}\left(\mathbf{M}\right)\ \otimes\bigotimes_{\mathbf{M}_{i,j}=?\ ,\ i\neq j}\min\left\{\mathbf{P}_{i,j},\mathbf{Q}_{i,j}\right\}, \tag{39}$$

$$\otimes_{nd}^{\max}\left(\mathbf{M}\right)\ :\ =\otimes_{nd}\left(\mathbf{M}\right)\ \otimes\bigotimes_{\mathbf{M}_{i,j}=?\ ,\ i\neq j}\max\left\{\mathbf{P}_{i,j},\mathbf{Q}_{i,j}\right\}. \tag{40}$$

If $\mathbf{M}$ is the matrix representation of a realized project structure, $\otimes_{nd}\left(\mathbf{M}\right)$ gives the *score value of* this *project structure*[5].

In all versions of the time-quality-cost trade-off functions, the maximal/minimal time, quality and cost demands can be determined for all activities. In this manner, the maximal/minimal total project time (TPT),

[5]The abbreviation nd in the index means "*no diagonal*".

total project quality and maximal/minimal total project cost (TPC) can be determined. This feature will be used when calculating the maximal and minimal demands for a project scenario and the maximal/minimal duration of a project structure.

In the case of (multi-mode) resource-constrained hybrid time-quality-cost trade-off problems, activities can be supplementary, and the dependencies can be flexible. If we exclude a task from the project, we also exclude the time/quality and cost/resource demands of this activity.

Instead, PHASES 1 through 3 are directed by (41)–(60), where the constants C_c, C_t, C_q, $\mathbf{C_r}$, C_{diag} and C_{nd} might be varied upon request. Clearly, some constant values allow for few or fewer solutions.

We are now ready to define the *problems* that we will solve in PHASES ONE, TWO and THREE.

Resource-constrained hybrid time-quality-cost trade-off problems:

Problem 1. PHASE ONE*: Let A be a finite set of activities and $\mathbf{M}$ be a matrix representation of A. Let $C_c, C_t, C_{diag} \in \mathbb{R}^+$ be given such that $C_{\min}(\mathbf{M}, \mathcal{W}) \leq C_c$, $T_{\min}(\mathbf{M}, \mathcal{W}) \leq C_t$ and $C_{diag} \leq \otimes_{diag}^{\max}(\mathbf{M})$. Now, find a* scenario *$S \subseteq A$, i.e., an* in-closure *$\mathbf{M}'$ of $\mathbf{M}$ such that*

$$\otimes_{diag}\left(\mathbf{M}'\right) \to \max \tag{41}$$

assuming

$$\begin{aligned} C_{\min}\left(\mathbf{M}', \mathcal{W}\right) &\leq C_c, &(42)\\ T_{\min}\left(\mathbf{M}', \mathcal{W}\right) &\leq C_t, &(43)\\ \otimes_{diag}\left(\mathbf{M}'\right) &\geq C_{diag}, &(44)\\ Q_{max}\left(\mathbf{M}', \mathcal{W}\right) &\geq C_q. &(45)\end{aligned}$$

The role of (44) is to stop the algorithm from searching for the maximum in (41) when (44), together with (42), (43) and (45), cannot be achieved.

Problem 2. PHASE TWO*: Let $\mathbf{M}'$ be a solution to Problem 1* PHASE ONE*, i.e., a matrix representation of a project scenario $S \subseteq A$. Let $C_t, C_{nd} \in \mathbb{R}^+$ be given such that $T_{\min}(\mathbf{M}', \mathcal{W}) \leq C_t$ and $C_{nd} \leq \otimes_{nd}^{\max}(\mathbf{M}")$. Now, find a* structure*, i.e., an* off-closure *$\mathbf{M}"$ of $\mathbf{M}'$ such that*

$$\otimes_{nd}\left(\mathbf{M}"\right) \to \max \tag{46}$$

assuming

$$\begin{aligned} T_{\min}\left(\mathbf{M}", \mathcal{W}\right) &\leq C_t, &(47)\\ \otimes_{nd}\left(\mathbf{M}"\right) &\geq C_{nd}. &(48)\end{aligned}$$

The role of (48) is to stop the algorithm from searching for the maximum in (46) when (48), together with (47), cannot be achieved.

After PHASE TWO, we are faced with a traditional time-cost trade-off problem; therefore, in PHASE THREE, we can specify different types of objective functions in Problem 3: PHASE THREE/1 and /2.

Problem 3. PHASE THREE*: Let* $\mathbf{M}$ *be a solution to Problem 1* PHASE TWO*, i.e., a matrix representation of a given project structure* $\mathcal{X} = (S, \prec, \sim)$*. Let* $C_c, C_t \in \mathbb{R}^+$ *be given such that* $C_{\min}(\mathbf{M}", \mathcal{W}) \leq C_c$ *and* $T_{\min}(\mathbf{M}", \mathcal{W}) \leq C_t$*.*
Problem 3 PHASE THREE***/1)*** *Find a project schedule* $\overrightarrow{w}$ *such that*

$$\mathbf{t}(\overrightarrow{w}) \to \min \tag{49}$$

assuming

$$\begin{aligned} \mathbf{c}(\overrightarrow{w}) &\leq C_c, & (50)\\ \mathbf{q}(\overrightarrow{w}) &\geq C_q, & (51)\\ \mathbf{r}(\overrightarrow{w}) &\leq \mathbf{C_r}. & (52) \end{aligned}$$

Problem 3 PHASE THREE***/2)*** *Find a project schedule* $\overrightarrow{w}$ *such that*

$$\mathbf{c}(\overrightarrow{w}) \to \min \tag{53}$$

assuming

$$\begin{aligned} \mathbf{t}(\overrightarrow{w}) &\leq C_t, & (54)\\ \mathbf{q}(\overrightarrow{w}) &\geq C_q, & (55)\\ \mathbf{r}(\overrightarrow{w}) &\leq \mathbf{C_r}. & (56) \end{aligned}$$

Problem 3 PHASE THREE***/3)*** *Find a project schedule* $\overrightarrow{w}$ *such that*

$$\mathbf{q}(\overrightarrow{w}) \to \max \tag{57}$$

assuming

$$\begin{aligned} \mathbf{c}(\overrightarrow{w}) &\leq C_c, & (58)\\ \mathbf{q}(\overrightarrow{w}) &\geq C_q, & (59)\\ \mathbf{r}(\overrightarrow{w}) &\leq \mathbf{C_r}. & (60) \end{aligned}$$

Note that the requirements for all the constants in each phase ensure that this phase has at least one solution and that this matrix can be treated in the next phase.

2.2. Modeling flexible project plans

Although the proposed model can be used for both discrete and continuous versions of trade-off problems, in simulations, we addressed continuous version of time-quality-cost trade-off problems to accelerate the computations. For *practical reasons*, we use a specific, deterministic, multimodal project domain matrix (PDM) to model the multi-mode resource-constrained hybrid continuous time-quality-cost trade-off problem. The PDM is an n by $n+6+r$ matrix (see, e.g., Table 1), where n is the number of tasks and r is the number of resources ([17]). The **PDM** has five domains: the logic domain (LD), time domain (TD), quality domain (QD), cost domain (CD) and resource domain (RD) (see Table 1).

In the initial step, $\mathbf{LD} := \mathbf{P}$, where $\mathbf{P} \in [0,1]^{n\times n}$ is a score matrix. The diagonal $[\mathbf{PDM}]_{i,i} = [\mathbf{LD}]_{i,i} = [\mathbf{P}]_{i,i}$ represents the task completion scores, and those terms off of the diagonal $[\mathbf{PDM}]_{i,j} = [\mathbf{LD}]_{i,j} = [\mathbf{P}]_{i,j}$ $(i \neq j)$ represent the task dependency scores between task a_i and task a_j.

TD, QD and **CD** contain of pair of columns of (minimum and maximum) values, whereas the resource domain (RD) contains columns of minimum/maximum values for every resource (see Table 1).

Ref. [17] shows how to address and resolve cycles in a PDM. Therefore, if the dependency scores can be characterized as probability values, then without loss of generality, we can assume that there is no cycle in the project net. In other words, the LD of the PDM can be rearranged as an upper triangular matrix.

Example 7. *Table 1 specifies a* **PDM** *matrix in which there are two modes and two resources. Suppose that* $\mathbf{LD} = \mathbf{P}$*. Additionally, suppose that the score values represents probability values; therefore,* $\otimes = \prod$ *and* $\mathbf{Q} = \mathbf{1} - \mathbf{P} = \mathbf{1} - \mathbf{LD}$*. We assumed that if task j was the successor of task i in every case* $\Rightarrow$ $[\mathbf{M}]_{i,j} :=$ "X"*, then* $[\mathbf{P}]_{i,j} = 1$ *(*$[\mathbf{Q}]_{i,j} = 0$*).*

If in every task j was not the successor of task i $\Rightarrow$ $[\mathbf{M}]_{i,j} :=$ "$\emptyset$"*, then* $[\mathbf{P}]_{i,j} = 0$ *(*$[\mathbf{Q}]_{i,j} = 1$*).*

If at least in one case task j was the successor of task i $\Rightarrow$ $[\mathbf{M}]_{i,j} :=$ "?"*, then* $[\mathbf{P}]_{i,j} \in (0,1)$ *(*$[\mathbf{Q}]_{i,j} = 1 - [\mathbf{P}]_{i,j}$*).*

$$\otimes_{nd}^{\max}(\mathbf{M}) := \otimes_{nd}(\mathbf{M}) \cdot \prod_{[\mathbf{M}]_{i,j}=?\ ,\ i\neq j} \max\left\{[\mathbf{P}]_{i,j}, [\mathbf{Q}]_{i,j}\right\}, \tag{61}$$

$$\otimes_{nd}^{\min}(\mathbf{M}) := \otimes_{nd}(\mathbf{M}) \cdot \prod_{[\mathbf{M}]_{i,j}=?\ ,\ i\neq j} \min\left\{[\mathbf{P}]_{i,j}, [\mathbf{Q}]_{i,j}\right\}. \tag{62}$$

We assumed that $[\mathbf{P}]_{i,j}$ *was the relative frequency of task/dependency occurrences. Therefore, the aim of the simulation was to determine the most probable project structure that can be completed given a specified deadline/budget/resource availability.*

$$\otimes_{diag}^{\max}(\mathbf{M}) := \otimes_{diag}(\mathbf{M}) \quad \cdot \prod_{\mathbf{M}_{i,i}=?} \max\left\{[\mathbf{P}]_{i,i}, [\mathbf{Q}]_{i,i}\right\}, \tag{63}$$

$$\otimes_{diag}^{\min}(\mathbf{M}) := \otimes_{diag}(\mathbf{M}) \quad \cdot \prod_{\mathbf{M}_{i,i}=?} \min\left\{[\mathbf{P}]_{i,i}, [\mathbf{Q}]_{i,i}\right\}. \tag{64}$$

Let the durations be in weeks, cost demands be in \$1,000 *and resource demands be in number of employees. The most-probable project scenario*

Table 1: Multi-modal specified PDM

PDM	Logic Domain					TD		CD		QD		RD			
	A	**B**	**C**	**D**	**E**	$t_{\min}$	$t_{\max}$	$c_{\min}$	$c_{\max}$	$q_{\min}$	$q_{\max}$	$r_{1\min}$	$r_{1\max}$	$r_{2\min}$	$r_{2\max}$
A	0.8	1.0	0.8	0.2	0.1	4	6	2.4	3.4	0.8	0.9	2.5	4.5	1.6	3.7
B	0.0	1.0	0.0	0.4	0.8	2	3	1.8	2.6	0.7	0.8	3.4	4.2	2.5	4.8
C	0.0	0.0	0.9	0.0	0.2	4	8	9.5	9.9	0.8	0.9	3.8	5.7	1.2	3.5
D	0.0	0.0	0.0	0.4	0.3	9	9	4.2	4.2	0.8	0.8	2.3	2.3	1.4	1.4
E	0.0	0.0	0.0	0.0	0.7	3	4	0.9	1.2	0.7	0.8	3.4	4.7	2.5	6.2
Constraints:						10.0		18.0		0.7		10.0		10.0	

contains tasks A,B,C and E, and the probability value of this project scenario is $\otimes(\mathbf{LD})_{diag}^{\max} = 0.8 \cdot 1.0 \cdot 0.9 \cdot (1-0.4) \cdot 0.7 = 0.3024$, *whereas* $\otimes(\mathbf{LD})_{diag}^{\min} = (1-0.8) \cdot 1.0 \cdot (1-0.9) \cdot 0.4 \cdot (1-0.7) = 0.0024$. *The minimal project duration is obtained if all flexible tasks and all flexible dependencies are excluded from this project and all tasks are completed within the shortest possible duration. Since only task* 2 *(task B) is a mandatory task,* $\mathrm{TPT}_{\min} = \min_j [\mathbf{TD}]_{2,j} = \min\{2,3\} = 2$ *weeks. This is a global minimum; therefore, there is no way to complete the project in less than 2 weeks. The minimal project cost also occurs if only the mandatory task is completed; however, the cost demands are assumed to be independent of the completion sequences of the tasks.* $\mathrm{TPC}_{\min} = \min_j [\mathbf{CD}]_{2,j} = \min\{1.8, 2.6\} = 1.8 \times \$1,000 = \$1,800$. *According to Eqs. (21)-(21),* $\mathrm{TPQ}_{\max} = 1$, *and the maximum of the resource demands is minimal when all supplementary tasks are ignored but all flexible dependencies between mandatory tasks are prescribed and the resource demands in the completion modes of mandatory tasks are minimal.* $\mathbf{TPR}_{\min} = \{3.4; 2.5\}$. *These bounds specify global minima/maxima; however, if a supplementary task or a flexible dependency is decided to be included in or excluded from the project, and in this manner, we obtain a* *restricted matrix, the minimal project cost/durations/resource demands and maximal project quality can be calculated for the restricted matrix.*

Remark 3. *Since Eqs. (63)–(64) give small values, if there are many tasks, the geometric mean of task scores is calculated and explored. The main advantage of using the geometric mean of task scores instead of multiplying the relative frequencies is that different project scores can be compared.*

If $\mathbf{M}$ *represents the initial hybrid logic plan and* $\mathbf{M}'$ *is the result (final project scenario) of* PHASE ONE*, then*

$$TPS\% = \otimes_{diag}(\mathbf{M})\% := \left(\frac{\sqrt[n]{\otimes_{diag}(\mathbf{M}')}}{\sqrt[n]{\otimes_{diag}^{\min}(\mathbf{M})}} - 1 \right) \times 100 \tag{65}$$

denotes ***the total project score (TPS) performance of project scenario M′****.*

TPS%=$\otimes_{diag}$% $\in [0, \infty]$. This value shows us how many times larger the project score is compared to the minimal requirement.

The performance schedules for the remaining parameters are specified as follows (similarly to Eq. 65):

$$TPQ\% := \left(\frac{TPQ(M,(W))}{TPQ_{\min}(M,(W))} - 1 \right) \times 100, \tag{66}$$

$$TPT\% := \left(\frac{TPT_{\max}(M,(W))}{TPT(M,(W))} - 1 \right) \times 100, \tag{67}$$

$$TPC\% := \left(\frac{TPC_{\max}(M,(W))}{TPC(M,(W))} - 1 \right) \times 100, \tag{68}$$

$$TPR_i\% := \left(\frac{TPR_{i_{\max}}(M,(W))}{TPR_i(M,(W))} - 1 \right) \times 100, \quad i = 1, 2.., r. \tag{69}$$

Remark 4. *Eqs. (65)–(69) are between* $[0, \infty]$ *if the project schedules are feasible. Otherwise, these values are 0.*

2.3. The Algorithms

2.3.1. PHASE ONE

We are given the matrix $\mathbf{M}_0 \in \{X, \emptyset, ?\}^{n \times n}$. Suppose that all the "?" symbols in the diagonal are in the first σ rows (columns). The algorithm sequentially changes these symbols to either "X" or "$\varnothing$" in *this* order; such a change is called a **step**. These changes are not final, and the original matrix $\mathbf{M}_0$ is also saved. We look for the optimum similarly to a "back-and-forth" method, saving information in the **buffer** B (a set) of possible manners we might investigate later. After replacing as many elements of $\mathbf{M}$ as we can (satisfying Eqs. (42), (43), and (45)), we go back to the cases in which B has higher scores $\otimes_{diag}$ than $\mathbf{M}$. (All the possible variations of $\mathbf{M}_0$ form a binary tree of size 2^σ with root $\mathbf{M}_0$.)

Here, $\mathbf{M}$ denotes the actual matrix *before* the next replacement, such that $[\mathbf{M}]_{j,j} = \text{“?”} \iff i \leq j \leq \sigma$ for some $1 \leq i \leq \sigma$, and denote by $\mathbf{M}[i, i = Y]$ the matrix *after* replacing $\mathbf{M}_{i,i}$ to Y, where $Y \in \{X, \emptyset\}$. *Before* replacing $[\mathbf{M}]_{i,i}$, we save the *other* possibility, which we do not follow in the present step, in B. The elements of B are of the form

$$\begin{aligned} \mathbf{b} &= \Big(i,\ [\mathbf{M}]_{1..n,1..n}\ ,\ Y,\ \otimes_{diag}^{\max}(\mathbf{M}[i, i = Y])\Big) && (70)\\ &= (i, \overrightarrow{m}, Y, \otimes_{\mathbf{b}}), && (71) \end{aligned}$$

where i denotes the elements in the diagonal of $\mathbf{M}$ that we are replacing, $\overrightarrow{m} = \mathbf{M}_{1..n,1..n}$ is the actual content of the diagonal of $\mathbf{M}$ (specifically $\mathbf{M}_{i,i} =$ "?"), $Y \in \{X, \emptyset\}$ and $\otimes_b = \otimes_{diag}^{\max}(\mathbf{M}[i, i = Y])$ is the"ideal" score that we may achieve by replacing $\mathbf{M}_{i,i}$ with Y.[6] B may contain several elements with the same $\otimes_{\mathbf{b}}$ value, but at this moment, we do not know which pf these elements can be realized later, i.e., satisfy (42)–(45)

Remark 5. *B contains only $\mathbf{M}[i, i = Y]$ extensions that have not yet been investigated but fulfill the bounds of (42)–(45). More precisely, for their* extension,

$$\begin{aligned} C_{\min}(\mathbf{M}[i, i = Y], \mathcal{W}) &\leq C_c, && (72)\\ T_{\min}(\mathbf{M}[i, i = Y], \mathcal{W}) &\leq C_t, && (73)\\ Q_{\max}(\mathbf{M}[i, i = Y], \mathcal{W}) &\geq C_q, && (74)\\ \otimes_{diag}^{\max}(\mathbf{M}[i, i = Y]) &\geq C_{diag}. && (75) \end{aligned}$$

Remark 6. *Before starting step i for $1 < i$, at the end of step $i - 1$, we have inserted Y into the $i - 1$-th entry of $\mathbf{M}$. Since we are now extending this configuration, B does* not *contain the corresponding record $\mathbf{b} = (i - 1, \overrightarrow{m}, Y, \otimes_{\mathbf{b}})$.*

[6] $\otimes_{\mathbf{b}}$ in (71) denotes a nonnegative real number, and $\otimes_{diag}^{\max}$ was defined in (10).

The algorithm starts a **new cycle** whenever it goes back to an element of B and starts to replace "?" from the $i = i_0 + 1$-th entry of the diagonal. Problem 1 may have a solution if, for at least one cycle, we are able to step i to σ (satisfying (42)–(45)). Of course, we store all the in-closures $\mathbf{M}'$ of $\mathbf{M}_0$ that were found by the algorithm and may be optimal solutions to Problem 1. If B contains an element $\mathbf{b}$ with a higher score than $\mathbf{M}'$ has (i.e., $\otimes_{\mathbf{b}} > \otimes_{diag}(\mathbf{M}')$), then we start a new cycle from $\mathbf{b}$. During this cycle, i cannot be increased to σ, $\otimes_b = \otimes_{diag}^{\max}(\mathbf{M}[i, i = Y])$ may fall below $\otimes_{diag}(\mathbf{M})$, or we might obtain a solution better than $\mathbf{M}'$.

START Let $i_0 := 0$, $i = 1$, $B := \emptyset$.

GENERAL STEP $(1 \leq i \leq \sigma)$, $\mathbf{M}$ is the actual matrix. Let

$$\mathbf{b}_X^i \; : \; = \left(i,\ \mathbf{M}_{1..n,1..n},\ X,\ \otimes_{diag}^{\max}(\mathbf{M}[i, i = X])\right), \tag{76}$$

$$\mathbf{b}_\emptyset^i \; : \; = \left(i,\ \mathbf{M}_{1..n,1..n},\ \emptyset,\ \otimes_{diag}^{\max}(\mathbf{M}[i, i = \emptyset])\right). \tag{77}$$

Case i) Neither $\mathbf{b}_\emptyset^i$ nor $\mathbf{b}_X^i$ fulfills (72)–(75) and $B = \emptyset$. Then, STOP since Problem 1 has no solution.

Case ii) Neither $\mathbf{b}_\emptyset^i$ nor $\mathbf{b}_X^i$ fulfills (72)–(75) but $B \neq \emptyset$. Recall that in Cases i) and ii), B may contain elements of type $\mathbf{b} = (j, \overrightarrow{m}, Y, \otimes_{\mathbf{b}})$ only if $j < i$ by Note 5 and (70), (71), and (76), (77). In Case ii), choose any element $\mathbf{b} \in B$ such that $\otimes_{\mathbf{b}}$ is maximal (in B). Then, reset the diagonal of $\mathbf{M}$ according to $\overrightarrow{m}$, set $i := j$, delete $\mathbf{b}$ from B, and proceed to the General Step.

Case iii) Exactly *one* of $\mathbf{b}_X^i$ or $\mathbf{b}_\emptyset^i$ fulfills (72)–(75), say, $\mathbf{b}_Y^i$. Let $\mathbf{M}_{i,i} :=$ "Y" and go to Step Increasing i.

Case iv) Both $\mathbf{b}_X^i$ and $\mathbf{b}_\emptyset^i$ fulfill (72)–(75).

If $\otimes_{diag}^{\max}(\mathbf{M}[i, i = X]) \leq \otimes_{diag}^{\max}(\mathbf{M}[i, i = \emptyset])$, then let $\mathbf{M}_{i,i} :=$ "$\emptyset$", $B := B \cup \left\{\mathbf{b}_X^i\right\}$, and go to Step Increasing i.

If $\otimes_{diag}^{\max}(\mathbf{M}[i, i = X]) > \otimes_{diag}^{\max}(\mathbf{M}[i, i = \emptyset])$, then let $\mathbf{M}_{i,i} :=$ "X", $B := B \cup \left\{\mathbf{b}_\emptyset^i\right\}$, and go to Step Increasing i.

STEP INCREASING (i) If $i < \sigma$, then let $i := i + 1$, and go to the General Step. In the case $i = \sigma$, go to the Check Step.

CHECK STEP $(i = \sigma)$ First, save the recent $\mathbf{M}$ with its $\otimes_{diag}(\mathbf{M})$. If B contains an element $\mathbf{b} = (i, \overrightarrow{m}, Y, \otimes_{\mathbf{b}})$ such that

$$\otimes_{\mathbf{b}} > \otimes_{diag}(\mathbf{M}), \tag{78}$$

then go back to $\mathbf{b}$ and start a new cycle, i.e., reset the diagonal of $\mathbf{M}$ according to $\overrightarrow{m}$, set $i := j$, delete $\mathbf{b}$ from B, and go to the General Step.

END of the Algorithm.

Remark 7. *If we want to find* all *optimal solutions to Problem 1, then replace (78) with*

$$\otimes_{\mathbf{b}} \geq \otimes_{diag}(\mathbf{M}) . \tag{79}$$

Of course, we must first save (in an output buffer*) the solution(s) we have found thus far. Then, we pick the* next *element* $\mathbf{b} \in B$ *in the buffer, reset the diagonal of* $\mathbf{M}$ *according to* $\overrightarrow{m}$*, set* $i := j$*, delete* $\mathbf{b}$ *from* B*, and go to the General Step.*
We call this algorithm the ***Hybrid Project Ranking*** *algorithm.*

Theorem 3. *The saved matrices (in the Check Step) of the above algorithm are exactly the optimal solutions to Problem 1. Specifically, there are no saved matrices if and only if (42)–(44) in Problem 1 has no solution at all.*

Proof. In each step, the algorithm chooses the best of (at most) two possibilities but buffers the other for further investigation. The value $\otimes_{diag}(\mathbf{M})$ for each $\mathbf{M}$ is a sharp *upper bound* for further continuation of $\mathbf{M}$. Therefore, all the buffered possibilities with smaller $\otimes$ than those of the finished (and saved) matrices (in the Check Step) could be deleted from the buffer. Since the algorithm checks *each* remaining element of the buffer (see the Check Step), at the end, we must obtain each optimal solution. ■

The result of PHASE ONE is to find (depending on the meaning of the completion scores and the aggregation functions (see Definition 4)) the most-desirable or most-probable project scenario. The goal of PHASE TWO is to find the most-desirable/most-probable project structure within a project scenario. However, the project structure always depends on the result of PHASE ONE. Therefore, the "highest"-scoring project structure can be interpreted only within a specified project scenario.

2.3.2. Phase two

In PHASE TWO, the goal is to find the most-probable or most-desirable project structure *within* the specified project scenario. The algorithm and most of the notation for PHASE TWO are the same as in PHASE ONE. We have to determine the "?" symbols off of the diagonal of $\mathbf{M}$ in a fixed (but arbitrary) order. *Before* each replacement, we save the other possibility in a buffer similarly to (70), checking the conditions corresponding to (47) and (48), such as (72)–(75) corresponded to (42)–(44). In each step, we have to refresh $T_{\min}(\mathbf{M}, \mathcal{W})$ and $\otimes_{nd}(\mathbf{M})$. The properties of the PHASE TWO algorithm can be proven along the lines of Theorem 3 and Subsection 2.3.4.

2.3.3. Phase three

After PHASE TWO we obtain a project structure, which represents a traditional resource-constrained time-quality-cost trade-off problem (RC-TQCTP). If there is no feasible solution to the given RC-TQCTP algorithm, we should go back to PHASE TWO and select the next project structure from the buffer.

The result of the RC-TQCTP will produce the optimal solution to the resource-constrained hybrid time-quality-cost trade-off problem.

The optimal output matrix is a domain mapping matrix (DMM) (see Table 2), in which flexible dependency and uncertain task completion are excluded or included. Therefore, the logic domain of the output matrix is a DSM.

Table 2: The output matrix of the proposed algorithm if the input is specified by Table 1

DMM	LD				TD	CD	QD	RD		SST
	A	**B**	**C**	**E**	t	c	q	r_1	r_2	
A	1	1	1		4	3.4	0.8000	4.5000	3.7000	0
B		1		1	2	2.6	0.7000	4.2000	4.8000	6
C			1		5	9.8	0.8250	5.2250	2.9250	6
E				1	3	1.2	0.7000	4.7000	6.2000	9
TPT,TPC,TPQ,**TPR**					**9**	**18.0**	**0.7202**	**9.9250**	**9.1250**	–

The optimal output matrix (furthermore, the project schedule matrix) contains one vector of time/cost demands (TD,CD) and one vector of quality parameters (QD). The PSM also contains an n by r submatrix of resource demands (RD) and a vector of scheduled start times (SST).

2.3.4. Algorithmic complexity

Briefly, in PHASE ONE, $\otimes_{diag}(\mathbf{M})$ is calculated and $T_{\min}(\mathbf{M},\mathcal{W})$ is refreshed (see Definition 13) such that each cycle is at most *quasilinear*, but we have no bounds on the *total* size of the buffer. Similarly, in PHASE TWO, we have to calculate $T_{\min}(\mathbf{M},\mathcal{W})$ and $\otimes_{nd}(\mathbf{M})$, which implies the same upper bound on time as in PHASE ONE. Perhaps some extreme counterexamples may cause exponential running time, but practical runs (see Section 3) provide quadratic runs.

In more detail, in general, the n^{th}-largest value can be determined within $O(n\log n)$ computation time (e.g., [22]); however, our decision tree is a special binary heap in which a quasilinear search algorithm can be specified. In Section 2.3, we saw that the best project structures can be found within

$O(t+d)$, where t is the number of supplementary task to be completed and d is the number of flexible task dependencies. If there are no supplementary tasks to be completed, the number of possible project structures depends only on the number of flexible dependencies. If there are d flexible dependencies, then there are 2^d possible project plans. The project network can be specified as an NDSM. In case of acyclic project networks, the maximal number of flexible dependency is $n(n-1)/2$, and in this case, the number of possible project structures is $2^{n(n-1)/2}$.

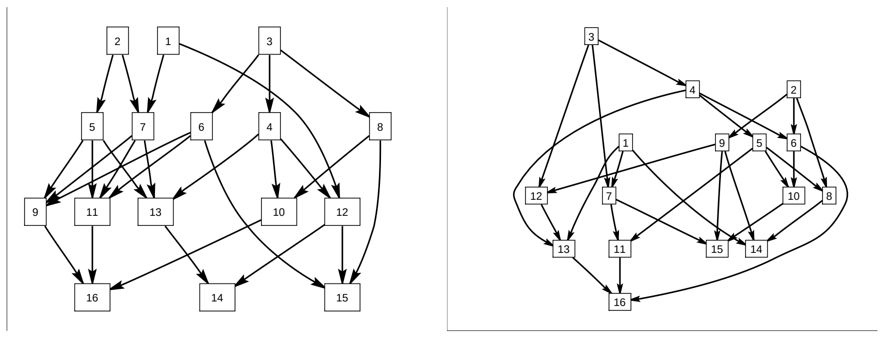
If there are t supplementary tasks, then 2^t project scenarios can be specified. In a special case, if $t = n$ and $d = n(n-1)/2$, then there are 2^n project scenarios and $\sum_{j=0}^{n} \binom{n}{j} 2^{j(j-1)/2}$ project plans.

The computational demand of the proposed hybrid algorithm with respect to the fulfilled PEM=LD of a specified deterministic PEM is $O(d) = O(n(n-1)/2) \approx O(n^2)$ in the case of a fulfilled upper-triangular NDSM, and the runtime $O(d+t) = O(n+n(n-1)/2) \approx O(n^2)$ is similar when considering a fulfilled upper-triangular PEM. For example, when $n = 50$, completely filled upper-triangular NDSM and PEM specify $5,78 \cdot 10^{368}$ possible project plans, whereas our algorithm finds a project structure within $O(50^2)$ steps. However, at the end of the project selection process, where we have a feasible project structure, the proposed algorithm has to utilize a RC-TQCTP method. In this case, the complexity of the RC-TQCTP method and the complexity of project selection are multiplied. As (multi-mode) resource-constrained versions of TQCTPs, DTQCTPs and STQCTPs, a stochastic version of DTQCTPs, are NP-hard problems. The hybrid versions of these problems, namely, hybrid discrete time-quality-cost trade-off problems (HDTQCTP), hybrid stochastic time-quality-cost trade-off problems (HSTQCTP) and (multi-mode) resource-constrained hybrid time-quality-cost trade-off problems, are also NP-hard problems.

3. Simulation results

The main goal of the simulation was to compare project scheduling performances (see Eqs. (65)–(69)) of the different types of project management approaches.

The first problem was to select an adequate project plan from project database because neither known project generators (such as ProGen [23], RanGen I[24], and II[25]) nor open project data sources (such as MMLIB[26], and PSPLIB[23]) distinguish mandatory and supplementary tasks or consider strict and flexible dependencies. Therefore, there are no score values



(a) n11_2 ($i_2 = (m-1)/(n-1) = 0.2$) (b) n16_1 ($i_2 = (m-1)/(n-1) = 0.267$)

Figure 1: Selected logic network from PSPLIB MRCPSP - n1 dataset

linked to task completion or task dependencies.

The second problem is that the quality parameters are neglected and usually the cost parameters are also missing from the project plans. Additionally, resources are either missing or specified in a discrete multi-mode resource allocation problem, in which there are more than two modes. Nevertheless, these project databases have been validated and applied in several publications for testing and comparing algorithms; therefore, we decided to use the logic network, and we extended the project plans with cost, quality, resource and score parameters in the case of the simulation. Fig. 1 shows the selected logic network from PSPLIB - Multi-Mode Resource Constrained Project Scheduling Problem (MRCPSP) n1 dataset: (`http://www.om-db.wi.tum.de/psplib/files/n1.mm.zip`).

Ref. [27] is specified a real-life project database; however, this database contains only five IT projects, and these projects did not follow the agile approach. In addition, this database did not contain completion modes. Therefore, project plans were selected from simulation database PSPLIB - MRCPSP - n1 dataset, where the main features of IT projects can be found.

3.1. Simulation database

The aim of the selection and the specification of initial project plans are to meet as much as possible the expectations for (IT) software project plans, especially the features of agile projects:

1. Since Ref. [28] and Ref. [27] showed that software projects usually contain more parallel tasks; therefore, according to the Ref. [28] and

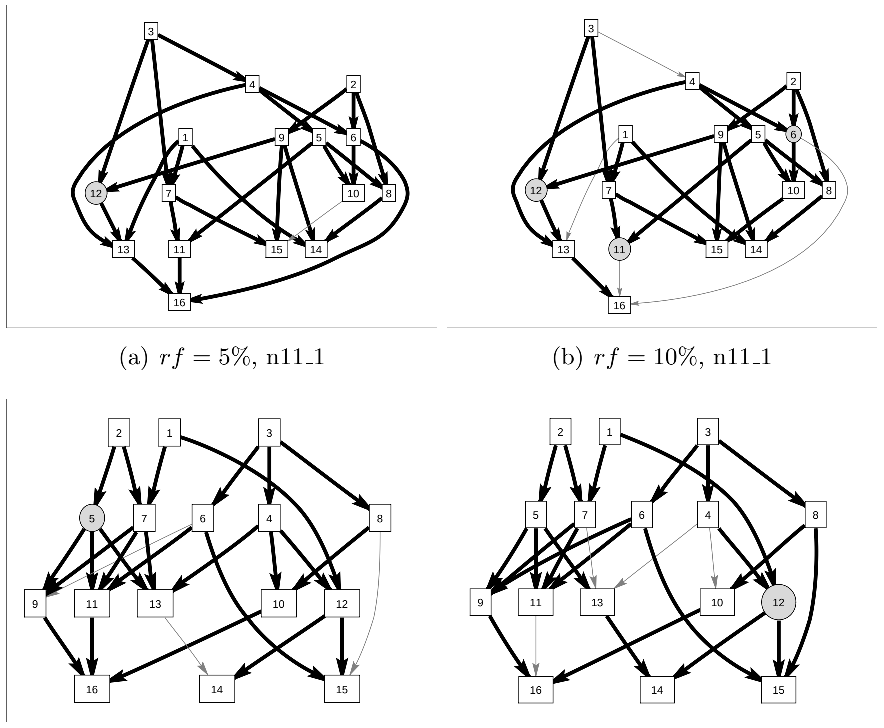
[27], the number of parallel tasks is greater than the number of serial tasks[7].

2. Projects are usually separated into smaller autonomous sub-projects (sprints) [see, e.g., Ref. 29] that should completed within 2-6 weeks; therefore, the number of tasks is limited and should not be greater than 20.
3. Contains at least two types of renewable resources (e.g., programmer or tester)
4. Contains two completion modes to apply continuous trade-off methods and in this manner also tests the performance of the hybrid approaches.

The selected n11_2 (see Fig. 1(a)) and n16_1 (see Fig. 1(b)) project plans satisfies criterion 1 (more parallel than serial completion) and criterion 2 ($n < 20$). After selecting logic network, 50-50 project plans are generated for both logic plans n11_2 and n16_1. The minimal value of work hours are set to $t_{\min} \in [20, 40]$ (in work hours) to keep the project duration to within 2-6 weeks. The minimal values of cost demands are $c_{\min} \in [1000, 3000]$ (in \$), which covers the direct costs. The minimal amounts of two types of resources $r_{\min} \in [3, 5]$ (in terms of number of employees, either testers or programmers) are also specified. At the end, the minimal value of quality parameters $q_{\min} \in [0.70, 0.9]$ are generated between the specified intervals. To simulate trade-offs, maximal values were between $110 - 120\%$ of the minimal values.

Known project databases (such as ProGen and RanGen I,II) do not distinguish mandatory and supplementary tasks and also do not distinguish fixed and flexible dependencies between two tasks. Consequently, after generating time/cost/resource demands, in our database, the supplementary and mandatory task completions or flexible and strict dependencies are not distinguished. Therefore, in order to simulate flexible environments, two types of datasets are specified. In both cases, first, the ***ratios of flexibility*** is specified as follows: $rf_1 \in \{5\%, 10\%\}$, $rf_2 \in \{25\%, 30\%\}$. This means that in the case of first dataset, 5% and 10%, whereas in the case of dataset 2, 25% and 30% of tasks and dependencies are selected to became supplementary tasks or flexible dependencies. In the case of both datasets, it is assumed that the score of including is greater than the score of excluding; therefore, the score values of supplementary tasks and flexible dependencies

[7]Following the simulations of Ref. [28], $i_2 = (m-1)/(n-1) \in [0.2, 0.3]$, where m is the number stages in a topological ordered network and n is the number of tasks. $i_2 = 1$ if all tasks are completed in a serial manner, and $i_2 = 0$ if all tasks are completed in parallel.



(a) $rf = 5\%$, n11_1

(b) $rf = 10\%$, n11_1

(c) $rf = 5\%$, n16_2

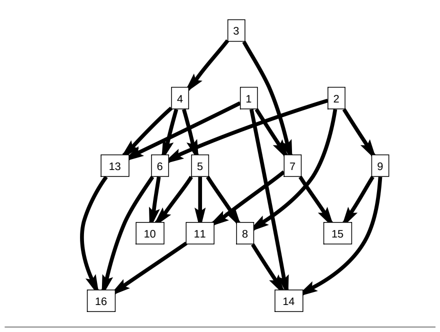
(d) $rf = 10\%$, n16_2

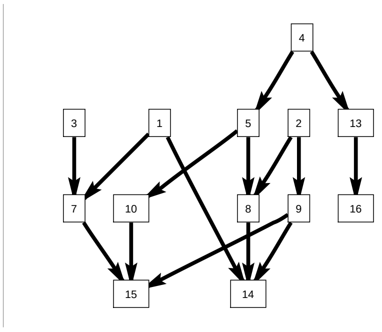
Figure 2: Stochastic logical networks (thin edges=flexible dependencies, circles=supplementary tasks).

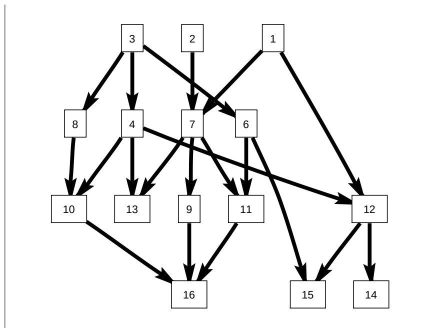
were greater than 0.5 but less than 1.0. Dataset 1 specifies a less-flexible environment, whereas dataset 2 specifies a more-flexible environment. Fig. 2 shows the logical networks of the hybrid project plan when $rf = 5\%$ and $rf = 10\%$.

In the case of dataset 1, the structure of the project is only slightly changed, even if all flexible dependency and all supplementary tasks are excluded from the project, whereas as Fig. 4 shows, when considering the more flexible environment, parallel autonomous subprojects (i.e., sprints) can be generated. It is an extreme case when all supplementary tasks are postponed and all flexible dependency are excluded from the project network (see Fig. 3).

Nevertheless, if we consider the minimal time or cost demands, Fig. 3 provides project plans when the total project time/total project cost is


(a) $rf = 5\%$, n11_1


(b) $rf = 10\%$, n11_1


(c) $rf = 5\%$, n16_2

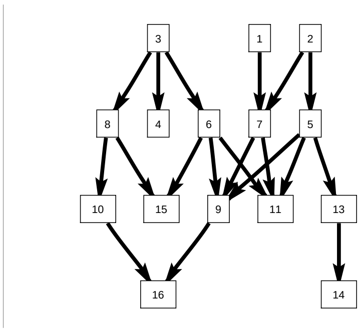
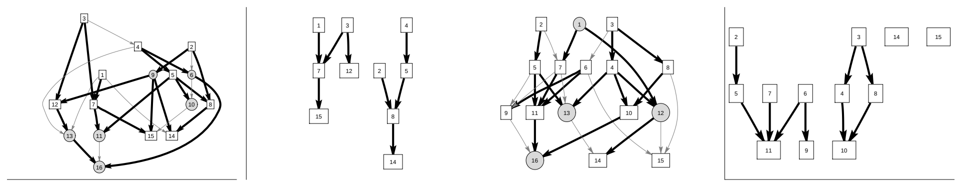

(d) $rf = 10\%$, n16_2

Figure 3: Logical networks if only fixed dependencies and mandatory tasks are considered (dataset 1).



(a) $i_2 = 0.2$ (b) $rf = 25\%$ (c) $i_2 = 0.27$ (d) $rf = 30\%$

Figure 4: Logic networks from dataset 2. Stochastic logic networks (a and c) from dataset 2 and the deterministic results (respectively b an d) if supplementary tasks and dependencies are omitted.

minimal.

3.2. The calculation of constraints and the evaluation scheduling performances

When calculating constraints, first, the minimal/maximal values of total project time/cost/quality/scores and resources are determined (see Definition 11-14 and Example 7). TPX $\in$ {TPT,TPC,TPQ,TPS,TPR}

The ratio of constraints is applied $C_x\% = \frac{C_x - \text{TPX}_{\min}}{\text{TPX}_{\max} - \text{TPX}_{\min}}$. In this simulation, all constraint ratios were 0.7 or 0.9, $(C_s\%, C_t\%, C_c\%, C_q\%, C_{r_1}\% = C_{r_2}\% \in \{0.7, 0.9\})$. Owing to the possible combination of constraints, $2^5 = 32$ types of constraints are specified for all project plans.

When comparing project management approaches, the ratio of feasible projects and, according to Eqs. (65)–(69), the scheduling performances are also calculated for all projects solved by three types of project management approaches.

3.3. Simulated project management approach

The proposed method contains three phases, but different types of project management approaches do not use all phases.

TPMa:. Traditional project management approaches (TPMa) do not allow for flexible dependencies and supplementary task completion. Therefore, the most-probable tasks and dependencies should be realized. In this case, the matrix representation ($[\mathbf{M}]_{i,j}^{\text{TPMa}}$) of a traditional project plan (from matrix representation ($\mathbf{M}$) of the original hybrid project plan) can be specified for every i, j as follows:

- $[\mathbf{P}]_{i,j} \geq 0.5 \Rightarrow [\mathbf{M}]_{i,j}^{\text{TPMa}} = $ "X",

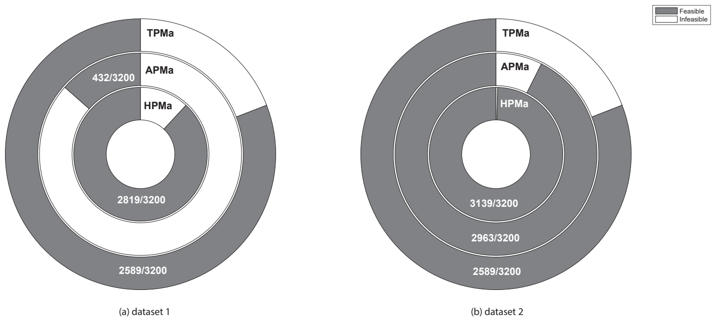


Figure 5: Feasibility results of project management approaches

- $[\mathbf{P}]_{i,j} < 0.5 \Rightarrow [\mathbf{M}]_{i,j}^{\text{TPMa}} = \text{“}\emptyset\text{”}$.

Since $[\mathbf{M}]_{i,j}^{\text{TPMa}}$ is a project structure, we go directly to PHASE THREE.

APMa:. Following the agile project management approach (APMa) allows us to specify flexible dependencies and supplementary task completion. However, in this case, we do not consider time-cost trade-offs. Only one mode, a *normal planning mode* $(t_{\max}, c_{\min}, q_{\max}, \mathbf{r}_{\min})$, is specified for every task. Thus, we cannot use any trade-off method to reduce the task duration.

HPMa:. The proposed algorithm assumes that project managers can consider the trade-offs and they can also reorganize or restructure the project. Therefore, the full PDMs are considered.

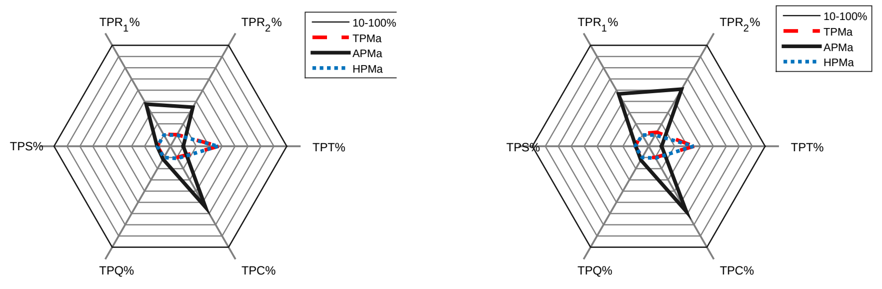
3.4. Number of simulations

2×100 types of PDM matrix, $2^5 = 32$ possible constraints, and 3 possible (i.e., traditional (TPMa), agile (APMa), or hybrid (HPMa)) project management approaches produced 19200 simulations.

To accelerate the computation of the time-cost and time-quality trade-off functions, they were considered as linear functions because the time-quality-cost trade-off problem can be treated as a cost flow problem, which can be solved in strongly polynomial manner (see, e.g., [30]).

Fig. 5 shows that in both datasets, HPMa produces the most-feasible solutions. However, if the flexibility ratio was low, the agile approach produces the least-feasible solutions. The agile approach can be more efficient if managers have more freedom in their decisions. They can parallelize tasks, or they can specify parallel sub-projects.

Fig. 6 shows performances schedules for different types of project management approaches. A larger covered area indicates better performance in scheduling. The largest covered area is produced by the agile approach; nevertheless, in a less-flexible project environment (dataset 1), APMa usually cannot find a feasible solution. The scheduling performances of TPMa and HPMa are very similar in both datasets. Both methods can better reduce project durations; however, HPMa produces more-feasible solutions, whereas TPMa keeps all tasks and therefore yields the highest quality and score values.



(a) less-flexible project environment (b) more-flexible project environment

Figure 6: Performance of schedules of different project management approaches for feasible projects.

A schedule performance indicator can be treated as a dependent variable in a regression model, in which independent variables are then applied in a project management approach (implemented by the proposed algorithm) and the constraints, the ratio of flexible dependencies ($F\%$) and the ratio of supplementary tasks ($S\%$) are varied. Table 3 presents the significant operators. The significance level was $\alpha = 0.001$. Table 3 reports the importance values for the independent variables, which are ratios of R^2 values from the regression models. Table 3 indicates that in 4 cases (i.e., for TPT%, TPC%, and TPR$_1$%, TPR$_2$%), the applied project management approach was the most important variable. The next-most-important variable was the adequate constraint, whereas in the case of considering the project scores and the quality performances, the most-important value was the ratio of supplementary tasks because ignoring or postponing supplementary tasks may reduce both the quality and the scores of the projects.

The results show that there is no perfect project management approach. Whereas the agile approach can reduce the cost and resources most significantly (see Fig. 6), HPMa can find the most-feasible solutions (see Fig. 5). The traditional approach can best maintain the score and quality.

Indeps. \ Deps.	**TPT%**	**TPC%**	**TPS%**	**TPQ%**	**TPR$_1$%**	**TPR$_2$%**
XPMa	78.0198	74.7481	0.5642	10.6008	39.8360	31.6482
C_t%	8.3171	1.1846	5.8063	5.8990	5.0401	5.1696
C_c%	1.4316	18.2283	5.4056	4.3890	3.6417	4.1025
C_q%	1.1519	1.2292	5.3624	33.0476	3.6555	4.1239
C_r%	0.9802	1.4805	5.0361	4.0924	22.4144	18.2393
C_s%	1.2538	1.3601	25.3817	4.7369	3.5497	4.0147
F%	2.3892	1.2322	4.9610	3.8254	12.2962	18.1735
S%	6.4564	0.5371	47.4826	33.4090	9.5663	14.5283
Adj. R^2	41.4300	50.8200	59.8000	43.1500	38.6500	42.5700

Table 3: Importance of the management approach and the constraints for schedule performance (the most-important values are highlighted).

4. Summary and conclusion

The proposed hybrid time-quality-cost trade-off approach may bridge the agile and traditional project management approaches. If there are no flexible dependencies or supplementary tasks, the problem will be a traditional TCQTP problem. However, if there is no trade-off between time and cost and/or time and quality demands, the task is to find a feasible project scenario and a feasible project plan given a set of time/quality/cost/resource constraints.

The hybrid project management (HPM) approach combines methods (e.g., time-cost and time-quality-cost trade-off methods) from traditional project management with structuring and scoring techniques from agile project management.

In this paper, resource-constrained hybrid time-quality-cost trade-off problems (RC-HTQCTPs) were examined to extend the traditional time-quality-cost trade-off model by drawing on the agile approach.

In this paper, a new matrix-based project planning method was proposed to model RC-HTQCTPs and find the most-desirable/least-time-consuming/lowest-cost project scenarios and project structures within the specified constraints.

The proposed algorithm is a fast, efficient method that supports the hybrid project management (HPM) approach. The algorithm is able find an optimal solution according to predefined preferences regarding factors such as time and cost.

The developed matrix-based method and proposed exact algorithm may be important and essential components of a project expert system supporting strategic decision-making, particularly in cases of large, complex, flexible projects.

4.1. Limitations and future works

The proposed model extends traditional resource-constrained time-quality-cost trade-off methods; however, in this model, only renewable resources (e.g., human resources) and one non-renewable resource (i.e., cost demand) are considered. In project management, renewable, non-renewable, and semi-renewable resources may also be important parameters. Therefore, this extension will be considered in future research. The paper presented a formal description of discrete and stochastic versions of hybrid trade-off problems; however, the paper focused only on continuous cases. The other extension is to address multi-mode resource constraint problems. However, these extensions require heuristic or meta-heuristic solvers because both discrete trade-off problems and multi-mode resource constraint resource allocation problems are NP-hard problems (see [4]).

Another possible application of this method is risk management and risk analysis. Supplementary task completion may model changes in management or client claims. Flexible task dependency may model technological changes. In this case, a more appropriate matrix-based model could be specified, and the efficiencies of TPM (e.g., trade-off methods) and APM (e.g., scoring and (re)structuring methods) could be compared for different types of project plans. This risk-management approach can be combined with current risk evaluation and mitigation methods, such as [20].

Acknowledgement

This research is supported by a János Bolyai Fellowship, Hungarian Academy of Science. We acknowledge the financial support of Széchenyi 2020 under the EFOP-3.6.1-16-2016-00015.

References

[1] D. R. Fulkerson, A network flow computation for project cost curves, Management science 7 (1961) 167–178.

[2] S. E. Elmaghraby, Activity networks: Project planning and control by network models, Wiley New York, 1977.

[3] J. J. Moder, C. R. Phillips, E. W. Davis, Project management with cpm, pert and precedence diagramming., 1983.

[4] P. Brucker, A. Drexl, R. Mohring, K. Neumann, E. Pesch, Resource-constrained project scheduling: Notation, classification, models, and methods, European Journal of Operational Research 112 (1999) 3–41.

[5] P. De, E. J. Dunne, J. B. Ghosh, C. E. Wells, The discrete time-cost tradeoff problem revisited, European Journal of Operational Research 81 (1995) 225–238.

[6] E. L. Demeulemeester, W. S. Herroelen, S. E. Elmaghraby, Optimal procedures for the discrete time/cost trade-off problem in project networks, European Journal of Operational Research 88 (1996) 50–68.

[7] H. R. Tareghian, S. H. Taheri, On the discrete time, cost and quality trade-off problem, Applied Mathematics and Computation 181 (2006) 1305–1312.

[8] A. Babu, N. Suresh, Project management with time, cost, and quality considerations, European Journal of Operational Research 88 (1996) 320 – 327.

[9] A. Salmasnia, H. Mokhtari, I. Nakhai Kamal Abadi, A robust scheduling of projects with time, cost, and quality considerations, The International Journal of Advanced Manufacturing Technology 60 (2012) 631–642.

[10] C. Feng, L. Liu, S. Burns, Stochastic construction time-cost trade-off analysis, Journal of Computing in Civil Engineering 14 (2000) 117–126.

[11] S. S. Said, M. Haouari, A hybrid simulation-optimization approach for the robust discrete time/cost trade-off problem, Applied Mathematics and Computation 259 (2015) 628 – 636.

[12] R. K. Wysocki, Effective Project Management: Traditional, Agile, Extreme, John Wiley & Sons, 5. auflage edition, 2009.

[13] M. Kuhrmann, P. Diebold, J. Mnch, P. Tell, K. Trektere, F. M. Caffery, G. Vahid, M. Felderer, O. Linssen, E. Hanser, C. Prause, Hybrid software development approaches in practice: A european perspective, IEEE Software PP (2018) 1–1.

[14] K. Schmitz, R. Mahapatra, S. Nerur, User engagement in the era of hybrid agile methodology, IEEE Software (2018) 1–1.

[15] V. Rahimian, R. Ramsin, Designing an agile methodology for mobile software development: A hybrid method engineering approach, in: Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on, pp. 337–342.

[16] M. Tyagi, S. Munisamy, L. Reddy, Traditional and hybrid software project tracking technique formulation: state space approach with initial state uncertainty, CSI Transactions on ICT 2 (2014) 141–151.

[17] Z. T. Kosztyán, Exact algorithm for matrix-based project planning problems, Expert Systems with Applications 42 (2015) 4460 – 4473.

[18] Z. T. Kosztyán, J. Kiss, Pem–a new matrix method for supporting the logic planning of software development projects, in: DSM 2010: Proceedings of the 12th International DSM Conference, Cambridge, UK, 22.-23.07. 2010.

[19] D. Tang, R. Zhu, J. Tang, R. Xu, R. He, Product design knowledge management based on design structure matrix, Advanced Engineering Informatics 24 (2010) 159–166. Enabling Technologies for Collaborative Design.

[20] C. Muriana, G. Vizzini, Project risk management: A deterministic quantitative technique for assessment and mitigation, International Journal of Project Management 35 (2017) 320 – 340.

[21] H.-l. Bi, X. Jia, F.-q. Lu, M. Huang, Schedule risk management of it outsourcing project using negotiation mechanism, in: E. Qi, J. Shen, R. Dou (Eds.), Proceedings of the 23rd International Conference on Industrial Engineering and Engineering Management 2016, Atlantis Press, Paris, 2017, pp. 29–33.

[22] J.-R. Sack, T. Strothotte, A characterization of heaps and its applications, Information and Computation 86 (1990) 69 – 86.

[23] R. Kolisch, A. Sprecher, {PSPLIB} - a project scheduling problem library: {OR} software - {ORSEP} operations research software exchange program, European Journal of Operational Research 96 (1997) 205 – 216.

[24] E. Demeulemeester, M. Vanhoucke, W. Herroelen, Rangen: A random network generator for activity-on-the-node networks, Journal of Scheduling 6 (2003) 17–38.

[25] 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 Journal of Operational Research 187 (2008) 511 – 524.

[26] V. V. Peteghem, M. Vanhoucke, An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances, European Journal of Operational Research 235 (2014) 62 – 72.

[27] M. Vanhoucke, Measuring the efficiency of project control using fictitious and empirical project data, International Journal of Project Management 30 (2012) 252 – 263.

[28] L. V. Tavares, J. A. Ferreira, J. S. Coelho, The risk of delay of a project in terms of the morphology of its network, European Journal of Operational Research 119 (1999) 510 – 537.

[29] T. Dingsøyr, S. Nerur, V. Balijepally, N. B. Moe, A decade of agile methodologies: Towards explaining agile software development, Journal of Systems and Software 85 (2012) 1213–1221. Special Issue: Agile Development.

[30] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, Some recent advances in network flows, SIAM Review 33 (1991) 175–219.