Doctoral Thesis

Synergy-Based Software Project Scheduling Problem: Formalization, Simulation, and Solution

Author:

Marcell Tamás Kurbucz

Supervisor:

Prof. **Zsolt Tibor Kosztyán**, *Ph.D.*

Thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy in the
Doctoral School in Management Sciences
and Business Administration

DOI:10.18136/PE.2021.781

University of Pannonia
Faculty of Business and Economics
Department of Quantitative Methods
Veszprém, Hungary

19 May 2021

Synergy-Based Software Project Scheduling Problem: Formalization, Simulation, and Solution

Thesis for obtaining a Ph.D. degree in the Doctoral School in Management Sciences and Business Administration of the University of Pannonia

in the branch of Management Sciences

Author: Marcell Tamás Kurbucz

Supervisor: Prof. Zsolt Tibor Kosztyán, Ph.D.

propose acceptance (yes / no) ....
(supervisor)

As a reviewer, I propose acceptance of the thesis:

Name of Reviewer .... yes / no

....
(reviewer)

Name of Reviewer: .... yes / no

....
(reviewer)

The Ph.D. candidate has achieved .. % at the public discussion.

Veszprém, ...

(Chairman of the Committee)

The grade of the Ph.D. Diploma …....................................... (%)

Veszprém,

....

(Chairman of UDHC)

Abstract

SYNERGY-BASED SOFTWARE PROJECT SCHEDULING PROBLEM: FORMALIZATION, SIMULATION, AND SOLUTION

The adequate allocation of human resources is one of the most important success factors in software projects. Although project teams can be regarded as complex systems where a team's performance is highly influenced by the interdependencies among team members, the allocation methods applied to date have focused only on the individual skills and consider project teams as units of isolated workers. The existing software project scheduling problem (SPSP) is extended to (1) consider different levels of skills of employees and (2) the pairwise synergies between them, as well as to (3) handle the flexible structure of a project, which is used in flexible management such as agile project management. To better understand the impact of synergies on the project's cost, the solutions of the classical and extended SPSP versions are analyzed and compared on the generated project networks. The results show not only that this factor has a highly significant impact but also that the project cost strongly depends on the structural parameters of the synergy network (e.g., topology, network size and degree centrality). Among these parameters, low degree centrality and some topologies, most notably star and circular networks, obtained the highest reduction in the projects' total cost.

Keywords: Software Project Scheduling; Staffing; Synergy Network; Social Network; Genetic Algorithm

Resumen

PROBLEMA DE PROGRAMACIÓN DE PROYECTOS DE SOFTWARE BASADO EN SINERGIAS: FORMALIZACIÓN, SIMULACIÓN Y SOLUCIÓN

La adecuada asignación de recursos humanos es uno de los factores de éxito más importantes en los proyectos de software. Aunque los equipos de proyecto pueden considerarse sistemas complejos en los que el desempeño de un equipo está muy influenciado por las interdependencias entre los miembros del equipo, los métodos de asignación aplicados hasta la fecha se han centrado solo en las habilidades individuales y consideran a los equipos de proyecto como unidades de trabajadores aislados. El problema de programación de proyectos de software existente (SPSP) se amplía para (1) considerar los diferentes niveles de habilidades de los empleados y (2) las sinergias por pares entre ellos, así como para (3) manejar la estructura flexible de un proyecto, que se utiliza en la gestión flexible como la gestión ágil de proyectos. Para comprender mejor el impacto de las sinergias en el costo del proyecto, las soluciones de las versiones SPSP clásicas y extendidas se analizan y comparan en las redes de proyectos generadas. Los resultados muestran no solo que este factor tiene un impacto muy significativo, sino también que el costo del proyecto depende en gran medida de los parámetros estructurales de la red de sinergia (por ejemplo, topología, tamaño de la red y grado de centralidad). Entre estos parámetros, la centralidad de bajo grado y algunas topologías, entre las que destacan las redes en estrella y circulares, obtuvieron la mayor reducción en el costo total de los proyectos.

Palabras Clave: Programación de Proyectos de Software; Dotación de Personal; Red de Sinergia; Red Social; Algoritmo Genético

Contents

List of Figures

List of Tables

List of Acronyms

APM: agile project management

CPM: critical path method

DMM: domain mapping matrix

DSM: dependency/design structure matrix

FIRO-(B): fundamental interpersonal relations orientation (behavior)

GA: genetic algorithm

HRAP: human resource allocation problem

IT: information technology

KSA: knowledge, skill and ability

MDM: multi-domain matrix

MOCell: multi-objective cellular genetic algorithm

MPx: emertxe project management

MS-RCPSP: multi-skill resource-constrained project scheduling problem

NMM: Nelder-Mead method

NSGA-II: non-dominated sorting genetic algorithm II

UML: unified modeling language

PAES: Pareto archived evolution strategy

PEM: project expert matrix

PERT: program/project evaluation and review technique

PSP: project scheduling problem

RCPSP: resource-constrained project scheduling problem

SMM: synergy-based multi-domain matrix

SNPM: stochastic network planning method

SPEA2: strength Pareto evolutionary algorithm 2

SPSP: software project scheduling problem

SSPSP: synergy-based software project scheduling problem

SynASF: synergy-based agile simulation framework

SynAPS: synergy-based agile project scheduling algorithm

TPM: traditional project management

xPM: extreme project management

Acknowledgments

I am grateful to my advisor, Prof. Zsolt Tibor Kosztyán, Ph.D., and to my co-author, István Szalkai, Ph.D. for their support and guidance. I also thank Prof. György Dósa, Ph.D., and reviewers, Nikolett Deutsch, Ph.D. habil., and Csaba Hegedűs, Ph.D. for their valuable advice. Special thanks to Péter Harta for making the data he collected available to me for the practical example.

CHAPTER 1

Introduction

1.1 Motivation of the Thesis

Agile development methods have been widely used in software engineering over the last decade (Lindsjørn et al., 2016). Contrary to the traditional planning approach, this methodology focuses on “individuals and interactions over processes and tools”, “working software over comprehensive documentation”, “customer collaboration over contract negotiation”, and “responding to change over following a plan” (Fowler et al., 2001, p. 2).[1] Since it emphasizes teamwork more than traditional development methods do (Nerur et al., 2005), it is not surprising that the tasks of allocating human resources and scheduling play a critical role in the success of software development projects (see, e.g., Jalote and Vishal, 2003), and consequently, in the competition in the information technology (IT) industry (Nan and Harter, 2009). To reduce development costs and beat the market, companies have to make reliable project plans; however, efficient allocation of workers is a particularly difficult and challenging problem, particularly for medium- to large-scale projects (see, e.g., Minku et al., 2013). For instance, in China alone, more than 40% of software projects were unsuccessful due to incoherent planning of project tasks and human resources (Ding and Jing, 2003).

In the literature on software development, the common issue of resource allocation and task scheduling is referred to as the software project scheduling problem (SPSP)

---

[1] Agile methodology is used as an umbrella term to describe a number of development methods (Dybå and Dingsøyr, 2008; Dingsøyr et al., 2012).

(see, e.g., Vega-Velázquez et al., 2018), which is related to the resource-constrained project scheduling problem (RCPSP) (Alba and Chicano, 2007; Vega-Velázquez et al., 2018) – or more specifically, to the multi-skill resource-constrained project scheduling problem (MS-RCPSP) (Myszkowski et al., 2019; Tirkolaee et al., 2019). The efficiency of solving this problem usually depends on several factors. On the one hand, the development process should be as short as possible, thus allowing the allocation of resources to other profitable processes as soon as possible. On the other hand, the associated cost should be minimal. This multi-objective nature makes planning even more complicated and, as a result of the increasing size of software projects, makes manual scheduling almost impossible (Shen et al., 2018).

Research on this topic has intensified rapidly in recent years; however, due to the above-mentioned reasons such research has mostly focused on the technical improvements of computer-aided planning (see, e.g., Chicano et al., 2011; Di Penta et al., 2011; Luna et al., 2014). Even though human aspects are an important factor of the success of software projects and should be a key research area within the field of (agile) software project planning, existing studies have only explored the human properties of the scheduling problem to a limited extent (Shen et al., 2018).

The goals of this dissertation are twofold: to proposing an agile approach that takes into account the impact of project team members on each other's performance during scheduling and to examining the effect of these interactions on project cost. To accomplish these goals, I extend the classical SPSP with synergies between employees and present a novel matrix-based approach that can handle employees' interactions and supports agile software development.[2] Then, I analyze and compare the solutions of the classical and the extended SPSP versions on projects from generated project networks to evaluate the impact of synergies on their costs.[3]

---

[2] For simplicity, in this study, pairwise synergies between employees are applied to model their interactions.

[3] Project networks, resources and skills are generated by iMOPSE multi-skill resource-constrained project scheduling problem generator (Myszkowski et al., 2019).

1.2 Research Questions

As a result of the dissertation, the following research questions (**RQs**) are answered:

RQ$_1$: Is it possible to determine a scheduling problem for traditional and flexible project planning environments that considers not only the skills of human resources but also the synergies between them?

RQ$_2$: Is it possible to develop a network- or matrix-based project scheduling model that takes into account the flexibility of project plans, the skills of human resources as well as the synergies between them?

RQ$_3$: Is there a(n optimal) solution for scheduling a flexible software project plan that considers the synergies between resources?

RQ$_4$: Is it possible to develop a simulation framework to examine the impact of the synergies between resources, the structures of synergy networks, the skills of human resources as well as the size, flexibility, and constraints of the project on the implementation of the project schedule?

1.3 Structure of the Thesis

The rest of the dissertation is organized as follows. Chapter 2 reviews the related studies and defines the research assumptions based on them. Chapter 3 introduces the extended SPSP, the hybrid genetic algorithm proposed to solve this problem as well as the proposed simulation framework. Chapter 4 presents the results, answers the research questions and defines the research theses. Chapter 5 introduces a practical example. Chapter 6 shows the threats to the validity. Chapter 7 concludes the dissertation. Finally, Chapter 8 proposes future research directions.

CHAPTER 2

Related Studies

This chapter provides a brief overview of the basic definitions and contexts of project and project management. Then it discusses the project planning problems and techniques related to the topic of the dissertation, followed by a detailed review of the main features and research directions of the SPSP. Finally, it presents some of the most popular theories of high-functioning teamwork in terms of industrial psychology and sociology, as well as the concept of synergy networks.

2.1 Project and Project Management

2.1.1 Basic Definitions

There are many definitions of the project in the project management literature, but since the field is practitioner dominated, these definitions are not constructed as rigorously as in established scientific fields (Chiocchio et al., 2015). Most of the definitions emphasize the uniqueness (see, e.g., Görög, 1999, 2007; Wysocki, 2011, 2019), complexity (see e.g., Cleland and King, 1983; Archibald, 2003), and temporary nature (see, e.g., Shenhar, 2001; Shenhar and Dvir, 2007; Vidal et al., 2011) of the project, while others focusing on its strategic role (see, e.g., Cleland, 2007; Leybourne, 2007; Cooke-Davies et al., 2009), constraints (see, e.g., Graham et al., 1979; Cleland and King, 1983), its significant human and non-human resource needs (see, e.g., Cleland and Kerzner, 1985; Jamieson and Morris, 2007), or more specifically, the teamwork in which the project is implemented (see, e.g., Schwab and Miner, 2008).

The most important two definitions regarding the topic of the dissertation are the following.

- "A complex effort to achieve a specific objective within a schedule and budget target, which typically cuts across organizational lines, is unique and is usually not repetitive within the organization" (Cleland and King, 1983, p. 70).
- "A project is a sequence of unique, complex, and connected activities that have one goal or purpose and that must be completed by a specific time, within budget, and according to specification." (Wysocki, 2019, p. 4).

As the present dissertation is related to project planning, more precisely to project scheduling and resource allocation, I will focus on the constraints rather than the uniqueness of projects when providing definitions. Although the uniqueness of projects is important from an organizational point of view, the planning methods presented in the dissertation can be applied regardless of the uniqueness of projects. For this reason, I will hereinafter use the definition of Wysocki (2019, p. 4) of what a project is, however, I disregard the uniqueness of activities emphasized in the original definition. The key concepts of the areas of scheduling and resource allocation can be defined as follows.

- The *activity* (or *task*) is "a distinct, scheduled portion of work performed during the course of a project" (PMI, 2012, p. 6).
- The *event* is "a point in time when an activity starts or ends" (Mubarak, 2019, p. 24).
- The *milestone* is "a significant point or event in a portfolio, program, or project" (PMI, 2012, p. 6).
- A *finish-to-start precedence relationship* between activities means that "a successor activity cannot start until a predecessor activity has finished. For example, installing the operating system on a PC (successor) cannot start until the PC hardware is assembled (predecessor)" (PMI, 2017, p. 190).
- A *resource* is a skilled employee "(specific disciplines either individually or in crews or teams), equipment, services, supplies, commodities, materials, budgets, or funds required to accomplish the defined work" (PMI, 2011, p. 2).

- The *project team* “consists of individuals with assigned roles and responsibilities who work collectively to achieve a shared project goal” (PMI, 2017, p. 309).

Similarly to the project term, there is no consensus definition of project management either. Two widely used, commonly formulated definitions are derived from PMI (2017) and Phillips (2018).

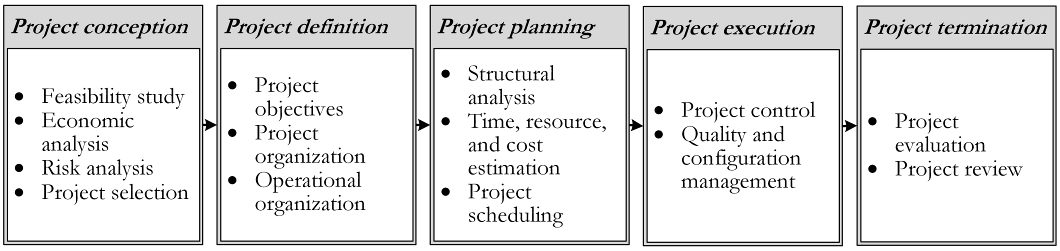
- Based on PMI (2017, p. 10), “project management is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements.” Moreover, “project management is accomplished through the appropriate application and integration of the project management processes identified for the project” and it “enables organizations to execute projects effectively and efficiently.”
- According to Phillips (2018, p. 13), “project management is the supervision and control of the work required to complete the project vision. The project team carries out the work needed to complete the project, while the project manager schedules, monitors, and controls the various project tasks. Project management requires that you apply your knowledge, skills, tools, and techniques, and do whatever it takes, generally speaking, to achieve the project objectives.”

In line with the above-mentioned definitions, the term project management is hereinafter used to mean “the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements” PMI (2017, p. 10).

2.1.2 Project Life Cycle

From a project management perspective, the life cycle of a project consists of five phases, each of which involves specific managerial tasks (see, e.g., Lewis James, 1997; Klein, 2012; Schwindt et al., 2015). These consecutive phases are illustrated in Fig. 1.

FIGURE 1. Project life cycle
(*Source:* *Schwindt et al., 2015, p. 27*)



In the first phase, so-called *project conception*, by using feasibility studies and economic and risk analysis, it is decided whether or not a project should be implemented. It is followed by the *project definition* phase, that defines the objectives and organizational form of the project as well as the milestone plan.[4] In the *project planning* phase, the project is decomposed into activities, then the precedence relations of these activities are specified.[5] Furthermore, for each task, the duration, the required resources, as well as the cost associated with the execution of that task are estimated. Finally, a project schedule is determined by some planning approach. At the end of the planning phase, the project is ready for implementation and the project execution phase begins (Schwindt et al., 2015). During the *execution phase*, project management continuously monitors and evaluates whether or not the project is performed according to the established baseline schedule. If significant deviations are detected, the plan has to be revised or an execution strategy – defined in the planning phase – is used to bring the project back to course.[6] Finally, the project is evaluated and documented in the *termination phase* (Schwindt et al., 2015).

---

[4] The milestone plan is a logical plan that presents the interconnections between milestones (Andersen, 2006).

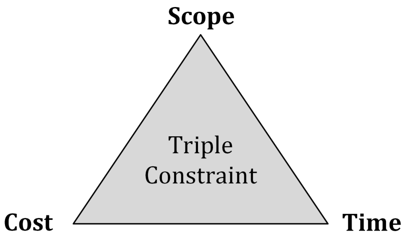
[5] This step is often called logical planning in the literature (see, e.g., Pecora and Cesta 2002; Kosztyán and Kiss 2010).

[6] Quality and configuration management are also performed in this phase (Turner 2009; PMI, 2017).

2.1.3 Triple Constraint of Project Management

As highlighted by the studies presented in Section 2.1.1 (Cleland and King, 1983; Archibald, 2003; Andersen et al., 2009), the primary constraints of projects are scope, time, and budget (PMI, 2017). The model referred to as the triple constraint, project triangle, or iron triangle in the literature, is focused on the interdependence between these three constraints.[7] To explain it more plainly if a change is made to the time taken to complete the project, one way or another it will have an impact on either the cost or scope of the project or both. Similarly, changing the scope of the project will impact the cost or the time is taken or both and so on (Dwyer et al., 2004). On the one hand, this model lays the foundation for the formation of project goals. On the other, the assessment of the success and failure of the project implementation after completion is fundamentally determined by the elements of the triangle (Pinto and Prescott, 1988; Atkinson, 1999). The triple constraint is illustrated in Fig. 2.

FIGURE 2. Triple constraint of project management
(Source: Hinde, 2018, p. 333)



2.1.4 Types of Projects

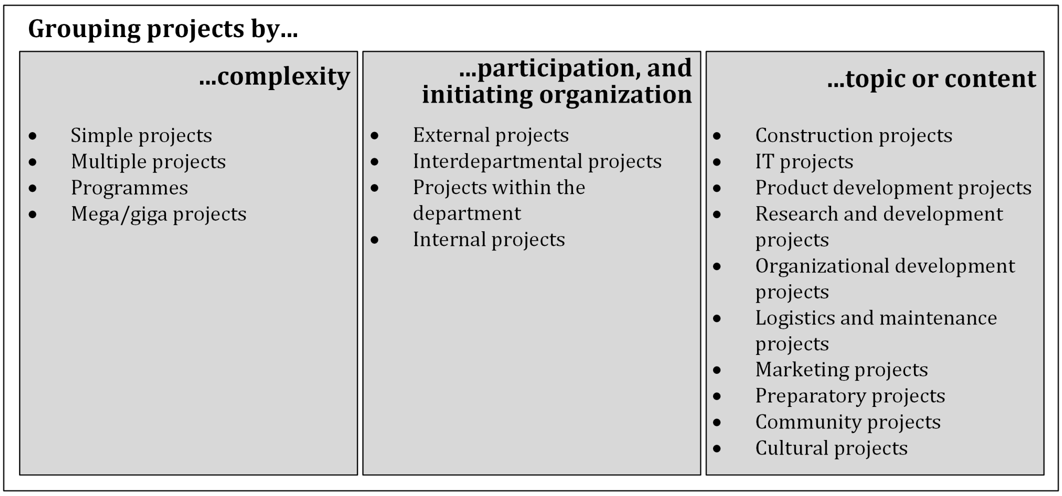
Grouping projects is essential to determine the appropriate methods for planning and managing them (Görög and Ternyik, 2001). According to Görög (2007), projects can be grouped based on their complexity, the nature of participation, and the initiating organization, as well as their topic or content (see Fig. 3).

---

[7] The origins of the triple constraint are unclear but based on Atkinson (1999), it has been used since at least the 1950s.

FIGURE 3. Typology of projects

(Source: based on Görög, 2003, p. 36)



Grouping projects by...

...complexity	**...participation, and initiating organization**	**...topic or content**
Simple projects Multiple projects Programmes Mega/giga projects	External projects Interdepartmental projects Projects within the department Internal projects	Construction projects IT projects Product development projects Research and development projects Organizational development projects Logistics and maintenance projects Marketing projects Preparatory projects Community projects Cultural projects

Since the dissertation is related to IT projects, the characteristics of this project type is discussed in detail. According to (Bannerman and Thorogood, 2012, p. 1), "IT projects are discrete and unique activities that serve as vehicles of multidimensional IT-based change." As Sheard et al. (2015) points out, the factor that most characterizes these projects is complexity, however, this complexity is a characteristic of more than just a technical system being developed. Following Rodriguez-Repiso et al. (2007) and Iriarte and Bayona (2020), IT projects are typically created in a complex environment by the numerous and continuous interaction of people whose work are highly interdependent. As a consequence, these projects are often canceled or reduced in scope because of overruns in cost and/or time or failure to produce anticipated benefits (Mehler, 1991; Lederer and Prasad, 1993; Kumar, 2002). Based on Rodriguez-Repiso et al. (2007, p. 2), the practical management of IT projects beyond the theories for success finds significant difficulties as follows:

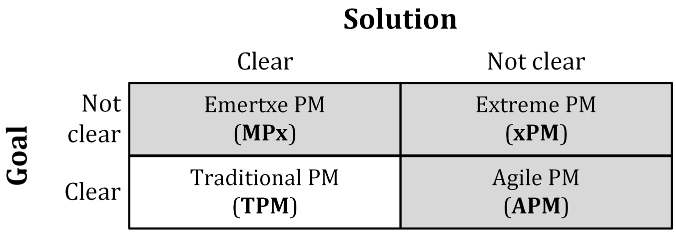
- "IT projects are often poorly defined, codes of practice are frequently ignored, and in some cases, not many lessons are learned from past experience."

- “Market pressures demand delivery in the shortest time frame even if it may result in a lower quality product.”
- “The rapid pace of technological progress in IT hinders the expertise in a particular technique and creates a culture where the use of tools not completely tested is acceptable and commonplace.”
- “The tendency to write new software code to perform well-established functions decreases reliability.”
- “IT projects contain a greater degree of a novelty than other engineering projects. In particular, IT projects related to product innovation development are an extremely complex, risky, and expensive endeavor.”
- “IT projects involve numerous iterations and continuous interaction between everyone involved in design and implementation. Their work is highly interdependent which necessitates efficient communication within the project team.”

2.1.5 Project Management Approaches

The term project management approach is mainly used as a set of principles and guidelines that define how specific project is managed (Introna and Whitley, 1997; Iivari et al., 2000). Principles developed in the 1950s require that methods and procedures be applied in a uniform manner, regardless of the type of project (Špundak, 2014). The basic idea behind this, so-called traditional project management (TPM) approach is that projects are relatively simple, predictable, and linear with clearly defined boundaries which all makes it easy to plan in detail and follow that plan without much changes (see, e.g., Wysocki, 2011, 2019; Boehm et al., 2000; Cicmil et al., 2006; Špundak, 2014). In recent decades, the objections regarding the rigidity of the TPM, together with the growing requests for continuous innovations that have impacted all industries and with the cost reduction trends, have led to the emergence of new project management approaches (Špundak, 2014). In line with Wysocki (2019), these new, so-called complex project management approaches (agile, extreme and emertxe) can be compared with TPM according to the clarity of their objectives and solutions (see Fig. 4).

FIGURE 4. Project management approaches in terms of goal and solution
(Source: Wysocki, 2019, p. 8)



		Solution	
		Clear	Not clear
Goal	Not clear	Emertxe PM **(MPx)**	Extreme PM **(xPM)**
	Clear	Traditional PM **(TPM)**	Agile PM **(APM)**

As Fig. 4 shows, TPM is suitable for managing well-structured projects with clear requirements and project scope. This approach accepts that actions affecting the project are foreseeable and that tools, techniques and actions are well-defined (Toljaga-Nikolic et al., 2017). When neither the project goal nor solution are known or not clearly defined, then the extreme project management (xPM) approach should be applied. Emertxe (MPx) is the inverse xPM approach, mainly used when a new technology is developed but does not have a known application yet (Toljaga-Nikolic et al., 2017). When the goal is clear but the solution is missing some or most parts, one can apply the agile project management (APM) approach (Wysocki, 2019). While xPM and MPx are related to research and development projects, the fourth approach, agile project management (APM) is mainly applied in software development. Given the topic of the dissertation, in the following, this approach is presented in detail and compared with TPM.

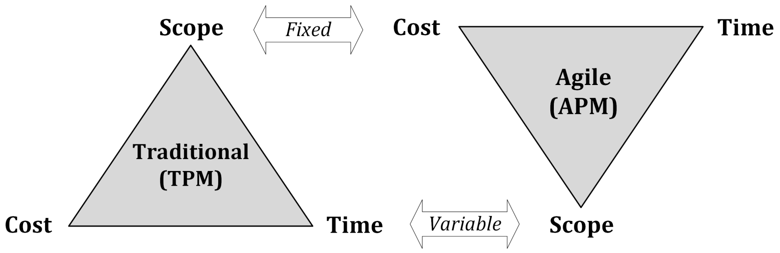
Contrary to the traditional approach, APM methods "are lightweight processes that employ short iterative cycles, actively involve users to establish, prioritize, and verify requirements, and rely on a team's tacit knowledge as opposed to documentation. A truly agile method must be iterative (takes several cycles to complete), incremental (not deliver the entire product at once), self-organizing (teams determine the best way to handle work), and emergent (processes, principles, and work structures are recognized during the project rather than predetermined)" (Boehm and Turner, 2005, p. 3). The key differences between TPM and APM are summarized in Table 1.

TABLE 1. Key differences between TPM and APM
(Source: Dybå and Dingsøyr, 2008, p. 4)

Area of interest	TPM	APM
Basic assumptions	The product can be fully described at the planning phase of the project.	A high quality product is worked out by small specialised teams on a continuous improvement basis.
Management style	Autocratic, Prescriptive	Affiliate, Democratic
Knowledge man.	Explicit	Tacit
Communication	Formal	Informal
Organ. structure	Bureaucratic, Highly formalised	Flexible, Cooperative
Quality control	Planned in time in details.	On-going control of the achieved sub-results toward the client's expectations.

Based on Dalcher (2009), in the TPM approach, the scope of the project is fixed, which, if necessary, must be achieved even at the cost of exceeding the planned costs and duration. In contrast, in the case of the APM, the available time and budget appear as a constraint, within which the scope must be achieved as much as possible (see Fig. 5).

FIGURE 5. Comparison of TPM and APM
(Source: Dalcher, 2009)



According to an international survey conducted by Wysocki (2011), only less than 20% of the projects belong to the traditional (like infrastructure) projects, and about 70% of the projects are handled as agile ones.[8] One of the main reasons for the popularity of the APM is that projects managed in this way are typically more successful than projects managed within traditional frameworks. Based on the results of Standish Group's recent survey, IT projects managed in the agile form are about

---

[8] The remaining about 10% are handled as extreme or emertxe project (Wysocki, 2011).

two times more successful than projects handled with traditional models and about a third time less likely to fail (SGI, 2019). The detailed results of the survey is presented in Table 2.

TABLE 2. Project success rates in traditional and agile IT projects
(Source: SGI, 2019)

Method	Successful	Challenged	Failed
Waterfall[9]	11%	60%	29%
Agile	39%	52%	9%

Successful: project that met all three of the triple constraints: schedule, cost, and scope.
Challenged: project that met two out of three constraints.
Failed: project that is canceled before it is completed, or completed but not used.

2.2 Project Planning Problems and Techniques

Project planning has been defined as "the process of choosing the one method and order of work to be adopted for a project from all the various ways and sequences in which it could be done" (Antill and Woodhead 1990, p. 8, as cited in Callahan et al. 1992, p. 2, Mubarak 2019, p. 4). According to PMI (2017, p. 554), the planning process group refers to "those processes required to establish the scope of the project, refine the objectives, and define the course of action required to attain the objectives that the project was undertaken to achieve". It serves as a foundation for several related functions, such as cost estimating, project control, quality control, safety management, scheduling or the allocation of human and non-human resources (Mubarak, 2019). Since both the project scheduling problem (PSP) and the human resource allocation problem (HRAP) are related to the SPSP, we briefly overview them before reviewing the literature of the SPSP in detail.

2.2.1 Project Scheduling Problem

Project scheduling is mainly related to selecting execution modes and fixing execution time intervals for the activities of a project (Schwindt et al., 2015). According to PMI (2011, p. 2), "it ensures the development of effective schedule models

---

[9] The waterfall model (Benington, 1983) is a traditional planning approach, widely used in software development.

through the application of skills, tools, techniques, and intuition acquired through knowledge, formal and informal training, and experience. A schedule model rationally organizes and integrates various project components (e.g., activities, resources, and logical relationships) to optimize the information available to the project management team and facilitate the likelihood of a successful project completion within the approved schedule baseline."

The first methods for solving the project scheduling problem (PSP) date back to the 1950s, when the widely known network-based models like the critical path method (CPM) or the project/program evaluation and review technique (PERT) were formulated and developed (Ratajczak-Ropel and Skakovski, 2018). These techniques allowed projects to be portrayed by networks in which activities are represented by arcs, events are represented by nodes, and the interrelations between activities are defined by the network structure (Icmeli et al., 1993). Their objective is to complete the project in the shortest time allowed by the priority relationships. CPM and PERT are referred to as complementary tools in the literature, because "CPM employs one time estimation and one cost estimation for each activity; PERT may utilize three time estimates (optimistic, expected, and pessimistic) and no costs for each activity" (Brennan, 1968, p. 1). These methods consider only the duration and precedence conditions of the activities and ignore the resource requirements (Mateo, 2016), which results in a favorable, so-called polynomial-time computation need on the one hand, and an oversimplified scheduling problem on the other (Özdamar and Ulusoy, 1995).[10]

In many real-life situations, there are delays in the implementation of certain activities when resources are not available in sufficient quantities during the time interval when they are scheduled to take place (Icmeli et al., 1993). The problem that complements the simple PSP with the scarcity of available resources is called

---

[10] An algorithm is said to be of polynomial time if its running time is upper bounded by a polynomial expression in the size of the input for the algorithm (see, e.g., Li et al., 2015).

the resource-constraint project scheduling problem (RCPSP) (Pritsker et al., 1969) and it has an NP-hard complexity.[11] Informally, the RCPSP considers resources of limited availability and activities of known duration and resource needs, linked by precedence relations. The problem consists of finding a schedule with a minimum duration by assigning a start time for each activity, while respecting priority conditions and resource availability (Artigues et al., 2008). Since the 1960s, a number of heuristics and many exact solution techniques have emerged to solve the RCPSP (Icmeli et al., 1993), and today a significant portion of scheduling problems are based on the RCPSP (Özdamar and Ulusoy, 1995).

2.2.2 Human Resource Allocation Problem

In the human resource allocation problem (HRAP), different project activities require employees with different skills, and the skill proficiency of employees significantly influences the efficiency of project execution (see, e.g., Chen and Zhang 2013). According to Kumar and Ganesh (1998) and Chen and Zhang (2013), techniques like PERT and CPM lack the consideration of resource allocation, and scheduling models like the basic RCPSP do not consider the allocation of employees with various skills. Consequently, tools based on these traditional planning techniques generally consider the scheduling of activities and the human resource allocation as two separate tasks. Thus, the HRAP must be solved manually by project managers (Kumar and Ganesh, 1998), which results in inefficient resource allocation and poor management performance (Chen and Zhang, 2013). As Yoshimura et al. (2006, p. 2) argues, "human resource allocation decisions are usually made according to the experience and intuition of project managers. However, as the contents of the projects become more complex and the required abilities to carry them out more diversified, there is an increasing need for logical support systems

---

[11] NP-hard problem means that there is no known algorithm which can solve the problem in polynomial time (see, e.g., Islam et al., 2019).

to assist decision makers when seeking the best possible deployment of the human resources." In the past twenty years, a number of methods have been developed to solve this complex, NP-hard problem (see, e.g., Cheng and Chu 2012; Almeida et al. 2016; Young et al. 2017; Wang and Zheng 2018). Among these, matrix-based planning methods have become increasingly popular.

2.2.3 Matrix-Based Flexible Planning

Unlike traditional project planning techniques, matrix-based methods provide a flexible planning environment and support the APM approach. Most of these methods are based on the so-called dependency/design structure matrix (DSM) developed by Steward (1981). The DSM is a binary square ($n \times n$) matrix that represents the strict successors of the project activities, and contrary to the majority of the network planning techniques, the circles in the dependency structure can be identified and handled by this method.[12] To augment the DSM method, Danilovic and Browning (2007) formalized the domain mapping matrix (DMM), which compares two DSMs from two different project domains. Contrary to a DSM, a DMM is a rectangular ($m \times n$) matrix, where m is the size of the first DSM and n is the size of the second. Another matrix proposed by Gorbea et al. (2008), the so-called the multi-domain matrix (MDM), is a fusion of DSM and DMM that allows for the integration of numerous different domains in one model (Deubzer et al., 2008) (see Fig. 6).

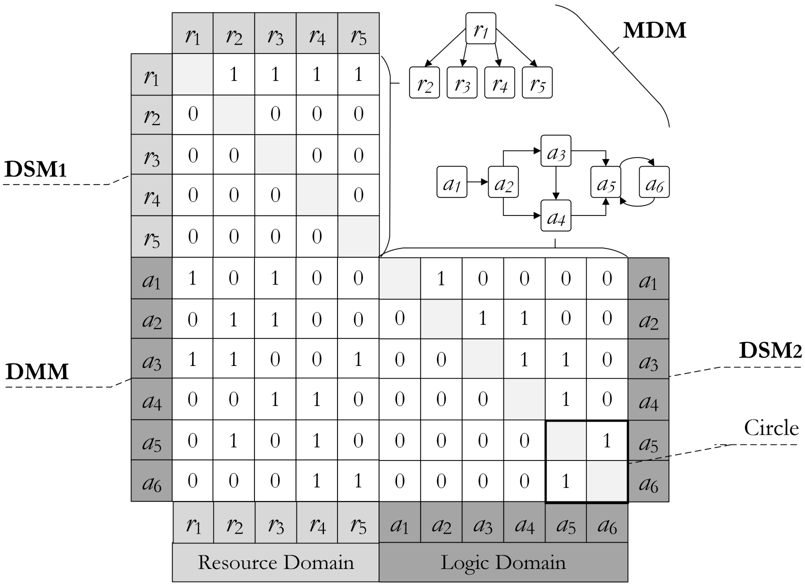
Although the original, binary DSM can only be used for logical planning, its improved forms can also be used for solving the PSP (Chen et al., 2003; Maheswari and Varghese, 2005; Gunawan and Ahsan, 2010; Shi and Blomquist, 2012; Mohammadi et al., 2014), as well as the RCPSP (Cho and Eppinger, 2005; Kosztyán,

---

[12] Note that even though the dissertation proposes a matrix-based method for software project planning, in its current form, the proposed method only handles acyclic project structures. Thus, we will herein-after only focus on projects with such a structure.

FIGURE 6. Multi-domain matrix (MDM)

(Source: own figure)



2015, 2020; Kosztyán and Szalkai, 2020), while providing a more flexible environment for project modeling compared to the original method. For instance, while the stochastic network planning method (SNPM) (Kosztyán et al., 2008) is able to model uncertain relations between activities, the project expert matrix (PEM) (Kosztyán et al., 2010) can also distinguish mandatory and supplementary activities based on the probability of their realization. The Project Domain Matrix (PDM) extends the PEM – in the model, it is called the logical domain – with cost, time and resource domains (Kosztyán, 2015; Kosztyán et al., 2020).[13] Furthermore, to transform the RCPSP into a more practical – and consequently, more complex – problem, Myszkowski et al. (2015a) complemented it with the skills domain and defined the multi-skill resource-constraint project scheduling problem (MS-RCPSP). According to Myszkowski et al. (2015a), in the MS-RCPSP, resources dispose of some given pool of skills, while every activity requires some skills in a given level to

---

[13] DSM-based methods can also be used in other areas of project planning, such as project monitoring and coordination (see, e.g., Kosztyán and Kurbucz, 2015; Kurbucz, 2016).

be performed. It means that not every resource is capable of performing every activity. In addition, the performance cost of the project schedule was added as an another criterion, transforming the classical single-objective (time) RCPSP into a multi-objective (time-cost trade-off) MS-RCPSP.

2.3 Software Project Scheduling Problem

As it is presented in Section 2.2, a number of methods have been proposed to solve the RCPSP (see, e.g., Hartmann and Briskorn, 2010; Węglarz et al., 2011), however, the common problem of scheduling and human resource allocation is a much newer and more complex area (see, e.g., Fernandez-Viagas and Framinan 2014).[14] In the software development literature, this problem is referred to as the software project scheduling problem (SPSP) (Alba and Chicano, 2007) and has been extensively studied (see, e.g., Xiao et al., 2013; Luna et al., 2014; Cheng et al., 2019; Guo et al., 2019; Rezende et al., 2019; da Silva et al., 2020). Alba and Chicano (2007) defined the differences between the SPSP and the RCPSP as follows. Firstly, in the SPSP there is a project cost and a cost associated with the workers, which must be minimized (in addition to the project duration). Moreover, while in the RCPSP there are several types of resources, the SPSP has only one (the employee) with several possible skills. Finally, while each activity in the RCPSP requires predefined quantities of each resource, skills in the SPSP are not quantifiable entities. Following Alba and Chicano (2007), these differences make the SPSP more realistic than the RCPSP, since it includes the concept of an employee with a salary and personal skills, also capable of performing several tasks during a regular working day. Note that the SPSP shows more similarities to the MS-RCPSP than to the RCPSP, however, there are also some differences between the first two. For instance, unlike in the MS-RCPSP, resources in the SPSP can perform multiple tasks over time,

---

[14] For a detailed review of the RCPSP literature, see, e.g., Hartmann and Briskorn (2010); Wu et al. (2014) and Fahmy (2016).

and it also takes into account the dedication of employees to activities (see, e.g., Myszkowski et al., 2015b, 2017; Laszczyk and Myszkowski, 2019).

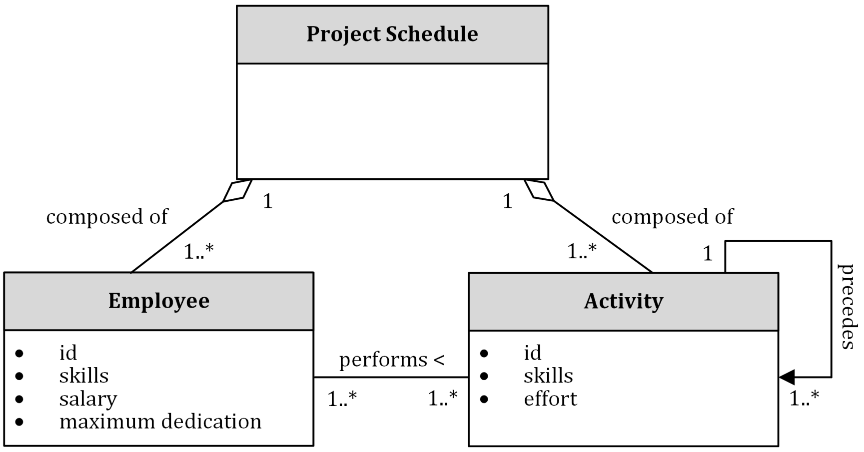
To present the basic concept of the SPSP and to review its literature, this section mainly relies on Vega-Velázquez et al. (2018), a survey dedicated to this problem.[15] Following the authors' logic, first, a reference model is defined, then 37 papers are compared with this model based on their structure, objective and optimization.[16]

2.3.1 Reference Model

Similarly to this study, most of the papers in the literature are based on the model proposed by Alba and Chicano (2007), even though their article did not contain the term SPSP.[17] Vega-Velázquez et al. (2018) illustrate the model of Alba and Chicano (2007) with an UML (Unified Modeling Language) class diagram, representing that a project schedule is composed of a set of activities and employees, and employees are associated with activities (see Fig. 7).[18]

FIGURE 7. UML class diagram of the SPSP

(Source: Vega-Velázquez et al., 2018, p. 11)



---

[15] This is justified not only by the detail of Vega-Velázquez et al. (2018), but also by the fact that the new approach presented in this study can thus be more easily compared with the literature.

[16] Note that the formal description of the SPSP (and its extension) is presented in Chapter 3.

[17] Alba and Chicano (2007) referred to the problem as the PSP.

[18] The UML "is a standardized general-purpose modeling language in the field of software engineering" (Suri and Jajoria, 2013, p. 4).

As Fig. 7 shows, in the SPSP, employees are characterized by their skills, their monthly salary, and their maximum dedication time to a project. Activities are the units of work that make up a project. They are characterized by a set of required skills and an effort which is expressed in person-months. An activity can be assigned to one or more employees, and each employee has a level of dedication associated with that activity. The duration of the activities can be calculated based on the efforts associated with those activities and the dedication of employees. In addition, activities can be interdependent. When these dependencies exist, an activity can only begin when its predecessors have completed. Once the duration of the activity has been determined, its start and end times can be calculated. The duration of the project is the sum of the duration of the longest consecutive activities.[19] The main constraints of the model are the following:

C_1: Each activity must be performed by at least one human resource.

C_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.

C_3: There must not be any human resource who exceeds his or her maximum dedication (allocation) to the project.

2.3.2 Structure

While almost half of the reviewed research papers reuse the model of Alba and Chicano (2007) without any changes, there are some studies that use only the first constraint (C_1) of the benchmark model. Other studies present a slightly modified version of Alba and Chicano's (2007). Table 3 presents the structural similarities and differences between the models of the examined papers and the reference model of Alba and Chicano (2007).

---

[19] This is called a critical path in the project management literature (see, e.g., Devaux, 1999).

TABLE 3. Structural differences between planning models

(Source: Vega-Velázquez et al., 2018, pp. 13-15, own update is denoted by ‡)

Paper	**Task attributes**		**Employee attributes**			**Constraints**	**Comments**
	Effort	**Req. Skills**	**Skills**	**Salary**	**Max. dedic.**		
Alreffaee and Alabajee (2020)‡	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
de Andrade et al. (2019)‡	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
Chicano et al. (2011)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
Chicano et al. (2012)	EQ	NR	NR	EQ	NR	- All of the tasks must be performed by at least one employee who has a productivity > 0 in the corresponding task. (this is the only constraint)	- Employees have a productivity attribute associated to tasks.
Luna et al. (2011)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
Luna et al. (2014)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
Chang et al. (2001)	EQ	EQ	EQ	EQ	DIF*	- A task must not be interrupted during its execution. - Employees must work on a task from its beginning to its end (that is, without interruption).	* It is called overtime limit overloading limit.
Chang et al. (2008)	EQ	EQ	DIF*	DIF**	DIF***	- Tasks must not finish after the established deadline. - There is a limit on the resources that can be assigned to a task.	* Employees skills are rated in a scale between 0 and 5. ** Employees are assigned a monthly base salary plus an additional payment for overwork. *** Maximum dedication corresponds to the amount of hours an employee can work with respect to an established maximum, expressed as a percentage. - Employees are also characterized by their availability in the project. - Tasks have an established deadline.
García-Nájera and del Carmen Gómez-Fuentes (2014)	EQ	DIF*	DIF**	EQ	NR	$\mathbf{C}_1$ is reused. - Additional constraint: Each task must be assigned to at least one employee that possesses the skill level required by the task (Similar to $\mathbf{C}_2$).	* Tasks are characterized by a required skill level. ** Employees possess a skill level: beginner, junior, senior and expert.

Reference model: Alba and Chicano (2007).
- EQ: The attribute is the same (equal) as in the reference model.
- DIF: There are differences with respect to the reference model.
- NR: The attribute was not reported or not considered in the model.

Continued on next page...

Paper	**Task attributes**		**Employee attributes**			**Constraints**	**Comments**
	Effort	**Req. Skills**	**Skills**	**Salary**	**Max. dedic.**		
García-Nájera and del Carmen Gómez-Fuentes (2014)	EQ	DIF*	DIF**	EQ	NR	- $\mathbf{C}_1$ is reused. - Additional constraint: Each task must be assigned to at least one employee that possesses the skill level required by the task (Similar to $\mathbf{C}_2$).	* Tasks are characterized by a required skill level. ** Employees possess a skill level: beginner, junior, senior and expert.
Minku et al. (2012)	EQ	EQ	EQ	EQ	EQ*	- Only $\mathbf{C}_1$.	* Maximum dedication of the employees is constant (equal to 1).
Minku et al. (2013)	EQ	EQ	EQ	EQ	EQ*	- Only $\mathbf{C}_1$.	* Maximum dedication of the employees is constant (equal to 1).
Ngo-The and Ruhe (2008)	EQ	NR	NR	NR	NR	- Additional constraint: All the tasks that are planned for a particular release must be finished before its due date. - Additional constraint: A feature must not require a larger amount of resources than those that are available.	- Employees are characterized by a productivity per type of task.
Duggan et al. (2004)	NR	NR	DIF*	NR	NR	NR	* Employees have a skill level for different types of tasks. Skill levels are the following: novice, average, good, very good and expert.
Xiao et al. (2013)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
Gueorguiev et al. (2009)	NR	DIF*	DIF**	NR	NR	NR	* Each task requires a particular skill to be performed. ** Each employee has a single skill.
Jin and Yao (2014)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$	Same as reference model.
Suri and Jajoria (2013)	NR	EQ	EQ	EQ	NR	NR	
Antoniol et al. (2004)	EQ	NR	NR	NR	NR	NR	
Rodríguez et al. (2011)	NR	NR	DIF*	NR	NR	NR	* Employees are of two types: novice or experienced. - Additional variables are considered including project size and scheduled time.

Continued on next page...

Reference model: Alba and Chicano (2007).
- EQ: The attribute is the same (equal) as in the reference model.
- DIF: There are differences with respect to the reference model.
- NR: The attribute was not reported or not considered in the model.

Paper	Task attributes		Employee attributes			Constraints	Comments
	Effort	Req. Skills	Skills	Salary	Max. dedic.		
Di Penta et al. (2011)	EQ	DIF*	EQ	NR	NR	- Additional constraint: Members of the development teams must be in disjoint sets. - Additional constraint: Members of a development team must possess a common set of skills. - Additional constraint: Each work package must be associated with a team that covers the skill required by the work package.	* Each work package is organized around a single skill.
Gonsalves and Itoh (2010)	DIF*	EQ	DIF**	DIF***	NR	- Additional constraint: No employee must be assigned simultaneously to more than one task at any given moment.	* Tasks are associated with the original processing time (initial). ** Employees skills are rated in a scale between 0 and 5. *** The cost of the employees is a daily cost.
Hanne and Nickel (2005)	NR	NR	DIF*	NR	NR	- Additional constraint: The author of a coding artifact cannot be the inspector or tester of said artifact. - Additional constraint: A task with a higher priority must be scheduled before a task with a lower priority, if possible.	* The skills of the employees are of coding inspection and testing and are rated in the interval [0-1]. - Tasks are characterized by size, complexity and domain.
Dupuy et al. (2013)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$	Same as reference model.
Shen et al. (2015)	DIF*	EQ	EQ	EQ	DIF**	$\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$ - Additional soft constraint: There is a maximum number of employees assigned to each task.	* The initial effort of a task can change. ** The availability of an employee can change during the project. - New tasks can appear during the project.
Wu et al. (2016)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$	Same as reference model.
Xiao et al. (2015)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$	Same as reference model.
Crawford et al. (2014)	EQ	EQ	EQ	EQ	EQ	$\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$	Same as reference model.

Continued on next page...

Reference model: Alba and Chicano (2007).
- EQ: The attribute is the same (equal) as in the reference model.
- DIF: There are differences with respect to the reference model.
- NR: The attribute was not reported or not considered in the model.

Paper	Task attributes		Employee attributes			Constraints	Comments
	Effort	Req. Skills	Skills	Salary	Max. dedic.		
Wena and Lin (2008)	NR	NR	NR	EQ	NR	- Additional constraint: The duration of a project cannot exceed an established maximum duration. - Additional constraint: The total cost of the project cannot exceed the established maximum cost.	- Additional input variables are considered, including the maximum duration and maximum cost for the project.
Crawford et al. (2016a)	EQ	EQ	EQ	EQ	EQ	C_1, C_2, C_3	Same as reference model.
Crawford et al. (2016b)	EQ	EQ	EQ	EQ	EQ	C_1, C_2, C_3	Same as reference model.
Shen et al. (2018)	DIF*	EQ	EQ	EQ	DIF**	C_1, C_2, C_3 - Additional soft constraint: There is a maximum number of employees assigned to each task.	* The initial effort of a task can change. ** The availability of an employee can change during the project.

Reference model: Alba and Chicano (2007).
- EQ: The attribute is the same (equal) as in the reference model.
- DIF: There are differences with respect to the reference model.
- NR: The attribute was not reported or not considered in the model.

2.3.3 Objective

The SPSP is considered as an NP-hard problem (see, e.g., Xiao et al., 2013), and in general, its two major goals are reducing both the cost and duration of the project; however, these goals are in conflict with each other (see, e.g., Alba and Chicano, 2007; Myszkowski et al., 2019). Similarly to other problems with multiple objectives, the general SPSP has no single solution. Instead, its solutions form a so-called Pareto-optimal set (Deb, 2001), where every point is optimal in the sense that neither the duration nor the cost objectives can be improved without worsening the other objective (see, e.g., Gonsalves and Itoh, 2010; Di Penta et al., 2011; García-Nájera and del Carmen Gómez-Fuentes, 2014; Luna et al., 2014). Alternatively, the SPSP can be treated as a single objective problem by using a composite objective function (see, e.g., Alba and Chicano, 2007; Suri and Jajoria, 2013; Xiao et al., 2013; Jin and Yao, 2014), and in some cases, other objectives may appear, such as minimizing overload (Chang et al., 2001; García-Nájera and del Carmen Gómez-Fuentes, 2014) or maximizing quality (Duggan et al., 2004; Hanne and Nickel, 2005). The optimization objectives of the reviewed papers are compared in Table 4.

TABLE 4. Differences in optimization objectives

(Source: Vega-Velázquez et al., 2018, pp. 20-21, own update is denoted by ‡)

Paper	**Objectives**						**Type of optimization**
	Duration	**Cost**	**Overload**	**Quality**	**Fragment.**	**Other**	
Alba and Chicano (2007)	X	X					S
Alreffaee and Alabajee (2020)‡	X	X					S
de Andrade et al. (2019)‡	X	X					M
Chicano et al. (2011)	X	X					M
Chicano et al. (2012)	X	X					M
Luna et al. (2011)	X	X					M
Luna et al. (2014)	X	X					M
Chang et al. (2001)	X	X	X				S
Chang et al. (2008)	X	X					S
García-Nájera and del Carmen Gómez-Fuentes (2014)	X	X	X				M
Minku et al. (2012)	X	X					S
Minku et al. (2013)	X				X		S
Ngo-The and Ruhe (2008)						X	S
Duggan et al. (2004)	X			X			M
Xiao et al. (2013)	X	X					S
Gueorguiev et al. (2009)	X					X	M
Jin and Yao (2014)	X	X					S
Suri and Jajoria (2013)	X	X					S
Di Penta et al. (2011)	X				X		M
Antoniol et al. (2004)	X						S
Rodríguez et al. (2011)	X	X				X	M
Gonsalves and Itoh (2010)	X	X					M
Dupuy et al. (2013)	X	X					S
Hanne and Nickel (2005)	X	X		X			M
Wena and Lin (2008)	X	X					M
Crawford et al. (2014)	X	X					S
Crawford et al. (2016a)	X	X					S
Crawford et al. (2016b)	X	X					S
Wu et al. (2016)	X	X					M
Xiao et al. (2015)	X	X					M
Shen et al. (2015)	X	X				X	M
Shen et al. (2018)	X	X				X	M

- S: Single objective problem.
- M: Multi-objective problem.

2.3.4 Optimization

To solve the SPSP, Coello et al. (2006) and Myszkowski et al. (2019) propose several meta-heuristics, while Chicano et al. (2011) and Luna et al. (2014) compare accuracy and scalability of these algorithms. Chicano et al. (2011) and Luna et al. (2014) observe that the algorithm called Pareto archived evolution strategy (PAES) (Knowles and Corne, 2000) has the best scalability and obtains the best approximate Pareto sets, while the most widely used non-dominated sorting genetic algorithm II (NSGA-II) (Deb et al., 2002) and strength Pareto evolutionary algorithm 2 (SPEA2) (Zitzler et al., 2001) are examples of the least accurate solvers in general. Nevertheless, PAES is outperformed by NSGA-II, SPEA2 and several recent algorithms, such as the multi-objective cellular genetic algorithm (MOCell) (Nebro et al., 2007) in high-cost short-duration project scheduling (Luna et al., 2014). Table 5 presents a comparison of the reviewed papers according to the optimization algorithm they used to solve the SPSP.

TABLE 5. Optimization algorithms

(Source: Vega-Velázquez et al., 2018, pp. 26-27, own update is denoted by ‡)

Paper	**Evolutionary algorithms**					**Swarm intelligence**		**Other algorithms**
	GA	**NSGA-II**	**SPEA-II**	**PAES**	**Other MOEAs**	**PSO**	**ACO**	
Alba and Chicano (2007)	X							
Alreffaee and Alabajee (2020)‡								WOA
de Andrade et al. (2019)‡		X		X	GE-HH			
Chicano et al. (2011)		X	X	X	NCRO MOCell GDE3			
Chicano et al. (2012)		X	X	X	MOCell			
Luna et al. (2011)		X		X	DEPT MO-FA			

Continued on next page...

- GA: Genetic Algorithm
- NSGA-II: Non-dominated Sorting Genetic Algorithm II
- SPEA2: Strength Pareto Evolutionary Algorithm 2
- PAES:Pareto Archived Evolution Strategy
- EHH: Evolutionary Hyper-heuristic
- MA: Memetic Algorithm
- PSO: Particle Swarm Optimization
- GE-HH: Grammatical Evolution Hyper-Heuristic
- NCRO: Non-dominated Chemical Reaction Optimization
- HC: Hill Climbing
- RS: Random Search
- SA: Simulated Annealing
- ACO: Ant Colony Optimization
- PBIL: Population-based incremental learning
- IWD: Intelligent Water Drops
- FA: Firefly Algorithm
- WOA: Whale Optimization Algorithm

Paper	**Evolutionary algorithms**					**Swarm intelligence**		**Other**
	GA	**NSGA-II**	**SPEA-II**	**PAES**	**Other MOEAs**	**PSO**	**ACO**	**algorithms**
Luna et al. (2014)		X	X	X	DEPT MOCell MOABC MO-FA GDE3			
Chang et al. (2001)	X							
Chang et al. (2008)	X							HC
García-Nájera and del Carmen Gómez-Fuentes (2014)	X	X						
Minku et al. (2012)	X							(1+1) EA
Minku et al. (2013)	X							(1+1) EA and RS
Ngo-The and Ruhe (2008)	X							
Duggan et al. (2004)		X (NSGA)						
Xiao et al. (2013)	X						X	
Gueorguiev et al. (2009)			X					RS
Jin and Yao (2014)	X							PBIL
Suri and Jajoria (2013)							X	
Di Penta et al. (2011)	X	X						HC and SA
Antoniol et al. (2004)	X							HC and SA
Rodríguez et al. (2011)		X						
Gonsalves and Itoh (2010)						X		
Dupuy et al. (2013)	X							
Hanne and Nickel (2005)					Custom MOEA			
Wena and Lin (2008)	X	X						
Crawford et al. (2014)							X	
Crawford et al. (2016a)								IWD
Crawford et al. (2016b)								FA
Wu et al. (2016)								EHH
Xiao et al. (2015)							X	
Shen et al. (2015)					Based on ϵ-MOEA			
Shen et al. (2018)								MA

- GA: Genetic Algorithm
- NSGA-II: Non-dominated Sorting Genetic Algorithm II
- SPEA2: Strength Pareto Evolutionary Algorithm 2
- PAES:Pareto Archived Evolution Strategy
- EHH: Evolutionary Hyper-heuristic
- MA: Memetic Algorithm
- PSO: Particle Swarm Optimization
- GE-HH: Grammatical Evolution Hyper-Heuristic
- NCRO: Non-dominated Chemical Reaction Optimization
- HC: Hill Climbing
- RS: Random Search
- SA: Simulated Annealing
- ACO: Ant Colony Optimization
- PBIL: Population-based incremental learning
- IWD: Intelligent Water Drops
- FA: Firefly Algorithm
- WOA: Whale Optimization Algorithm

2.4 Project Team Composition and Effectiveness

The previous sections focused on the formalized problems of project scheduling and staff allocation. To understand the human aspects of these problems, we step out of these formalized frameworks, and briefly review the implications of project team effectiveness from the perspectives of industrial psychology. While traditional job analysis methods identify the knowledge, skills, and abilities (KSAs) needed for individual job performance, they tend to be insensitive to the social context in which work occurs (Muchinsky, 2006). As many studies in the field of team composition emphasize, choosing team members on the basis of individual-task KSAs alone is not enough to ensure team effectiveness (see, e.g, Klimoski and Jones 1995; Salas and Burke 2002; Muchinsky 2006; Chen et al. 2014). Note that, similar to traditional job analysis methods, the SPSP follows this oversimplified selection mechanism, which may result in low-performing project teams. To overview potential directions of improvement, the two basic approaches of assembling high-functioning project teams are discussed in the following. According to Chiocchio et al. (2015), these are the individual characteristics approach and the jigsaw puzzle approach.

2.4.1 Individual Characteristics Approach

This approach is based on the general assumption that when we predict the performance of a team, some personal characteristics matter much more than others (Chiocchio et al., 2015). Determining what these relevant characteristics are – and how the distribution of each, within a team, relates to the team's performance – becomes the aim of the team composition research within this approach (Chiocchio et al., 2015). Based on the literature, these characteristics are mainly related to individual KSAs and personality types.

2.4.1.1 Knowledge, Skills, and Abilities

Since a team, by definition, is a social entity that interact in a larger social context Muchinsky (2006), team composition based on individual KSAs requires a team-level approach (Klimoski and Jones, 1995). According to Salas and Burke (2002), successful team members need two general types of skills: taskwork skills that are needed to perform the actual task, and different behavioral, cognitive, and attitudinal skills – so-called teamwork skills – to coordinate their actions and work independently. As they argue, "although taskwork skills are the foundation for the operational side of performance, teamwork skills are the foundation for the necessary synchronization, integration, and social interaction that must occur between members for the team to complete the assigned goal" (Salas and Burke, 2002, p. 240). In line with Konak et al. (2019), effective teamwork requires KSAs in a set of diverse areas including leadership, communication, group decision-making, negotiation skills, team motivation, conflict resolution, social skills, understanding of diversity, responsibility, and accountability. In addition, Stevens and Campion (1999) developed a paper-and-pencil selection test for staffing work teams. The KSAs, measured by the test, reflected conflict resolution, collaborative problem solving, communication, goal setting and performance management, and planning and task coordination.[20] According to their findings, the consideration of individual level KSAs can have both conceptual and practical value in the staffing of work teams.

2.4.1.2 Personality

As another key factor of the individual characteristics approach, researchers have investigated the personalities of team members (see, e.g., Hogan et al., 1988; Smith-Jentsch et al., 1996; Barry and Stewart, 1997; Yilmaz et al., 2017). Two most widely applied and studied models are the Big Five (see, e.g., Goldberg 2013) and the Five-Factor Model (Costa Jr and McCrae, 2008). Although these models were

---

[20] The conceptual framework of teamwork KSAs introduced by Campion et al. (1994).

developed by applying two different methodologies, they converged on the same five factors of personality: agreeableness (warm, polite, trusting), conscientiousness (achievement-driven, diligent, organized), extraversion (sociable, gregarious, active), emotional stability (low anxiety, anger, and self-consciousness), and openness to experience (intellectual, artistic). According to Neuman and Wright (1999), the validity of Big Five personality factors can be extended to the prediction of team performance. Based on their findings, conscientiousness and agreeableness predicted various dimensions of team performance, while agreeableness predicted ratings of the interpersonal skills of team members. Bell (2007) supports the relevance of team personality composition operationalized as the group means, since it reported that for each of the Big Five personality traits, group means were positively correlated with team performance. Yilmaz et al. (2017) investigates the personality traits of 216 employees from a middle-sized software company to explore effective software team structures. They experienced that not all traits were equally present in the company. While individuals with traditional tasks matched with the traditional characteristics, agile teams are found to be more extroverted. Based on Schmitt et al. (1984); Hunter et al. (1990), considering personalities is useful in general, but purely personality-based allocation strategies provide weaker predictions on the effectiveness of teamwork than strategies based on individual KSAs.

2.4.2 Jigsaw Puzzle Approach

The jigsaw puzzle approach (see, e.g., Allen and West 2005; Chiocchio et al. 2015 – or configuration perspective (see, e.g., Schneider and Smith 2004) – provides an even more complex view of teamwork effectiveness than the individual approach. According to Chiocchio et al. (2015), here the question is not whether a team's means or variability on a single variable influences performance, but whether the members of a team "fit together" – or complement each other – based on the particular combination of several variables associated with each member. "Just as a given puzzle piece will not fit into all existing sets of interlocking pieces, a given person will fit successfully into teams with some people configurations, but not others" (Chiocchio et al., 2015, p. 303). Within this approach, team roles and personal compatibility are the most extensively studied topics.

2.4.2.1 Team Roles

Based on (Sarbin, 1954, p. 223), role theory "is an interdisciplinary theory in that its variables are drawn from studies of culture, society, and personality". Roles are important in teams because they represent patterns of behavior that relate to the activities of other team members in pursuit of the overall team goal (Driskell et al., 2017). The most prominent among the team role theories is Belbin's (1993; 2010; 2014) team role model. This model states that in teamwork, individuals tend to play different type of roles beyond the usual functional roles associated with their technical activities (Fernandes, 2007; Branco et al., 2015). In his early publication (Belbin, 1981), Belbin defined eight roles that are necessary for a team to be successful: plant, monitor evaluator, shaper, completer finisher, company worker, chairman, resource investigator, teamworker. Later he added a ninth role, the specialist, and renamed the chairman to coordinator and the company worker to implementer (Belbin, 1993, 2010). Furthermore, the nine roles may also be categorized as thinking, action, and social roles (Belbin, 2014). Based on his findings, teams work best when there is diversity in team members, and individual roles mixed in a team can make a team more effective. The nine team roles are detailed in Table 6.

TABLE 6. Descriptions of Belbin team roles

(Sources: Belbin, 2010, p. 22 and Belbin, 2014, p. 12)

Team role (notation)	Category	Contribution	Allowable weakness
Plant (PL)	Thinking roles	Creative, imaginative, free-thinking. Generates ideas and solves difficult problems.	Ignores incidentals. Too preoccupied to communicate effectively.
Specialist (SP)		Single-minded, self-starting, dedicated. Provides knowledge and skills in rare supply.	Contributes only on a narrow front. Dwells on technicalities.
Monitor evaluator (ME)		Sober, strategic and discerning. Sees all options and judges accurately.	Lacks drive and ability to inspire others. Can be overly critical.
Shaper (SH)	Action roles	Challenging, dynamic, thrives on pressure. Has the drive and courage to overcome obstacles.	Prone to provocation. Offends peoples feelings.
Completer finisher (CF)		Painstaking, conscientious, anxious. Searches out errors. Polishes and perfects.	Inclined to worry unduly. Reluctant to delegate.
Implementer (IMP)		Practical, reliable, efficient. Turns ideas into actions and organises work that needs to be done.	Somewhat inflexible. Slow to respond to new possibilities.
Coordinator (CO)	Social roles	Mature, confident, identifies talent. Clarifies goals. Delegates effectively.	Can be seen as manipulative. Offloads own share of the work.
Resource investigator (RI)		Outgoing, enthusiastic, communicative. Explores opportunities and develops contacts.	Over-optimistic. Loses interest once initial enthusiasm has passed.
Teamworker (TW)		Co-operative, perceptive and diplomatic. Listens and averts friction.	Indecisive in crunch situations. Avoids confrontation.

Note that: strength of contribution in any one of the roles is commonly associated with particular weaknesses. These are called allowable weaknesses. Executives are seldom strong in all nine Team Roles.

Belbin's theory has been applied in several software-related investigations to evaluate the impact of the different team roles on project teams (see, e.g., Henry and Stevens 1999; Thomas 1999; Schoenhoff 2001; Stevens and Henry 2002; Simeunovic and Landelius 2017; Karabeleski and Avdic 2018). From these, Thomas (1999) found evidence that Belbin's roles provide useful information to form software teams, and also emphasized that teams that contain one – and only one – leader perform better than teams with no leader or multiple leaders. Similarly to Simeunovic and Landelius (2017), Karabeleski and Avdic (2018) experienced that not all roles are needed for a successful project. They also found that the most effective teams contained different roles that still had some personality traits in common. In line with these results, to support cooperation within project teams, Kurbucz (2013) presents a formalized method for team composition, which – in addition to individual KSAs – also takes the roles of team members into account.

2.4.2.2 Personal Compatibility

Other jigsaw puzzle theories emphasize the personal compatibility among team members. The main idea behind this approach is that team members' personalities may need to be complementary – or "fit together" – in order for the team to achieve its potential (Chiocchio et al., 2015). One of the most widely adopted theories in this field is the Fundamental Interpersonal Relationship Orientation (FIRO) theory proposed by Schutz (1955, 1959, 1992). According to Schutz (1955, 1959, 1992), people's intention to interact with others can be measured by three phases – inclusion, control, and affection. In line with (Anop and Aldaghi, 2010, p. 3), these phases describe and explain a team's development as follows:

1 Inclusion: "The team is formed and the members get to know each other during the Inclusion phase. The project manager maps out the objectives of the project, which have to be achieved by the team."

2 Control: "The project manager identifies each team members' values during this phase. This phase satisfies the members' social need or individual attention and their sense of contributing to the overall project."

3 Openness: "By this phase the team is working well and strong group identity has developed. An atmosphere of openness has developed within the team. In the last phase the group becomes highly efficient. All teams are not progress to the Openness phase. When changes in personnel occur in the team, it is common for it to take one or more steps backwards in its development."

To be employed empirically, FIRO was operationalized as FIRO-B (FIRO behavior) (Schutz, 1959, 1992). As reported by Furnham (1990, 1996), FIRO-B was one of the three most widely used surveys in occupational psychology at the date of publication of his studies. Nevertheless, as Chiocchio et al. (2015) argue, the most relevant research has not found robust correlations involving compatibility on the needs measured by the FIRO-B and team performance (Moos and Speisman, 1962; Shaw and Webb, 1982). Moreover, there is limited literature of FIRO-B on the composition of software development teams.

2.5 Sociometry and Synergy Networks

Although it is simpler to predict a team's outcome based on the aggregate skills of its members, as presented in the previous section, interactions among employees may have a great effect on team performance (Hsu et al., 2016). The concept that captures this phenomenon is called synergy (see, e.g., Tannenbaum et al., 1992; Mears and Voehl, 1994; Hoopes and Postrel, 1999; Hong et al., 2004; Ruiz and Fuentes, 2017) or synergy effect (see, e.g., Raluca, 2012; Scholtes et al., 2016; Ren et al., 2018) between team members. According to (Scholtes et al., 2016, p. 2), "the collaboration of developers in a team can give rise to synergy effects, which result in the team being more productive than one would expect from simply adding up the individual productivities of its members." In addition, its opposite form – the so-called negative synergy (effect) – can also emerge within a team, and leads to weaker than expected productivity (see, e.g, Carbonell and Rodríguez Escudero, 2019; Ruiz and Fuentes, 2017). The aim of this section is to present a network model that can quantify the positive and negative synergies of employees. Since industrial sociology deals with the analysis of models similar to the synergy network, and since industrial sociologists, similarly to industrial psychologists (see Section 2.4) have put a significant effort into investigating the reasons behind teamwork effectiveness, we will first briefly review their results.

Industrial sociology examines the effectiveness of teamwork (synergy of team members) in the light of formal and informal relationships between team members. When software developers know each other well and understand each other's needs, their collective sense-making enables them to perform their individual activities in ways that take into account the activity needs of other team members (Crowston and Kammerer, 1998; Espinosa et al., 2007). In line with this, Espinosa et al. (2007) found that the benefit of team familiarity on the software development team's performance is enhanced when team coordination is more challenging, i.e., when teams are larger or geographically dispersed. To investigate the social structure or, more generally, interdependence among group members, researchers use sociometry (Moreno, 1960; Sorenson, 1971).

Although we have limited information on how the structural properties of a sociometric network – the so-called sociogram (see, e.g., Zorrilla and de Lima Silva, 2019) – affect collective performance, several publications have focused on this issue (see, e.g., Sparrowe et al., 2001; Ahuja et al., 2003; Cummings and Cross, 2003). Based on Ahuja et al. (2003), centrality indicators of the social network are stronger direct predictors of performance than the individual characteristics, e.g., functional role, status or communication role.[21] Sparrowe et al. (2001) observed that groups with decentralized structures performed better at complex tasks than groups with centralized structures, and as stated in Cummings and Cross (2003), more hierarchical structure and greater core-periphery discrepancies were negatively related to performance.

In contrast with these results, Shaw (1964) demonstrated that groups with decentralized communication nets took less time to finish complex tasks than groups with centralized communication nets. Sanchez et al. (2017) analyzed 899 IT projects of a leading bank, and they found that the success of project management was positively impacted by not only the size, duration and the postponement of the project, but also the formal power of the project manager. Furthermore, they observed that smaller and less dispersed teams have better results than larger and sparse teams in addressing a multiplicity of projects. Yang and Tang (2004) and Wu et al. (2008) found positive relationship between the cohesion and centrality of software development teams and their overall effectiveness. Furthermore, Yang and Tang (2004) emphasized that conflict indexes were not significantly correlated with the final performance of the teams examined.[22]

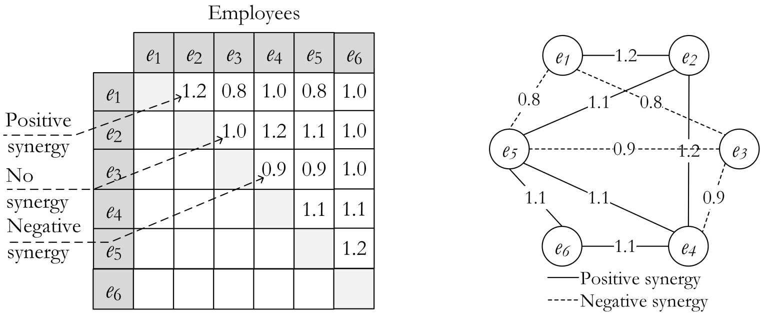
---

[21] Typically, four measures of centrality are used in the literature: degree, betweenness, closeness and eigenvector centrality (Mote, 2005).

[22] Sparrowe et al.'s (2001) and Yang and Tang's (2004) calculations are based on in- and outdegree centrality. Sanchez et al. (2017) used closeness and eigenvector, while Wu et al. (2008) applied betweenness centrality (see, e.g., Sridharan and Balakrishnan, 2019).

Over the past decade, multiple network models of synergies between team members have emerged (see, e.g., Liemhetcharat and Veloso, 2014; Melo and Sardinha, 2016; Dzvonyar et al., 2018). In these networks, nodes indicate the employees and the weighted edges between them typically represent the estimated effectiveness of their joint work. For the sake of simplicity, these networks typically represent employees' pairwise synergies (see, e.g., Liemhetcharat and Veloso, 2012, 2014; Brown, 2020) instead of groupwise ones, which makes team composition easier. Note that although these models are similar in structure to sociograms, synergies are difficult to measure and can be primarily used to support team composition rather than to perform an empirical analysis. In line with Källo et al. (2013), the adjacency matrix of the synergy network can be visualized by a DSM method. Fig. 8 presents both the network and the DSM representation of the same pairwise synergy structure.

FIGURE 8. DSM and network representation of pairwise synergies *(Source: own figure)*



2.6 Research Assumptions

Although most psychological and sociological approaches emphasize the complexity of the project team, none of the applied models can handle the interdependence of employees. Moreover, while the MS-RCPSP has already been extended to support flexible project planning, the SPSP cannot handle logical planning uncertainties (Cram and Marabelli, 2018). Since this flexibility and employee interdependencies are particular characteristics of IT projects (Rodriguez-Repiso et al., 2007; Iriarte and Bayona, 2020) as well as their APM practices (Fowler et al., 2001; Jalote and Vishal, 2003), this dissertation is focused on the study and elimination of these shortcomings. According to the literature related to the effectiveness of project teams, formal and informal relationships between employees can be a source of positive or negative synergies that significantly affect the performance of the project team (Ahuja et al., 2003) or, consequently, the outcome of the project (Sanchez et al., 2017). The structure of these relationships are often studied by using sociometric networks, however, the results in this area are contradictory. While Ahuja et al. (2003) and Cummings and Cross (2003) emphasize the beneficial impact of decentralized, less hierarchical structures on performance, Sanchez et al. (2017) found a positive connection between the formal power of the project manager, as well as the smaller, less dispersed teams, and the success of IT projects. Although employee interdependencies have a significant impact on project outcomes – especially for (software) projects managed by an APM approach –, no planning method has yet been developed to study or apply the phenomenon in practice.

Based on a review of the literature, one research assumption is formulated for each of the four research questions (see **RQs** in Chapter 1.2). The four research assumptions (**RAs**) of the dissertation are as follows:

RA$_1$: The classical software project scheduling problem can be extended by considering flexible task dependencies and synergies between resources.

RA$_2$: The multi-domain matrix (MDM) can be specified to a flexible multi-domain matrix whose interconnected domains model the flexible project plan, the skills of human resources as well as the synergies between them.

RA$_3$: Using metaheuristic algorithms, it is possible to find a feasible solution to the project scheduling problem that takes into account flexible task dependencies and synergies between resources.

RA$_4$: By supplementing existing or generated project databases with flexible task dependencies and resource synergies, it is possible to create a simulation environment to examine the impact of human resource synergies and skills, as well as project size, flexibility, and constraints, on project feasibility.

CHAPTER 3

Methods

This chapter first gives a detailed introduction of classical and synergy-based SPSPs. It then presents a hybrid genetic algorithm as well as a simulation framework that can be used to solve and study these problems.

3.1 Formal Description of the (S)SPSP

This section contains a formal description of the SPSP as well as that of its extension, i.e., the SSPSP. Unlike other reported studies of this topic, for clarity and flexible planning, I use a matrix-based method to define the problem. The proposed matrix-based method is a specification of the MDM method (see Danilovic and Browning, 2007). As it was discussed in Section 2.2.3, the original MDM version allows several domains that can interact with one another; however, the original MDM only handles fixed dependencies and task occurrences (see, e.g., Danilovic and Browning, 2007; Browning, 2014). Contrary to the original method, the proposed synergy-based multi-domain matrix (SMM) considers flexible dependencies and supplementary task completions in order to support the synergy-based software project scheduling problem (SSPSP). The SSPSP is based on a combination of the agile approach and sociometric – or more precisely, synergy – graphs. To formulate the problem, I extend the notation of Alba and Chicano (2007) and Luna et al. (2014). Since it is solved via the proposed multi-domain matrix-based method, the necessary domains (submatrices) are also specified.

3.1.1 Notation

First, the mathematical definitions, necessary for stating the problem and the solution algorithm, are determined. Here, I follow the formulation proposed by Alba and Chicano (2007) and Luna et al. (2014), but unlike these models, I also consider the levels of skills and synergy between employees.

Briefly: We are given a set of employees with ± synergies among them and possessing certain (individual) levels of some skills, in order to solve certain tasks that require certain levels of these skills. We must decide which tasks should be done (possibly not all of them) and their order, and we must distribute (allocate) the employees (possibly in part time) to solve the chosen tasks, fulfilling several other requirements and achieving some optimums (see Eqs. (32) - (35) for details). The set of all of these decisions made by the algorithm is called a project scenario.

All of the data are stored in a large matrix called SMM, containing several blocks that are called domains, as shown in Fig. 9.

In detail:

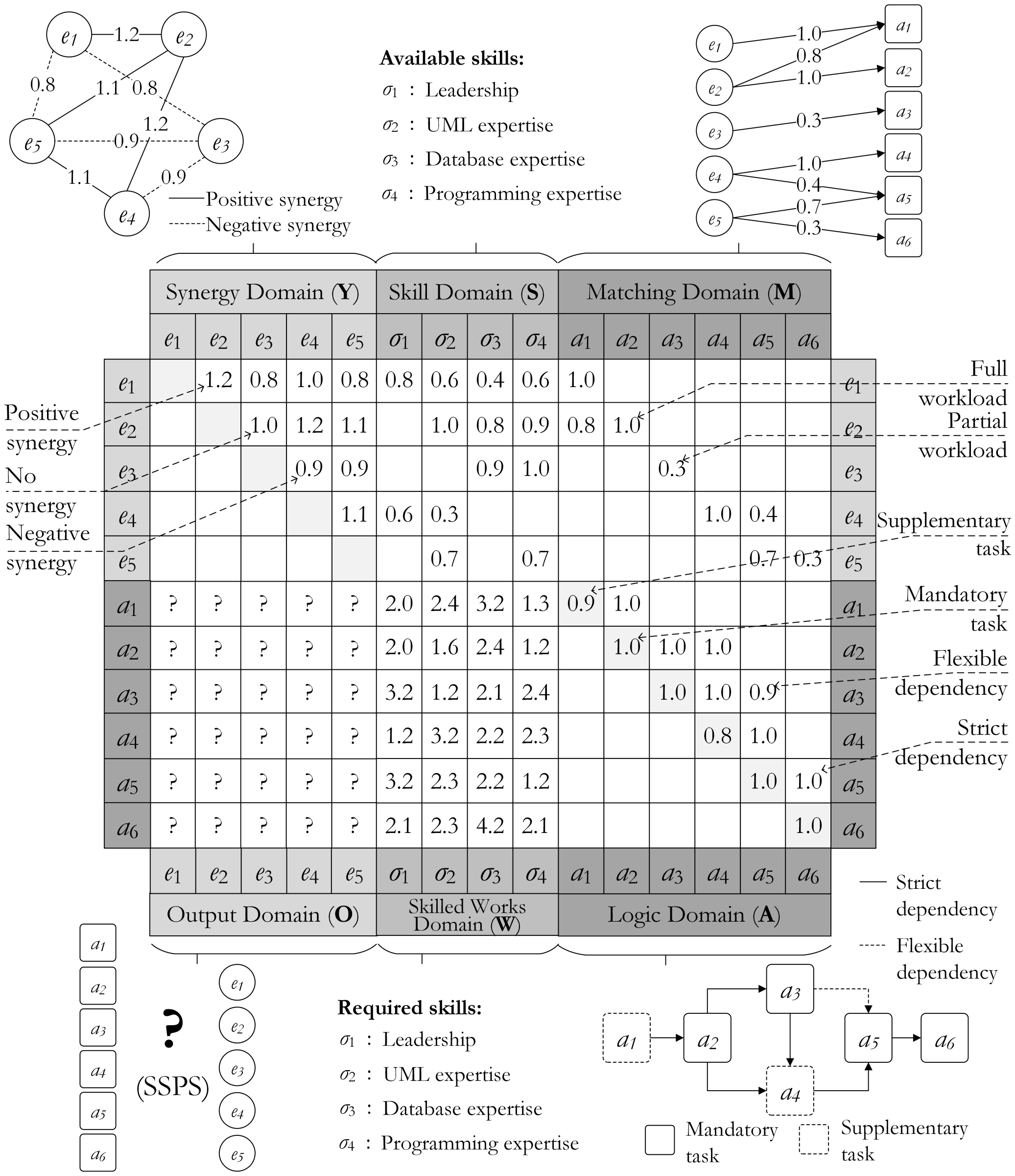
- $E = \{e_1, \ldots, e_m\}$ is the set of employees ($m \in \mathbb{N}^+$).
- $\mathbf{Y}$ is called the synergy domain in the proposed SMM. It is a symmetric m by m matrix of nonnegative real numbers ($\mathbf{Y} \in (\mathbb{R}^+)^{m \times m}$), denoting the synergies among the employees as (for $i, j = 1, 2, ..., m$):
  - $[\mathbf{Y}]_{i,j} > 1$ represents positive,
  - $[\mathbf{Y}]_{i,j} = 1$ represents neutral,
  - $0 < [\mathbf{Y}]_{i,j} < 1$ represents negative synergy between employees e_i and e_j, and $[\mathbf{Y}]_{i,i} = 1$ and $[\mathbf{Y}]_{i,j} = [\mathbf{Y}]_{j,i}$ are assumed.[23]

---

[23] Observe that both the positive and negative synergies are represented by positive real numbers, where $\mathbf{Y}$: $0 < [\mathbf{Y}]_{i,j} < 1$ stand for negative and $1 < [\mathbf{Y}]_{i,j}$ for positive synergies. By default, $[\mathbf{Y}]_{i,j} = 1$, which is assumed in Alba and Chicano (2007) and Luna et al. (2014).

FIGURE 9. Synergy-based multi-domain matrix (SMM)

(Source: own figure)



Available skills:

σ_1 : Leadership

σ_2 : UML expertise

σ_3 : Database expertise

σ_4 : Programming expertise

	Synergy Domain (Y)					Skill Domain (S)				Matching Domain (M)						
	e_1	e_2	e_3	e_4	e_5	σ_1	σ_2	σ_3	σ_4	a_1	a_2	a_3	a_4	a_5	a_6	
e_1		1.2	0.8	1.0	0.8	0.8	0.6	0.4	0.6	1.0						e_1
e_2			1.0	1.2	1.1		1.0	0.8	0.9	0.8	1.0					e_2
e_3				0.9	0.9			0.9	1.0			0.3				e_3
e_4					1.1	0.6	0.3						1.0	0.4		e_4
e_5							0.7		0.7					0.7	0.3	e_5
a_1	?	?	?	?	?	2.0	2.4	3.2	1.3	0.9	1.0					a_1
a_2	?	?	?	?	?	2.0	1.6	2.4	1.2		1.0	1.0	1.0			a_2
a_3	?	?	?	?	?	3.2	1.2	2.1	2.4			1.0	1.0	0.9		a_3
a_4	?	?	?	?	?	1.2	3.2	2.2	2.3				0.8	1.0		a_4
a_5	?	?	?	?	?	3.2	2.3	2.2	1.2					1.0	1.0	a_5
a_6	?	?	?	?	?	2.1	2.3	4.2	2.1						1.0	a_6
	e_1	e_2	e_3	e_4	e_5	σ_1	σ_2	σ_3	σ_4	a_1	a_2	a_3	a_4	a_5	a_6	
	Output Domain (O)					Skilled Works Domain (W)				Logic Domain (A)						

Required skills:

σ_1 : Leadership

σ_2 : UML expertise

σ_3 : Database expertise

σ_4 : Programming expertise

- For any subset $\varepsilon \subseteq E$, we let:

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

  the (geometric) mean of synergies among the employees in ε.

- $S = \{\sigma_1, \ldots, \sigma_s\}$ is the set of skills ($s \in \mathbb{N}$).

- Each employee may have a set of skills, i.e., person e_i has skills:

$$S(e_i) := \left\{\sigma_1^{(i)}, \ldots, \sigma_{\rho_i}^{(i)}\right\} \subseteq S. \tag{2}$$

- The proposed model also handles the levels of skills: $\ell(e_i, \sigma_k) \geq 0$ is the level of e_i in σ_k ($1 \leq i \leq m$, $1 \leq k \leq s$); clearly, $\sigma_k \in S(e_i) \iff 0 < \ell(e_i, \sigma_k)$.[24] These levels can be added, e.g., e_{i_1} and e_{i_2} working together achieve σ_k:

$$[\mathbf{Y}]_{i_1,i_2} \cdot (\ell(e_{i_1}, \sigma_k) + \ell(e_{i_2}, \sigma_k)). \tag{3}$$

  For a larger set $\varepsilon \subseteq E$, we can only use the approximate formula:[25]

$$\ell(\varepsilon, \sigma_k) := \overline{Y}_{\varepsilon} \cdot \sum_{i \in \varepsilon} \ell(e_i, \sigma_k). \tag{4}$$

  (Note that this formula will be modified by the matrix **O** later.)

---

[24] Note that the set of skills (S) are defined in light of the activities associated with them. For instance, if an employee (e_i) has a given level of Python programming skills ($\ell(e_i, \sigma_k)$) that is insufficient to participate in the given task (a_i), where intermediate skill is required, then $\ell(e_i, \sigma_k) = 0$ and the label of the skill should reflect the required level of skill, such as intermediate Python programming.

[25] We may think $\ell(e_i, \sigma_k) = 0$ or $\ell(e_i, \sigma_k) = 1$ in Alba and Chicano (2007) and Luna et al. (2014), without a summing possibility.

- S is the m by s matrix $[\mathrm{S}]_{i,k} := \ell(e_i, \sigma_k)$ is called the skill domain in the SMM matrix.

- $A = \{a_1, \ldots, a_n\}$ is the set of tasks (or activities) to be performed ($n \in \mathbb{N}$). $A^c \subseteq A$ is the subset of mandatory (or compulsory) and $A^- := A \setminus A^c$ is the set of supplementary tasks. Supplementary tasks can be removed from the project or postponed to a later project if they cannot be implemented due to constraints.

- The algorithm will choose which supplementary tasks will be carried out, but it must perform each compulsory task. The final set of tasks to be carried out is denoted by $A^{c(O)}$; clearly, $A^c \subseteq A^{c(O)} \subseteq A$ must hold.

- Among all of the tasks, we have dependencies $\prec, \sim, \bowtie$ with the following meanings. For any $i, j \leq n$, $i \neq j$:
  - $a_i \prec a_j$ means a strict (or required) dependency: a_j must not be started unless a_i has been completed,
  - $a_i \sim a_j$ means no dependency: the starting time of a_j is not affected by a_i,
  - $a_i \bowtie a_j$ means an uncertain (or flexible) dependency: the algorithm must turn each $a_i \bowtie a_j$ into either (i) $a_i \prec a_j$ or $a_j \prec a_i$ or (ii) $a_i \sim a_j$. In case (i), we say that the dependency $a_i \bowtie a_j$ is included in the project, in case (ii) it is excluded.

- Clearly, $\prec$ is a partial order that excludes cycles such as $a_1 \prec a_2 \prec \ldots \prec a_1$, while $\bowtie$ and $\sim$ are symmetric relations.[26]

- $\mathbf{A}$ is called the logic domain in the SMM.[27] It is the n by n matrix storing the above information as:[28]
  - $[\mathbf{A}]_{i,i} = 1 \iff a_i$ is mandatory,

---

[26] By a standard topological ordering algorithm, we may assume that $a_i \prec a_j \implies i < j$.

[27] Note that PEM (Kosztyán et al., 2010) and PDM (Kosztyán, 2015; Kosztyán et al., 2020) methods contain a similar domain (see Section 2.2.3).

[28] $i < j$ and $\mathbf{A}$ is an upper triangle matrix by footnote 26.

  - $0 < [\mathbf{A}]_{i,i} < 1 \iff a_i$ is supplementary (score value or relative priority of a_i),
  - $[\mathbf{A}]_{i,j} = 1 \iff a_i \prec a_j$,
  - $[\mathbf{A}]_{i,j} = 0 \iff a_i \sim a_j$,
  - $0 < [\mathbf{A}]_{i,j} < 1 \iff a_i \bowtie a_j$ (score value or relative priority of $a_i \bowtie a_j$). (The values $[\mathbf{A}]_{i,j}$ will also be called probabilities in constraint $\mathbf{C}_5$.)

- The algorithm must modify the elements of $\mathbf{A}$, such that $0 < [\mathbf{A}]_{i,i} < 1$ and $0 < [\mathbf{A}]_{i,j} < 1$ (and leave the others unchanged), where the final matrix is denoted by $\mathbf{A}(\mathbf{O})$, which contains only the 0 and 1 entries.

- The set of skills that are required to perform activity a_j is denoted by $S(a_j) := \left\{\sigma_1^{(j)}, \ldots, \sigma_{\rho_j}^{(j)}\right\} \subseteq S$ $(j = 1, 2, ..., n)$.

- More specifically, if the *minimum* level of σ_k required for a_j is a nonnegative real number. $L(a_j, \sigma_k) \in \mathbb{R}$, then we must have $\sigma_k \in S(a_j) \iff 0 < L(a_j, \sigma_k)$ and $L(a_j, \sigma_k) \leq \ell(\varepsilon_j, \sigma_k)$ ($\varepsilon_j \subseteq E$ will be chosen by the algorithm).

- $\mathbf{W}$ is the n by s matrix storing L, i.e., $[\mathbf{W}]_{j,k} := L(a_j, \sigma_k)$, $\mathbf{W}$ is called the skilled work domain (in SMM), its elements $w_{j,k} = [\mathbf{W}]_{j,k}$ are called skilled work elements.

- $\mathbf{M}$ is an m by n matrix, called the matching domain, where $[\mathbf{M}]_{i,j} \in [0, 1]$ is the maximal (allowed) ratio of the working time of employee e_i allocated to (working on) task a_j.[29]

- The solution of the SSPSP that must be determined by the algorithm is an n by m matrix (of nonnegative real numbers), denoted by $\mathbf{O}$, where the element $[\mathbf{O}]_{j,i} > 0$ represents the (final) allocation of employee e_i to activity a_j.

---

[29] At this point, the literature assumes the equivalent effectiveness of human resources who have the skills to perform the task. However, the proposed model also addresses both the level of skills and synergy as multiplicative factors that can increase or reduce the effectiveness.

- The value $[\mathbf{O}]_{j,i}$ is the proposed ratio of the working time of e_i 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 $j = 1, 2, ..., n$ and $i = 1, \ldots, m$, while $\sum_{j=1}^{n} [M]_{i,j} \leq 1$ are not required for any $i = 1, \ldots, m$.

- $[\mathbf{O}]_{j,i}$ will sometimes be denoted by $a_j^{e_i}$.

- The duration of activity a_j is denoted by $a_j^{dur}(\mathbf{O})$. (This depends on resources modified by the synergy factor, as calculated in Eqs. (11) and (12). The starting time of a_j is $a_j^{start}(\mathbf{O})$, and the finishing time is $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + a_j^{dur}(\mathbf{O})$ (see Eq. (13)).[30]

- The duration of the project is denoted by p_{dur} or TPT (the total project time), and its cost is by p_{cost} or TPC (the total project cost).

- Each employee e_i can be allocated partially or entirely to the *project*, where the total of $e_i^w := \sum_{j=1}^{n} [\mathbf{O}]_{j,i}$, not exceeding its maximum value $e_i^{maxw} := \sum_{j=1}^{n} [\mathbf{M}]_{i,j}$. Clearly, $0 \leq e_i^w \leq 1$ by $\sum_{j=1}^{n} [\mathbf{O}]_{j,i} \leq 1$. (See the matching domain (M) in Fig. 9.)

- The monthly salary of employee e_i is denoted by e_i^{salary}.

- The notations of structural parameters of synergy networks are summarized in Table 7.[31]

TABLE 7. Analyzed centrality and proximity metrics
(Source: own table)

Notation	**Metrics** *(node level, average)*
BC	Betweenness centrality
CC	Closeness centrality
DC	Degree centrality
PP	Proximity prestige

---

[30] Recall that $a_i \prec a_j$ implies $a_i^{end}(\mathbf{O}) \leq a_j^{start}(\mathbf{O})$.

[31] The average of node-level centrality metrics and proximity prestige are calculated based on (Saxena and Iyengar, 2020, p. 10) and (Musiał et al., 2009, p. 2), respectively.

3.1.2 Formalism Related to Project Duration

Assume that the algorithm has already fixed all of the supplementary tasks and flexible dependencies (stored in $\mathbf{A}$ and in $\mathbf{A}(\mathbf{O})$), as well as the allocations of e_i to a_j (stored in $\mathbf{O}$). In the following, all of the a_j mentioned below have already been decided by the algorithm to be compulsory. Note that Alba and Chicano (2007) assumed that there was no change in the allocation of a certain employee to a certain activity while it was being performed.

The total effort that is allocated to a_j $(j = 1, 2, ..., n)$ is:

$$A_j := \sum_{i=1}^{m} a_j^{e_i} = \sum_{i=1}^{m} [\mathbf{O}]_{j,i}. \tag{5}$$

For any task a_j $(j = 1, ..., n)$ let:

$$\varepsilon_j := \{i \leq m : 0 < [\mathbf{O}]_{j,i}\} \tag{6}$$

be the set of employees who are effectively working on (allocated to) a_j.[32]

Since we do measure the levels of skills, which must be summed separately, we have to consider all the skills separately. For any skill σ_k, the amount (level) of work on σ_k that the team ε_j completes in a_j is (without synergies):[33]

$$A_j^w(k) := \sum_{i=1}^{m} \left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right) = \sum_{i \in \varepsilon_j} \ell(e_i, \sigma_k) \cdot [\mathbf{O}]_{j,i}. \tag{7}$$

Considering the synergies, the adjusted amount of work done by σ_k is:

$$A_j^{w,adj}(k) := \overline{Y}_{\varepsilon_j} \cdot A_j^w(k). \tag{8}$$

---

[32] The employees are assumed to work together, i.e., parallel.

[33] The sum that may be written for all i since $[\mathbf{O}]_{j,i} = 0$ for $i \notin \varepsilon_j$.

Since task a_j requires $L\left(a_j, \sigma_k\right) = [\mathbf{W}]_{j,k}$ amount of skill σ_k, the required time (duration) for completing σ_k in a_j by ε_j without synergies is:

$$a_{j,k}^{dur}\left(\mathbf{O}\right) = \frac{L\left(a_j, \sigma_k\right)}{A_j^w\left(k\right)} = \frac{[\mathbf{W}]_{j,k}}{\sum\limits_{i=1}^{m}\left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right)}, \tag{9}$$

and the adjusted required time (with synergies) is:

$$a_{j,k}^{dur,adj}\left(\mathbf{O}\right) = \frac{L\left(a_j, \sigma_k\right)}{A_j^{w,adj}\left(k\right)} = \frac{[\mathbf{W}]_{j,k}}{\overline{Y}_{\varepsilon_j} \cdot \sum\limits_{i=1}^{m}\left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right)}. \tag{10}$$

Assuming that each e_i uses all of his/her skills simultaneously:

$$a_j^{dur}\left(\mathbf{O}\right) = \max_{k \in S(a_j)} \left\{a_{j,k}^{dur}\left(\mathbf{O}\right)\right\}, \tag{11}$$

and

$$\overline{a}_j^{dur}\left(\mathbf{O}\right) := a_j^{dur,adj}\left(\mathbf{O}\right) = \max_{k \in S(a_j)} \left\{a_{j,k}^{dur,adj}\left(\mathbf{O}\right)\right\}. \tag{12}$$

Of course, completing a_j requires all necessary skills to be covered.[34] This value is used to calculate the ending times of the activities $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + \overline{a}_j^{dur}(\mathbf{O})$, where:

$$a_j^{start}(\mathbf{O}) \geq \begin{cases} 0 & \text{if } \nexists\, a_i \in A,\, a_i \prec a_j \\ \max\{a_i^{end}(\mathbf{O}) : a_i \prec a_j\} & \text{otherwise} \end{cases}. \tag{13}$$

At this point, I also note that the referenced studies have not addressed the cases in which an activity cannot be started because there are no available resources for performing that activity, even though all of its prerequisite activities have been finished. Moreover, I assume that the starting time of the project is 0. (Clearly, a_i and former $\bowtie$ in Eq. (13) and hereinafter are decided by the algorithm to be carried out and be converted to $\prec$.)

---

[34] I.e. $S\left(a_j\right) \subseteq \bigcup\limits_{i \in \varepsilon_j} S\left(e_i\right)$, since for $k \notin \bigcup\limits_{i \in \varepsilon_j} S\left(e_i\right)$ the denumerators of Eqs. (9) and (10) are zero. See also Constraint 2 ($\mathbf{C}_2$) in Eq. (22) in Section 3.1.4.

The values calculated above enable calculating the duration of the project (p_{dur}) as follows:

$$\text{TPT} := p_{dur} = \max\{a_j^{end}(\mathbf{O}) : j = 1, \ldots, n\}. \tag{14}$$

I must emphasize that the values $a_j^{start}(\mathbf{O})$ in Eq. (13) and TPT in Eq. (14) are minimal: no algorithm can start a_j and finish the project earlier than in Eqs. (13) and (14), so they can be denoted by $a_j^{start}(\mathbf{O})_{\min}$ and $\text{TPT}_{\min}$. However, in practice, it is possible that some activities cannot be started at $a_j^{start}(\mathbf{O})_{\min}$ (e.g., because of the lack of human resources). Therefore, the proposed algorithm is allowed to schedule some (even all) tasks a_j later than $a_j^{start}(\mathbf{O})_{\min}$, as described by:

$$a_j^{start}(\mathbf{O})_{ALG} \geq a_j^{start}(\mathbf{O})_{\min}, \tag{15}$$

where $a_j^{start}(\mathbf{O})_{ALG}$ is the real starting time for the task a_j. Clearly, $\overline{a}_j^{dur}(\mathbf{O})_{ALG} = \overline{a}_j^{dur}(\mathbf{O})_{\min}$, $a_j^{end}(\mathbf{O})_{ALG} = a_j^{start}(\mathbf{O})_{ALG} + \overline{a}_j^{dur}(\mathbf{O})_{ALG}$ and:

$$a_j^{start}(\mathbf{O})_{ALG} \geq \begin{cases} 0 & \text{if } \nexists\, a_i \in A,\, a_i \prec a_j \\ \max\{a_i^{end}(\mathbf{O})_{ALG} : a_i \prec a_j\} & \text{otherwise} \end{cases} \tag{16}$$

must also hold.[35] We also require:

$$\text{TPT}_{ALG} \geq \text{TPT}_{\min}. \tag{17}$$

The sequence (of real numbers) is called:

$$(a_1^{start}(\mathbf{O})_{ALG}, \ldots, a_n^{start}(\mathbf{O})_{ALG}) \tag{18}$$

scheduled start time sequence (**SST**). In the following, I omit the subscripts $\min$ and ALG, and I always mean ALG, unless stated otherwise.

---

[35] An explicit formula can be obtained for TPT from the recursive assumptions in Eqs. (11)-(16), mainly based on $\prec$, called the critical or longest min paths (see Kosztyán and Szalkai, 2018, 2020 and Kosztyán et al., 2019 for details).

Fig. 9 presents several networks such as a single project (see the logic domain, **A** and the project graph on the bottom right corner of Fig. 9), a synergy network (see the synergy domain, **S** and the synergy graph in the top left corner of Fig. 9), possible matches between employees and tasks (see the matching domain, **M** and the employee-task matching graph in the top right corner of Fig. 9), and the output domain (**O**). The skill domain (**S**) represents the level of skills, while the amount of required (skilled) works are specified in the skilled works domain (**W**). A prerequisite for project success is that the required skills are available. The proposed matrix-based model only represents the required available skills. The goal is to assign employees to tasks to achieve a good feasible solution with respect to the composite objective function (see Eq. (35)) and constraints (see $\mathbf{C}_1$-$\mathbf{C}_8$ in Section 3.1.4).

3.1.3 Formalism Related to the Project Cost

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

$$\text{TPC}_{syn} = \text{TPC} := p_{cost} = \sum_{i=1}^{m} \sum_{j=1}^{n} (e_i^{salary} \times [\mathbf{O}]_{j,i} \times \overline{a}_j^{dur}(\mathbf{O})), \tag{19}$$

$$\text{TPC}_{nosyn} := \sum_{i=1}^{m} \sum_{j=1}^{n} (e_i^{salary} \times [\mathbf{O}]_{j,i} \times a_j^{dur}(\mathbf{O})). \tag{20}$$

3.1.4 Constraints

While a solution to the SSPSP is calculated, several constraints must be taken into account and be satisfied. First, these constraints are listed, and then I explain each of the constrains in detail.

C$_1$: Each activity must be performed by at least one human resource.

C$_2$: The set of skills that an activity requires must be a subset of the union of skills of the employees who perform this activity.

C$_3$: There must not be any human resource who exceeds his or her maximum dedication (allocation) to the project (roughly, $e_i^w := \sum_{j=1}^{n} [\mathbf{O}]_{j,i} \leq e_i^{maxw}$ for $i = 1, \ldots, m$).

There are two new constraints: the first specifies the set of implemented tasks, and the second considers both the skill levels and the synergies among employees.

C$_4$: The score of the project scenario (total project score, TPS; see Eq. (34)) is greater than a specified (score) constraint C_s.

C$_5$: The probability of the project structure is greater than a specified (probability) constraint C_p.

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

C$_6$: General overwork is not allowed (roughly $E^w = \sum_{i=1}^{m} e_i^w \leq K^w$ for some constant K^w).

C$_7$: The total project cost (TPC) must be less than the cost constraint (C_c).

C$_8$: The duration of the project (the total project time, TPT) must be less than the time constraint (C_t).

In the proposed model, a complex objective (target) function is specified. The goal is to specify the most likely project structure and a resource allocation scheme that minimizes the project duration in the most desired project scenario.

Now, we describe **C**$_1$-**C**$_8$ in detail.

C$_1$: for each $a_j \in A^{c(O)}$,

$$\varepsilon_j := \{e_i \in E : 0 < [\mathbf{O}]_{j,i}\} \neq \emptyset. \tag{21}$$

$\mathbf{C}_2$: for each $a_j \in A^{c(O)}$,

$$S\left(a_j\right) \subseteq \bigcup_{e_i \in \varepsilon_j} S\left(e_i\right). \tag{22}$$

$\mathbf{C}_3$: Since several tasks cannot be solved simultaneously, the rate of the allocation of e_i may vary with time. Therefore, I create a function $e_i^{work}(\tau)$ (for $0 \leq \tau \leq p_{dur}$) that determines how much work by employee e_i is dedicated (allocated) to the project for all of the parallel activities at time τ:

$$e_i^{work}(\tau) := \sum_{\{j \mid a_j^{start} \leq \tau \leq a_j^{end},\ a_j \in A^{c(O)}\}} [\mathbf{O}]_{j,i}. \tag{23}$$

(Here, I mean $a_j^{start}(\mathbf{O})_{ALG} \leq \tau \leq a_j^{end}(\mathbf{O})_{ALG}$, according to SST of the algorithm.) So, C_3 is:

$$e_i^{work}(\tau) \leq e_i^{maxw} \quad \text{for } i = 1, \ldots, m \text{ and } \tau. \tag{24}$$

For $\mathbf{C}_4$ through $\mathbf{C}_6$, we need to define some additional terminology and notation.[36] Let the score values of the implemented activity $a_i \in A^{c(O)}$ be $\mathbb{S}_i := [\mathbf{A}]_{i,i}$ and the score values of omitted one $(a_i \in A \backslash A^{c(O)})$ $\mathbb{S}_i := 1 - [\mathbf{A}]_{i,i}$ ($i = 1, 2, ..., n$).

The probability $p_{i,j}$ of the (input) dependency $a_i \bowtie a_j$ for $a_i, a_j \in A^{c(O)}$ is $p_{i,j} := [\mathbf{A}]_{i,j}$ if that dependency will be included in the project plan (i.e., changed to $a_i \prec a_j$), and $p_{i,j} := 1 - [\mathbf{A}]_{i,j}$ if not (i.e., changed to $a_i \sim a_j$).[37]

The proposed model allows decision-makers to omit several supplementary activities from this project and allocate them to the next project (or the next sprint), i.e., $A^c \subseteq A^{c(O)} \subseteq A$.

---

[36] We must be careful to distinguish the input data in A^c and in $\mathbf{A}$ from the output solution in $A^{c(O)}$ and in $\mathbf{A}\left(\mathbf{O}\right)$.

[37] $i < j$ by footnote 26.

For $\mathbf{C}_4$ through $\mathbf{C}_6$ we are given the (suitable) constants (positive real numbers) C_s, C_p, C_c, C_t, K^w and ϵ^K.

$\mathbf{C}_4$:

$$\text{TPS} := \sqrt[n]{\prod_{i=1}^{n} \mathbb{S}_i} \;\geq\; C_s. \tag{25}$$

$\mathbf{C}_5$:

$$\sum_{a_i, a_j \in A^{c(O)},\ i \neq j} p_{i,j} \geq C_p. \tag{26}$$

For $\mathbf{C}_6$, first, we construct the function $overwork(\tau)$ for $0 \leq \tau \leq p_{dur}$ as:

$$overwork(\tau) := \begin{cases} \sum\limits_{i=1}^{m} e_i^{work}(\tau) - K^w & \text{if } \sum\limits_{i=1}^{m} e_i^{work}(\tau) > K^w \\ & \\ 0 & \text{otherwise} \end{cases}, \tag{27}$$

and the total overwork p_{over} of the project:

$$p_{over} := \int\limits_{\tau=0}^{\tau=p_{dur}} overwork(\tau) d\tau \tag{28}$$

Now, we set:

$\mathbf{C}_6$:

$$p_{over} < \epsilon^K. \tag{29}$$

$\mathbf{C}_7$:

$$\text{TPC} := p_{cost} \leq C_c. \tag{30}$$

$\mathbf{C}_8$:

$$\text{TPT} := p_{dur} \leq C_t. \tag{31}$$

Next, we must find TPT_{min}, TPC_{min} and TPS_{max}. From these, the minimum TPT_{min} is reached if all of the uncertain tasks and flexible dependencies are omitted from the project (i.e., $A^{c(O)} = A^c$ and each $\bowtie$ is changed to $\sim$), and if the maximum number of employees is dedicated (allocated) to the activities (i.e., $[\mathbf{O}]_{j,i} = [\mathbf{M}]_{i,j}$).[38]

3.1.5 Objective Function

Now, we state the objective functions that we seek to optimize simultaneously (in Eq. (35)) using the algorithm:

$$\text{TPT} \to \min, \tag{32}$$

and

$$\text{TPC} \to \min, \tag{33}$$

and

$$\text{TPS} \to \max. \tag{34}$$

These objective (target) functions can be considered a multi-objective problem or a composite objective (target) function and can be specified as follows (here, C_s, C_p, C_c and C_t are given reasonable constants):

$$z := 1 - \sqrt[3]{\left(\frac{C_t - \text{TPT}}{C_t - \text{TPT}_{\text{min}}}\right) * \left(\frac{C_c - \text{TPC}}{C_c - \text{TPC}_{\text{min}}}\right) * \left(\frac{\text{TPS} - C_s}{\text{TPS}_{\text{max}} - C_s}\right)} \to \min, \tag{35}$$

assuming the constraints $\mathbf{C}_1 - \mathbf{C}_8$. Finally, similar to most of the SPSP literature, I assume constant skills of the human resources for simplicity. However, several studies address improvements in human skills, and the proposed model can also be extended to take this into account. For example, Chang et al. (2008) introduce an employee experience and training model that accounts for the learning speed of employees and the time interval of training when calculating the improvement in

---

[38] See Kosztyán and Szalkai, 2018, 2020 and Kosztyán et al., 2019 for details).

employee skills. The model in Chang et al. (2008) influences how quickly employees can perform a specific task.

3.1.6 Summary of Notations

The notations are summarized as follows:

- $E = \{e_1, \ldots, e_m\}$ = employees, $e_i \in E$,
- $[\mathbf{Y}]_{i,j}$ = synergy between e_i and e_j,
- $\overline{Y}_\varepsilon = \sqrt[\eta]{\prod_{i,j \in \varepsilon} \prod_{i<j} [\mathbf{Y}]_{i,j}}$ geometric mean of synergies (see Eq. (1)),
- $S = \{\sigma_1, \ldots, \sigma_s\}$ = skills, $\sigma_k \in S$,
- $S(e_i) := \left\{\sigma_1^{(i)}, \ldots, \sigma_{\rho_i}^{(i)}\right\}$ = skills of e_i, $S(e_i) \subseteq S$,
- $[\mathbf{S}]_{i,k} = \ell(e_i, \sigma_k)$ = the level of e_i in σ_k, $\ell(\varepsilon, \sigma_k) := \overline{Y}_\varepsilon \cdot \sum_{i \in \varepsilon} \ell(e_i, \sigma_k)$,
- $A = \{a_1, \ldots, a_n\}$ = tasks (activities), $a_j \in A$:
  - $A^c =$ mandatory (compulsory), given, $A^- = A \setminus A^c$ supplementary,
  - $A^{c(O)} =$ compulsory tasks decided by the algorithm, $A^c \subseteq A^{c(O)} \subseteq A$,
  - $a_i \prec a_j$ strict (or required) dependency, $a_i \sim a_j$ no dependency,
  - $a_i \bowtie a_j$ uncertain (or flexible) dependency,
- $\mathbf{A}$ = input matrix:
  - $[\mathbf{A}]_{i,i} = 1 \iff a_i$ is mandatory,
  - $0 < [\mathbf{A}]_{i,i} < 1 \iff a_i$ is supplementary,
  - $[\mathbf{A}]_{i,j} = 1 \iff a_i \prec a_j$,
  - $[\mathbf{A}]_{i,j} = 0 \iff a_i \sim a_j$,
  - $0 < [\mathbf{A}]_{i,j} < 1 \iff a_i \bowtie a_j$,

- $\mathbf{A}(\mathbf{O}) = \mathbf{A}$ as modified by the algorithm,
- $S(a_j) := \left\{\sigma_1^{(j)}, \ldots, \sigma_{\rho_j}^{(j)}\right\}$ = skills required to a_j, $S(a_j) \subseteq S$,
- $[\mathbf{W}]_{j,k} = w_{j,k} = L(a_j, \sigma_k) =$ the minimum level of σ_k required to a_j,
- $[\mathbf{M}]_{i,j}$ = the maximal (allowed) ratio of the working time of e_i allocated to a_j,
- $[\mathbf{O}]_{j,i}$ = the (proposed) working time ratio of e_i allocated to a_j,
- $a_j^{e_i} := [\mathbf{O}]_{j,i}$,
- $A_j := \sum\limits_{i=1}^{m} a_j^{e_i}$ = the total effort allocated to a_j (in terms of human resources),
- $\varepsilon_j := \left\{e_i \in E : 0 < a_j^{e_i}\right\}$,
- $e_i^w := \sum\limits_{j=1}^{n} [\mathbf{O}]_{j,i} \leq e_i^{maxw} := \sum\limits_{j=1}^{n} [\mathbf{M}]_{i,j}$,
- $a_j^{dur}(\mathbf{O})$ = duration of a_j, see Eq. (11),
- $\overline{a}_j^{dur}(\mathbf{O})$ = adjusted duration of a_j, see Eq. (12),
- $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + \overline{a}_j^{dur}(\mathbf{O})$,
- $a_j^{start}(\mathbf{O})_{\min}$ = minimal starting time of a_j, see Eq. (13),
- $a_j^{start}(\mathbf{O}) = a_j^{start}(\mathbf{O})_{ALG}$ = the scheduled starting time, **SST**, decided by the algorithm,
- **SST** = $(a_1^{start}(\mathbf{O})_{ALG}$, ... , $a_n^{start}(\mathbf{O})_{ALG})$, see Eq. (18),
- $\text{TPT} = p_{dur}$ = total project duration, see Eq. (14),
- $\text{TPC}_{syn} = \text{TPC} = p_{cost}$ = total project cost with synergies, see Eq. (19),
- TPC_{nosyn} = total project cost without synergies, see Eq. (20),
- $e_i^{work}(\tau)$:= how much e_i is allocated to the project at time τ, see Eq. (23),

- $\mathbb{S}_i$ = score values of a_i,
- $p_{i,j}$ = probability of the dependency $a_i \bowtie a_j$,
- TPS = total project score, see Eq. (25),
- $overwork(\tau)$ = general overwork at time τ, see Eq. (27),
- p_{over} = the total overwork of the project, see Eq. (28),
- z = the composite objective function to be minimized, see Eq. (35).

3.2 Proposed Hybrid Genetic Algorithm

Since SPSP is NP-hard (Xiao et al., 2013), which is a special case of synergy-based SPSP, the SSPSP is also NP-hard. There are exact methods that can solve small instances of SPSP to optimality (Vega-Velázquez et al., 2018) (see Section 2.3.4); however, these methods are not practical for larger instances, and their resolution requires other kinds of techniques such as metaheuristics (Yang, 2010). Thus, a metaheuristic method of solving it is proposed. This section provides an overview of this algorithm.

Although most variables of the objective (target) function (i.e., dedications to activities and the scheduled start time of activities, referred to as SST) are continuous (with real variables), the model also contains several binary variables, namely, decisions regarding task/dependency exclusion/inclusion. Therefore, a mixed-integer genetic algorithm is used to seek a good feasible solution. All of the default operators (i.e., crossover, mutation, and selection) of the genetic algorithm must be modified because an excluded task has no dependency, duration, or cost demands.

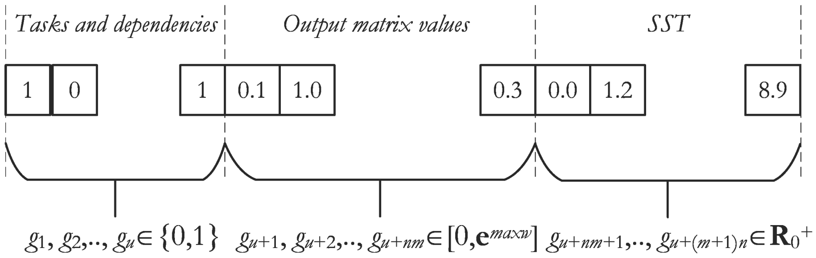
The results of the genetic algorithm are refined using a Nelder-Mead minimization (NMM) method. The NMM optimization function continues the optimization after the termination of the GA. The NMM function can refine only the real values such as the values of the output matrix ($\mathbf{O}$) and the scheduled start time (SST) of activities. The MATLAB Global Optimization toolbox is used to implement the hybrid genetic algorithm; however, the standard mutation, crossover and selection function as well as the hyperparameters must be modified (see Section 3.2.1). I hereinafter refer to this hybrid genetic algorithm as the synergy-based agile project scheduling algorithm (SynAPS).

Generally, the sets of excluded/included flexible task occurrences and flexible task dependencies (see the logic domain $(\mathbf{A})$), the values of allocations (see the output domain $(\mathbf{O})$) and the scheduled start time (SST) for all tasks must be specified. After the final specification, the resulting matrix $\mathbf{A}'$ contains only values $\{0, 1\}$, where $[\mathbf{A}]_{ii} = 1$ ($[\mathbf{A}]_{ii} = 0$) means that task a_i will be included in (excluded from) the project. Nevertheless, if a task is excluded from the project, the dependencies of the (excluded) tasks and all the (time/cost/resource) requirements are also excluded from the project.

3.2.1 Parameters of the SynAPS

Fitness function: In our case, the fitness function is a composite function (see Eq. (35)). We seek the elements of the output matrix $(\mathbf{O} \in \mathbb{R}_{+}^{n\times m})$, the decision results of the flexible dependencies and supplementary task occurrences that are represented in the final logic domain $\mathbf{A}' \in \{0, 1\}^{n\times n}$, and the scheduled start time for all activities such that the resource constraint can be satisfied. It is assumed that a potential solution to a problem may be represented as a set of parameters/values. These values (known as genes) are joined together to form a vector (referred to as a chromosome, shown in Fig. 10). In genetic terminology, the set of values represented by a particular chromosome is referred to as an individual.

FIGURE 10. Structure of a chromosome
(Source: own figure)



If u is the number of uncertain tasks $+$ dependencies, m is the number of employees, and n is the number of activities, then a chromosome vector with $u + (m + 1)n$ elements can be constructed. For ease of use, the first part of the chromosome is the decision part, and the numbers are binary values. The second part is the output, which codes the output matrix row by row. The last part is the scheduling part, where the values are also real and positive. The fitness of an individual depends on its chromosome and is evaluated by the fitness function. During the reproductive phase, individuals are selected from the population and are recombined, producing offspring that compose the next generation. Parents are then randomly selected from the population using a scheme that favors fitter individuals. After two parents have been selected, their chromosomes are recombined, typically using the mechanisms of crossover and mutation. The latter is usually applied to some individuals to guarantee population diversity.

Population: In the first step, a number of possible solutions must be generated. First, the elements of the logic domain $\mathbf{A}'$ will be generated because if $[\mathbf{A}']_{ii} = 0$, then $[\mathbf{O}]_{ij:=1,2..,m} := 0$, i.e., activity $a_i \in A$ will be excluded from the project; therefore, the excluded task has no time, cost or resource requirements. Since an excluded task has no dependencies, $[\mathbf{A}']_{ji} = [\mathbf{A}']_{ij} := 0$ if $[\mathbf{A}']_{ii} = 0$. I denoted the initial population by P_0 and the population of the G^{th} generation by P_G.

Selection mechanism: One of the main operators in a genetic algorithm is the selection operator. First, feasible solutions must be selected by a tournament. Because we usually have many feasible solutions, we use a tournament selection mechanism. In this case, each parent is determined by choosing a random number of tournament players and then choosing the best individual from that set to be a parent. The tournament size must be at least 2. In our case, I set the tournament size to 10. The set of selected chromosomes in the G^{th} generation was denoted by S_G.

Elite count: This is a positive integer specifying how many individuals in the current generation are guaranteed to survive to the next generation. It was set to 5% in this work, which means there were 5% so-called elite children in every generation.

Crossover fraction: The crossover fraction specifies the fraction of each population (other than elite children) that consists of crossover children. A crossover fraction of 1 means that all of the children other than elite individuals are crossover children, while a crossover fraction of 0 means that all of the children are mutation children. The best results were obtained when this parameter was set to 0.8. This means that 80% of the selected children (excluding elite children) were parents used in the crossover function (so-called crossover children) and 20% of the selected children (excluding elite children) were used in the mutation function (so-called mutation children).

Crossover operator: The (fractionated) selected chromosomes were used. Since a chromosome has a binary or decision part and two continuous parts, two kinds of crossover functions must be combined. For the continuous parts, the arithmetic crossover function is used. Such a function creates children that are the weighted arithmetic mean of the two parents (i.e., depending on the fitness function). For the continuous part (called recombined), this crossover function can be very effective. At the same time, this crossover mechanism cannot be used for the binary or decision parts of the chromosome. In this case, a uniform crossover function is used. However, the parents may be infeasible; thus, here I assume that the feasible parents' genes are 10 times as dominant. In other words, a gene is ten times more likely to originate from feasible parents than from infeasible parents.[39] After the set of children chromosomes has been determined, the requirements of the excluded tasks and their task dependencies must be eliminated (set to 0). The set of recombined children chromosomes in the G^{th} generation was denoted by $C_G(S_G)$.

Mutation operator: The mutation is a two-step process where the first step is general and is carried out 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 rate of being mutated. According to the results of the settings, this rate is specified as 0.01. 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. In the presence of constraints, directions that are adaptive with respect to the preceding successful or unsuccessful generation are randomly generated. 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). The set of mutated chromosomes in the G^{th} generation was denoted by $M_G(S_G)$.

Next generation: The mutated and crossover individuals are considered together with the old population, and the best $N = 100$ individuals are selected for the next generation.

Stopping criteria: A genetic algorithm terminates if we reach the maximum number of generations (set at 100 in this case) or if the average relative change in the best

---

[39] If all the parents are feasible or all the parents are infeasible, the standard uniform crossover function is used.

fitness function value over generations is less than or equal to the function tolerance ($1E-8$).

3.3 Proposed Simulation Framework

The purpose of the simulation is to determine the parameter(s) that influence(s) the (changes in) project duration and the cost demands of the project while considering the synergies between the employees. The proposed, so-called synergy-based agile simulation framework (SynASF) can be separated into two stages:

- **Stage 1:** Specifying problem sets,
- **Stage 2:** Solving problems.

3.3.1 Specifying Problem Sets

A problem set contains the following:

(1) SMMs that numerically represent the synergy-based software scheduling problem,

(2) Minimum (δ_{min}) and maximum (δ_{max}) values of the possible synergies between two members,[40]

(3) Constant ratios (see Section 3.3.1.2).

3.3.1.1 Specification of the SMM

To determine the SMM matrices, first, the logic and skill domains of the SMM matrices are generated by the iMOPSE project generator (Myszkowski et al., 2019). The iMOPSE generator randomly selects which skills are required for the different tasks. The levels of these skills (skilled work elements; $[\mathbf{W}]_{j,k}$) are determined

---

[40] In the simulation, the average (pairwise) synergy between two employees is $AvgSyn := (\delta_{\text{min}} + \delta_{\text{max}})/2$. Note that unlike $\overline{Y}_\varepsilon$, we already know the elements of the **O** domain when defining $AvgSyn$, thus $AvgSyn$ measures synergy more accurately than $\overline{Y}_\varepsilon$ does.

by Monte Carlo simulation (Chan, 2013) in such a way that the resulting project scenarios are still feasible at the maximum of skilled work elements.[41]

The aim of the selection and the generation of the initial project plans is to meet the expectations for (IT) software project plans to the greatest extent possible, particularly the features of agile projects. Therefore, the following selection criteria (CR) were defined as follows:

CR$_1$: Since Tavares et al. (1999) and Vanhoucke (2012) showed that software projects usually contain more parallel tasks, in the case of selected project structures, the number of parallel tasks should be greater than the number of serial tasks.[42] Nevertheless, several agile methods such as the KANBAN and SCRUMBAN methods limit the number of parallel work-in-progress (WIP) tasks and allow only $3-5$ WIP tasks. Therefore, in the simulation, the number of WIP tasks must be lower than 5.

CR$_2$: Projects are usually separated into smaller autonomous subprojects (so-called sprints; see, e.g., Dingsøyr et al. 2012) that should be completed within $2-5$ weeks; therefore, the number of tasks is limited and should not be greater than 50.

Therefore, the number of tasks were 30 and 50 (N_a), and the number of employers (resources) was 10. The other parameters were selected as the default values with a minimal duration=1, maximal duration=8, minimal resource skill type=1, and maximal resource skill type=9 (the number of skills is N_{sk}). Ten projects that fulfilled criteria CR$_1$-CR$_2$ were selected from the generated project networks.

---

[41] Note that unit salaries are used in the calculations.

[42] Following the simulations of Tavares et al. (1999), $i_2 = (m-1)/(n-1) \in [0.2, 0.3]$, where m is the number of stages in a topological ordered network and n is the number of tasks. $i_2 = 1$ if all of the tasks are completed in a serial manner, and $i_2 = 0$ if all of the tasks are completed in parallel.

Nevertheless, these project networks cannot be used directly. Neither known project generators – such as ProGen (Kolisch and Sprecher, 1997), RanGen I (Demeulemeester et al., 2003), and II (Vanhoucke et al., 2008) or iMOPSE (Myszkowski et al., 2019) – nor open project data sources – such as PSPLIB (Kolisch and Sprecher, 1997) and MMLIB (Peteghem and Vanhoucke, 2014) – distinguish mandatory and supplementary tasks or consider strict and flexible dependencies. Thus, there are no score values linked to task completion or task dependencies. Therefore, to simulate project flexibility, $ff \times 100\%$ (where $ff := \{0.10, 0.15, 0.20\}$) of the matrix values are reduced from 1 to a lower value in the range of $[0.5, 1.0]$, causing them to represent either uncertain tasks or flexible dependencies.

The original database does not contain synergies between the employees, which therefore must be specified in order to follow a sociometric structure (see Fig. 13. The default value of synergy between two employees is 1 and is known as neutral synergy. For given sociometric structures (see Fig. 13 in Section 4.1), the synergy values (weights of synergy networks) can be either greater or less than 1. Values $\delta_{\text{min}} := \{-0.3, -0.1, ..., 0.2\}$, $\delta_{\text{max}} := \delta_{\text{min}} + 0.1$ represent the minimum and maximum differences between the generated and neutral synergies, respectively.

3.3.1.2 Calculation of Constraints

After SMM matrices are generated the constraints are calculated. The ratio of the project score is $C_{s\%} = \frac{C_s - \text{TPS}_{\text{min}}}{\text{TPS}_{\text{max}} - \text{TPS}_{\text{min}}} := \{0.6, 0.8., 1.0\}$. In addition to the score constraints, both time ($C_t := \{1.00, 1.25, 1.50\} \cdot \text{TPT}_{\text{min}}$) and cost ($C_c := \{1.00, 1.25, 1.50\} \cdot \text{TPC}_{\text{min}}$) are calculated. As a result, a set of $n_{C_t\%} \times n_{C_c\%} \times n_{C_s\%} \times n_{ff\%} \times n_{\delta_{\text{max}}} \times n_n \times n_{proj} = 3^5 \times 2 \times 144 = 69,984$ SMM matrices and constraints are generated, where $n_{C_t\%}, n_{C_c\%}, n_{C_s\%}$ are the numbers of time, cost and score constraints, respectively; $n_{ff\%}$ is the number of flexible parameters; $n_{\delta_{\text{max}}}$ is the number of δ_{max} values; n_n is the number of task numbers $(30, 50)$; and n_{proj} is the number of selected project structures.

3.3.2 Solving Problems

In stage 2, the problem sets are solved by the SynAPS, where the objective (target) function is Eq. (35). Given this complex target function, in addition to the maximization of the project scores, the project duration, project cost, and thus the resource demands must be reduced simultaneously. The project durations (TPT), project costs (TPC), and total project scores (TPS) are calculated for every problem set both considering and ignoring synergies, obtaining $\mathrm{TPX}_{\mathrm{syn}}$ and $\mathrm{TPX}_{\mathrm{nosyn}}$, respectively. The differences between $\mathrm{TPX}_{\mathrm{nosyn}}$ and $\mathrm{TPX}_{\mathrm{syn}}$ are calculated, and the cost differences observed in the results are studied. In this study, both positive (or favorable) and negative (or unfavorable) synergy effects are considered. A positive $\Delta\mathrm{TPC}=\mathrm{TPC}_{\mathrm{nosyn}}-\mathrm{TPC}_{\mathrm{syn}}$ means that the positive synergy effect is greater than the negative synergy effect.

CHAPTER 4

Results and Discussion

This section provides an answer to the research questions (see $\mathbf{RQ}$s in Section 1.2), then formulates the research theses of the dissertation.

4.1 Answering the Research Questions

$\mathbf{RQ}_1$: Is it possible to determine a scheduling problem for traditional and flexible project planning environments that considers not only the skills of human resources but also the synergies between them?

In line with the $\mathbf{RA}_1$ (see Section 2.6), the classical software project scheduling problem (SPSP) was extended by considering flexible task dependencies and pairwise synergies between resources (see Section 3.1). The so-called synergy-based software project scheduling problem (SSPSP) thus defined reflects the APM approach (see Section 2.1.5) widely used in software development practice, and consequently outlines a more realistic planning problem than the classical SPSP.

$\mathbf{RQ}_2$: Is it possible to develop a network- or matrix-based project scheduling model that takes into account flexible project plans, the skills of human resources as well as the synergies between them?

To model both classical and synergy-based SPSPs, I proposed an extended form of the multi-domain matrix (MDM) according to the $\mathbf{RA}_2$ (see Section 2.6). The new, so-called synergy-based multi-domain matrix (SMM) contains multiple interconnected domains that model the flexible logical structure of the project, the amount of (skilled) work to be performed within the project, the skills of human resources and the positive and negative pairwise synergies between them, as well as the maximum resource assignments (see Fig. 9 in Section 3.1.1).

$\mathbf{RQ}_3$: Is there a(n optimal) solution for scheduling a flexible software project plan that considers the synergies between resources?

Using the proposed simulation framework (SynASF) (see Section 3.3) and the proposed hybrid genetic algorithm (SynAPS) (see Section 3.2), $69,984$ classical and synergy-based SPSPs were simulated and solved with respect to the given objective function (see Eq. (35) in Section 3.1.5) and given constraints (see $\mathbf{C}_1$-$\mathbf{C}_8$ in Section 3.1.4). For all optimization problems, SynAPS found a feasible solution. Not only does this verify the $\mathbf{RA}_3$ (see Section 2.6), but it also shows that by applying the proposed multi-domain model (SMM) and the SynAPS, both classical and synergy-based SPSPs can be solved even in a flexible project planning environment.

$\mathbf{RQ}_4$: Is it possible to develop a simulation framework to examine the impact of the synergies between resources, the structures of synergy networks, the skills of human resources as well as the size, flexibility, and constraints of the project on the implementation of the project schedule?

In order to decide whether the proposed simulation framework is suitable for performing the examination referred to in the $\mathbf{RQ}_4$, I analyze the optimization results of $69,984$ classical and synergy-based SPSPs simulated by the SynASF (see Section 3.3). The analysis is based on the research model presented in Fig. 11.[43]

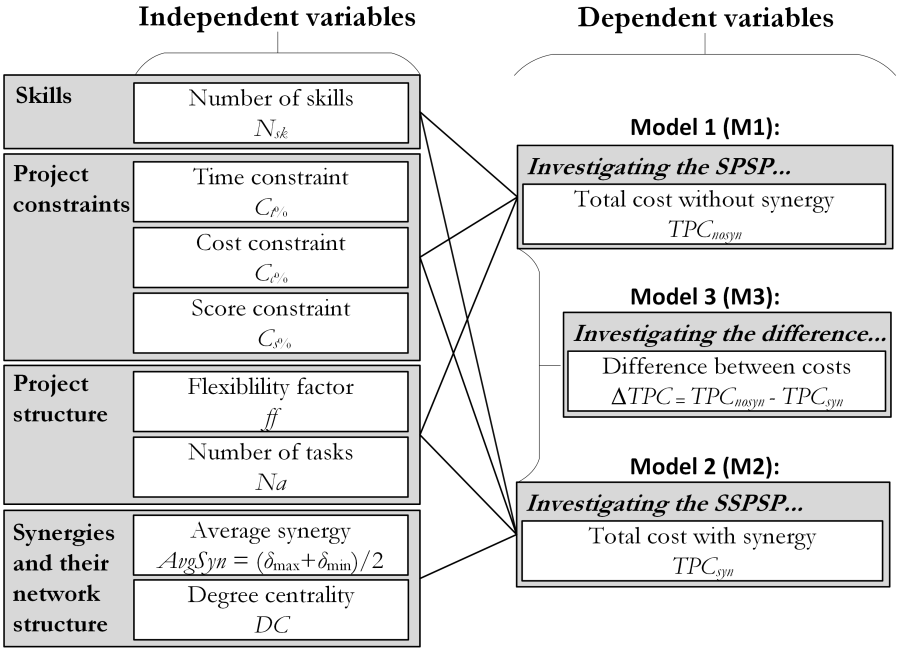
This model is focused on three cases: the case of SPSP, in which synergies are ignored (M1), the case of SSPSP, in which synergies considered (M2), and the difference between these two approaches (M3). Since the cost of the project is a function of the duration and the employees' salary, in the main text, TPC_{nosyn}, TPC_{syn}

---

[43] Note that contrary to Table 7, only one structural parameter is considered in Fig. 11. Reducing the model is justified by the high correlations between the degree centrality (DC) and other structural parameters (see Table A.1 in Appendix A.1). DC is the average of the node-level centrality values calculated by dividing the actual degree of a node by the number of nodes reduced by 1.

FIGURE 11. Research model

(Source: own figure)



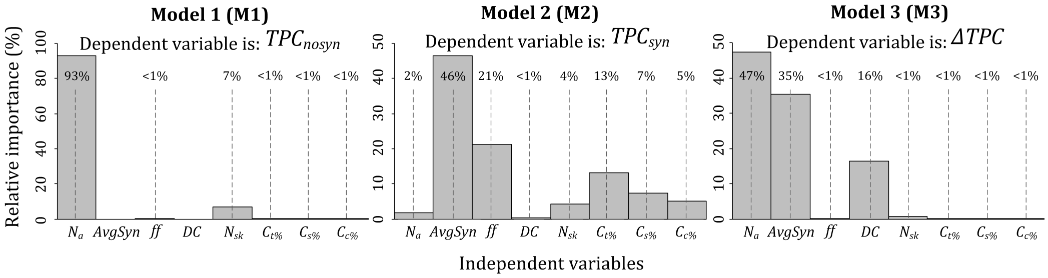
and ΔTPC are considered as the only dependent variables. To derive a complete picture of the operation of the proposed method, I also perform calculations for the project durations in the Appendices (see Fig. A.1 in Appendix A.2).[44] During the analysis of the optimization results, I employ the regression tree ensemble model of the MATLAB Regression Learner App (MathWorks, 2019b). For both the main (see Fig. 11) and the additional (see Fig. A.1 in Appendix A.2) models, 10-fold cross-validation was used, and the hyperparameters were tuned by Bayesian optimization. Fig. 12 shows the relative importance of the independent variables (predictors) for all three cases.[45]

---

[44] Note that the scores for the project scenarios are the same for classical and synergy-based SPSPs, therefore the impact of independent variables on TPS is not examined separately.

[45] The relative importance is calculated using the *predictorImportance* MATLAB function (see MathWorks, 2019a). The full details of the optimization processes can be found in Appendix A.3.

FIGURE 12. Relative importance of the various predictors
(Source: own figure)



In case of the SPSP (see Fig. 12 – M1), the size of the project (N_a) and the various skills of employees (N_{sk}) are the main factors impacting project costs; however, if synergies between two employees are considered (see Fig. 12 – M2), the principal effect is due to the average pairwise synergy ($AvgSyn$) itself. In this case, changes in the time, score and cost constraints ($C_{t\%}, C_{s\%}, C_{c\%}$) and flexibility ($ff$) also influence the project cost, while the previously important size (N_a) and skills (N_{sk}), as well as degree centrality (DC), have only a small impact on the cost. Model 3 specifies the parameters that explain the cost differences of these two approaches. According to this model, the project size (N_a) has the highest explanatory power of 47%, followed by the average synergy ($AvgSyn$ – 35%) and the structural parameter (C_D – 16%). These results have two main implications. First, the synergy-related parameters have a very strong effect on projects' costs even though, based on the current parameterization of the model, the interdependence of two employees can only change their performance to a relatively small extent (see Section 3.3.1.1). Second, the high impact of the structural parameter (DC) appears to be consistent with the relevant literature (see Section 2.5).

To determine, which structures of synergy networks reduce the project cost the most, I examine how ΔTPC (see Fig. 11 – M3) is influenced by the sociometric structure and how their relation changes based on the structural parameter (DC; the respective results are shown in shades of gray) or how project flexibility (ff – Case $1-3$) varies (see Fig. 13).

FIGURE 13. Effect of sociometric structures on the project cost

(Source: own figure)

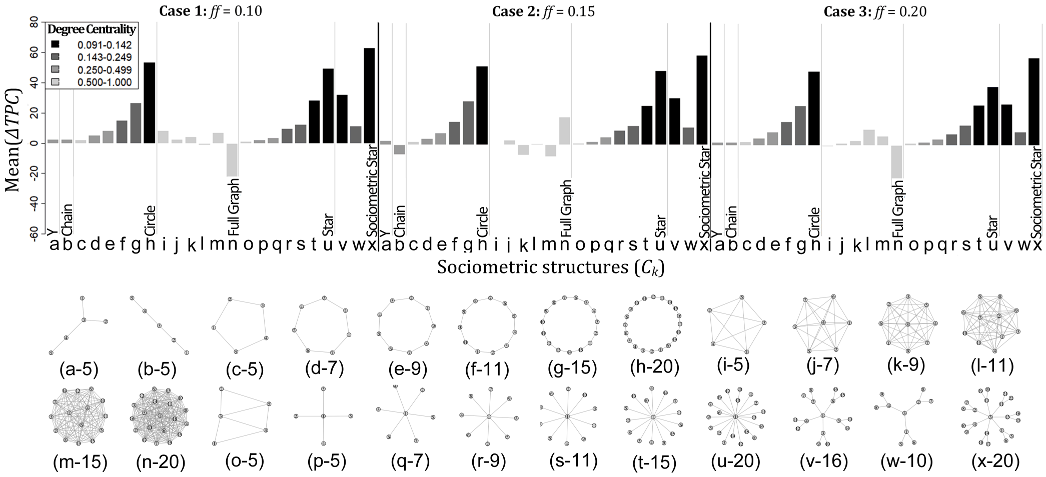


Fig. 13 shows that structures with a low degree centrality (DC) generally lead to a greater reduction in the project cost; however, the veracity of this statement depends on the topology of the sociometric network. Although I observe that the flexibility of the project (ff) has a negligible effect on ΔTPC (see Fig. 12), I find that the chain and full graph networks are highly sensitive even to small changes of this parameter (see Fig. 13 Case 1-3). In some cases involving these topologies, the TPC$_{\text{syn}}$ is greater than the TPC$_{\text{nosyn}}$, resulting in a negative ΔTPC. Furthermore, the most decentralized topology (the full graph) leads to the worst results because of its high sensitivity to negative synergies. On the one hand, these findings are contrary to that of Sparrowe et al. (2001), since, in their model, decentralized networks (such as circle and full graph networks) are unable to reduce costs by an amount greater than that of the centralized networks (such as star and sociometric star networks). On the other hand, it is in line with the empirical findings of Sanchez et al. (2017), who observed that the formal power of the project manager as well as the smaller, less dispersed teams have a positive impact on the outcome of projects (see Section 2.4). The results presented in Fig. 13 are also in accordance with Shaw

(1964), since I found that teams with decentralized networks took less time to finish complex tasks than groups with centralized networks. However, while synergy can be proxied by theories like Belbin's on the effective team roles, or findings on the relation between teamwork effectiveness and the hierarchy or communication (see Sections 2.4-2.5), note that, due to the complex nature of the synergy, the results of the simulation are difficult to compare with the empirical results presented above (see Section 2.5). Given the simulation results, the $\mathbf{RA}_4$ (see Section 2.6) is accepted.

4.2 Research Theses

Based on the presented results (see Section 4.1) – and their validity (see Chapter 6) –, one research thesis is formulated for each of the four research questions (see **RQ**s in Section 1.2) and assumptions (see **RA**s in Section 2.6). The four theses (**RT**s) of the dissertation are the following:

$\mathbf{RT}_1$: The proposed synergy-based software scheduling problem (SSPSP) extends the classical software scheduling problem (SPSP) to take into account the flexibility of project plans, as well as the pairwise synergies between resources.

$\mathbf{RT}_2$: The proposed synergy-based multi-domain matrix (SMM) contains multiple interconnected domains that model the flexible logical structure of the project, the amount of (skilled) work to be performed within the project, the skills of human resources and the positive and negative pairwise synergies between them, as well as the maximum resource assignments. The proposed matrix is able to model all solutions of both the classical (SPSP) and the synergy-based (SSPSP) software scheduling problems.

$\mathbf{RT}_3$: The proposed synergy-based agile project scheduling algorithm (SynAPS) finds a feasible solution for both the classical (SPSP) and the synergy-based (SSPSP) software project scheduling problems with respect to the given objective function (that minimizes the duration and cost of the project while simultaneously maximizing its score) and given constraints (in relation to the duration, cost, resource, and score of the project).

- $\mathbf{RT}_4$: The proposed synergy-based agile simulation framework (SynASF) is suitable for examining the impact of pairwise synergies between resources, synergy structures, skills as well as the size, flexibility, and constraints of the project on the implementation of project scheduling. According to the synergy-based agile simulation framework (SynASF):
  - $\mathbf{RT}_{4.1}$: The costs of projects are most sensitive to the pairwise synergies of human resources.
  - $\mathbf{RT}_{4.2}$: The impact of pairwise synergies on project costs is mainly influenced by the size of the project, the average pairwise synergy, and the structural parameter (degree centrality) of the synergy network.
  - $\mathbf{RT}_{4.3}$: Synergy networks with low degree centrality lead to a greater reduction in the project cost; however, the impact of synergies is also influenced by the topology of networks. The highest costs are obtained by the synergy networks with the most decentralized topology (full graph) because of their high sensitivity to negative synergies.

CHAPTER 5

Practical example

This chapter presents an empirical example for defining and solving a real-life software project scheduling problem, both in a classical (SPSP) and in a synergy-based (SSPSP) context. The source of the data is a multinational automotive manufacturing company that is a market leader in automotive safety, automated driving, and electric mobility.[46] This company develops embedded software for automotive equipment, and in order to fulfill rapidly changing customer needs, places great emphasis on applying an APM approach. The practical example is based on a scheduling problem of a software development sprint contained in a product development project. It was selected based on two criteria. On the one hand, the selected sprint must be a good representative of other similar sprints managed by the company, not only in terms of its logical structure but also in terms of the proportion of workloads. On the other hand, it should rely as much as possible on teamwork, so in addition to generalizability, I also took into account the number of team members planned to be involved in the implementation.

In the following, I will first describe the planning problem of the selected sprint in detail. Then by using SMM matrices (see Fig. 9 in Section 3.1.1), this problem is defined as a classical (SPSP) and as a synergy-based (SSPSP) project scheduling problem. Finally, after both problems are solved by the proposed SynAPS (see Section 3.2), results are discussed.

---

[46] The company's name is withheld at their request. The anonymized data was collected by Péter Harta, who made it available to me. Some of this data was used in Harta (2021).

5.1 Problem Definition

The studied problem is a typical example for the software project scheduling problem defined in Section 2.3. As a combination of scheduling and human resource allocation, the goal was to define a schedule that meets both pre-defined time and cost constraints. The start date of the sprint was scheduled for October 16, 2020 and it has to be implemented by November 8, 2021.[47] In other words, the sprint has to be completed in 268 business days. In addition, the planned budget may not exceed 1960 (measured in an employee's daily salary).[48]

The logical structure of the studied software development sprint is defined by the company's product development template. It contains 9 mandatory and 3 supplementary tasks, which were scheduled along the following logic. The sprint begins with the translation of customer requirements (a_1), followed by the parallel development of function models related to the communication (a_3) and steering (a_4) systems. The development of another two functional models in connection with the brake (a_2) and engine systems (a_5) are scheduled as supplementary tasks. As a result of these development tasks, the codes related to each functional area and their documents are completed. Once the flash loader (a_6) is also ready, codes are integrated and uploaded to the developed device (a_7). These tasks are followed by the integration test (a_8), then by the labor (a_9) and road (a_{10}) tests performed in parallel. Finally, after the quality control of the implementation process (a_{11}), the sprint is closed by the preparation of the release request document and the handover of the software (a_{12}). These tasks and their relations are described in Table 8.

The examined company manages its projects in a multi-project environment. A total of 20 employees are available for implementation, of which 16 employees can

---

[47] Data related to task and project durations as well as to implementation constraints were linearly transformed at the company's request.

[48] Since salaries are considered confidential data, unit salaries are used in the calculations.

TABLE 8. Tasks of the software development sprint

(Source: own table)

Notation	**Name**	**Probability**	**Workload** (*in* 8 *hours*)	**Direct Predecessors**
a_1	Translation of requirements	1.0	20	
a_2	Function model 1	1.0	80	a_1
a_3	Function model 2	0.6	110	a_1
a_4	Function model 3	1.0	40	a_1
a_5	Function model 4	0.8	30	a_1
a_6	Flash loader development	1.0	8	a_2, a_3, a_4, a_5
a_7	Software integration	1.0	4	a_2, a_3, a_4, a_5
a_8	Integration test	1.0	5	a_6, a_7
a_9	Laboratory test	1.0	15	a_8
a_{10}	Road test	0.6	10	a_8
a_{11}	Quality control	1.0	4	a_9, a_{10}
a_{12}	Software handover	1.0	6	a_{11}

be involved in the project full-time (8 hours per business day) and 4 employees part-time (4 hours per business day). In addition to the data originally considered during scheduling, Belbin's team roles (Belbin, 1981, 2010) of employees are also collected (see Table 6 in Section 2.4.2.1). Information on available employees is found in Table 9.

5.2 Specification of the SMM

Synergy Domain (Y): To define synergy domain, Belbin's team role test (Belbin, 1981, 2010) is applied (see Section 2.4.2.1). In line with Belbin's theory, I sought to compose a high-functioning, heterogeneous team in terms of role. Positive pairwise synergies are assumed (1.3) between those employees that have different role categories (thinking, action, social). If the role category of the two employees is the same but their roles are different, I assume a moderately negative synergy (0.85). Finally, in the case of the same categories and roles, a lower synergy value is assumed than in the previous case (0.7).[49]

---

[49] In the case of the SPSP, all pairwise synergy values are 1. Note that in this example, SPSP differs from SSPSP only in this domain.

TABLE 9. Available employees

(Source: own table)

Notation	**Dedication** (*hours per BD*[‡])	**Experience in...**	**Belbin's roles**[*]	
			Category	**Role**
e_1	8	$\sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_{11}, \sigma_{12}$	Action	SH
e_2	4	$\sigma_2, \sigma_3, \sigma_4, \sigma_5$	Social	SH, CF
e_3	8	$\sigma_2, \sigma_3, \sigma_4, \sigma_5$	Social	SP
e_4	8	$\sigma_2, \sigma_6, \sigma_{11}, \sigma_{12}$	Action	TW
e_5	8	$\sigma_1, \sigma_2, \sigma_6, \sigma_7, \sigma_8$	Social	CF
e_6	8	$\sigma_3, \sigma_6, \sigma_{11}$	Action	SH
e_7	8	$\sigma_2, \sigma_3, \sigma_4, \sigma_6$	Thinking	PL
e_8	4	$\sigma_1, \sigma_2, \sigma_4, \sigma_7, \sigma_8, \sigma_{12}$	Action	SH
e_9	8	$\sigma_3, \sigma_4, \sigma_9$	Action	SP
e_{10}	8	$\sigma_2, \sigma_5, \sigma_9$	Social	SP
e_{11}	8	$\sigma_3, \sigma_{10}, \sigma_{11}$	Social	SP
e_{12}	8	$\sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6$	Social	SP
e_{13}	8	σ_{10}, σ_{11}	Social	SP
e_{14}	8	$\sigma_1, \sigma_2, \sigma_3, \sigma_4$	Action	SH
e_{15}	8	$\sigma_2, \sigma_3, \sigma_4, \sigma_5$	Action	ME
e_{16}	4	$\sigma_1, \sigma_4, \sigma_5$	Action	PL
e_{17}	8	$\sigma_2, \sigma_3, \sigma_5, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}$	Social	SP
e_{18}	8	$\sigma_2, \sigma_3, \sigma_5, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}$	Action	SH
e_{19}	4	$\sigma_2, \sigma_3, \sigma_6$	Action	CO
e_{20}	8	$\sigma_2, \sigma_4, \sigma_6$	Action	ME

[*] This information was not considered in the original scheduling problem.

[‡] BD: business day.

Experience (formerly Skill) Domain (S): As the skills of employees are considered confidential, I define this domain based on the experience of the employees. To this end, I examined which employees worked in which activities and how many times in the last 3 years. The experience of employees is 1 if their participation number deviates from the average by no more than half the standard deviation. Above/below these limits of half the standard deviation, experience values are 1.15 and 0.85, respectively. With a participation number one standard deviation higher/lower than the average, the values of experience are 1.3 and 0.7, respectively.

Matching Domain (M): Of the 20 employees, 16 work full-time (a value of 1 means 8 hours per workday) and 4 employees work part-time (a value of 0.5 means 4 hours per workday) on the project.

Logic Domain (A): This domain contains the logical order in which the activities are performed, as well as the probability of the activities being carried out.

Skilled Works Domain (W): This domain contains the workloads required by each activity – measured in 8 hours. The SMM matrix of the SPSP is illustrated with Fig. 14.

5.3 Results

After both SPSP and SSPSP are solved by the proposed SynAPS, the TPT and TPC values of the shortest project scenarios are compared. We distinguish cases in which supplementary activities may have been dropped ($\text{TPS} \leq \text{TPS}_{\max}$), and those cases in which these activities had to be carried out ($\text{TPS} = \text{TPS}_{\max}$). Results are presented in Table 10.

TABLE 10. Comparison of the shortest project scenarios
(Source: own table)

Case*	**TPX**	**SPSP**	**SSPSP**	**Difference**	**Feasible**‡**?**
I	TPT	248.47	239.82	3.48%	yes
	TPC	1668.40	1599.10	4.15%	yes
II	TPT	288.95	284.60	1.51%	no
	TPC	3009.80	2901.90	3.58%	no

***I**: $\text{TPS} \leq \text{TPS}_{\max}$, **II**: $\text{TPS} = \text{TPS}_{\max}$
‡**Constraints**: time: 268, cost: 1960.

Based on Table 10, both the time and cost of feasible software development sprints are lower when synergies are considered, however, the difference between the two models' results is not significant (Case I: 3.48% and 4.15%, Case II: 1.51% and 3.58%, respectively). Note that in projects implemented by real cross-functional teams, ignoring synergies may have a much greater impact on the quality of project plans. Scenarios that contain every supplementary task are infeasible under the predetermined constraints.

FIGURE 14. SMM of the practical example

(Source: own figure)

	Synergy Domain (**Y**)																				Skill Domain (**S**)												Matching Domain (**M**)												
	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11	e12	e13	e14	e15	e16	e17	e18	e19	e20	σ1	σ2	σ3	σ4	σ5	σ6	σ7	σ8	σ9	σ10	σ11	σ12	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12	
e1		1.3	1.3	0.85	1.3	0.7	1.3	0.7	0.85	1.3	1.3	1.3	1.3	0.7	0.85	0.85	1.3	0.7	0.85	0.85		1.3	1	1	1.3						0.7	1		1	1	1	1						1	1	e1
e2			0.85	1.3	0.85	1.3	1.3	1.3	1.3	0.85	0.85	0.85	0.85	1.3	1.3	1.3	0.85	1.3	1.3	1.3		1.3	1.15	1	0.85									0.5	0.5	0.5	0.5								e2
e3				1.3	0.85	1.3	1.3	1.3	0.7	0.7	0.7	0.7	0.7	1.3	1.3	1.3	0.7	1.3	1.3	1.3		1.3	1.3	1.15	1.15									1	1	1	1								e3
e4					1.3	0.85	1.3	0.85	0.85	1.3	1.3	1.3	1.3	0.85	0.85	0.85	1.3	0.85	0.85	0.85		0.7				0.85					1	1.3		1				1					1	1	e4
e5						1.3	1.3	1.3	1.3	0.85	0.85	0.85	0.85	1.3	1.3	1.3	0.85	1.3	1.3	1.3	0.85	1				1	0.85	0.85					1	1				1	1	1					e5
e6							1.3	0.7	0.85	1.3	1.3	1.3	1.3	0.7	0.85	0.85	1.3	0.7	0.85	0.85			0.7			0.7					1.3				1			1					1		e6
e7								1.3	1.3	1.3	1.3	1.3	1.3	1.3	1.3	0.7	1.3	1.3	1.3	1.3		0.85	1	0.7		0.85								1	1	1		1							e7
e8									0.85	1.3	1.3	1.3	1.3	0.7	0.85	0.85	1.3	0.7	0.85	0.85	1.3	0.85		0.7			0.85	0.85				0.85	0.5	0.5		0.5			0.5	0.5				1	e8
e9										0.7	0.7	0.7	0.7	0.85	0.85	0.85	0.7	0.85	0.85	0.85			1.3	1					0.7						1	1					1				e9
e10											0.7	0.7	0.7	1.3	1.3	1.3	0.7	1.3	1.3	1.3		0.85			0.85				0.85					1			1				1				e10
e11												0.7	0.7	1.3	1.3	1.3	0.7	1.3	1.3	1.3			0.7							0.85	1				1							1	1		e11
e12													0.7	1.3	1.3	1.3	0.7	1.3	1.3	1.3		1	1	1	1.15	1								1	1	1	1	1							e12
e13														1.3	1.3	1.3	0.7	1.3	1.3	1.3										1	0.85											1	1		e13
e14															0.85	0.85	1.3	0.7	0.85	0.85	0.85	1.3	0.85	1									1	1	1	1									e14
e15																0.85	1.3	0.85	0.85	0.7		0.85	1.3	0.85	0.7									1	1	1	1								e15
e16																	1.3	0.85	0.85	0.85	1.15			1.3	1								0.5			0.5	0.5								e16
e17																		1.3	1.3	1.3		0.85	0.85		1		1.3	1.15	1.3	1.3				1	1		1		1	1	1	1			e17
e18																			0.85	0.85							1	1.15	1.15	1									1	1	1	1			e18
e19																				0.85		1	1			1								0.5	0.5			0.5							e19
e20																						1		1		1.3								1		1		1							e20
a1	Translation of requirements																				20												1	1	1	1	1								a1
a2	Function model 1																					80												1				1	1						a2
a3	Function model 2																						110												0.6			1	1						a3
a4	Function model 3																							40												1		1	1						a4
a5	Function model 4																								30												0.8	1	1						a5
a6	Flash loader development																									8												1		1					a6
a7	Software integration																										4												1	1					a7
a8	Integration test																											5												1	1	1			a8
a9	Laboratory test																												15												1		1		a9
a10	Road test																													10												0.6	1		a10
a11	Quality control																														4												1	1	a11
a12	Software handover																															6												1	a12
	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11	e12	e13	e14	e15	e16	e17	e18	e19	e20	σ1	σ2	σ3	σ4	σ5	σ6	σ7	σ8	σ9	σ10	σ11	σ12	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12	
	Output Domain (**O**)																				Skilled Works Domain (**W**)												Logic Domain (**A**)												

CHAPTER 6

Threats to Validity

Internal validity threats in this work can occur due to the randomness of the results obtained from the simulation and SynAPS, as well as a lack of treatment of several variables such as synergy structures for the optimization. To avoid such a threat, different actions were taken:

- First and foremost, the number of generations, elite count, crossover fraction, mutation rate, and population size needed by SynAPS were carefully calibrated. The chosen values were determined ensuring that further changes do not significantly affect the results. Hyperparameters were then used where the convergence was best.
- Similarly, the number of iterations required by the entire approach was calibrated. As described in detail in Section 3.2, further increases over 50 iterations do not produce improvements in the applied fitness function; nevertheless, the maximum iterations are specified to 100 (see Section 3.2).
- To avoid the effect of randomness on the results, GAs were executed 40 times, and I verified that the obtained fitness function value at the last stage does not change among the iterations.
- Last but not least, Nelder-Mead optimization was used to refine the continuous part of the chromosome.

Regarding the *external validity*, the proposed approach and the obtained results can be extended to non-IT project structures. I applied CR_1-CR_2 selection criterias (see Section 3.3.1.1) to select project structures that are specific for the IT projects merely because flexible approaches are still only widely used in IT projects. However, with the proliferation of flexible approaches, this study may also be interesting for projects with different structures.

Construct validity threats may be due to the simplifications of the software project process. To mitigate this threat, all small social network structures were explored, which can be reviewed in the literature. Software projects are generated by the iMOPSE generator (Myszkowski et al., 2019). The selection criteria (see CR_1 and CR_2 in Section 3.3.1.1) were then followed. Therefore, considering the available literature regarding the structure of the IT projects, the generated project structure characterizes the features of an IT project.

To improve the *conclusion validity*, the optimization results are analyzed by a highly robust method, the so-called regression tree ensemble model of the MATLAB Regression Learner App (MathWorks, 2019b). During the calculation, 10-fold cross-validation was used, and hyperparameters were tuned by Bayesian optimization.[50] In addition, large-scale simulation increases the validity of the conclusion.

---

[50] The details of the optimization processes can be found in Appendix A.3.

CHAPTER 7

Summary and Conclusion

7.1 Summary

While cross-validation of solvers and other technical aspects of the software project scheduling problem (SPSP) have been extensively explored in the literature, significantly fewer studies consider the definition of the problem itself. This study was focused on two possible approaches of extending the classical SPSP. First, a general form of the SPSP assumes fixed logic plans; however, applying flexible dependencies and using task priorities instead of fixed occurrences will result in more flexible project plans consistent with the agile approach. Despite the existence of agile project scheduling algorithms (see, e.g., Kosztyán, 2015), to date SPSP has not yet been extended to incorporate this feature. Second, while software development projects and particularly those that are software development projects using the agile approach (Wysocki, 2011, 2019) place a greater emphasis on teamwork than the traditional methods (Nerur et al., 2005), in SPSP, employees are regarded as independent resources. This by definition assumes that the best (i.e., the most skilled) workers will perform tasks within the shortest timespan and with the highest quality; however, none of the extensions address the interdependence of resources.

In this dissertation, the classical SPSP was formulated in a flexible, multi-domain model and it was extended with pairwise synergies that can influence the employees' performance during project implementation. Using simulations based on the new approach, I searched for project indicators that have the largest influence on changes in project costs. The main results of the analysis were as follows. Based on the proposed simple model, (1) the costs of projects are extremely sensitive to the interdependencies of resources; (2) synergy networks with low degree centrality significantly reduce the project costs, and (3) synergy networks with full graph topology are most sensitive to unfavorable synergies (e.g., conflicts). Since the impact of positive or negative pairwise synergies and the structure of sociometric networks can also be modeled, the proposed method can be a novel element in risk analysis tools, particularly in the context of human resources-critical projects. The research questions (**RQ**s) and assumptions (**RA**s), as well as the theses (**RT**s) formulated for each are summarized in Table 11.

7.2 Conclusion

This section concludes the dissertation with a view to its contribution to the literature and practical implications.

7.2.1 Contribution to the Literature

While most of the software projects are managed in an agile framework (see, e.g., Wysocki 2011, 2019), the SPSP ignores the two main features of this approach: the flexibility of planning and the complexity of teamwork. To make the SPSP more realistic and practical, the present dissertation offers a solution to both shortcomings. First, the SPSP was formulated in a multi-domain, matrix-based structure that allows flexible project planning. Then, to model the effect of employee interdependencies on the common performance and, consequently, on the outcome of the project, the SPSP was extended with pairwise synergies. Since the dissertation provides a general framework for modeling different sources of synergies – such as the formal structure of the team, communication between team members, team roles, and shared knowledge or experience –, it may prove suitable for bridging the gap between human-centered and methodological research.

TABLE 11. Research questions, assumptions and theses

(Source: own table)

N*	Description
$\mathbf{RQ}_1$	Is it possible to determine a scheduling problem for traditional and flexible project planning environments that considers not only the skills of human resources but also the synergies between them?
$\mathbf{RA}_1$	The classical software project scheduling problem can be extended by considering flexible task dependencies and synergies between resources. **(Verified)**
$\mathbf{RT}_1$	The proposed synergy-based software scheduling problem (SSPSP) extends the classical software scheduling problem (SPSP) to take into account the flexibility of project plans, as well as the pairwise synergies between resources.
$\mathbf{RQ}_2$	Is it possible to develop a network- or matrix-based project scheduling model that takes into account the flexibility of project plans, the skills of human resources as well as the synergies between them?
$\mathbf{RA}_2$	The multi-domain matrix (MDM) can be specified to a flexible multi-domain matrix whose interconnected domains model the flexible project plan, the skills of human resources as well as the synergies between them. **(Verified)**
$\mathbf{RT}_2$	The proposed synergy-based multi-domain matrix (SMM) contains multiple interconnected domains that model the flexible logical structure of the project, the amount of (skilled) work to be performed within the project, the skills of human resources and the positive and negative pairwise synergies between them, as well as the maximum resource assignments. The proposed matrix is able to model all solutions of both the classical (SPSP) and the synergy-based (SSPSP) software scheduling problems.
$\mathbf{RQ}_3$	Is there a(n optimal) solution for scheduling a flexible software project plan that considers the synergies between resources?
$\mathbf{RA}_3$	Using metaheuristic algorithms, it is possible to find a feasible solution to the project scheduling problem that takes into account flexible task dependencies and synergies between resources. **(Verified)**
$\mathbf{RT}_3$	The proposed synergy-based agile project scheduling algorithm (SynAPS) finds a feasible solution for both the classical (SPSP) and the synergy-based (SSPSP) software project scheduling problems with respect to the given objective function (that minimizes the duration and cost of the project while simultaneously maximizing its score) and given constraints (in relation to the duration, cost, resource, and score of the project).
$\mathbf{RQ}_4$	Is it possible to develop a simulation framework to examine the impact of the synergies between resources, the structures of synergy networks, the skills of human resources as well as the size, flexibility, and constraints of the project on the implementation of the project schedule?
$\mathbf{RA}_4$	By supplementing existing or generated project databases with flexible task dependencies and resource synergies, it is possible to create a simulation environment to examine the impact of human resource synergies and skills, as well as project size, flexibility, and constraints, on project feasibility. **(Verified)**
$\mathbf{RT}_4$	The proposed synergy-based agile simulation framework (SynASF) is suitable for examining the impact of pairwise synergies between resources, synergy structures, skills as well as the size, flexibility, and constraints of the project on the implementation of project scheduling. According to the synergy-based agile simulation framework (SynASF): $\mathbf{RT}_{4.1}$: The costs of projects are most sensitive to the pairwise synergies of human resources; $\mathbf{RT}_{4.2}$: The impact of pairwise synergies on project costs is mainly influenced by the size of the project, the average pairwise synergy, and the structural parameter (degree centrality) of the synergy network; $\mathbf{RT}_{4.3}$: Synergy networks with low degree centrality lead to a greater reduction in the project cost; however, the impact of synergies is also influenced by the topology of networks. The highest costs are obtained by the synergy networks with the most decentralized topology (full graph) because of their high sensitivity to negative synergies.

***Notations:** $\mathbf{RQ}$: research question, $\mathbf{RA}$: research assumption, $\mathbf{RT}$: research thesis.

In order to facilitate the comparison of the proposed model with the literature (see Table 3-5 in Section 2.3), a summary of its features is hereby presented. Like most models presented in related studies, it was formulated based on Alba and Chicano (2007) and Luna et al. (2014); however the new model has a matrix-based, multi-domain structure and allows for the consideration of pairwise synergies of human resources in scheduling and resource allocation. To address flexible planning and employee interdependencies, the original model was complemented with five new constraints (see C_4-C_8 in Section 3.1.4). The new model has a single composite target function (see Eq. (35) in Section 3.1.5) that minimizes the duration and cost of the project, while simultaneously maximizing its score. To find a feasible good solution with respect to this target function and given constraints, a hybrid genetic algorithm called SynAPS was proposed that combines a mixed-integer genetic algorithm with the Nelder-Mead minimization method (see Section 3.2).

7.2.2 Practical Implications

As a planning and decision-supporting tool, the proposed method may be particularly beneficial for software development companies that adopt the agile project management (APM) approach and already have the expertise and technical background to solve a complex software project scheduling problem (SPSP). In contrast to other approaches found in the literature, the new multi-domain method supports flexible project planning, and provides an opportunity to model employee interdependencies by introducing the concept of pairwise synergies. As it is not limited to one source of synergy, it can be used to model the impact of different synergy sources – such as the formal structure of the team, communication between team members, team roles, and shared knowledge or experience – on the implementation of the project schedule, depending on the available data and the characteristics of the projects managed by the company.

CHAPTER 8

Limitations and Future Research

To simplify the model, the performed simulations were based on only pairwise synergies; nevertheless, I believe that the importance of human resource interdependencies may motivate researchers to explore this aspect in greater detail and test presented statements in practice. The presented simple model disregards several important human factors that could affect the results (e.g., employees may prefer working in groups with a decentralized sociometric structure). In this work, I focused on single projects; however, such software projects are usually pursued in a multi-project environment. Therefore, my next study will address the impacts of synergy effects in software projects in a multi-project environment.

Bibliography

Ahuja, M. K., Galletta, D. F., and Carley, K. M. (2003). Individual centrality and performance in virtual r&d groups: An empirical study. *Management Science*, 49(1):21–38.

Alba, E. and Chicano, J. F. (2007). Software project management with {GAs}. *Information Sciences*, 177(11):2380–2401.

Allen, N. and West, M. (2005). Selecting for teamwork. *The Blackwell handbook of personnel selection*, pages 476–494.

Almeida, B. F., Correia, I., and Saldanha-da Gama, F. (2016). Priority-based heuristics for the multi-skill resource constrained project scheduling problem. *Expert Systems with Applications*, 57:91–103.

Alreffaee, T. R. and Alabajee, M. A.-A. (2020). Solving software project scheduling problem using whale optimization algorithm. In *IOP Conference Series: Materials Science and Engineering*, volume 928, page 032084. IOP Publishing.

Andersen, E. S. (2006). Milestone planning—a different planning approach. In *PMI® Global Congress*.

Andersen, E. S., Grude, K. V., and Haug, T. (2009). *Goal directed project management: effective techniques and strategies*. Kogan Page Publishers.

Anop, S. and Aldaghi, S. (2010). Management of communication risks: Evidencefrom international projects case study. In *International conference PM-05-Advancing project management for the 21st century "Concepts, tools & techniques for managing successful projects", Heraklion, Crete, Greece.*

Antill, J. and Woodhead, R. (1990). *Critical Path Methods in Construction Practice*. Wiley-Interscience publication. Wiley.

Antoniol, G., Di Penta, M., and Harman, M. (2004). Search-based techniques for optimizing software project resource allocation. In *Genetic and Evolutionary Computation Conference*, pages 1425–1426. Springer.

Archibald, R. (2003). *Managing High-Technology Programs and Projects*. Wiley.

Artigues, C., Demassey, S., and Neron, E. (2008). *Resource-constrained project scheduling*. Wiley Online Library.

Atkinson, R. (1999). Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. *International journal of project management*, 17(6):337–342.

Bannerman, P. L. and Thorogood, A. (2012). Celebrating it projects success: a multi-domain analysis. In *2012 45th Hawaii International Conference on System Sciences*, pages 4874–4883. IEEE.

Barry, B. and Stewart, G. L. (1997). Composition, process, and performance in self-managed groups: the role of personality. *Journal of Applied Psychology*, 82(6):62–78.

Belbin, R. (1981). *Management Teams: Why They Succeed Or Fail*. Elsevier Science & Technology Books.

Belbin, R. (1993). Roles at work: a strategy for human resource management.

Belbin, R. (2010). *Management Teams: Why They Succeed Or Fail*. Butterworth-Heinemann.

Belbin, R. (2014). Method, reliability & validity, statistics & research: A comprehensive review of belbin team roles. *Belbin UK*, pages 1–26.

Bell, S. T. (2007). Deep-level composition variables as predictors of team performance: a meta-analysis. *Journal of applied psychology*, 92(3):595.

Benington, H. D. (1983). Production of large computer programs. *Annals of the History of Computing*, 5(4):350–361.

Boehm, B. and Turner, R. (2005). Management challenges to implementing agile processes in traditional development organizations. *IEEE software*, 22(5):30–39.

Boehm, B. W., Clark, Horowitz, Brown, Reifer, Chulani, Madachy, R., and Steece, B. (2000). *Software Cost Estimation with Cocomo II with Cdrom*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition.

Branco, D. T. M. C., de Oliveira, E. C. C., Galvão, L., Prikladnicki, R., and Conte, T. (2015). An empirical study about the influence of project manager personality in software project effort. In *ICEIS (2)*, pages 102–113.

Brennan, M. (1968). Pert and cpm: a selected bibliography. *Council of Planning Librarians*.

Brown, C. (2020). *Networks and Economic Behavior: Theory and Experiments*. PhD thesis, The Florida State University.

Browning, T. R. (2014). Managing complex project process models with a process architecture framework. *International Journal of Project Management*, 32(2):229–241.

Callahan, M. T., Quackenbush, D. G., and Rowings, J. E. (1992). *Construction project scheduling*.

Campion, M. A., Campion, J. E., and Hudson Jr, J. P. (1994). Structured interviewing: A note on incremental validity and alternative question types. *Journal of Applied Psychology*, 79(6):998.

Carbonell, P. and Rodríguez Escudero, A. I. (2019). The dark side of team social cohesion in npd team boundary spanning. *Journal of Product Innovation Management*, 36(2):149–171.

Chan, W. K. V. (2013). *Theory and applications of Monte Carlo simulations*. BoD–Books on Demand.

Chang, C. K., Christensen, M. J., and Zhang, T. (2001). Genetic algorithms for project management. *Annals of Software Engineering*, 11(1):107–139.

Chang, C. K., Jiang, H.-y., Di, Y., Zhu, D., and Ge, Y. (2008). Time-line based model for software project scheduling with genetic algorithms. *Inf. Softw. Technol.*, 50(11):1142–1154.

Chen, C.-H., Ling, S. F., and Chen, W. (2003). Project scheduling for collaborative product development using {DSM}. *International Journal of Project Management*, 21(4):291–299.

Chen, T. R., Shore, D. B., Zaccaro, S. J., Dalal, R. S., Tetrick, L. E., and Gorab, A. K. (2014). An organizational psychology perspective to examining computer security incident response teams. *IEEE Security & Privacy*, 12(5):61–67.

Chen, W.-N. and Zhang, J. (2013). Ant colony optimization for software project scheduling and staffing with an event-based scheduler. *IEEE Transactions on Software Engineering*, 39(1):1–17.

Cheng, H. and Chu, X. (2012). Task assignment with multiskilled employees and multiple modes for product development projects. *The International Journal of Advanced Manufacturing Technology*, 61(1):391–403.

Cheng, J., Ji, J., Guo, Y.-n., and Ji, J. (2019). Dynamic multiobjective software project scheduling optimization method based on firework algorithm. *Mathematical Problems in Engineering*, 2019.

Chicano, F., Cervantes, A., Luna, F., and Recio, G. (2012). A novel multiobjective formulation of the robust software project scheduling problem. In *European Conference on the Applications of Evolutionary Computation*, pages 497–507. Springer.

Chicano, F., Luna, F., Nebro, A. J., and Alba, E. (2011). Using multi-objective metaheuristics to solve the software project scheduling problem. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pages 1915–1922, New York, NY, USA. ACM.

Chiocchio, F., Kelloway, E. K., and Hobbs, B. (2015). *The psychology and management of project teams*. Oxford University Press.

Cho, S.-H. and Eppinger, S. D. (2005). A simulation-based process model for managing complex design projects. *IEEE Transactions on engineering management*, 52(3):316–328.

Cicmil, S., Williams, T., Thomas, J., and Hodgson, D. (2006). Rethinking project management: researching the actuality of projects. *International journal of project management*, 24(8):675–686.

Cleland, D. and Kerzner, H. (1985). *A Project Management Dictionary of Terms*. Van Nostrand Reinhold.

Cleland, D. and King, W. (1983). *Project Management Handbook*. Van Nostrand Reinhold Company.

Cleland, D. I. (2007). *Project management: strategic design and implementation*. McGraw-Hill Education.

Coello, C. A. C., Lamont, G. B., and Veldhuizen, D. A. V. (2006). *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Cooke-Davies, T. J., Crawford, L. H., and Lechler, T. G. (2009). Project management systems: Moving project management from an operational to a strategic discipline. *Project Management Journal*, 40(1):110–123.

Costa Jr, P. T. and McCrae, R. R. (2008). *The Revised NEO Personality Inventory (NEO-PI-R)*. Sage Publications, Inc.

Cram, W. A. and Marabelli, M. (2018). Have your cake and eat it too? simultaneously pursuing the knowledge-sharing benefits of agile and traditional development approaches. *Information & Management*, 55(3):322–339.

Crawford, B., Soto, R., Astorga, G., and Olguín, E. (2016a). An alternative solution to the software project scheduling problem. In *Artificial Intelligence Perspectives in Intelligent Systems*, pages 501–510. Springer.

Crawford, B., Soto, R., Johnson, F., Monfroy, E., and Paredes, F. (2014). A max-min ant system algorithm to solve the software project scheduling problem. *Expert Syst. Appl.*, 41(15):6634–6645.

Crawford, B., Soto, R., Johnson, F., Valencia, C., and Paredes, F. (2016b). Firefly algorithm to solve a project scheduling problem. In *Artificial Intelligence*

Perspectives in Intelligent Systems, pages 449–458. Springer.

Crowston, K. and Kammerer, E. E. (1998). Coordination and collective mind in software requirements development. *IBM Systems Journal*, 37(2):227–245.

Cummings, J. N. and Cross, R. (2003). Structural properties of work groups and their consequences for performance. *Social Networks*, 25(3):197 – 210.

da Silva, G. F., Silva, L., and Britto, A. (2020). Dynamic software project scheduling problem with pso and dynamic strategies based on memory. In *Brazilian Conference on Intelligent Systems*, pages 79–94. Springer.

Dalcher, D. J. (2009). Aipm book series & research at the ncpm. In *PMUni conference, Vienna*.

Danilovic, M. and Browning, T. R. (2007). Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management*, 25(3):300–314.

de Andrade, J., Silva, L., Britto, A., and Amaral, R. (2019). Solving the software project scheduling problem with hyper-heuristics. In *International Conference on Artificial Intelligence and Soft Computing*, pages 399–411. Springer.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley Interscience Series in Systems and Optimization. Wiley.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Demeulemeester, E., Vanhoucke, M., and Herroelen, W. (2003). Rangen: A random network generator for activity-on-the-node networks. *Journal of Scheduling*, 6(1):17–38.

Deubzer, F., Lindemann, U., et al. (2008). Functional modelling for design synthesis using mdm methodology. In *DSM 2008: Proceedings of the 10th International DSM Conference, Stockholm, Sweden, 11.-12.11. 2008*, pages 403–411.

Devaux, S. A. (1999). *Total project control: A manager's guide to integrated project planning, measuring, and tracking*. Wiley New York, NY.

Di Penta, M., Harman, M., and Antoniol, G. (2011). The use of search-based optimization techniques to schedule and staff software projects: an approach and an empirical study. *Software: Practice and Experience*, 41(5):495–519.

Ding, R. and Jing, X. (2003). Five principles of project management in software companies. *Project Management Technology*, 1.

Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, 85(6):1213 – 1221. Special Issue: Agile Development.

Driskell, T., Driskell, J. E., Burke, C. S., and Salas, E. (2017). Team roles: A review and integration. *Small Group Research*, 48(4):482–511.

Duggan, J., Byrne, J., and Lyons, G. J. (2004). A task allocation optimizer for software construction. *IEEE software*, 21(3):76–82.

Dupuy, G., Stark, N., and Salto, C. (2013). Algoritmo evolutivo para el problema de planificación en proyectos de desarrollo de software. In *XVIII Congreso Argentino de Ciencias de la Computación*.

Dwyer, J., Stanton, P., and Thiessen, V. (2004). Why project management. *Project management in health and community services: getting good ideas to work*, pages 3–22.

Dybå, T. and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and software technology*, 50(9-10):833–859.

Dzvonyar, D., Alperowitz, L., Henze, D., and Bruegge, B. (2018). Team composition in software engineering project courses. In *2018 IEEE/ACM International Workshop on Software Engineering Education for Millennials (SEEM)*, pages 16–23. IEEE.

Espinosa, J. A., Slaughter, S. A., Kraut, R. E., and Herbsleb, J. D. (2007). Familiarity, complexity, and team performance in geographically distributed software development. *Organization science*, 18(4):613–630.

Fahmy, A. M. (2016). Optimization algorithms in project scheduling‖. *Optimization Algorithms-Methods and Applications*.

Fernandes, F. L. M., S. F. Q. B. d. (2007). Relationship between personal competences and project management personality types (in portuguese). In *Proceedings of the 2º Congresso Brasileiro de Gerenciamento de Projetos. Salvador, Bahia.*

Fernandez-Viagas, V. and Framinan, J. M. (2014). Integrated project scheduling and staff assignment with controllable processing times. *The Scientific World Journal*, 2014.

Fowler, M., Highsmith, J., et al. (2001). The agile manifesto. *Software Development*, 9(8):28–35.

Furnham, A. (1990). The fakeability of the 16 pf, myers-briggs and firo-b personality measures. *Personality and Individual Differences*, 11(7):711–716.

Furnham, A. (1996). The firo-b, the learning style questionnaire, and the five-factor model. *Journal of Social Behavior and Personality*, 11(2):285.

García-Nájera, A. and del Carmen Gómez-Fuentes, M. (2014). A multi-objective genetic algorithm for the software project scheduling problem. In *Mexican International Conference on Artificial Intelligence*, pages 13–24. Springer.

Goldberg, L. R. (2013). An alternative "description of personality": The big-five factor structure. *Personality and Personality Disorders: The Science of Mental Health*, 7:34.

Gonsalves, T. and Itoh, K. (2010). Multi-objective optimization for software development projects. In *Lecture Notes in Engineering and Computer Science: International Multiconference of Engineers and Computer Scientist 2010*, pages 1–6.

Gorbea, C., Spielmannleitner, T., Lindemann, U., and Fricke, E. (2008). Analysis of hybrid vehicle architectures using multiple domain matrices. In *DSM 2008:*

Proceedings of the 10th International DSM Conference, Stockholm, Sweden, 11.-12.11. 2008.

Görög, M. (1999). *Bevezetés a projektmenedzsmentbe*. Aula Kiadó, Budapest.

Görög, M. (2003). *A projektvezetés mestersége*. Aula.

Görög, M. (2007). *Általános projektmenedzsment*. Aula Kiadó, Budapest.

Görög, M. and Ternyik, L. (2001). *Informatikai projektek vezetése*. Kossuth.

Graham, R. L., Lawler, E. L., Lenstra, J. K., and Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics*, volume 5, pages 287–326. Elsevier.

Gueorguiev, S., Harman, M., and Antoniol, G. (2009). Software project planning for robustness and completion time in the presence of uncertainty using multi objective search based software engineering. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1673–1680.

Gunawan, I. and Ahsan, K. (2010). Project scheduling improvement using design structure matrix. *International Journal of Project Organisation and Management*, 2(4):311–327.

Guo, Y., Ji, J., Ji, J., Gong, D., Cheng, J., and Shen, X. (2019). Firework-based software project scheduling method considering the learning and forgetting effect. *Soft computing*, 23(13):5019–5034.

Hanne, T. and Nickel, S. (2005). A multiobjective evolutionary algorithm for scheduling and inspection planning in software development projects. *European Journal of Operational Research*, 167(3):663–678.

Harta, P. (2021). Agilis projektcsapat optimális kiválasztásának vizsgálata szoftver projekt környezetben. In *XXXV. Országos Tudományos Diákköri Konferencia Közgazdaságtudományi Szekció*.

Hartmann, S. and Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of operational research*, 207(1):1–14.

Henry, S. M. and Stevens, K. T. (1999). Using belbin's leadership role to improve team effectiveness: An empirical investigation. *Journal of Systems and Software*, 44(3):241–250.

Hinde, D. (2018). *PRINCE2 Study Guide: 2017 Update, Second Edition*. Sybex.

Hogan, R., Raza, S., and Driskell, J. E. (1988). *Personality, Team Performance, and Organizational Context*, pages 93–103. Springer US, Boston, MA.

Hong, P., Doll, W. J., Nahm, A. Y., and Li, X. (2004). Knowledge sharing in integrated product development. *European journal of innovation management*.

Hoopes, D. G. and Postrel, S. (1999). Shared knowledge,"glitches," and product development performance. *Strategic management journal*, 20(9):837–865.

Hsu, S.-C., Weng, K.-W., Cui, Q., and Rand, W. (2016). Understanding the complexity of project team member selection through agent-based modeling. *International Journal of Project Management*, 34(1):82 – 93.

Hunter, J. E., Schmidt, F. L., and Judiesch, M. K. (1990). Individual differences in output variability as a function of job complexity. *Journal of Applied Psychology*, 75(1):28–42.

Icmeli, O., Erenguc, S. S., and Zappe, C. J. (1993). Project scheduling problems: a survey. *International Journal of Operations & Production Management*.

Iivari, J., Hirschheim, R., and Klein, H. K. (2000). A dynamic framework for classifying information systems development methodologies and approaches. *Journal of management information systems*, 17(3):179–218.

Institute, P. (2017). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)–Sixth Edition*. PMBOK® Guide. Project Management Institute.

Institute, P. M. (2011). Practice standard for scheduling. Project Management Institute.

Institute, P. M. (2012). Pmi lexicon of project management terms. Project Management Institute.

Introna, L. D. and Whitley, E. A. (1997). Against method-ism. *Information Technology & People*.

Iriarte, C. and Bayona, S. (2020). It projects success factors: a literature review. *SciKA-Association for Promotion and Dissemination of Scientific Knowledge*.

Islam, M. R., Islam, M. S., and Sakeef, N. (2019). Rna secondary structure prediction with pseudoknots using chemical reaction optimization algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Jalote, P. and Vishal, B. (2003). Optimal resource allocation for the quality control process. In *14th International Symposium on Software Reliability Engineering, 2003. ISSRE 2003.*, pages 26–33. IEEE.

Jamieson, A. and Morris, P. W. (2007). Moving from corporate strategy to project strategy. In *The Wiley Guide to Project, Program, and Portfolio Management*, volume 3, pages 34–62. John Wiley & Sons Hoboken, NJ.

Jin, N. and Yao, X. (2014). Heuristic optimization for software project management with impacts of team efficiency. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 3016–3023. IEEE.

Källo, R., Eerme, M., and Reedik, V. (2013). Ways of increasing synergy in automated factory design and commissioning teamwork. *Journal of Materials Science and Engineering. B*, 3(9B):597.

Karabeleski, S. and Avdic, D. (2018). Pairing belbin's team roles towards productivity.

Klein, R. (2012). *Scheduling of resource-constrained projects*, volume 10. Springer Science & Business Media.

Klimoski, R. and Jones, R. G. (1995). Staffing for effective group decision making: Key issues in matching people and teams. *Team effectiveness and decision making in organizations*, 29:1–332.

Knowles, J. D. and Corne, D. W. (2000). Approximating the nondominated front using the pareto archived evolution strategy. *Evol. Comput.*, 8(2):149–172.

Kolisch, R. and Sprecher, A. (1997). {PSPLIB} - a project scheduling problem library: {OR} software - {ORSEP} operations research software exchange program. *European Journal of Operational Research*, 96(1):205 – 216.

Konak, A., Kulturel-Konak, S., and Cheung, G. W. (2019). Teamwork attitudes, interest and self-efficacy between online and face-to-face information technology students. *Team Performance Management: An International Journal*.

Kosztyán, Z. T. (2015). Exact algorithm for matrix-based project planning problems. *Expert Systems with Applications*, 42(9):4460 – 4473.

Kosztyán, Z. T. (2020). An exact algorithm for the flexible multilevel project scheduling problem. *Expert Systems with Applications*, 158:113485.

Kosztyán, Z. T., Fejes, J., and Judit, K. (2008). Sztohisztikus hálóstruktúrák kezelése projektütemezési feladatokban. *SZIGMA Matematikai-közgazdasági folyóirat*, 39(1-2):87–105.

Kosztyán, Z. T., Jakab, R., Novák, G., and Hegedűs, C. (2020). Survive it! survival analysis of it project planning approaches. *Operations Research Perspectives*, 7:100170.

Kosztyán, Z. T. and Kiss, J. (2010). Stochastic network planning method. In *Advanced techniques in computing sciences and software engineering*, pages 263–268. Springer.

Kosztyán, Z. T., Kiss, J., et al. (2010). Pem–a new matrix method for supporting the logic planning ofsoftware development projects. In *DSM 2010: Proceedings of the 12th International DSM Conference, Cambridge, UK, 22.-23.07. 2010*, pages 97–110.

Kosztyán, Z. T. and Kurbucz, M. T. (2015). Projektek nyomon követése mátrixokkal= comprehensive planning and coordinating matrix. *Taylor*, 7(1-2):144–151.

Kosztyán, Z. T., Pribojszki-Németh, A., and Szalkai, I. (2019). Hybrid multimode resource-constrained maintenance project scheduling problem. *Operations Research Perspectives*, 6:100129.

Kosztyán, Z. T. and Szalkai, I. (2018). Hybrid time-quality-cost trade-off problems. *Operations Research Perspectives*, 5:306–318.

Kosztyán, Z. T. and Szalkai, I. (2020). Multimode resource-constrained project scheduling in flexible projects. *Journal of Global Optimization*, 76(1):211–241.

Kumar, A. and Ganesh, L. (1998). Use of petri nets for resource allocation in projects. *IEEE Transactions on Engineering Management*, 45(1):49–56.

Kumar, R. L. (2002). Managing risks in it projects: an options perspective. *Information & management*, 40(1):63–74.

Kurbucz, M. T. (2013). Emberi erőforrások optimális kiválasztásának vizsgálata a projekttervezésben. *E-CONOM*, 2(2):58–78.

Kurbucz, M. T. (2016). Projektek átfogó tervezésének és koordinálásának támogatása mátrixokkal= comprehensive planning and coordinating by matrix-based methods. *E-CONOM*, 5(1):148–160.

Laszczyk, M. and Myszkowski, P. B. (2019). Improved selection in evolutionary multi–objective optimization of multi–skill resource–constrained project scheduling problem. *Information Sciences*, 481:412–431.

Lederer, A. L. and Prasad, J. (1993). Information systems software cost estimating: a current assessment. *Journal of information technology*, 8(1):22–33.

Lewis James, P. (1997). Fundamentals of project management. *American Management Association*.

Leybourne, S. A. (2007). The changing bias of project management research: a consideration of the literatures and an application of extant theory. *Project Management Journal*, 38(1):61–73.

Li, H., Guo, Q.-L., Wang, Q., and Xi, L.-F. (2015). Algorithms to test open set condition for self-similar set related to pv numbers. *Journal of Mathematical*

Analysis and Applications, 421(1):453–473.

Liemhetcharat, S. and Veloso, M. (2012). Modeling and learning synergy for team formation with heterogeneous agents. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 365–374.

Liemhetcharat, S. and Veloso, M. (2014). Weighted synergy graphs for effective team formation with heterogeneous ad hoc agents. *Artificial Intelligence*, 208:41–65.

Lindsjørn, Y., Sjøberg, D. I., Dingsøyr, T., Bergersen, G. R., and Dybå, T. (2016). Teamwork quality and project success in software development: A survey of agile development teams. *Journal of Systems and Software*, 122:274–286.

Luna, F., González-Álvarez, D. L., Chicano, F., and Vega-Rodríguez, M. A. (2011). On the scalability of multi-objective metaheuristics for the software scheduling problem. In *2011 11th International Conference on Intelligent Systems Design and Applications*, pages 1110–1115. IEEE.

Luna, F., González-Álvarez, D. L., Chicano, F., and Vega-Rodríguez, M. A. (2014). The software project scheduling problem: A scalability analysis of multi-objective metaheuristics. *Appl. Soft Comput.*, 15:136–148.

Maheswari, J. U. and Varghese, K. (2005). Project scheduling using dependency structure matrix. *International Journal of Project Management*, 23(3):223–230.

Mateo, J. R. S. C. (2016). *Management Science, Operations Research and Project Management: Modelling, Evaluation, Scheduling, Monitoring*. Routledge.

MathWorks (2019a). Predictor importance. URL: https://www.mathworks.com/help/stats/compactregressionensemble.predictorimportance.html. [Accessed: 25 August 2020].

MathWorks (2019b). Regression learner app. URL: https://www.mathworks.com/help/stats/regression-learner-app.html. [Accessed: 25 August 2020].

Mears, P. and Voehl, F. (1994). *Team building: a structured learning approach*. CRC Press.

Mehler, M. (1991). Reining in runaway systems. *Information Week*, pages 20–24.

Melo, F. S. and Sardinha, A. (2016). Ad hoc teamwork by learning teammates' task. *Autonomous Agents and Multi-Agent Systems*, 30(2):175–219.

Minku, L. L., Sudholt, D., and Yao, X. (2012). Evolutionary algorithms for the project scheduling problem: runtime analysis and improved design. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 1221–1228.

Minku, L. L., Sudholt, D., and Yao, X. (2013). Improved evolutionary algorithm design for the project scheduling problem based on runtime analysis. *IEEE Transactions on Software Engineering*, 40(1):83–102.

Mohammadi, M., Sajadi, S. M., and Tavakoli, M. M. (2014). Scheduling new product development projects using simulation-based dependency structure matrix. *International Journal of Logistics Systems and Management*, 19(3):311–328.

Moos, R. H. and Speisman, J. C. (1962). Group compatibility and productivity. *The Journal of Abnormal and Social Psychology*, 65(3):190.

Moreno, J. L. (1960). *The Sociometry Reader*. Glencoe, Illinois: The Free Press.

Mote, J. E. (2005). R&d ecology: using 2-mode network analysis to explore complexity in r&d environments. *Journal of Engineering and Technology Management*, 22(1):93 – 111. Research on Social Networks and the Organization of Research and Development.

Mubarak, S. (2019). *Construction Project Scheduling and Control*. Wiley.

Muchinsky, P. (2006). *Psychology Applied to Work: An Introduction to Industrial and Organizational Psychology*. Thomson/Wadsworth.

Musiał, K., Kazienko, P., and Brodka, P. (2009). User position measures in social networks. In *Proceedings of the 3rd workshop on social network mining and analysis*, pages 1–9.

Myszkowski, P. B., Laszczyk, M., and Lichodij, J. (2017). Efficient selection operators in nsga-ii for solving bi-objective multi-skill resource-constrained project scheduling problem. In *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 83–86. IEEE.

Myszkowski, P. B., Laszczyk, M., Nikulin, I., and Skowroński, M. (2019). Imopse: a library for bicriteria optimization in multi-skill resource-constrained project scheduling problem. *Soft Computing*, 23(10):3397–3410.

Myszkowski, P. B., Skowroński, M. E., Olech, Ł. P., and Oślizło, K. (2015a). Hybrid ant colony optimization in solving multi-skill resource-constrained project scheduling problem. *Soft Computing*, 19(12):3599–3619.

Myszkowski, P. B., Skowroński, M. E., and Sikora, K. (2015b). A new benchmark dataset for multi-skill resource-constrained project scheduling problem. In *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 129–138. IEEE.

Nan, N. and Harter, D. E. (2009). Impact of budget and schedule pressure on software development cycle time and effort. *IEEE Transactions on Software Engineering*, 35(5):624–637.

Nebro, A. J., Durillo, J. J., Luna, F., Dorronsoro, B., and Alba, E. (2007). Design issues in a multiobjective cellular genetic algorithm. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 126–140. Springer.

Nerur, S., Mahapatra, R., and Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Commun. ACM*, 48(5):72–78.

Neuman, G. A. and Wright, J. (1999). Team effectiveness: beyond skills and cognitive ability. *Journal of Applied psychology*, 84(3):376.

Ngo-The, A. and Ruhe, G. (2008). Optimized resource allocation for software release planning. *IEEE Transactions on Software Engineering*, 35(1):109–123.

Özdamar, L. and Ulusoy, G. (1995). A survey on the resource-constrained project scheduling problem. *IIE transactions*, 27(5):574–586.

Pecora, F. and Cesta, A. (2002). Planning and scheduling ingredients for a multi-agent system. In *Proceedings of UK PLANSIG02 Workshop, Delft, The Netherlands*, volume 371.

Peteghem, V. V. and Vanhoucke, M. (2014). An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. *European Journal of Operational Research*, 235(1):62 – 72.

Phillips, J. and Education, M.-H. (2018). *PMP®: Project Management Professional Study Guide*. McGraw Hill.

Pinto, J. K. and Prescott, J. E. (1988). Variations in critical success factors over the stages in the project life cycle. *Journal of management*, 14(1):5–18.

Pritsker, A. A. B., Waiters, L. J., and Wolfe, P. M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management science*, 16(1):93–108.

Raluca, Z. (2012). A conceptual framework for team social capital as basis for organizational team synergy. *Annals of the University Dunarea de Jos of Galati: Fascicle: I, Economics & Applied Informatics*, 18(2).

Ratajczak-Ropel, E. and Skakovski, A. (2018). *Population-Based Approaches to the Resource-Constrained and Discrete-Continuous Scheduling*. Springer.

Ren, M., Ren, L., and Jain, H. (2018). Manufacturing service composition model based on synergy effect: a social network analysis approach. *Applied Soft Computing*, 70:288–300.

Rezende, A. V., Silva, L., Britto, A., and Amaral, R. (2019). Software project scheduling problem in the context of search-based software engineering: A systematic review. *Journal of Systems and Software*, 155:43–56.

Rodríguez, D., Ruiz, M., Riquelme, J. C., and Harrison, R. (2011). Multiobjective simulation optimisation in software project management. In *Proceedings of the*

13th annual conference on Genetic and evolutionary computation, pages 1883–1890.

Rodriguez-Repiso, L., Setchi, R., and Salmeron, J. L. (2007). Modelling it projects success: Emerging methodologies reviewed. *Technovation*, 27(10):582–594.

Ruiz, M. and Fuentes, G. (2017). Applying extreme engineering and personality factors to improve software development under a heavyweight methodology. In *International Conference on Product-Focused Software Process Improvement*, pages 470–481. Springer.

Salas, E. and Burke, C. (2002). Simulation for training is effective when.... *BMJ Quality & Safety*, 11(2):119–120.

Sanchez, O. P., Terlizzi, M. A., et al. (2017). Cost and time project management success factors for information systems development projects. *International Journal of Project Management*, 35(8):1608–1626.

Sarbin, T. R. (1954). Role theory'in lindzey, g.(ed.) handbook of social psychology. *Reading, Mass., G. Addison-Wesley*.

Saxena, A. and Iyengar, S. (2020). Centrality measures in complex networks: A survey. *arXiv preprint arXiv:2011.07190*.

Schmitt, N., Gooding, R. Z., Noe, R. A., and Kirsch, M. (1984). Metaanalyses of validity studies published between 1964 and 1982 and the investigation of study characteristics. *Personnel Psychology*, 37(3):407–422.

Schneider, B. and Smith, D. B. (2004). *Personality and organizations*. Psychology Press.

Schoenhoff, P. K. (2001). *Belbin's Company Worker, The Self-Perception Inventory, and Their Application to Software Engineering Teams*. PhD thesis, Virginia Tech.

Scholtes, I., Mavrodiev, P., and Schweitzer, F. (2016). From aristotle to ringelmann: a large-scale analysis of team productivity and coordination in open source software projects. *Empirical Software Engineering*, 21(2):642–683.

Schutz, W. (1992). Beyond firo-b—three new theory-derived measures—element b: Behavior, element f: Feelings, element s: Self. *Psychological reports*, 70(3):915–937.

Schutz, W. C. (1955). What makes groups productive? *Human Relations*, 8(4):429–465.

Schutz, W. C. (1959). Firo, a three-dimensional theory of interpersonal behavior. *Journal of Symbolic Logic*, 24(3):216–217.

Schwab, A. and Miner, A. S. (2008). Learning in hybrid-project systems: The effects of project performance on repeated collaboration. *Academy of Management Journal*, 51(6):1117–1149.

Schwindt, C., Zimmermann, J., et al. (2015). *Handbook on project management and scheduling vol. 2*. Springer.

SGI (2019). Chaos manifesto. Standish Group International.

Shaw, M. E. (1964). Communication networks. In *Advances in experimental social psychology*, volume 1, pages 111–147. Elsevier.

Shaw, M. E. and Webb, J. N. (1982). When compatibility interferes with group effectiveness: Facilitation of learning in small groups. *Small Group Behavior*, 13(4):555–564.

Sheard, S., Cook, S., Honour, E., Hybertson, D., Krupa, J., McEver, J., McKinney, D., Ondrus, P., Ryan, A., Scheurer, R., et al. (2015). A complexity primer for systems engineers. *INCOSE Complex Systems Working Group White Paper*, 1:1–10.

Shen, X., Minku, L. L., Bahsoon, R., and Yao, X. (2015). Dynamic software project scheduling through a proactive-rescheduling method. *IEEE Transactions on Software Engineering*, 42(7):658–686.

Shen, X.-N., Minku, L. L., Marturi, N., Guo, Y.-N., and Han, Y. (2018). A q-learning-based memetic algorithm for multi-objective dynamic software project scheduling. *Information Sciences*, 428:1–29.

Shenhar, A. and Dvir, D. (2007). *Reinventing Project Management: The Diamond Approach To Successful Growth And Innovation*. Harvard Business Review Press.

Shenhar, A. J. (2001). One size does not fit all projects: Exploring classical contingency domains. *Management science*, 47(3):394–414.

Shi, Q. and Blomquist, T. (2012). A new approach for project scheduling using fuzzy dependency structure matrix. *International Journal of Project Management*, 30(4):503–510.

Simeunovic, A. and Landelius, T. (2017). Belbin's team roles in agile software development.

Smith-Jentsch, K. A., Salas, E., and Baker, D. P. (1996). Training team performance-related assertiveness. *Personnel Psychology*, 49(4):909–936.

Sorenson, J. R. (1971). Task demands, group interaction and group performance. *Sociometry*, 34(4):483–495.

Sparrowe, R. T., Liden, R. C., Wayne, S. J., and Kraimer, M. L. (2001). Social networks and the performance of individuals and groups. *Academy of Management Journal*, 44(2):316–325.

Špundak, M. (2014). Mixed agile/traditional project management methodology–reality or illusion? *Procedia-Social and Behavioral Sciences*, 119:939–948.

Sridharan, S. and Balakrishnan, R. (2019). *Discrete Mathematics: Graph Algorithms, Algebraic Structures, Coding Theory, and Cryptography*. CRC Press.

Stevens, K. and Henry, S. (2002). Analysing software teams using belbin's innovative plant role. *Department of Computer and Information Science, University of Mississippi and Department of Computer Science, Virginia Tech*.

Stevens, M. J. and Campion, M. A. (1999). Staffing work teams: Development and validation of a selection test for teamwork settings. *Journal of Management*, 25(2):207–228.

Steward, D. V. (1981). *Systems analysis and management: structure, strategy, and design*. Petrocelli books.

Suri, B. and Jajoria, P. (2013). Using ant colony optimization in software development project scheduling. In *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2101–2106. IEEE.

Tannenbaum, S. I., Beard, R. L., and Salas, E. (1992). Team building and its influence on team effectiveness: An examination of conceptual and empirical developments. In *Advances in psychology*, volume 82, pages 117–153. Elsevier.

Tavares, L. V., Ferreira, J. A., and Coelho, J. S. (1999). The risk of delay of a project in terms of the morphology of its network. *European Journal of Operational Research*, 119(2):510 – 537.

Thomas, R. N. (1999). Group dynamics and software engineering. In *Object Oriented Programming Systems Languages and Applications: Educators' Symposium*.

Tirkolaee, E. B., Goli, A., Hematian, M., Sangaiah, A. K., and Han, T. (2019). Multi-objective multi-mode resource constrained project scheduling problem using pareto-based algorithms. *Computing*, 101(6):547–570.

Toljaga-Nikolic, D., Petrovic, D., and Mihic, M. (2017). How to choose the appropriate project management approach? In *2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT)*, volume 2, pages 1–5. IEEE.

Turner, J. (2009). *The Handbook of Project-based Management: Leading Strategic Change in Organizations*. McGraw-Hill's AccessEngineering. McGraw-Hill.

Vanhoucke, M. (2012). Measuring the efficiency of project control using fictitious and empirical project data. *International Journal of Project Management*, 30(2):252 – 263.

Vanhoucke, M., Coelho, J., Debels, D., Maenhout, B., and Tavares, L. V. (2008). An evaluation of the adequacy of project network generators with systematically sampled networks. *European Journal of Operational Research*, 187(2):511 – 524.

Vega-Velázquez, M. Á., García-Nájera, A., and Cervantes, H. (2018). A survey on the software project scheduling problem. *International Journal of Production Economics*, 202:145–161.

Vidal, L.-A., Marle, F., and Bocquet, J.-C. (2011). Measuring project complexity using the analytic hierarchy process. *International Journal of Project Management*, 29(6):718–727.

Wang, L. and Zheng, X.-l. (2018). A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem. *Swarm and Evolutionary Computation*, 38:54–63.

Węglarz, J., Józefowska, J., Mika, M., and Waligóra, G. (2011). Project scheduling with finite or infinite number of activity processing modes–a survey. *European Journal of operational research*, 208(3):177–205.

Wena, F. and Lin, C.-M. (2008). Multistage human resource allocation for software development by multiobjective genetic algorithm. *The Open Applied Mathematics Journal*, 2(1).

Wu, C., Wang, X., and Lin, J. (2014). Optimizations in project scheduling: A state-of-art survey. In *Optimization and control methods in industrial engineering and construction*, pages 161–177. Springer.

Wu, L., Waber, B. N., Aral, S., Brynjolfsson, E., and Pentland, A. (2008). Mining face-to-face interaction networks using sociometric badges: Predicting productivity in an it configuration task. *Available at SSRN 1130251*.

Wu, X., Consoli, P., Minku, L., Ochoa, G., and Yao, X. (2016). An evolutionary hyper-heuristic for the software project scheduling problem. In *International Conference on Parallel Problem Solving from Nature*, pages 37–47. Springer.

Wysocki, R. K. (2011). *Effective project management: traditional, agile, extreme*. John Wiley & Sons.

Wysocki, R. K. (2019). *Effective project management: traditional, agile, extreme, hybrid*. John Wiley & Sons.

Xiao, J., Ao, X.-T., and Tang, Y. (2013). Solving software project scheduling problems with ant colony optimization. *Comput. Oper. Res.*, 40(1):33–46.

Xiao, J., Gao, M.-L., and Huang, M.-M. (2015). Empirical study of multi-objective ant colony optimization to software project scheduling problems. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 759–766.

Yang, H.-L. and Tang, J.-H. (2004). Team structure and team performance in is development: a social network perspective. *Information & management*, 41(3):335–349.

Yang, X.-S. (2010). *Engineering optimization: an introduction with metaheuristic applications*. John Wiley & Sons.

Yilmaz, M., O'Connor, R. V., Colomo-Palacios, R., and Clarke, P. (2017). An examination of personality traits and how they impact on software development teams. *Information and Software Technology*, 86:101–122.

Yoshimura, M., Fujimi, Y., Izui, K., and Nishiwaki, S. (2006). Decision-making support system for human resource allocation in product development projects. *International journal of production research*, 44(5):831–848.

Young, K. D., Feydy, T., and Schutt, A. (2017). Constraint programming applied to the multi-skill project scheduling problem. In *International Conference on Principles and Practice of Constraint Programming*, pages 308–317. Springer.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report*, 103.

Zorrilla, M. and de Lima Silva, M. (2019). Sociograms: An effective tool for decision making in social learning. *Technology, Knowledge and Learning*, 24(4):659–681.

Appendix

A.1 Correlation of Independent Variables

TABLE A.1. Kendall rank correlation of independent variables

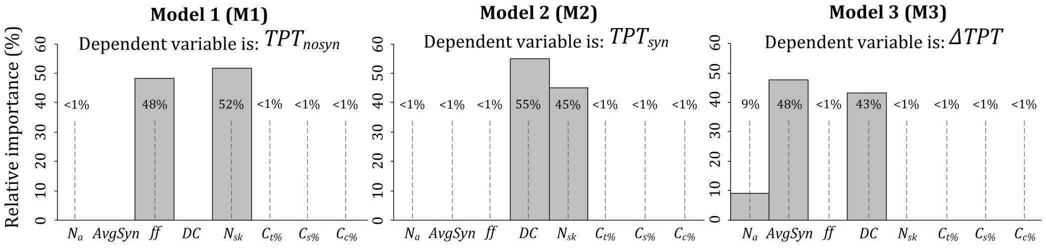
(Source: own table)

Variable	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
(1) N_{sk}	1.00										
(2) $AvgSyn$	0.00	1.00									
(3) $C_{t\%}$	0.00	0.00	1.00								
(4) $C_{c\%}$	0.00	0.00	0.00	1.00							
(5) $C_{s\%}$	0.00	0.00	0.00	0.00	1.00						
(6) ff	0.00	0.00	0.00	0.00	0.00	1.00					
(7) N_a	0.92	0.00	0.00	0.00	0.00	0.00	1.00				
(8) DC	-0.52	0.00	0.00	0.00	0.00	0.00	-0.52	1.00			
(9) CC	-0.37	0.00	0.00	0.00	0.00	0.00	-0.37	0.87	1.00		
(10) BC	-0.20	0.00	0.00	0.00	0.00	0.00	-0.20	-0.44	-0.73	1.00	
(11) PP	-0.37	0.00	0.00	0.00	0.00	0.00	-0.37	0.87	1.00	-0.73	1.00

A.2 Predictor Importance in Additional Model

FIGURE A.1. Relative importance of various predictors (additional model)

(Source: own figure)



Remark: dependent variables are related to time.

A.3 Electronic Supplementary Material

The Electronic Supplementary Material is available at *https://github.com/IHFSPP/Electronic_Supplementary_Material.git*. It contains the following files:

(1) **Calculations for Fig. 12** (*fig10_calc.m*): Calculations used for preparing Fig. 12.

(2) **Workspace for Fig. 12** (*fig10_workspace.mat*): Workspace of calculations used for preparing Fig. 12.

(3) **Calculations for Fig. A.1** (*figa1_calc.m*): Calculations used for preparing Fig. A.1.

(4) **Workspace for Fig. A.1** (*figa1_workspace.mat*): Workspace of calculations used for preparing Fig. A.1.