TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Technische Dienstleistungen und Operations Management

Quantitative approaches to the management of human resources in IT–projects

Dipl.-Ing. Univ., Dipl.-Wirtsch.Ing. Univ.
Christian M. Heimerl

Vollständiger Abdruck der von der Fakultät für Wirtschaftswissenschaften der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Wirtschaftswissenschaften (Dr. rer. pol.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr. Gunther Friedl
Prüfer der Disseration:	1.	Univ.-Prof. Dr. Rainer Kolisch
	2.	a.o. Univ.-Prof. Dr. Walter J. Gutjahr, Universität Wien, Österreich

Die Dissertation wurde am 23.09.2009 bei der Technischen Universität München eingereicht und durch die Fakultät für Wirtschaftswissenschaften am 21.10.2009 angenommen.

Table of Contents

Table of Contents i

List of Tables ii

List of Figures iv

1 Managing human resources in IT–projects 1
- 1.1 Introduction . . . 1
- 1.2 Aspects treated in this thesis . . . 3
- 1.3 Outline . . . 7

2 Staffing and scheduling projects 8
- 2.1 Introduction . . . 8
- 2.2 Literature Review . . . 9
- 2.3 Model . . . 12
  - 2.3.1 Model description . . . 12
  - 2.3.2 MIP Formulation . . . 14
  - 2.3.3 Model extensions . . . 16
  - 2.3.4 Improved MIP Formulation . . . 17
- 2.4 Computational study . . . 20
  - 2.4.1 Test instances . . . 20
  - 2.4.2 Solution gaps . . . 22
  - 2.4.3 Computation time . . . 24
- 2.5 Managerial insight . . . 25
  - 2.5.1 Influences of the factors on the costs . . . 25
  - 2.5.2 Optimal vs. heuristic planning . . . 29
  - 2.5.3 Central vs. decentral planning . . . 30
- 2.6 Summary . . . 33

3 Staffing and scheduling serial projects 35
- 3.1 Introduction . . . 35

3.2 Model . . . 36
3.2.1 Model description . . . 36
3.2.2 MIP Formulation . . . 37
3.2.3 Additional cuts . . . 39
3.3 Staffing subproblem . . . 40
3.3.1 LP Formulation . . . 41
3.3.2 Generalized minimum cost flow formulations . . . 42
3.3.3 Generalized network simplex algorithm . . . 47
3.4 Hybrid metaheuristic . . . 48
3.4.1 Genetic algorithm . . . 48
3.4.2 Tabu search . . . 49
3.5 Experimental investigation . . . 52
3.5.1 Experimental setup . . . 52
3.5.2 Computation times . . . 53
3.5.3 Solution gaps . . . 56
3.6 Summary . . . 59

4 Qualification of human resources **61**
4.1 Introduction . . . 61
4.2 Literature Review . . . 62
4.3 Model . . . 64
4.4 Implementation and Test setup . . . 68
4.5 Results . . . 70
4.5.1 Computational study . . . 70
4.5.2 Managerial insights . . . 71
4.5.2.1 Analysis methods . . . 71
4.5.2.2 Outsourcing decisions . . . 74
4.5.2.3 Impact of the learning curve . . . 75
4.5.2.4 Impact of company skill level targets . . . 77
4.5.2.5 Impact of knowledge depreciation . . . 79
4.5.2.6 Impact of internal costs . . . 81
4.6 Summary . . . 81

5 Conclusion and Outlook **83**

A Notation **vii**

Bibliography **ix**

List of Tables

2.1 Parameters for the base case 20
2.2 Factors and levels for the experimental test design 22
2.3 Solution gaps Δ of the two LP–Relaxations 22
2.4 Adjusted solution gaps Δ' of the two LP–Relaxations 24
2.5 Solution times (seconds) for different time window sizes and relative MIP gap tolerances 24
2.6 Solution times (in seconds) for different factors 25
2.7 Optimal costs (in millions) for the different factor levels 28
2.8 Comparison of heuristics 30
2.9 Ratio of feasible solutions with heuristics RND and MCF . . . 31
2.10 Factors and levels for the experimental test design 32
2.11 Cost improvement κ by central planning depending on the number of departments $|\mathcal{D}|$ and their specialization δ 33

3.1 Parameters of the test instances 52
3.2 Solution methods applied 53
3.3 Average evaluation times for instances with $TWS = 5$ in milliseconds 54
3.4 Average computation times for combinations of δ_a^{min} and δ_a^{max} and $(|\mathcal{P}|, |\mathcal{R}^i|) = (5, 5)$ in seconds 55
3.5 Average computation times for different time window sizes TWS in seconds 56
3.6 Average solution gaps Δ for combinations of δ_a^{min} and δ_a^{max} and $(|\mathcal{P}|, |\mathcal{R}^i|) = (5, 5)$ 57
3.7 Average solution gaps compared to the LP–relaxation for different time windows sizes TWS 59
3.8 Percentage of optimally solved instances with $(|\mathcal{P}|, |\mathcal{R}^i|) = (5, 5)$ 60

4.1 Parameters and skill matrix of the base case 69
4.2 Variations of the base case 71
4.3 Computational results 72
4.4 Additional computational results 72

4.5 Fraction ρ_s of work done internally, specialization of the company CV($X^{\mathcal{C}}$) and specialization $\overline{\sigma}_k$ of internal resources . . . 74
4.6 Fraction ρ_s of work done internally and specialization of the company CV($X^{\mathcal{C}}$) for instances 1, 5, 9 and 13 75
4.7 Specialization $\overline{\sigma}_k$ of internal resources for instances 1, 5, 9 and 13 77

A.1 Notation for the MIP vii
A.2 Notation for the MIP viii

List of Figures

1.1 Example of the basic problem setting 2
1.2 Consecutive planning levels of human resource allocation within a project–context 5

2.1 Analogy to the multiprocessor scheduling problem 16
2.2 Visualization of $\mathcal{T}_{pt}$ 19
2.3 Normalized availability of an internal resource as a step function 21
2.4 Impact of the time window size on the costs 26
2.5 Impact of the number of projects on the costs 27
2.6 Impact of the average number of skills per resource on the costs 27
2.7 Impact of the workload on the costs 28
2.8 Cost improvement κ by central planning depending on the number of departments $|\mathcal{D}|$ and their specialization δ 34

3.1 Notation for graphs 42
3.2 Graph G_1 for relaxed single–period problem LP5 43
3.3 Graph G_2 for LP6 45
3.4 Relevant part of Graph G_3 for LP6 46
3.5 Typical evolution of the objective function value during the GA over time 58

4.1 Production possibility frontier 66
4.2 Plots of the flat ($\lambda = 0.012$) and the steep ($\lambda = 0.02$) learning curve $f(z)$ 70
4.3 Solution of instance 1 with $\lambda_{ks} = 0.012$, $\phi_s = 0$, $\beta_{kst} = 0$, and $c_k^i = 0$ 76
4.4 Solution of instance 9 with $\lambda_{ks} = 0.02$, $\phi_s = 0$, $\beta_{kst} = 0$, and $c_k^i = 0$ 77
4.5 Solution of instance 3 with $\lambda_{ks} = 0.012$, $\phi_s = 2.5$, $\beta_{kst} = 0$, and $c_k^i = 0$ 78
4.6 Influence of company skill level targets ϕ_s on costs 79

4.7 Solution of instance 5 with $\lambda_{ks} = 0.012$, $\phi_s = 0$, $\beta_{kst} = 10$, and $c_k^i = 0$ 80
4.8 Solution of instance 7 with $\lambda_{ks} = 0.012$, $\phi_s = 2.5$, $\beta_{kst} = 10$, and $c_k^i = 0$ 81

Chapter 1

Challenges of managing human resources in IT–projects

1.1 Introduction

Assigning human resources to work (also known as "staffing", cf. Barreto et al. [9]) taking into account resource–specific skills and efficiencies is a general planning task which has to be performed in any organization.[1] It is of particular importance for service firms where, compared to manufacturing firms, the labour intensity is high (cf. e.g. Brusco [20]). The emphasis of this thesis is on the context of IT–projects. Nevertheless the problem is common in many other project environments as well.

The problem in focus arose at the departments of information technology (IT) of a major semiconductor manufacturer (cf. Heimerl [42]) and of a large telecommunication company. There, multiple projects such as the selection, development, installation and configuration of new IT–systems for different departments have to be done on a recurrent basis. Projects arrive dynamically over time and have then to be accomplished within a given time span. For the processing of the projects internal human resources with different skills and different unit costs have to be used (cf. Figure 1.1). Furthermore, fractions of the projects may be processed by (usually more expensive) external human resources, i.e. project work can be outsourced. Some examples for skills required in IT–projects are programming, software architecture, security, or hardware skills. Note that resources can have multiple skills, e.g. as depicted in Figure 1.1 one resource has programming and architecture skills,

[1] Since only human resources will be considered in this thesis, the expression "resources" will be used equivalently to "human resources" henceforth.

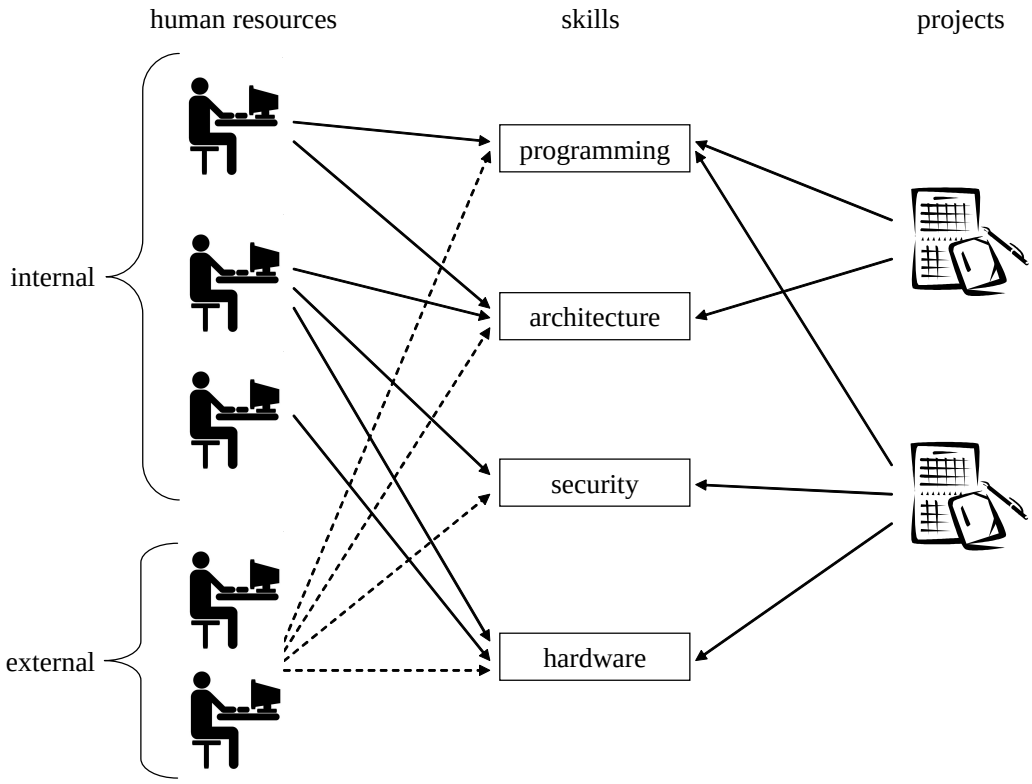


Figure 1.1: Example of the basic problem setting

another has architecture, security and hardware skills while a third resource only has hardware skills.

Projects consist of several work packages. Each work package can only be processed by resources having (at least) the skill defined by the work package. E.g. in Figure 1.1 a work package requiring the programming skill can be processed by the first internal resource while a work package requiring the architecture skill can be processed by the first and second internal resource. Alternatively, external resources may be assigned to the work packages. Furthermore, resources usually work at different performance levels (i.e. efficiencies) w.r.t. a skill. Performance levels determine the time required to process a work package. If an internal resource is very efficient in performing a certain skill it can do more work requiring this skill in the same amount of time. This reduces the need to outsource work packages. The goal pursued throughout this thesis is to assign project work packages to resources efficiently such that total assignment costs of internal and external resources are minimized. In the course of the thesis this basic problem setting will be extended w.r.t. several aspects.

1.2 Aspects treated in this thesis

Optimization goal This thesis introduces several mathematical optimization models for the management of human resources in IT–projects. Mathematical optimization models can represent real decision problems using several variables, parameters, constraints, and (usually) a single objective function. The problem is to find the values of the decision variables that optimize (i.e. minimize or maximize) a goal expressed in terms of the objective function. At the same time one or more constraints need to be taken into account.

When solving staffing problems previous research pursued company– as well as resource–specific goals: *Company–specific goals* can be the maximization of the company's revenue (cf. Abboud et al. [1], Kwak et al. [55]), the maximization of the company's utility (cf. Campbell and Diaby [23], Sayin and Karabati [65]), the maximization of the company's performance or productivity (cf. Chu and Lin [25], Kwak et al. [55], Nembhard and Osothsilp [57]), the minimization of costs for (external) resources (cf. Bassett [10], Bhatnagar et al. [16], Cai and Li [21], Eiselt and Marianov [32], Wu and Sun [76], Wu [77]) and/or overtime (cf. Bhatnagar et al. [16], Eiselt and Marianov [32]), the minimization of project duration (cf. Bellenguez-Morineau and Néron [15], Vairaktarakis [69]) or the optimization of other performance indicators such as the so–called project scatter factor or the resource dedication profile (cf. Hendriks et al. [48]). Typical *resource–specific goals* are the maximization of the resources' satisfaction (cf. Abboud et al. [1], Yoshimura et al. [79]), the maximization of preferences or priorities for a task (cf. Corominas et al. [27]), the maximization of appropriateness for a task (cf. Ballou and Tayi [8]), the minimization of the resources' boredom (cf. Eiselt and Marianov [32]), the maximization of skill improvement (cf. Gutjahr et al. [40], Sayin and Karabati [65]), equalizing workload among resources (cf. Cai and Li [21], Eiselt and Marianov [32]) or compliance with soft–constraints like a limit on the maximum number of similar tasks per resource (cf. Corominas et al. [28]).

This thesis concentrates on the company–specific goal of cost minimization. Costs accrue from staffing projects with internal and external human resources. More specifically the costs are caused by a human resource for the time spent on processing a project work package. Each time unit spent on project work is charged according to a resource specific cost rate.

Next to the goal of cost minimization an additional rather strategic goal will be considered in Chapter 4. There the goal is to qualify internal human resources according to requirements on the company's skill portfolio.

Planning levels Usually not only a single project but multiple projects raise demand for resources. Multi–project–environments can be found particularly in software development (cf. Wu and Sun [76]), software maintenance (cf. Ballou and Tayi [8]) as well as research and development (cf. Bassett [10], Yoshimura et al. [79]). Compared to ongoing operations, project environments imply additional complexity w.r.t. the staffing problem. In the case of ongoing operations such as in health care, call centers or many manufacturing environments the planning task can be done in two steps: First, the number of resources required per skill and period (demand) is determined. Second, staffing is done by assigning human resources to the demand. Typically, this is done based on shifts (cf. Brunner et al. [19], Ernst et al. [33]). In a project–context, the two levels cannot be separated as easily because the demand for resources in one period depends on the schedules of the projects.

Within a multi–project–context human resource allocation is usually embedded in the following three consecutive planning levels (cf. Figure 1.2, Heimerl and Kolisch [45]): project selection, project scheduling, and project staffing. The decisions on these planning levels are intertwined in both directions, e.g. optimal decisions on the project staffing level depend on the decisions on the project scheduling level and vice versa. By scheduling the projects in a smart manner the demand for skills can be aligned with the supply reducing the need for external resources and the associated assignment costs. Thus, an integrating multi–project–perspective of more than one planning level offers advantages w.r.t. to objective function value. This observations holds no matter what the overall goal is (e.g. minimization of costs, maximization of revenue, minimization of project duration). However, complexity grows extraordinarily with the number of planning levels involved. Furthermore, goals might be very heterogeneous on the considered planning levels and, thus, hard to be integrated.

This thesis will focus on an integrated view to the planning levels of scheduling and staffing. The evaluation of the potential for cost reductions that is achievable by the combination of these planning levels will be part of this thesis. The cost savings that can be obtained by adding flexibility regarding the project start period will be analysed, too.

Staffing Whether resources need to be assigned completely to one task or project within one period of time or whether the assignment can be spread across multiple tasks/projects has been treated differently in previous research: Several authors (e.g. [1, 15, 16, 20, 21, 23, 25, 27, 56, 57, 68–70, 76]) assume a *discrete 0–1 assignment* of resources to tasks. This approach is appropriate if e.g. transfer times between tasks (e.g. due to travelling) are

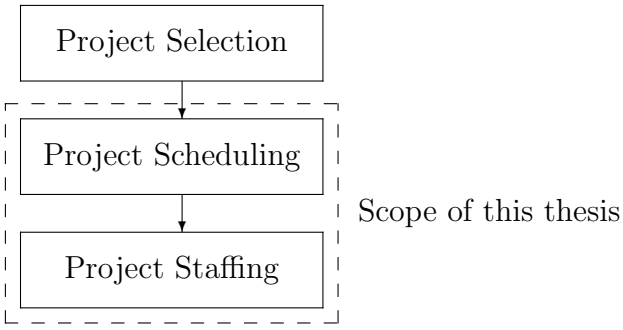


Figure 1.2: Consecutive planning levels of human resource allocation within a project–context

relatively high and cannot be neglected (cf. Krüger and Scholl [54]), the lengths of the periods are relatively short, or the transfer between tasks must occur concurrently (e.g. due to a minimum number of resources per task constraint). Other publications (cf. e.g. [3, 8, 39–41, 45, 51, 79]) allow *fractional assignments* of resources to tasks. Fractional assignments allow to model multi–tasking of resources within a period. This is a realistic assumption for longer period lengths of e.g. months or weeks. While *discrete* fractional assignments only allow predefined assignment shares (e.g. 0.0, 0.5, 1.0), in the case of *continuous* fractional assignments every real number between 0 and 1 describes a valid assignment. In the aforementioned practical problem settings the use of continuous fractional assignments is a requirement. While from a mathematical point of view, the continuous fractional problem is easier to solve than the discrete problems, the interpretation of the fractional assignment results within a period remains as a subsequent problem in practice.

Skills With respect to the ability of human resources to perform different types of work, three different assumptions are made in the literature: *i*) Single–skilled resources possess exactly one skill out of a set of different skills which are required (within the organizational unit and the time horizon of relevance). *ii*) On the contrary, completely–skilled resources possess each of the skills required. *iii*) A multi–skilled resource possesses a subset of the relevant skills. The multi–skill case is the most general one and includes the two other ones as special cases.

Classical project planning models (e.g. the resource–constrained project scheduling problem (RCPSP), cf. Brucker et al. [18]) implicitly assume the single–skill case. I.e. each resource has one specific skill and for each skill and each period the capacity of all resources can be aggregated to an overall

capacity. This does not take into account that human resources usually have multiple skills, i.e. are capable of doing different types of work (cf. Heimerl and Kolisch [45]). However, the multi–skill case is of particular importance in the IT. In the IT–departments where this problem arose, a few dozens of skills were identified and the vast majority of the resources had more than one skill (cf. Heimerl [42]).

Efficiencies, Learning, and Forgetting When considering multi–skilled resources it can be of particular importance to take efficiencies into account. The efficiency of a human resource is defined as the amount of work that can be accomplished by a human resource in a given time span (cf. Heimerl and Kolisch [45]). Efficiencies have a direct impact on the overall amount of work that can be performed and, hence, an indirect impact on costs. Furthermore, a resource's qualification level can also be described by its efficiencies. If efficiencies are equal for all resources they are called *homogeneous*. However, efficiencies can also differ w.r.t. the person and the skill performed (*heterogeneous*) as each person has different strengths and weaknesses. If efficiencies are constant over time they are called *static*. Due to time–dependent effects they may also float over time (*dynamic*). A prominent efficiency–increasing effect is learning. Amongst others learning is caused by the repetition of similar tasks, i.e. in a project context by the assignment of similar work packages to the same resource. As a consequence subsequent tasks can be processed more quickly. The inverse effect is forgetting. It is usually caused by breaks when a skill is not used for a certain period of time. Furthermore, technological progress is extremely rapid especially in IT–environments. If a resource is not keeping up to date with evolving technologies its efficiency will decrease. Obviously, time–varying efficiencies must be considered when making cost–optimal assignment decisions. Aspects of time–dependent efficiencies will be treated in more detail in Chapter 4.

Outsourcing Classical project planning approaches (cf. Brucker et al. [18]) usually rely on the assumption that resources are scarce. However, especially in the IT sector outsourcing is a commonly used option to supplement the internal workforce. Against this background scarcity of resources is often no hard constraint. Nevertheless, the degree of outsourcing is often limited by other (soft) constraints. E.g. risks of not achieving the deliverables might increase if outsourcing is exaggerated. About 25% of all IT–projects are cancelled before completion and one of the reasons for failing is outsourcing (cf. Asay [7]). Thus, a restriction on the amount of outsourced work is necessary. Limited budgets are an indirect boundary for the degree of outsourced work

since costs of short–term external resources are usually higher than those of internal resources. However, depending on the project type (e.g. complexity, size) a more explicit limit on the amount of outsourced work is required. In the IT–department of the semiconductor manufacturer the ratio of internally and externally performed project work acts as a limit for the maximum degree of outsourcing (cf. Heimerl [42]). This ratio was also adapted in this thesis as outsourcing limit.

1.3 Outline

The thesis is organized as follows. In Chapter 2 the integrated project scheduling and staffing problem is addressed. The assumption in this problem is that only the project start can be scheduled. The project itself is already scheduled and only shifts with the project start. A detailed problem description, the mathematical optimization model, a computational study and managerial insights regarding this type of problem will be presented.

The problem is generalized in Chapter 3 as each project will be modelled as a serial stream of activities. Hence, not only the project start but each of the serially linked activities has to be scheduled. Next to the mathematical optimization model an efficient solution algorithm will be presented. It will be compared with standard solution procedures via a computational study.

Chapter 4 will focus on the staffing problem and will assume the project schedules as given. The goal is to staff the projects considering learning and depreciation of knowledge. Under the goal of cost–minimization it is required to qualify human resources towards a desired skill portfolio. This problem is modelled as a non–linear mathematical optimization model. A computational study and managerial insights will be presented.

This thesis concludes with a summary and an outlook in Chapter 5.

Chapter 2

Staffing and scheduling aggregated IT–projects

2.1 Introduction

The assignment of human resources to work is a common but complex task. It is particularly complex if it has to be done in a project environment since the temporal distribution of the demand relies on the schedules of the projects. Furthermore, especially in IT–projects resources are multi–skilled and outsourcing is very common. As stated in the introductory chapter the consideration of more than one planning level promises better results than a separated (hierarchic) view. Thus, in this chapter an optimization model for simultaneous scheduling and staffing multiple projects will be presented (cf. Heimerl [42], Heimerl and Kolisch [43, 45]).

The optimization models in this chapter will assume that only the start of each project can be scheduled. The remainder of the project, i.e. subsequent work packages, is already scheduled w.r.t. the project start and only depends on and shifts with the project start. The human workforce is assumed to be multi–skilled with heterogeneous but static efficiencies. Outsourcing by means of assigning external human resources to project work is allowed but limited.

In the following Section 2.2 the relevant literature will be reviewed. A detailed problem description and the optimization model will be presented afterwards in Section 2.3. Section 2.4 will provide results of a computational study. Managerial insights, e.g. advantages and disadvantages of central vs. decentral planning, will be given in Section 2.5. The advantages of an integrated optimization of project scheduling and staffing will be evaluated there, too. Finally, the chapter concludes with a summary in Section 2.6.

2.2 Literature Review

A number of papers have treated assignment decisions in a multi–skill environment explicitly. In what follows the literature will be reviewed following a grouping according to the considered planning levels.

Project staffing. Papers which only consider the project staffing problem assume a project schedule as given. The latter defines for each period how many units of each skill are required. Cai and Li [21] treat the assignment of days–off schedules to multi–skilled resources with two skills in order to cover the given resource demand within a service setting. Skill efficiencies are assumed to be static and homogeneous. Campbell [22] as well as Campbell and Diaby [23] consider the assignment of multi–skilled human resources with heterogeneous efficiencies to given demand in a health care service setting for a single period (static case). Corominas et al. [27] assume a single period where demand for different skills has to be covered by human resources with multiple skills as well as homogeneous and static efficiencies. The objective function, next to minimizing shortage and surplus assignments, maximizes the sum of the priorities arising from the assignment. Corominas et al. [28] relate to a service setting where multi–skilled human resources with homogeneous efficiencies have to be assigned to activities over the periods of the planning horizon such that different objectives are met in the best way. A heuristic is proposed which solves a sequence of assignment problems where the elements of the assignment matrix take into account the different objectives. Valls et al. [70] propose a model for the assignment of multi–skilled human resources with homogeneous efficiencies to operate a set of machines for the periods of a planning horizon. Each machine requires a specific skill and has to be operated by one human resource. The objective is to minimize the number of human resources deployed. The problem is modeled as a restricted vertex coloring problem which is solved with a branch–and–bound procedure.

Project scheduling and staffing. One of the most thoroughly discussed project planning problems in literature is the resource–constrained project scheduling problem (RCPSP) (cf. e.g. Brucker et al. [18]). There a project is modelled as a set of activities linked by precedence relationships. Each activity has a demand for one or more resource types and each resource type has a limited capacity. A resource type represents an aggregated view of several resources with equal properties (e.g. the same skill). The problem consists of scheduling the activities while matching resource demand and supply. The

RCPSP implicitly assumes the single–skill case with homogeneous efficiencies of all resources and discrete 0–1 assignments of resources to activities.

The extension of the RCPSP, the multi–mode RCPSP (MMRCPSP) is, in general, capable to depict the multi–skill case. But, as Bellenguez-Morineau and Néron [15] point out, the large number of emerging modes make the MMRCPSP not a viable option to model multiple skills. Hence, they developed a proprietary multi–skill model (cf. Bellenguez and Néron [12, 13, 14], Bellenguez-Morineau and Néron [15]). Project activities demand specific skills (and not resources as in the RCPSP) and human resources own multiple skills. The problem is then to schedule activities and match skill demand and skill supply. Skill efficiencies are assumed to be homogeneous and static. As the classical RCPSP the model assumes discrete 0–1 assignments of resources to activities.

Alfares and Bailey [4] consider construction projects where they minimize project duration and staffing costs. Only one type of resource, where each resource has the same single skill with homogeneous and static efficiency, is considered. An integer linear program is proposed which simultaneously schedules activities and determines two–days–off tours for the human resources. Bassett [10] considers R&D–projects in the chemical industry where independent activities have to be scheduled within time windows and internal as well as external human resources have to be assigned to the activities. Efficiencies are assumed to be homogeneous and static. Valls et al. [71] consider the scheduling of tasks and their assignment to a multi–skilled workforce with heterogeneous efficiencies in a service center under multiple criteria. Wu and Sun [76] present a mixed–integer non–linear program for multi–project scheduling and staffing. The work of the activities has to be allocated within the respective time windows (whereas precedence relations between activities are not taken into account). Efficiencies are assumed to be heterogeneous and dynamic and it is further assumed that each human resource can only be assigned to one project per period. The problem is solved with a genetic algorithm. Vairaktarakis [69] defines a measure for the amount resources are capable of doing multi–skill work (with the extremes single– and completely–skilled). Based on an MIP and heuristics he investigates how the increase of multi–skill capability improves the project performance measure project duration. Efficiencies are assumed to be homogeneous and static. Drexl [31] considers the completely–skilled case where each resource (with homogeneous and static efficiency) has an overall capacity for the entire planning horizon and cannot process more than one activity within each period. A number of papers treat the audit scheduling problem (see e.g. Dodin and Elimam [30] and the literature cited there) which can be categorized as multi–skill case with static and homogeneous efficiencies. Alba and Chicano [3] present a

project scheduling problem which is similar to an RCPSP. However, multiple objectives (time, cost, quality) are considered, resources can be assigned fractionally, resources are multi–skilled with homogeneous efficiencies, and tasks require multiple skills. The time required for completing a task depends on the number and fractions of assigned resources. The problem is not explicitly modelled as a mathematical program but only qualitatively described. A genetic algorithm is proposed for solving the problem.

Project selection and staffing Taylor et al. [68] and Yoshimura et al. [79] assume a single period and thus consider project selection and staffing with static efficiencies. Taylor et al. [68] consider the single–skill case with homogeneous efficiencies. Project performance measures such as project success and project duration depend on the amount of resources assigned to a project. A goal programming model takes the different success measures for the project portfolio into account. Yoshimura et al. [79] propose a three step approach within an R&D–context (where only the following two steps are, in the context of this paper, relevant): First, projects are selected taking into account the availability of different skills. Afterwards, human resources are assigned to the project work by considering heterogeneous efficiencies.

Project selection, scheduling, and staffing Ballou and Tayi [8], Gutjahr and Reiter [39] as well as Gutjahr et al. [40, 41] treat the problem of project selection, scheduling, and staffing. All of them decompose the entire problem into cascaded subproblems which are then solved separately. Ballou and Tayi [8] solve the project selection and scheduling with a binary program where columns relate to predefined schedules. Afterwards, they solve a sequence of transportation problems, each for one period of the planning horizon, in order to assign human resources to capacity demand of projects. Thereby, heterogeneous and dynamic efficiencies are considered implicitly by setting the cost–parameter of the transportation problem for each combination of project and resource accordingly. The cost–parameter reflects how good the human resource can accomplish the project work. Learning and forgetting is considered by altering the cost–parameter from one period to the next. Gutjahr et al. [40] propose a mixed–integer non–linear program where the work of selected projects is allocated within pre–specified time windows and human resources are assigned to the work. Again, heterogeneous and dynamic efficiencies are considered. The problem is decomposed by applying a metaheuristic for project selection and, subsequently, a greedy priority–based heuristic for project scheduling and staffing. Gutjahr et al.

[41] consider a multi-objective generalization of the problem while Gutjahr and Reiter [39] consider a bi–objective stochastic generalization.

The approach presented in this chapter differs from the reviewed papers as follows: Other than Wu and Sun [76] the problem is modelled as a mixed–integer linear program which, due to a tight LP–relaxation, can be solved close to optimality with standard MIP–solvers even for large real–world problems (results will be presented in Section 2.4). In contrast to Alfares and Bailey [4], Bassett [10] as well as Bellenguez-Morineau and Néron [15] this model assumes heterogeneous efficiencies. Neither address project selection (cf. Ballou and Tayi [8], Yoshimura et al. [79]) nor learning and forgetting (i.e. dynamic efficiencies) are considered in contrast to Gutjahr and Reiter [39] as well as Gutjahr et al. [40, 41]. Opposed to [39–41] projects are discretely scheduled but as in Alfares and Bailey [4], Bassett [10], and Wu and Sun [76] internal and external resources (outsourcing) are explicitly considered. A fundamental assumption of the following model is that each human resource can work in each period at multiple projects, possibly using different skills (continuous fractional assignments, cf. e.g. Alba and Chicano [3], Ballou and Tayi [8], Bassett [10], Gutjahr and Reiter [39], Gutjahr et al. [40, 41], and Yoshimura et al. [79]). This assumption differs from those of e.g. Valls et al. [71] as well as Wu and Sun [76].

2.3 Model

2.3.1 Model description

Projects. Let $\mathcal{P}$ be a set of projects which have to be processed. In total the projects require the set of skills $\mathcal{S}$. In the context of IT–projects skills are for example programming, architecture, security, or hardware. The schedule but not the start of each project is assumed to be given. More precisely, for each project $p \in \mathcal{P}$ a duration d_p is given and in project period $q = 1, \ldots, d_p$ project p requests r_{psq} work units of skill s. The amount of skill s requested in the q–th period of project p, r_{psq}, is termed as work package. Project p must start within the time window $[ES_p, LS_p]$ where ES_p denotes the earliest and LS_p denotes the latest start period of project p. The latest finish period LF_p of project p can then be calculated as $LF_p = LS_p + d_p - 1$. The time line goes from period $t = 1$ to period $t = T$ where T denotes the planning horizon. Time period t is defined as the time span between the points in

time $t-1$ and t, i.e. $[t-1, t[$ for $t = 1, \ldots, T$. W.l.o.g. for the remainder of this chapter a period length of one month is assumed.

Resources. As stated above, the projects do in total require skills out of the set $\mathcal{S}$ in order to be processed. The source of skills are human resources within the company (internal resources $\mathcal{R}^i$) and outside of the company (external resources $\mathcal{R}^e$). $\mathcal{R} = \mathcal{R}^i \cup \mathcal{R}^e$ denotes the total set of human resources. The subset of skills that resource $k \in \mathcal{R}$ is capable of is denoted with $\mathcal{S}_k \subseteq \mathcal{S}$. From the skill point of view $\mathcal{R}_s = \mathcal{R}^i_s \cup \mathcal{R}^e_s$ denotes the set of resources which do have skill $s \in \mathcal{S}$.

Even if two resources have the same skill they might differ w.r.t. the level the skill is performed. In order to depict this the concept of efficiency is employed. The efficiency that resource $k \in \mathcal{R}_s$ performs work which requires skill s is denoted by $\eta_{sk} > 0$. The reciprocal $\frac{1}{\eta_{sk}}$ is the time needed by resource $k \in \mathcal{R}_s$ when performing one unit of work requiring skill s. The efficiencies are static across the planning horizon (cf. Chapter 4 for time–varying efficiencies due to learning and forgetting). For performing work an internal human resource $k \in \mathcal{R}^i$ has a regular capacity of R^r_{kt} and an overtime capacity of R^o_{kt} time units in period t. The cost per time unit of internal resource $k \in \mathcal{R}^i$ working in regular– and overtime–mode is c^r_k and c^o_k, respectively.

External resources depict resources which are hired on a temporarily basis or who perform outsourced work. It is assumed that for each skill $s \in \mathcal{S}$ external resources can be acquired in an unlimited amount for a cost of c^e_s per time unit. External resources will henceforth be aggregated and not considered individually. An obvious assumption is $c^e_s > c^o_k > c^r_k, \forall s, k$ because otherwise, in the case of cost minimization, all work would be assigned to external resources.

The use of external resources is limited in each project. A minimum ratio $e_p \geq 0$ of the work performed by internal resources to the work performed by external resources in project p has to be maintained. The reason for these constraints are manifold: Core competencies shall be kept within the company, the management of the project shall be performed internally, and a minimum of internal knowledge shall be employed.

Variables. The following decision variables are employed: $x^r_{ptsk} \geq 0$ is the amount of work (typically measured in full time equivalents) that internal resource k performs during regular hours in period t on the work packages of project p which require skill s. Accordingly, $x^o_{ptsk} \geq 0$ is the amount of overtime work of the internal resource k. $y_{pts} \geq 0$ is the amount of work of

project p requiring skill s performed by external resources in period t. These decision variables are continuous. The aggregation of resource demand to periods justifies the use of continuous assignment decisions where in each period a fraction of the resource capacity may be assigned to a work package. This approach is common practice in IT–projects (cf. e.g. Alba and Chicano [3], Ballou and Tayi [8]). Finally, the binary decision variables $z_{pt} \in \{0, 1\}$ are required for setting project start times: z_{pt} equals 1 if project p is started at the beginning of period t and 0 otherwise. Table A.1 in the Appendix provides a summary of the notation.

The question is how the start times of the projects should be scheduled and which human resources should be assigned to each work package such that the different requirements are met and costs are minimized.

2.3.2 MIP Formulation

The following mixed–binary linear program MIP1 represents the aforementioned problem.

$$\text{Min} \quad \sum_{p\in\mathcal{P}} \sum_{t=ES_p}^{LF_p} \sum_{s\in\mathcal{S}} \left(c^e_s y_{pts} + \sum_{k\in\mathcal{R}^i_s} \frac{1}{\eta_{sk}} \left(c^r_k x^r_{ptsk} + c^o_k x^o_{ptsk} \right) \right) \tag{2.1}$$

subject to

$$\sum_{t=ES_p}^{LS_p} z_{pt} = 1 \qquad p \in \mathcal{P} \tag{2.2}$$

$$r_{psq} z_{p\tau} \le y_{pts} + \sum_{k\in\mathcal{R}^i_s} \left(x^r_{ptsk} + x^o_{ptsk} \right) \qquad \begin{array}{l} p \in \mathcal{P} \\ \tau = ES_p, \ldots, LS_p \\ q = 1, \ldots, d_p \\ t = \tau + q - 1 \\ s \in \mathcal{S} \end{array} \tag{2.3}$$

$$\sum_{p\in\mathcal{P}} \sum_{s\in\mathcal{S}} \frac{1}{\eta_{sk}} x^r_{ptsk} \le R^r_{kt} \qquad \begin{array}{l} t = 1, \ldots, T \\ k \in \mathcal{R}^i \end{array} \tag{2.4}$$

$$\sum_{p\in\mathcal{P}} \sum_{s\in\mathcal{S}} \frac{1}{\eta_{sk}} x^o_{ptsk} \le R^o_{kt} \qquad \begin{array}{l} t = 1, \ldots, T \\ k \in \mathcal{R}^i \end{array} \tag{2.5}$$

$$\sum_{t=ES_p}^{LF_p} \sum_{s\in\mathcal{S}} \sum_{k\in\mathcal{R}_s^i} \left(x_{ptsk}^r + x_{ptsk}^o\right) \geq e_p \sum_{t=ES_p}^{LF_p} \sum_{s\in\mathcal{S}} y_{pts} \qquad p \in \mathcal{P} \tag{2.6}$$

$$x_{ptsk}^r, x_{ptsk}^o \geq 0 \qquad \begin{array}{l} p \in \mathcal{P} \\ t = ES_p, \ldots, LF_p \\ s \in \mathcal{S} \\ k \in \mathcal{R}_s^i \end{array} \tag{2.7}$$

$$y_{pts} \geq 0 \qquad \begin{array}{l} p \in \mathcal{P} \\ t = ES_p, \ldots, LF_p \\ s \in \mathcal{S} \end{array} \tag{2.8}$$

$$z_{pt} \in \{0,1\} \qquad \begin{array}{l} p \in \mathcal{P} \\ t = ES_p, \ldots, LS_p \end{array} \tag{2.9}$$

The objective function (2.1) minimizes the labor costs of internal and external resources which accrue by processing the work packages of the projects. Note that the time required by the internal resource k to perform x_{ptsk} work units depends on the efficiency η_{sk}. I.e. in case of a high efficiency ($\eta_{sk} > 1$) less time is needed while in case of a low efficiency ($\eta_{sk} < 1$) more time is needed. For all external resources an efficiency of $\eta = 1$ is assumed. Due to constraints (2.2) each project p is forced to start exactly once within its start time window $[ES_p, LS_p]$. Constraints (2.3) ensure that the work packages of project p are performed by internal and external resources. More precisely, for each period $t = ES_p, \ldots, (LS_p + d_p - 1)$ the required work of skill s has to be covered by internal or external resources. The work requiring skill s which has to be performed in period t depends on the start time z_{pt} and the schedule r_{psq} of the projects. The available capacities of the internal resources for regular time and overtime are considered by constraints (2.4) and (2.5), respectively. Constraints (2.6) ensure for each project p a minimum ratio e_p of the work performed by internal resources to the work performed by external resources. Finally, the constraints (2.7), (2.8) and (2.9) define the decision variables.

Proposition 1. *The problem of simultaneous scheduling and staffing multiple projects is NP–hard.*

Proof. The considered problem can be restricted to a multiprocessor scheduling problem by allowing only instances with $T = 2$, $ES_p = 1$, $LS_p = 2$, $d_p = |\mathcal{R}^i| = |\mathcal{S}| = |\mathcal{S}_k| = \eta_{sk} = 1$, $c_s^e > 0$, $c_k^r = e_p = R_{kt}^o = 0$, and $R_{kt}^r = \frac{1}{T}\sum_{p,s,q} r_{psq}$. Figure 2.1 depicts the correspondence between the restricted scheduling and staffing problem (left) and the multiprocessor scheduling

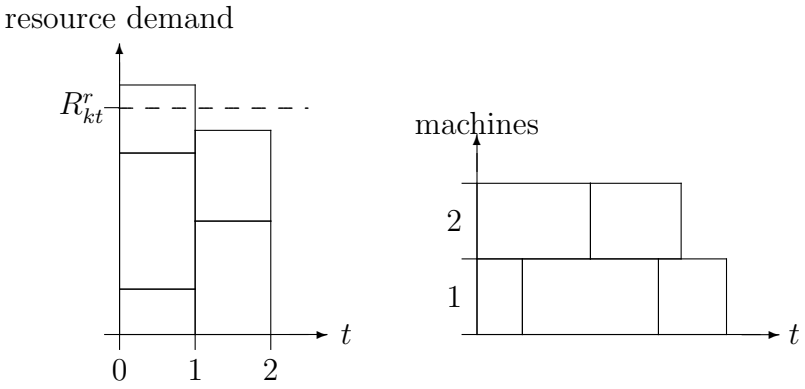


Figure 2.1: Analogy to the multiprocessor scheduling problem

problem (right): Periods correspond to machines, projects correspond to jobs, and resource demands correspond to the jobs' lengths. The objective of minimizing costs corresponds to minimizing makespan of the multiprocessor scheduling problem. Since the multiprocessor scheduling problem is known to be NP–hard (cf. SS8 in Garey and Johnson [34]), the considered problem as a generalization of the multiprocessor scheduling problem must also be NP–hard.

□

2.3.3 Model extensions

A possible extension of MIP1 are the following budget–constraints for each project p where B_p denotes the budget of project p:

$$\sum_{t=ES_p}^{LF_p} \sum_{s\in\mathcal{S}} \left(c^e_s y_{pts} + \sum_{k\in\mathcal{R}^i_s} \frac{1}{\eta_{sk}} \left(c^r_k x^r_{ptsk} + c^o_k x^o_{ptsk} \right) \right) \leq B_p \quad p \in \mathcal{P} \tag{2.10}$$

If external resources are considered to be scarce, MIP1 could be extended with the following constraints, where R^e_{ts} denotes the external capacity of skill s in period t.

$$\sum_{p\in\mathcal{P}} y_{pts} \leq R^e_{ts} \qquad \begin{matrix} t = 1,\ldots,T \\ s \in \mathcal{S} \end{matrix} \tag{2.11}$$

Finally, labor contracts might require, that external resources may only be hired in discrete quantities, e.g. multiples of u. To ensure this requirement

non–negative integer decision variables y^i_{pts} need to be added for project p, period t and skill s. Additionally, the following constraints have to be appended to MIP1.

$$y_{pts} = y^i_{pts} u \qquad \begin{array}{l} p \in \mathcal{P} \\ t = 1, \ldots, T \\ s \in \mathcal{S} \end{array} \tag{2.12}$$

$$y^i_{pts} \in \mathbb{Z}^+_0 \qquad \begin{array}{l} p \in \mathcal{P} \\ t = 1, \ldots, T \\ s \in \mathcal{S} \end{array} \tag{2.13}$$

2.3.4 Improved MIP Formulation

MIP1 has the drawback that its LP–relaxation (LP1) is not very tight due to the constraints (2.3). The latter ensure for each project p and skill s that the work scheduled in period $t = ES_p, \ldots, (LS_p + d_p - 1)$ is assigned to internal or external resources. Note that in the case of strictly positive cost coefficients c the constraints (2.3) are always binding in LP1. It is now shown that LP1 does assign less than the total demand of project p to the resources if the size of the start time window is positive and the solution is non–integer. For this let γ_{pts} denote the right hand side (rhs) of constraints (2.3), i.e.

$$\gamma_{pts} := y_{pts} + \sum_{k \in \mathcal{R}^i_s} \left(x^r_{ptsk} + x^o_{ptsk} \right) . \tag{2.14}$$

Without loss of generality only a single project p and a single skill s is considered such that the indices p and s can be omitted, i.e. $\gamma_t \equiv \gamma_{pts}$. Then, the following proposition can be stated.

Proposition 2. *For $LS - ES > 0$, $d > 1$ and $e = 0$, LP1 with a non–integer solution, i.e. $\exists\, t : 0 < z_t < 1$, assigns less than the total demand $\sum_{q=1}^{d} r_q$ to the resources. I.e. $\sum_{q=1}^{d} r_q > \sum_{t=ES}^{LF} \gamma_t$ holds.*

Proof. The proposition is proved by contradiction: Let $d = 2$, $0 < r_1 < r_2$, $ES = 1$, $LS = 2$, and hence $t = 1, \ldots, 3$. Using matrix notation the following constraints (2.3) are then obtained.

$$\begin{pmatrix} r_1 & 0 \\ 0 & r_1 \\ r_2 & 0 \\ 0 & r_2 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \leq \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_2 \\ \gamma_3 \end{pmatrix} \tag{2.15}$$

Each row of (2.15) represents a combination of a project start period τ and a project period q which results in at least one resource demand for each calendar period t. The number of constraints that are obtained for one project p and calendar period t is bounded from above by $\min(d, LS-ES+1)$. For a MIP–solution not more than one of these constraints will have a non–zero left hand side (lhs) where the resource demand has to be assigned to a resource. However, for an LP–relaxation usually $0 < z_t < 1$ holds. Hence up to $\min(d, LS - ES + 1)$ constraints will have a non–zero lhs where only the constraints with maximum lhs will hold whereas for the remaining constraints no resources will be assigned to demand.

Assume for the example an LP–solution $z_1 = z_2 = 0.5$. Since $r_1 < r_2$ for period 2 the third constraint will hold while the second constraint becomes redundant. The total capacity allocated to demand is then $\sum_{t=ES}^{LF} \gamma_t = 0.5r_1 + 0.5r_2 + 0.5r_2$ which is less than total demand $\sum_{q=1}^{d} r_q = r_1 + r_2$. □

Note that for some $e > 0$ constraints (2.6) eventually force $\sum_{q=1}^{d} r_q \leq \sum_{t=ES}^{LF} \gamma_t$ even for LP1. In what follows a modification of constraints (2.3) is proposed which gives a tighter LP–relaxation for any value of e. Let

$$\mathcal{T}_{pt} = \{(\tau, q) \in \{ES_p, \ldots, LS_p\} \times \{1, \ldots, d_p\} \,|\, \tau + q - 1 = t\} \tag{2.16}$$

be a set which consists of all combinations (τ, q) of project start periods τ and project periods q of project p that lead to a resource demand in calendar period t. Figure 2.2 visualizes the four sets which exist for $[ES_p, LS_p] = [2, 3]$ and $d_p = 3$. Employing $\mathcal{T}_{pt}$ the new assignment–constraints can be formulated as follows.

$$\sum_{(\tau,q)\in\mathcal{T}_{pt}} r_{psq} z_{p\tau} \leq y_{pts} + \sum_{k\in\mathcal{R}_s^i} \left(x^r_{ptsk} + x^o_{ptsk}\right) \qquad \begin{array}{l} p \in \mathcal{P} \\ t = ES_p, \ldots, LF_p \\ s \in \mathcal{S} \end{array} \tag{2.17}$$

The improved MIP2 is then (2.1)–(2.2), (2.17), and (2.4)–(2.9). It is now shown that MIP2 gives a tighter LP–relaxation LP2 for the problem.

Proposition 3. *LP2 always assigns the total demand* $\sum_{q=1}^{d} r_q$ *to the resources. I.e.* $\sum_{q=1}^{d} r_q \leq \sum_{t=ES}^{LF} \gamma_t$ *holds.*

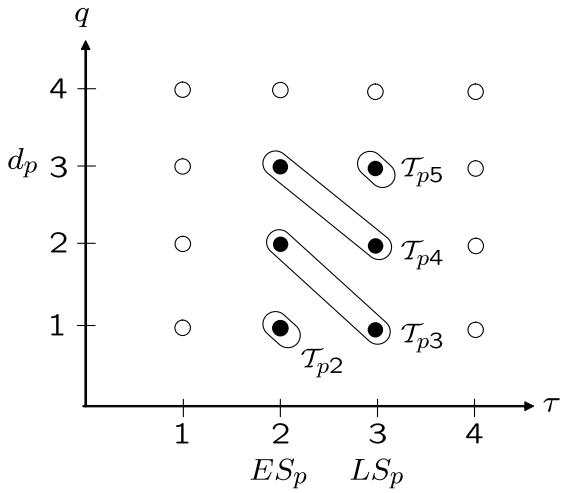


Figure 2.2: Visualization of $\mathcal{T}_{pt}$ for $d_p = 3$, $ES_p = 2$, and $LS_p = 3$

Proof. To prove this proposition the matrix formulation of constraints (2.17) is analysed for one arbitrary combination of project p and skill s:

$$\underbrace{\left\{ \begin{pmatrix} r_1 & 0 & 0 & \cdots & 0 \\ r_2 & r_1 & 0 & & \vdots \\ \vdots & r_2 & r_1 & \ddots & \vdots \\ r_d & \vdots & r_2 & \ddots & 0 \\ 0 & r_d & \vdots & \ddots & r_1 \\ 0 & 0 & r_d & & r_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & r_d \end{pmatrix} \right\}}_{LS-ES+1} LF-ES+1 \qquad \begin{pmatrix} z_{ES} \\ z_{ES+1} \\ \vdots \\ z_{LS} \end{pmatrix} \leq \begin{pmatrix} \gamma_{ES} \\ \gamma_{ES+1} \\ \vdots \\ \gamma_{LF} \end{pmatrix} \tag{2.18}$$

Since each r_q $(q = 1, \ldots, d)$ appears in all elements of exactly one secondary diagonal (and nowhere else), each r_q is multiplied exactly once with a different z_t $(t = ES, \ldots, LS)$. Summing up the lhs of (2.18) gives

$$\sum_{t=ES}^{LS} \sum_{q=1}^{d} (r_q z_t) = \sum_{t=ES_p}^{LS_p} z_t \sum_{q=1}^{d} r_q = 1 \sum_{q=1}^{d} r_q = \sum_{q=1}^{d} r_q \tag{2.19}$$

Summing up the rhs side of (2.18) gives $\sum_{t=ES}^{LF} \gamma_t$. Thus, $\sum_{q=1}^{d} r_q \leq \sum_{t=ES}^{LF} \gamma_t$ holds which proves the proposition.

□

2.4 Computational study

2.4.1 Test instances

Test instances were generated inspired by data of the IT–department of a large semiconductor manufacturer (cf. Heimerl [42]). In Table 2.1 parameters for a base case are defined which is varied using a factorial ceteris paribus design with four factors.

$\lvert\mathcal{P}\rvert$	$=$	20	$\lvert\mathcal{S}\rvert$	$=$	25
TWS_p	$=$	0	$\lvert\mathcal{S}_k\rvert$	$=$	4
ES_p	$\sim$	$U(1,7)$	$\lvert\mathcal{R}^i\rvert$	$=$	100
d_p	$=$	6	R^r_{kt}	$=$	20 $(t = 6,\ldots,12)$
T	$=$	12	R^o_{kt}	$=$	$0.3R^r_{kt} = 6$ $(t = 6,\ldots,12)$
$\lvert\mathcal{S}\rvert$ per period and project	$=$	3	η_{sk}	$\sim$	$TN_{0.5,1.5}(1, 0.25)$
$\lvert\mathcal{S}\rvert$ per project	$\leq$	4	c^r_k	$=$	500
e_p	$=$	0.2	c^o_k	$=$	$1.2c^r_k = 600$
B_p	$=$	∞	c^e_s	$\sim$	$TN_{600,1000}(800, 100)$
			ρ	$=$	2.0

Table 2.1: Parameters for the base case

The time window size of project p is defined as

$$TWS_p = LS_p - ES_p \quad . \tag{2.20}$$

The earliest start periods ES_p of the projects were drawn from a discrete uniform distribution between period 1 and 7. Together with a deterministic project length of $d_p = 6$ this leads to a planning horizon of $T = 12$. The skills required by the projects and owned by the internal resources were chosen randomly. The efficiencies of the internal resources were drawn from a truncated normal distribution $TN_{a,b}(\mu, \sigma)$ with the expected value $\mu = 1$, a standard deviation of $\sigma = 0.25$, a minimum efficiency of $a = 0.5$, and a maximum efficiency of $b = 1.5$. The cost rates for external resources were also drawn from a truncated normal distribution $TN_{a,b}(\mu, \sigma)$ with $\mu = 800$, $\sigma = 100$, $a = 600$, and b =1,000.

To mimic a dynamic situation, where projects arrive with a constant arrival rate and where work packages of projects which have been started in the past have already been assigned to resources, the availability of internal resources was modelled according to the function depicted in Figure 2.3.

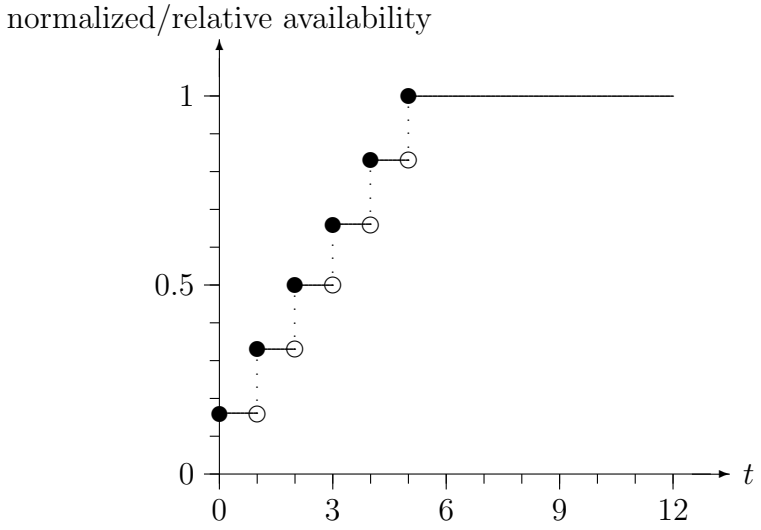


Figure 2.3: Normalized availability of an internal resource as a step function

Thus in period $t = 0$, which is directly preceding the planning horizon, internal resources do have no left–over capacity for projects with $ES_p \geq 0$. Since the duration of all projects is $d_p = 6$ the available capacity increases in a step–wise fashion until the full capacity is provided in period $t = 6$.

The workload ρ is defined by the ratio of the expected resource demand to the availability of internal resources in period $t = d_p = 6$ (when all projects which have been started before the planning horizon are finished) as

$$\rho = \frac{\sum_{p \in \mathcal{P}} \sum_{s \in \mathcal{S}} \mathbb{E}\left(r_{psq} | t = d_p\right)}{\sum_{k \in \mathcal{R}^i} (R^r_{kd_p} + R^o_{kd_p})}. \tag{2.21}$$

$\mathbb{E}\left(r_{psq} | t = d_p\right)$ denotes the expected demand of project p for skill s in period $t = d_p$. With $r_{psq} = 0$ for all $q < 1$ and $q > d_p$ the expected demand can be calculated according to

$$\mathbb{E}\left(r_{psq} | t = d_p\right) = \frac{1}{TWS_p + 1} \sum_{q=d_p - LS_p + 1}^{d_p - ES_p + 1} r_{psq}. \tag{2.22}$$

Given a workload level ρ and resource supply R^r and R^o the required expected resource demand $\mathbb{E}\left(r_{psq}\right)$ of the work packages is calculated according to Equation (2.21). The actual demand of the individual work packages was then drawn from a normal distribution with $\mu = \mathbb{E}\left(r_{psq}\right)$ and a coefficient of variation $COV = 0.1$ (i.e. $\sigma = 0.1\mathbb{E}\left(r_{psq}\right)$).

For the ceteris paribus design the base case was altered by varying the following four factors one at a time: The expected time window size $\mathbb{E}(TWS_p)$, the number of projects $|\mathcal{P}|$, the number of skills per resource $|\mathcal{S}_k|$, and the workload ρ.

Table 2.2 lists the factors and their employed levels. For each factor level 10 instances were generated which led to a total of $10 \cdot (5 + 5 + 6 + 5) = 210$ test instances.

Factors	Experimental levels
$\mathbb{E}(TWS_p)$	$0.0, 0.5, 1.0, 1.5, 2.0$
$\|\mathcal{P}\|$	$10, 20, 30, 40, 50$
$\|\mathcal{S}_k\|$	$1, 2, 4, 6, 8, 10$
ρ	$1.0, 1.5, 2.0, 2.5, 3.0$

Table 2.2: Factors and levels for the experimental test design

The time window sizes were generated using minimum variance, i.e. for $\mathbb{E}(TWS_p) = 1.0$ all projects had $TWS_p = 1$ and if $\mathbb{E}(TWS_p) = 1.5$, one half of the projects had $TWS_p = 1$ and the other half had $TWS_p = 2$. Furthermore, to keep $T = 12$ even with larger time window sizes, $LS_p \leq 7$ holds.

The tests were performed on an Intel P4 with 2.4 GHz and 512 MB RAM using ILOG CPLEX 10.1.

2.4.2 Solution gaps

The solution gaps of the LP–Relaxations for MIP1 and MIP2 were compared. The solution gaps are defined as $\Delta = \frac{Opt-LB}{Opt}$, where Opt is the objective function value of the optimal solution of the MIP and LB is the lower bound which is calculated by solving the LP–Relaxation to optimality. Table 2.3 shows the mean μ and the standard deviation σ of the gap for both MIP–formulations and different time window sizes.

	LP1		LP2	
$\mathbb{E}(TWS_p)$	μ	σ	μ	σ
0.0	0.00%	0.00%	0.00%	0.00%
0.5	17.77%	0.84%	0.11%	0.06%
1.0	36.90%	1.47%	0.28%	0.13%
1.5	44.48%	1.45%	0.30%	0.15%
2.0	51.64%	1.22%	0.34%	0.15%

Table 2.3: Solution gaps Δ of the two LP–Relaxations

For a time window size of 0 project p has exactly one feasible start period $ES_p = LS_p$ and hence both LP–relaxations give the optimal value of the MIP. For an increasing time window size it can be seen that the gaps of both MIPs increase, but that the size of the gap and the increase of the gap are much smaller for MIP2. A close look at the parameters explains why the gaps of MIP2 are surprisingly small. The cost rates for internal resources are $c_k^r = 500 < c_k^o < c_s^e$. Since all work packages must be processed, there is a lower bound $C_{\min}$ of considerable size for the costs which are inevitable regardless of the scheduling and assignment decisions. The latter distorts the results in favour of the solution gaps of both MIPs. The modified solution gap

$$\Delta' = \frac{(Opt - C_{\min}) - (LB - C_{\min})}{Opt - C_{\min}} = \frac{Opt - LB}{Opt - C_{\min}} \tag{2.23}$$

takes into account the lower bound $C_{\min}$ in order to correct the distortion. The lower bound

$$C_{\min} = \sum_{s \in \mathcal{S}} \left(X_s^r \min_{k \in \mathcal{R}^i} \left(\frac{c_k^r}{\eta_{sk}} \right) + X_s^o \min_{k \in \mathcal{R}^i} \left(\frac{c_k^o}{\eta_{sk}} \right) + Y_s c_s^e \right) \tag{2.24}$$

is calculated using

$$X_s^r = \min \left\{ \sum_{p \in \mathcal{P}} \sum_{q=1}^{d_p} r_{psq}, \sum_{k \in \mathcal{R}_s^i} \sum_{t=1}^{T} \eta_{sk} R_{kt}^r \right\} \tag{2.25}$$

$$X_s^o = \min \left\{ \sum_{p \in \mathcal{P}} \sum_{q=1}^{d_p} r_{psq} - X_s^r, \sum_{k \in \mathcal{R}_s^i} \sum_{t=1}^{T} \eta_{sk} R_{kt}^o \right\} \tag{2.26}$$

and

$$Y_s = \sum_{p \in \mathcal{P}} \sum_{q=1}^{d_p} r_{psq} - X_s^r - X_s^o \,. \tag{2.27}$$

X_s^r as calculated in Equation (2.25) is an upper bound for the amount of work which can be performed by the internal resources in regular time. Taking into account X_s^r, Equation (2.26) calculates an upper bound for the amount of work which can be performed by the internal resources in overtime. Equation (2.27) assigns the remaining work to be done by external resources.

Table 2.4 provides the values for the adjusted gap Δ'. Much more clearly than in the case of Δ it can be seen that MIP1 performs rather poor while MIP2 generates extremely sharp bounds.

	LP1		LP2	
$\mathbb{E}(TWS_p)$	μ	σ	μ	σ
0.0	0.00%	0.00%	0.00%	0.00%
0.5	124.11%	21.28%	0.78%	0.48%
1.0	282.50%	48.79%	2.21%	1.42%
1.5	341.38%	57.29%	2.46%	1.55%
2.0	387.79%	62.94%	2.74%	1.57%

Table 2.4: Adjusted solution gaps Δ' of the two LP–Relaxations

2.4.3 Computation time

In what follows the results regarding the computation time needed for calculating the optimal solution of MIP2 are presented. The size of the time window and the relative MIP gap tolerances of CPLEX was altered according to the values given in Table 2.5. For increasing time windows and hence an increasing number of binary variables a sharp increase of computation times is observed. However, a solution gap of 1% leads to sufficiently good solutions in less than 3 seconds.

	relative MIP gap tolerances									
	0.01%		0.1%		1%		5%		10%	
$\mathbb{E}(TWS_p)$	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
0.5	2.6	0.9	1.4	1.0	0.8	0.1	0.8	0.1	0.8	0.1
1.0	14.1	16.6	10.9	13.9	1.5	0.4	1.5	0.4	1.5	0.4
1.5	26.2	33.2	22.2	33.9	2.2	0.6	2.1	0.6	2.1	0.7
2.0	62.2	121.6	49.3	92.4	2.8	1.1	2.7	0.9	2.7	0.9

Table 2.5: Solution times (seconds) for different time window sizes and relative MIP gap tolerances

The results for the other factors are shown in Table 2.6: The number of projects $|\mathcal{P}|$ and the number of skills per internal resource $|\mathcal{S}_k|$ have, due to the growing number of continuous variables, a linear influence on the computation times. The impact of the workload–level on the computation time is not linear. It can only be observed that instances with a workload of $\rho = 1$ are faster to solve than those with higher workloads.

$\|\mathcal{P}\|$	μ	σ	$\|\mathcal{S}_k\|$	μ	σ	ρ	μ	σ
10	0.20	0.02	1	0.10	0.02	1.0	0.44	0.05
20	0.48	0.06	2	0.22	0.02	1.5	0.68	0.04
30	0.84	0.07	4	0.55	0.07	2.0	0.67	0.09
40	1.30	0.13	6	0.81	0.15	2.5	0.66	0.04
50	1.76	0.18	8	1.07	0.17	3.0	0.60	0.05
			10	1.37	0.15			
			25	4.32	0.85			
Total	0.91	0.58		1.20	1.39		0.61	0.11

Table 2.6: Solution times (in seconds) for different factors

2.5 Managerial insight

2.5.1 Influences of the factors on the costs

In what follows the impact of the systematically varied factors on the optimal costs will be analysed. Table 2.7 provides the results in a compact form. In addition Figures 2.4 – 2.7 visualize the results by providing the mean and the range of ± 1 standard deviation of the optimal objective function value for each level of the varied factor, respectively.

Time window size. The time window size is expected to have a negative influence on the costs. I.e. increasing time window sizes of the projects are used by MIP2 for levelling the demand profile to avoid excess of demand and thus the use of expensive overtime and external resources. As can be seen in Figure 2.4 and Table 2.7, the latter holds when increasing the average time window size from 0 to 1. The costs decrease by 3.2%. But a further increase of the time window size does not show significant effects. The latter observation is not expected. The explanation for this is as follows: Since ES_p has been distributed uniformly and d_p has been set to be constant, the demand profile is, by the way the data has been generated, already levelled to a large amount. Hence, a small increase of the start time window suffices to level the demand profile almost to optimality. A further increase of the time window size will not be of any use. Applying an analysis of variance (ANOVA) a significance level of $\alpha = 0.07$ is obtained for the whole range of time window sizes.

Number of projects. Figure 2.5 displays the influence of the number of projects on the optimal costs. The observed level of significance is $\alpha < 0.001$. An explanation of this effect is as follows: The level of the number of projects

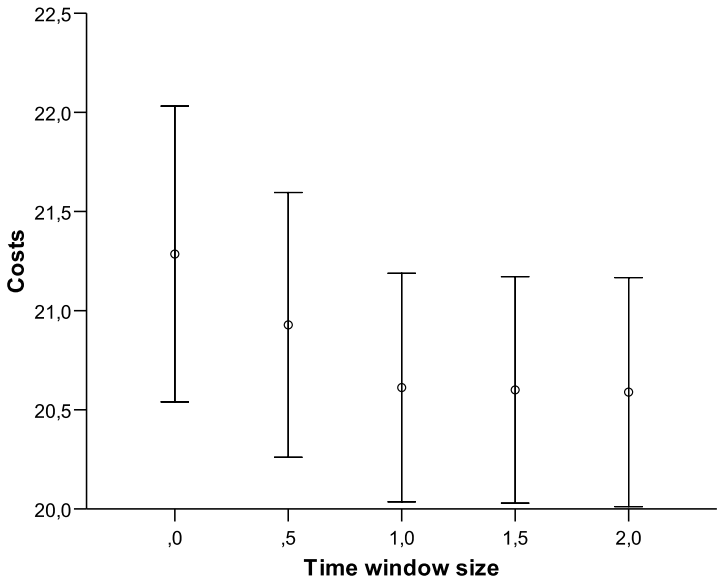


Figure 2.4: Impact of the time window size on the costs

was altered while keeping the total resource demand constant. An increasing number of projects is then equivalent to a decrease in the resource demand per project. This opens for MIP2 more planning flexibility which leads to less costs.

Number of skills of internal resources. Since the test data does not account costs for different skill levels of the internal resources (higher cost rates for employees with more skills) decreasing costs are expected for an increase in the average number of skills per resource. This is affirmed in Figure 2.6 with an $\alpha < 0.001$ level of significance. The costs decrease monotonically with increasing number of skills per resource. Note that the marginal decrease of costs is diminishing and hence one could determine an "cost–optimal qualification level" for internal resources if the costs of qualification would be considered. The dashed line represents the costs if each resource possesses all skills.

Workload. The workload level ρ shows an $\alpha < 0.001$ level of significance. Since the capacity of internal resources is fixed, external resources have to be assigned to the resource demand which exceeds the capacity of the internal resources. This leads to a linear increase of the costs.

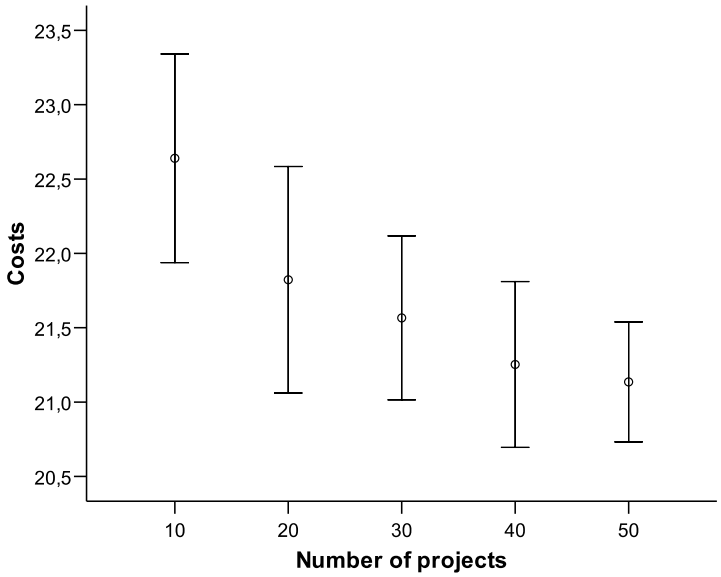


Figure 2.5: Impact of the number of projects on the costs

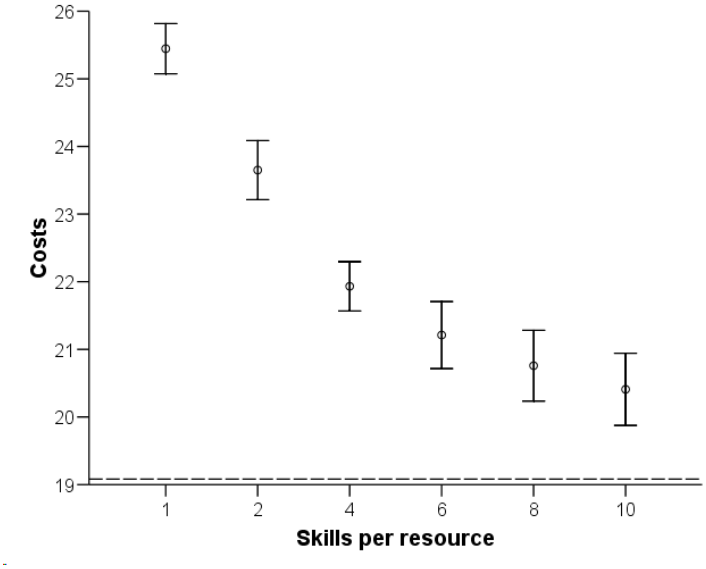


Figure 2.6: Impact of the average number of skills per resource on the costs

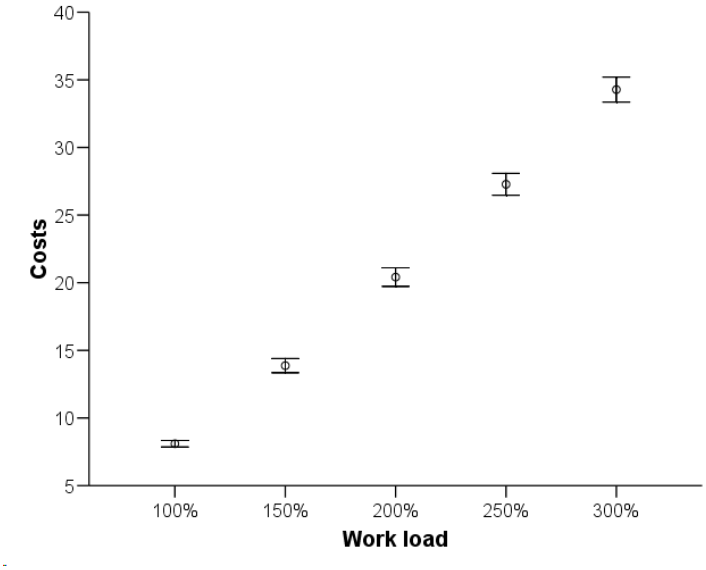


Figure 2.7: Impact of the workload on the costs

$\mathbb{E}(TWS_p)$	μ	σ	$\lvert\mathcal{P}\rvert$	μ	σ
0.0	21.29	0.75	10	22.64	0.70
0.5	20.93	0.67	20	21.82	0.76
1.0	20.61	0.58	30	21.57	0.55
1.5	20.60	0.57	40	21.25	0.56
2.0	20.59	0.58	50	21.14	0.40
Total	20.80	0.66		21.68	0.80

$\lvert\mathcal{S}_k\rvert$	μ	σ	ρ	μ	σ
1	25.45	0.37	1.0	8.10	0.24
2	23.65	0.44	1.5	13.87	0.53
4	21.93	0.36	2.0	20.41	0.68
6	21.21	0.50	2.5	27.28	0.81
8	20.76	0.52	3.0	34.28	0.92
10	20.41	0.53			
25	19.08	0.62			
Total	21.78	2.05		20.79	9.42

Table 2.7: Optimal costs (in millions) for the different factor levels

2.5.2 Optimal vs. heuristic planning

The vast majority of firms performs the allocation of human resources and the scheduling of projects, if at all, not with an optimization–based approach such as MIP2 but manually, maybe with the help of spreadsheet– and database–programs. In what follows the results of the MIP are compared with two heuristic approaches which are currently applied in the IT–department of the semiconductor manufacturer where the problem was encountered: Random Assignment (RND) (cf. Algorithm 1) and Maximum Cost First Assignment (MCF) (cf. Algorithm 2).

Algorithm 1 Random Assignment

1: Set RetryCounter := 5 and $\mathcal{WP} = \emptyset$. For all $p \in \mathcal{P}$ set project start to ES_p
2: Add all work packages r_{pst} to the set $\mathcal{WP}$ in an arbitrary order.
3: **for all** r_{pst} in $\mathcal{WP}$ (sequentially) **do**
4: **for all** internal resources $k \in \mathcal{R}^i_s$ (in arbitrary order) **do**
5: Set $x^r_{ptsk} := \min(R^r_{kt}, r_{pst})$.
6: Set $R^r_{kt} := R^r_{kt} - x^r_{ptsk}$ and $r_{pst} := r_{pst} - x^r_{ptsk}$.
7: **end for**
8: **for all** internal resources $k \in \mathcal{R}^i_s$ (in arbitrary order) **do**
9: Set $x^o_{ptsk} := \min(R^o_{kt}, r_{pst})$.
10: Set $R^o_{kt} := R^o_{kt} - x^o_{ptsk}$ and $r_{pst} := r_{pst} - x^o_{ptsk}$.
11: **end for**
12: Set $y_{pts} := y_{pts} + r_{pst}$.
13: **end for**
14: **if** any of constraints (2.10) or (2.6) is violated by the assignments **then**
15: Set RetryCounter := RetryCounter − 1.
16: **if** RetryCounter == 0 **then**
17: Mark instance as infeasible.
18: **else**
19: Goto Line 1
20: **end if**
21: **end if**

Both heuristics staff the work packages in the order of a list. For the next work package on the list the cost minimal assignment of internal and external resources is done by taking into account the left over capacity of the resources. RND and MCF mainly differ in the way the list is generated. While RND orders the work packages in random order, MCF orders the work packages in the order of decreasing costs of external resources. Due to the

Algorithm 2 Maximum Cost First Assignment

1: Set RetryCounter := 5 and $\mathcal{WP} = \emptyset$. For all $p \in \mathcal{P}$ set project start to ES_p
2: Add all work packages r_{pst} to the *ordered* set $\mathcal{WP}$. *Order the set according to descending values of* c^e_s*, such that externally expensive work packages appear first.*
3: ...

minimum–ratio–of–internal–work constraints (2.6) a solution of each of the heuristics might be infeasible. In the latter case the heuristic starts anew. Since the human resources which are assigned to a work package are selected in arbitrary order (cf. line 4 and 8), a new start might lead to a feasible solution. As soon as a feasible solution has been determined the heuristic stops. The number of restarts allowed has been set to 5.

Heuristic	% feasible solutions	avg. % deviation from opt. (DEV1)	avg. % deviation from opt. (DEV2)
RND	58.09	15.57	16.55
MCF	33.81	13.29	13.46

Table 2.8: Comparison of heuristics

Table 2.8 gives the percentage of feasible solutions, the average deviation from the optimum of the feasible solutions obtained by each heuristic (DEV1), and the average deviation from the optimum of the instances solved by both heuristics (DEV2). Table 2.9 provides further details on the ratio of feasible solutions for different factors. It can be seen that the drawback of the heuristic approaches is not only the higher costs when compared to the optimum solution but the fact that no feasible solution could be generated for many instances. This holds especially for MCF where the order of the work package list is identical for each retry. If MCF finds a solution, the latter is, on average, superior to the one found by RND. The results clearly show that the use of the proposed MIP is advisable.

2.5.3 Central vs. decentral planning

Contingent to their formal organization, companies usually practise a decentral planning where the project scheduling and staff assignment is done within the departments but not across department borders. This approach lowers the complexity of the planning task and thus enables the application of manual planning. In what follows the drawback of the decentral planning

$\mathbb{E}(TWS_p)$	RND	MCF	ρ	RND	MCF
0.0	50%	60%	1.0	100%	100%
0.5	60%	50%	1.5	100%	100%
1.0	50%	50%	2.0	100%	100%
1.5	60%	20%	2.5	100%	50%
2.0	10%	10%	3.0	90%	20%
avg.	46%	38%		98%	74%

$\lvert\mathcal{P}\rvert$	RND	MCF	$\lvert\mathcal{S}_k\rvert$	RND	MCF
10	80%	60%	1	0%	10%
20	70%	0%	2	30%	20%
30	50%	0%	4	50%	30%
40	40%	0%	6	50%	10%
50	20%	0%	8	50%	10%
			10	60%	10%
avg.	52%	12%		40%	15%

Table 2.9: Ratio of feasible solutions with heuristics RND and MCF

approach is assessed from the cost–based perspective. More precisely, the influence of the number of departments and the level of specialization on the staffing costs for central and decentral planning is evaluated.

The company is divided into a set $\mathcal{D}$ of distinct departments. This is done by assuming that the company's set of projects $\mathcal{P}$ and internal resources $\mathcal{R}^i$ can feasibly be partitioned into disjoint subsets $\mathcal{P}_d^D$ and $\mathcal{R}_d^D, d \in \mathcal{D}$, respectively, with the following properties:

(i) The skills that are required by the projects of each department d

$$\mathcal{S}_d^D = \left\{ s \in \mathcal{S} \,|\, r_{psq} > 0 : p \in \mathcal{P}_d^D, q = 1, \ldots, d_p \right\} \tag{2.28}$$

are disjoint, i.e. $\mathcal{S}_i^D \cap \mathcal{S}_j^D = \emptyset \quad \forall i, j \in \mathcal{D}, i \neq j$.

(ii) The cardinalities of the sets $\mathcal{P}_d^D$ are as equal as possible, i.e.

$$\max_{d \in \mathcal{D}} |\mathcal{P}_d^D| \leq 1 + \min_{d \in \mathcal{D}} |\mathcal{P}_d^D| \tag{2.29}$$

holds. Analogously, (2.29) holds for $\mathcal{R}_d^D$ and $\mathcal{S}_d^D$.

(iii) With

$$\delta_{kd} = \frac{\left|\mathcal{S}_k \cap \mathcal{S}_d^D\right|}{|\mathcal{S}_d^D|} \tag{2.30}$$

the degree is measured to which the skills $\mathcal{S}_k$ of resource $k \in \mathcal{R}_d^D$ are required by projects assigned to department d. For $\delta_{kd} = 1$ there is a perfect match, i.e. all skills of resource k are required by the projects of department d. For $\delta_{kd} = 0$ resource k does have none of the skills required by the projects of department d. For $0 < \delta_{kd} < 1$ only a subset of the resource's skills $\mathcal{S}_k$ is required by projects of department d and the resource's other skills are required by some projects of other departments $d' \neq d$.

In what follows decentral planning is defined as solving MIP2 for the internal resources and projects of each department $d \in \mathcal{D}$ separately and merging the separate plans afterwards. Contrary, central planning solves MIP2 once by including the resources and projects of all departments at once. In the latter case it is possible to utilize the skills of resources which are required by projects of other departments. Z^d denotes the optimal objective function value obtained by decentral planning while Z^c represents the optimal objective function value of the central planning approach. Thus, $\kappa = \frac{Z^d - Z^c}{Z^d}$ is the cost improvement which can be leveraged by central vs. decentral planning. Note that $Z^c \leq Z^d$ and hence $\kappa \geq 0$ always holds.

Using the parameters of the base case (cf. Table 2.1) the number of departments $|\mathcal{D}|$ and the specialization measure δ is systematically varied according to Table 2.10. Ten instances were generated for each factor combination of $|\mathcal{D}|$ and δ which lead to $(2 + 2 + 3 + 3) \cdot 10 = 100$ instances in total.

Factors	Experimental levels	
$\|\mathcal{D}\|$	$2, 3, 4, 5$	
δ	$0.5, 0.75$	for $\|\mathcal{D}\| \in \{2, 3\}$
	$0.25, 0.5, 0.75$	for $\|\mathcal{D}\| \in \{4, 5\}$

Table 2.10: Factors and levels for the experimental test design

Table 2.11 and Figure 2.8 provide κ for the different combinations of $|\mathcal{D}|$ and δ. The influence of δ on the cost improvement is with $\alpha < 0.001$ highly significant while the influence of $|\mathcal{D}|$ has only a influence at the $\alpha > 0.14$ level of significance. The interaction between δ and $|\mathcal{D}|$ cannot be considered to be significant ($\alpha > 0.45$). From the results it can be concluded that the more specialized the internal resources are (i.e. the higher the values of δ are) the smaller is the gap between central and decentral planning. This is caused by the fact that the central planning process cannot derive a big benefit from cross–utilization of departmental resources.

	$\delta = 0.25$		$\delta = 0.50$		$\delta = 0.75$		Total	
$\lvert\mathcal{D}\rvert$	μ	σ	μ	σ	μ	σ	μ	σ
2			6.97%	1.04%	3.05%	0.42%	5.01%	2.20%
3			6.30%	1.32%	3.29%	0.61%	4.79%	1.86%
4	13.91%	1.32%	7.40%	0.55%	3.38%	0.26%	8.23%	4.56%
5	13.66%	1.86%	8.19%	1.21%	3.63%	0.87%	8.49%	4.43%
Total	13.78%	1.53%	7.21%	1.21%	3.34%	0.58%	6.98%	4.00%

Table 2.11: Cost improvement κ by central planning depending on the number of departments $|\mathcal{D}|$ and their specialization δ

2.6 Summary

In this chapter the simultaneous scheduling of multiple projects and their staffing with a multi–skilled human workforce with heterogeneous and static efficiencies has been addressed. Only the start of each project is scheduled while the remainder of the project is assumed to be scheduled w.r.t. the project start already. The problem is typical for development and maintenance of IT–systems and IT–services as well as research and development projects. A mixed–integer linear program with a sharp LP–bound has been proposed. The latter enables the optimal solution of real–world problems with the standard–solver CPLEX as long as the size of the project start time windows is moderate. It has been shown how overall costs decrease with increasing start time windows of the projects, decreasing project size (keeping the total amount of work constant) and increasing number of skills per resource. For a constant size of the internal workforce an increasing workload leads to higher cost due to an increased use of external resources. Comparing the MIP–solution with the ones derived by simple heuristics currently used in an IT–service company it can be observed that costs can be decreased substantially. Moreover, if there are minimum ratios for the amount of project work which has to be done by internal resources the heuristics often fail to derive feasible solutions while the MIP generates the latter as long as they do exist. Finally, the benefit of using the MIP for central planning compared to decentral planning has been demonstrated. In the case of central planning resources can be deployed across department borders which leads to higher utilization of internal resources and thus substantial cost savings. This effect increases as more as the assignment of human resources to departments is arbitrary and does not reflect specific skills.

Up to now projects have been aggregated and scheduled as a block of work packages. Only the project start times have been subject to optimization. This limit is going to be relaxed in the following chapter. There, projects

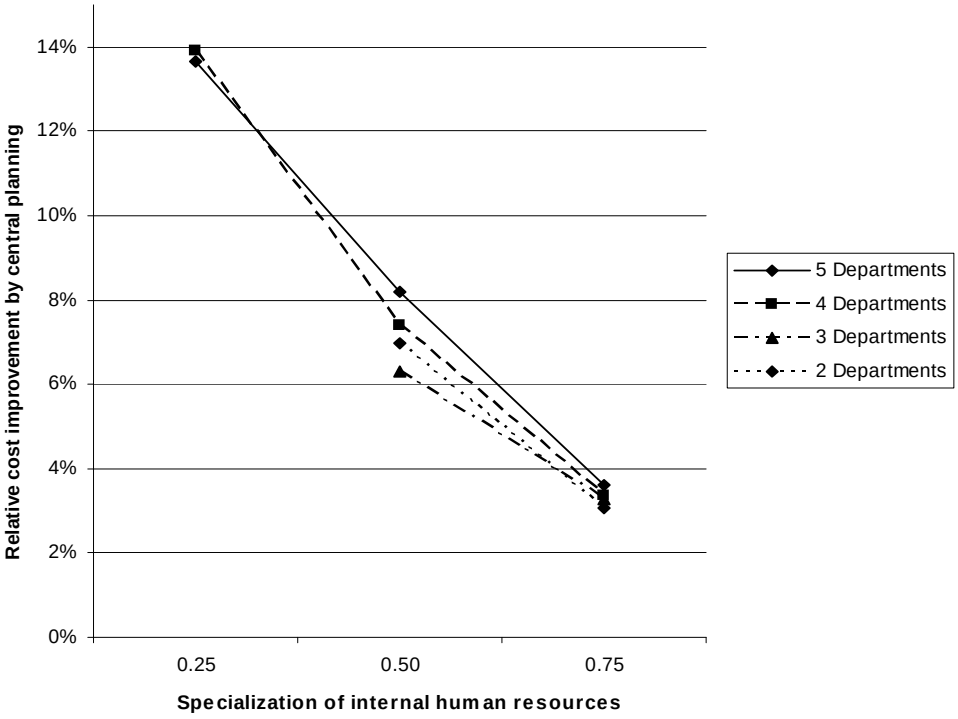


Figure 2.8: Cost improvement κ by central planning depending on the number of departments $|\mathcal{D}|$ and their specialization δ

will be modelled as a serial stream of phases or activities and the starting time of each phase can be optimized individually.

Chapter 3

Staffing and scheduling disaggregated IT–projects

3.1 Introduction

In this chapter the focus is still on the problem of simultaneous scheduling and staffing multiple projects as it has been considered in Chapter 2. However, the model of the preceding chapter is extended w.r.t. the structure of the projects (cf. Heimerl and Kolisch [47]). IT–projects usually have a serial network structure consisting of 5 to 7 activities (phases) where each activity has a duration of a few weeks. Instead of treating each project as one aggregated block of work packages as in Chapter 2, each project will be disaggregated into these activities and the starting times of the activities will be subject to optimization. The resource demand of each activity is given and consists of several work packages. Each work package defines the demand for a certain skill in each period the activity is performed. The period in which the demand actually arises depends on the schedule of each project. Often, activities of a project directly succeed each other. However, they may also be separated by breaks of limited length or overlap each other partially. Thus, activities are linked with its predecessor by minimum and maximum time–lags (cf. e.g. Neumann et al. [61]). The question is how the projects' activities are scheduled and resources are assigned to project work such that costs are minimized.

One of the results of the preceding chapter was that the aggregated problem is NP–hard. Furthermore, the computation time for obtaining optimal solutions increases rapidly with the number of projects (objects to be scheduled) and the time window size. Due to the disaggregation of projects to activities the number of objects to be scheduled increases and the period

length usually needs to be reduced, too. Hence, time window sizes of the activities of disaggregated projects are often larger. Obviously, for disaggregated projects the development of a heuristic solution method becomes inevitable.

The remainder of this chapter is organized as follows: In order to address the aforementioned problem a model for simultaneous scheduling and staffing multiple projects with serial structures is proposed in Section 3.2. The model is a generalization of the problem treated in Chapter 2. Thus, the literature reviewed in Chapter 2 is also relevant for the problem presented in this chapter. In Section 3.3 several generalized network flow representations of the staffing subproblem will be presented. Afterwards, in Section 3.4 a hybrid metaheuristic is proposed which employs these network representations to solve the optimization problem efficiently. The experimental investigation will be presented in Section 3.5. Finally, the chapter concludes with a summary in Section 3.6.

3.2 Model

3.2.1 Model description

Projects. Let $\mathcal{P}$ be a set of projects which have to be processed. Each project $p \in \mathcal{P}$ consists of activities $\mathcal{A}_p \subseteq \mathcal{A}$ with $\mathcal{A}_p \cap \mathcal{A}_{p'} = \emptyset$ for $p, p' \in \mathcal{P}, p \neq p'$ and $\bigcup_{p \in \mathcal{P}} \mathcal{A}_p = \mathcal{A}$. The projects have serial network structures, i.e. each activity $a \in \mathcal{A}_p$ of project p is assumed to have exactly one preceding activity $pred(a)$. The predecessor of the first activity of each project is a dummy activity $a_0 \notin \mathcal{A}$. The successor of activity a is denoted by $succ(a)$. Each activity a is linked with its predecessor by maximum and minimum start–to–start time–lags δ_a^{max} and δ_a^{min} ($\delta_a^{max} \geq \delta_a^{min} \geq 0$), respectively (cf. e.g. Neumann et al. [61]). Activities are assumed to be non–preemptive.

The time line is divided into periods of length one. Time period t is defined as the time span between the points in time $t-1$ and t, i.e. $[t-1, t[$ for $t = 1, \ldots, T$ where T denotes the planning horizon. W.l.o.g. for the remainder of this chapter a period length of one week is assumed. An activity always starts at the beginning of the start period and ends at the end of the finish period. These definitions were chosen in order to ease the notation of the upcoming models. In general network structures the earliest and latest start periods ES_a and LS_a of each activity a can be calculated using the Floyd–Warshall algorithm (cf. Ahuja et al. [2]). For serial project structures the calculation is far easier by initializing $ES_{a_0} = 1$ and calculating

recursively

$$ES_a = ES_{pred(a)} + \delta_a^{min} \quad (a \in \mathcal{A}) \tag{3.1}$$

as well as

$$LS_a = LS_{pred(a)} + \delta_a^{max} \quad (a \in \mathcal{A}). \tag{3.2}$$

Given the duration $d_a \geq 1$ of each activity a the earliest and latest finish periods are then $EF_a = ES_a + d_a - 1$ and $LF_a = LS_a + d_a - 1$. The planning horizon is defined by $T = \max_{a \in \mathcal{A}} LF_a$.

In total the activities require the set of skills $\mathcal{S}$. In the context of IT–projects skills are for example programming, architecture, security, or hardware. More precisely each activity a requires r_{asq} work units of skill s in executing period $q = 1, \ldots, d_a$. The amount of skill s requested in the q–th period of activity a, r_{asq}, can be seen as a work package and will henceforth be termed as such.

Resources. The definitions and notation concerning internal and external resources, skills and minimum internal work ratios are equivalent to those of Section 2.3.

Variables. The following decision variables are employed: $x^r_{atsk} \geq 0$ is the amount of work (typically measured in full time equivalents) that internal resource k performs during regular hours in period t on the work packages of activity a which require skill s. Accordingly, $x^o_{atsk} \geq 0$ is the amount of overtime work of the internal resource k. $y_{ats} \geq 0$ is the amount of work of activity a requiring skill s performed by external resources in period t. Both decision variables are continuous. Finally, the binary decision variables $z_{at} \in \{0, 1\}$ are required for setting activity start times. z_{at} equals 1 if activity a is started at the beginning of period t and 0 otherwise. Table A.1 in the Appendix provides a summary of the notation.

The question is how the projects' activities should be scheduled and which human resources should be assigned to each work package such that the different requirements are met and costs are minimized.

3.2.2 MIP Formulation

The following mixed–binary linear program MIP3 represents the problem. Its LP–relaxation is denoted with LP3.

$$\text{Min} \quad \sum_{a \in \mathcal{A}} \sum_{t=ES_a}^{LF_a} \sum_{s \in \mathcal{S}} \left(c^e_s y_{ats} + \sum_{k \in \mathcal{R}^i_s} \frac{1}{\eta_{sk}} \left(c^r_k x^r_{atsk} + c^o_k x^o_{atsk} \right) \right) \tag{3.3}$$

subject to

$$\sum_{t=ES_a}^{LS_a} z_{at} = 1 \qquad a \in \mathcal{A} \tag{3.4}$$

$$\delta_a^{min} \leq \sum_{t=ES_a}^{LS_a} t \cdot z_{at} - \sum_{t=ES_{pred(a)}}^{LS_{pred(a)}} t \cdot z_{pred(a)t} \leq \delta_a^{max} \qquad a \in \mathcal{A} \tag{3.5}$$

$$\sum_{(\tau,q)\in\mathcal{T}_{at}} r_{asq} \cdot z_{a\tau} \leq y_{ats} + \sum_{k\in\mathcal{R}_s^i} x_{atsk}^r + x_{atsk}^o \qquad \begin{array}{l} a \in \mathcal{A} \\ t = ES_a, \ldots, LF_a \\ s \in \mathcal{S} \end{array} \tag{3.6}$$

$$\sum_{a\in\mathcal{A}} \sum_{s\in\mathcal{S}} \frac{1}{\eta_{sk}} x_{atsk}^r \leq R_{kt}^r \qquad \begin{array}{l} t = 1, \ldots, T \\ k \in \mathcal{R}^i \end{array} \tag{3.7}$$

$$\sum_{a\in\mathcal{A}} \sum_{s\in\mathcal{S}} \frac{1}{\eta_{sk}} x_{atsk}^o \leq R_{kt}^o \qquad \begin{array}{l} t = 1, \ldots, T \\ k \in \mathcal{R}^i \end{array} \tag{3.8}$$

$$\sum_{a\in\mathcal{A}_p} \sum_{t=ES_a}^{LF_a} \sum_{s\in\mathcal{S}} \sum_{k\in\mathcal{R}_s^i} x_{atsk}^r + x_{atsk}^o \geq e_p \sum_{a\in\mathcal{A}_p} \sum_{t=ES_a}^{LF_a} \sum_{s\in\mathcal{S}} y_{ats} \qquad p \in \mathcal{P} \tag{3.9}$$

$$x_{atsk}^r, x_{atsk}^o \geq 0 \qquad \begin{array}{l} a \in \mathcal{A} \\ t = ES_a, \ldots, LF_a \\ s \in \mathcal{S} \\ k \in \mathcal{R}_s^i \end{array} \tag{3.10}$$

$$y_{ats} \geq 0 \qquad \begin{array}{l} a \in \mathcal{A} \\ t = ES_a, \ldots, LF_a \\ s \in \mathcal{S} \end{array} \tag{3.11}$$

$$z_{at} \in \{0, 1\} \qquad \begin{array}{l} a \in \mathcal{A} \\ t = ES_a, \ldots, LS_a \end{array} \tag{3.12}$$

The objective function (3.3) minimizes the labor costs of internal and external resources which accrue by processing the work packages of the activities. Note that the time required by the internal resource k to perform x_{atsk} work units depends on the efficiency η_{sk}. I.e. in case of a high efficiency ($\eta_{sk} > 1$) less time is needed while in case of a low efficiency ($\eta_{sk} < 1$) more

time is needed. For all external resources an efficiency of $\eta = 1$ is assumed. Due to constraints (3.4) each activity a is forced to start exactly once within its absolute start time window $[ES_a, LS_a]$. Constraints (3.5) ensure that each activity a is started within its start time window $[\delta_a^{min}, \delta_a^{max}]$ relative to the start time of its predecessor $pred(a)$. Constraints (3.6) ensure that the work packages of activity a are performed by internal and external resources. More precisely, for each period $t = ES_a, \ldots, LF_a$ the required work of skill s has to be covered by internal or external resources. The work requiring skill s which has to be performed in period t depends on the start time variable z_{at} and the demand profile r_{asq} of the activities. The set

$$\mathcal{T}_{at} = \{(\tau, q) \in \{ES_a, \ldots, LS_a\} \times \{1, \ldots, d_a\} \,|\, \tau + q - 1 = t\} \tag{3.13}$$

consists of all combinations (τ, q) of activity start periods τ and executing periods q of activity a that lead to a resource demand in calendar period t (cf. Heimerl and Kolisch [45]). To illustrate this with an example let $ES_a = 1$, $LS_a = 2$, and $d_a = 2$. Then $\mathcal{T}_{a1} = \{(1,1)\}$, $\mathcal{T}_{a2} = \{(1,2),(2,1)\}$, and $\mathcal{T}_{a3} = \{(2,2)\}$.

The available capacities of the internal resources for regular time and overtime are considered by constraints (3.7) and (3.8), respectively. For each project p constraints (3.9) ensure a minimum ratio $e_p \geq 0$ of the work performed by internal resources to the work performed by external resources. Finally, the constraints (3.10), (3.11) and (3.12) define the decision variables.

Note that constraints (3.6) are formulated as inequalities. This allows oversupply of work packages by internal resources in order to obtain feasible solutions w.r.t. constraints (3.9). Oversupply can be used to improve the quality of deliverables by e.g. additional code reviews in IT–projects or to further qualify resources for existing or additional skills (cf. Heimerl and Kolisch [46]).

Proposition 4. *The problem of simultaneous scheduling and staffing serial projects is NP–hard.*

Proof. MIP3 is a generalization of the NP–hard problem depicted in Chapter 2 as it can be restricted to that case by allowing only instances with $|\mathcal{A}_p| = 1$ for all $p \in \mathcal{P}$. Hence, the considered problem must also be NP–hard. □

3.2.3 Additional cuts

Additional constraints proposed by Zhu et al. [80] for the multi–mode resource–constrained project scheduling problem are employed in order to tight-

en the formulation MIP3. The binary decision variables z_{at} for the determination of activity start periods need to be defined for all periods $t = ES_a, \ldots, LS_a$. It is easy to see that $LS_a - ES_a \geq LS_{pred(a)} - ES_{pred(a)}$ holds, i.e. the time window size of an activity increases (not strictly) with the activity's rank (i.e. the number of predecessor activities, cf. Davis [29] and Kløvstad [53]). Thus, for activities with high rank the number of binary decision variable can be large. However, some combinations of the starting times of an activity and its predecessor are invalid which can be exploited by the following cuts. After having derived start time windows $[ES_a, LS_a]$ for each activity a according to (3.1) and (3.2) and setting the start period of its predecessor $pred(a)$ to some $\tau \in [ES_{pred(a)}, LS_{pred(a)}]$ then it follows that activity a cannot start after period $\tau + \delta_a^{max}$. Analogously, activity a cannot start before period $\tau + \delta_a^{min}$. In general the two cuts can be formulated as follows.

$$\sum_{\tau=ES_{pred(a)}}^{t} z_{pred(a)\tau} + \sum_{\tau=t+\delta_a^{max}+1}^{LS_a} z_{a\tau} \leq 1 \quad \begin{array}{l} a \in \mathcal{A} \\ t = ES_{pred(a)}, \ldots, LS_{pred(a)} - 1 \end{array} \tag{3.14}$$

$$\sum_{\tau=t}^{LS_{pred(a)}} z_{pred(a)\tau} + \sum_{\tau=ES_a}^{t+\delta_a^{min}-1} z_{a\tau} \leq 1 \quad \begin{array}{l} a \in \mathcal{A} \\ t = ES_{pred(a)} + 1, \ldots, LS_{pred(a)} \end{array} \tag{3.15}$$

To give a short example for cut (3.14) consider the case that $\delta_a^{max} = 2$ for activity a. Thus, if $pred(a)$ starts in period $t = 5$ or earlier, activity a may not start in period 8 or later, i.e.

$$\sum_{\tau=ES_{pred(a)}}^{5} z_{pred(a)\tau} + \sum_{\tau=8}^{LS_a} z_{a\tau} \leq 1.$$

As proved in Zhu et al. [80] only the direct predecessor of an activity needs to be considered for each cut. Additional cuts for all indirect predecessors will not further tighten MIP3. In preliminary tests the effectiveness of the cuts (3.14) and (3.15) was verified and hence they were used in the experimental investigation. The tightened formulation (3.3)—(3.12) and (3.14)—(3.15) is denoted MIP4 and its LP–relaxation is denoted LP4.

3.3 Staffing subproblem

The hybrid metaheuristic that will be presented in Section 3.4 depends on an efficient evaluation of intermediate solutions. This requires to solve the

staffing subproblem which arises in MIP3 if the activities' starting times are fixed. In this section a generalized minimum cost flow formulation of the staffing subproblem is proposed. A linear program formulation of the staffing subproblem is given in Section 3.3.1. Corresponding network flow formulations are presented in Section 3.3.2. Finally, the generalized network simplex algorithm that can solve generalized network flow problems faster than standard solvers is described in Section 3.3.3.

3.3.1 LP Formulation

Let the values of the variables z_{at} be binary and valid (i.e. obeying constraints (3.4) and (3.5)), then the starting period of activity a can be calculated by $S_a = \sum_{t=ES_a}^{LS_a} t z_a$ (cf. Pritsker et al. [63]). The vectors of starting and finishing periods are denoted by $\mathbf{S} = (S_a)$ and $\mathbf{F} = (F_a) = (S_a + d_a - 1)$, respectively. With

$$r'_{ast} = \sum_{(\tau,q)\in\mathcal{T}_{at}} r_{asq} z_{a\tau} = r_{as(q+S_a-1)} \tag{3.16}$$

$$T' = \min_{a\in\mathcal{A}} S_a \tag{3.17}$$

and

$$T'' = \max_{a\in\mathcal{A}} F_a \tag{3.18}$$

MIP3 then reduces to the following linear program that minimizes costs while assigning resources to work packages.

$$\text{Min} \quad \sum_{a\in\mathcal{A}} \sum_{t=S_a}^{F_a} \sum_{s\in\mathcal{S}} \left(c^e_s y_{ats} + \sum_{k\in\mathcal{R}^i_s} \frac{1}{\eta_{sk}} \left(c^r_k x^r_{atsk} + c^o_k x^o_{atsk} \right) \right) \tag{3.19}$$

subject to

$$r'_{ast} \le y_{ats} + \sum_{k\in\mathcal{R}^i_s} x^r_{atsk} + x^o_{atsk} \qquad \begin{array}{l} a\in\mathcal{A} \\ t = S_a,\ldots,F_a \\ s\in\mathcal{S} \end{array} \tag{3.20}$$

$$\sum_{a\in\mathcal{A}} \sum_{s\in\mathcal{S}} \frac{1}{\eta_{sk}} x^r_{atsk} \le R^r_{kt} \qquad \begin{array}{l} t = T',\ldots,T'' \\ k\in\mathcal{R}^i \end{array} \tag{3.21}$$

$$\sum_{a\in\mathcal{A}} \sum_{s\in\mathcal{S}} \frac{1}{\eta_{sk}} x^o_{atsk} \le R^o_{kt} \qquad \begin{array}{l} t = T',\ldots,T'' \\ k\in\mathcal{R}^i \end{array} \tag{3.22}$$

$$\sum_{a\in\mathcal{A}_p}\sum_{t=S_a}^{F_a}\sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{R}_s^i} x^r_{atsk} + x^o_{atsk} \geq e_p \sum_{a\in\mathcal{A}_p}\sum_{t=S_a}^{F_a}\sum_{s\in\mathcal{S}} y_{ats} \qquad p\in\mathcal{P} \tag{3.23}$$

$$x^r_{atsk}, x^o_{atsk} \geq 0 \qquad \begin{array}{l} a\in\mathcal{A} \\ t=S_a,\ldots,F_a \\ s\in\mathcal{S} \\ k\in\mathcal{R}_s^i \end{array} \tag{3.24}$$

$$y_{ats} \geq 0 \qquad \begin{array}{l} a\in\mathcal{A} \\ t=S_a,\ldots,F_a \\ s\in\mathcal{S} \end{array} \tag{3.25}$$

The linear program (3.19)—(3.22), (3.24)—(3.25) eliminating constraints (3.23) is denoted LP5. Note that LP5 can be decomposed into T independent linear programs for each period t. The linear program (3.19)—(3.25) is denoted LP6.

3.3.2 Generalized minimum cost flow formulations

Both LP5 and LP6 can be modelled as generalized minimum cost flow problems. First a generalized minimum cost flow formulation for LP5 will be presented. Then this network will be expanded in order to obtain a generalized minimum cost flow formulation for LP6. Finally, a modified network formulation for LP6 will be presented which is advantageous especially when repeatedly evaluating similar solutions in the course of a metaheuristic.

A generalized network $G(N, A)$ consists of nodes $i \in N$ with supply $b_i \geq 0$ or demand $b_i < 0$ and directed arcs $(i, j) \in A$ with upper bounds u_{ij}, gains $\mu_{ij} > 0$, and unit costs c_{ij} (cf. Ahuja et al. [2]). The networks presented in this chapter will only contain unbounded arcs, i.e. $u_{ij} = \infty$. Figure 3.1 provides the notation where entries for the following default values will be omitted: $b_i = 0$, $c_{ij} = 0$, and $\mu_{ij} = 1$.

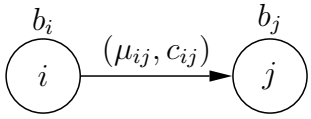


Figure 3.1: Notation for graphs

Decomposed single–period network representation of LP5. Graph G_1 represents the decomposed problem LP5 for period t (cf. Figure 3.2). On the internal supply side the graph consists of overtime and regular working

time nodes k^o and k^r for each internal resource $k \in \mathcal{R}^i$, respectively. The supply of these nodes is $b_{k^o} = R^o_{kt}$ and $b_{k^r} = R^r_{kt}$, respectively. Arcs (k^o, k^r) connect overtime and regular work time with unit costs $c_{k^o k^r} = c^o_k - c^r_k$. Since constraints (3.21) and (3.22) are inequalities, excess supply does not need to be used and, thus, can be consumed by zero–cost loops (k^o, k^o) and (k^r, k^r) with gains $\mu_{k^o k^o} < 1$ and $\mu_{k^r k^r} < 1$, respectively. An arc with $\mu_{ij} < 1$ is called lossy since the flow leaving the arc is smaller than the flow entering the arc. The demand of activities requiring skill s in period t can be aggregated and depicted by node s such that $b_s = -\sum_{a \in \mathcal{A}} r'_{ast}$. An arc (k^r, s) with gain $\mu_{ks} = \eta_{sk}$ and unit costs $c_{k^r s} = c^r_k$ is added to the graph, if $k \in \mathcal{R}^i_s$. Finally, external supply can be modelled by loops (s, s) with gain $\mu_{ss} = 2$ and unit costs $c_{ss} = c^e_s$. An arc with $\mu_{ij} > 1$ is called gainy since the flow leaving the arc is greater than the flow entering the arc. The supply created by a loop with $\mu_{ij} = 2$ equals the flow x_{ij} over the loop (cf. Ahuja et al. [2]): One unit of flow over the loop creates two units of supply at the node. One of these two units is required to flow over the loop in order to hold the mass–balance constraints.

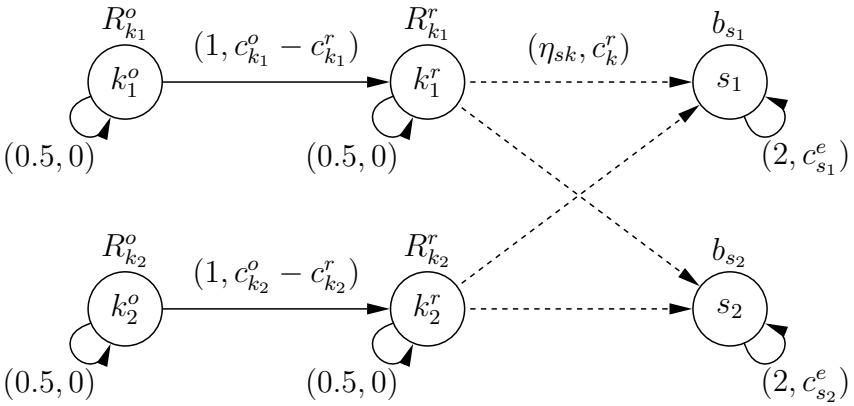


Figure 3.2: Graph G_1 for relaxed single–period problem LP5 with $\mathcal{R}^i = \{k_1, k_2\}$ and $\mathcal{S} = \{s_1, s_2\}$. Indices for period t have been omitted.

For each period $t = T', \ldots, T''$ the network contains

$$2 \cdot |\mathcal{R}^i| + |\mathcal{S}| \tag{3.26}$$

nodes and

$$3 \cdot |\mathcal{R}^i| + |\mathcal{S}| + \sum_{s \in S} |\mathcal{R}^i_s| \tag{3.27}$$

arcs. Of course, the network can be reduced if e.g. overtime capacity $R^o_{kt} = 0$. LP5 can be used if $e_p = 0 \; \forall \, p \in \mathcal{P}$ or to obtain a lower bound on LP6. This lower bound can even be tightened by incorporating constraints (3.23) into Graph G_1 using Lagrangian relaxation.

Network representation of LP6. For LP6 the corresponding Graph G_2 is given in Figure 3.3. The structure of the network for the internal supply side is the same as in G_1 except for the fact that it is repeated for each period t. Thus, the indices of t have to be taken into account and the nodes are denoted $k^o t$, $k^r t$, and st, respectively. Since constraints (3.23) are project–specific, the demand cannot be aggregated over all projects such that the demand $b_{pst} = -\sum_{a \in \mathcal{A}_p} r'_{ast}$ arises in node pst. Nodes st only act as transshipment nodes reducing the number of arcs in the network. Constraints (3.23) limit the ratio of external work within a project. Thus, only a limited capacity $b_p = -\frac{1}{1+e_p} \sum_{s \in \mathcal{S}} \sum_{t=T'}^{T''} b_{pst}$ is available as external supply in each node p. If constraints (3.20) were equalities the graph described up to this point would already represent the linear program. However, since the constraints (3.20) are inequalities it is valid to oversupply a work package with internal resources in order to use more external resources for other work packages of the same project and concurrently hold constraints (3.23). Hence, oversupply by internal resources allows further external resources such that additional arcs (pst, p) with gains $\mu_{pst,p} = \frac{1}{1+e_p}$ need to be added. These arcs depict that one unit of oversupply by internal resources in a work package of project p allows the use of $\frac{1}{1+e_p}$ additional units of external resources for this project. Note that the cycle P along the nodes p, pst, and p is not gainy since $\mu_{pst,p} \leq 1$, $\mu_{p,pst} = 1$ and the gain μ_P along a path P is defined as $\mu_P = \prod_{(i,j) \in P} \mu_{ij}$ (cf. Ahuja et al. [2]). Hence, external supply cannot create additional external supply. In order to keep the figure simple only two out of eight arcs (pst, p) are depicted in Figure 3.3.

Let

$$\hat{T} = T'' - T' + 1 \tag{3.28}$$

denote the number of periods. Then the network contains

$$|\mathcal{P}| + \hat{T} \cdot (2|\mathcal{R}^i| + |\mathcal{S}| \cdot (1 + |\mathcal{P}|)) \tag{3.29}$$

nodes and

$$|\mathcal{P}| + \hat{T} \cdot \left(3 \cdot |\mathcal{R}^i| + 3 \cdot |\mathcal{S}| \cdot |\mathcal{P}| + \sum_{s \in S} |\mathcal{R}^i_s| \right) \tag{3.30}$$

arcs. Again, the network can be reduced if e.g. overtime capacity $R^o_{kt} = 0$ or demand $b_{pst} = 0$ holds.

Modified network representation of LP6. The hybrid metaheuristic described in Section 3.4 has to solve the minimum cost flow problem for many different vectors of starting periods $\mathbf{S}$. The values of b_{pst} for the Graph G_2 have to be changed for each evaluated vector of starting periods. As a

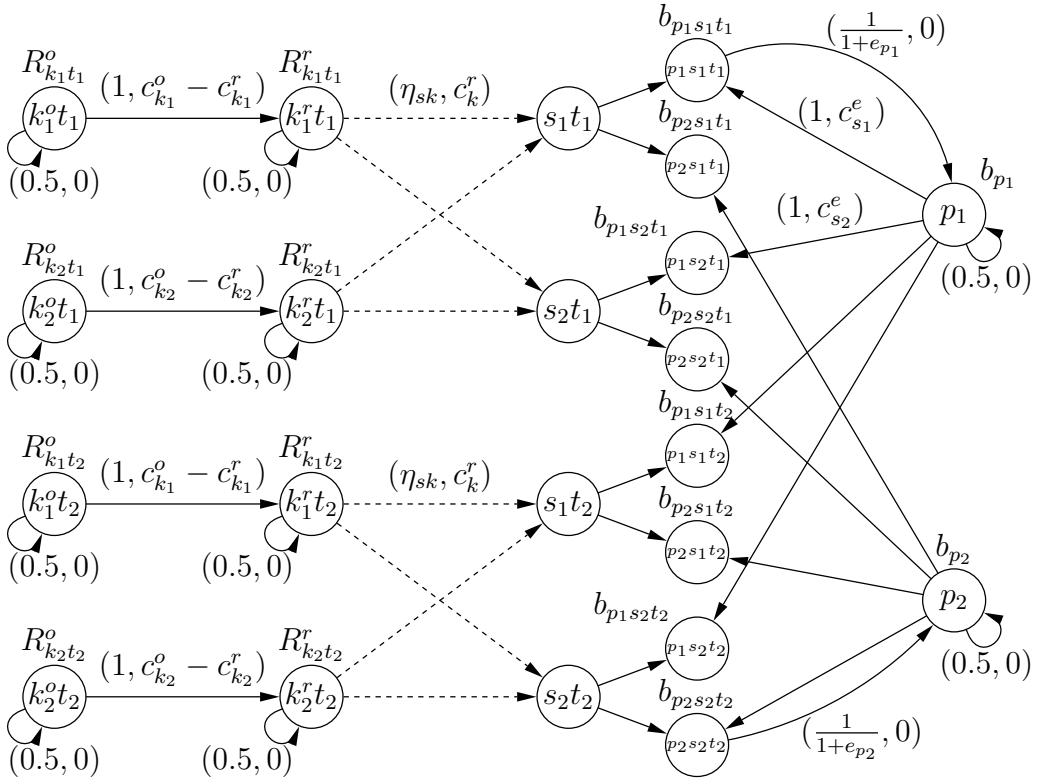


Figure 3.3: Graph G_2 for LP6 with $\mathcal{R}^i = \{k_1, k_2\}$, $\mathcal{S} = \{s_1, s_2\}$, $\mathcal{P} = \{p_1, p_2\}$, and $t = \{t_1, t_2\}$

consequence the (primal) generalized network simplex algorithm described in Section 3.3.3 always needs to be started from scratch. However, the vectors of starting periods $\mathbf{S}$ are often similar. E.g. when the neighbourhood defined in the tabu search procedure in Section 3.4.2 is evaluated, starting periods of two arbitrary solutions of the neighbourhood only differ for a few activities. Hence, the evaluation procedure can be expedited by constructing and solving Graph G_3 (cf. Figure 3.4). Every vector of starting periods $\mathbf{S}$ can be represented in G_3 by modifying only the cost parameters of at most $2 \cdot |\mathcal{S}| \cdot \sum_{a \in \mathcal{A}} d_a$ arcs. Modifying the cost parameters keeps the solution primal feasible. The (feasible) solution for an evaluated vector of starting periods $\mathbf{S}$ can act as an initial (feasible) basis for the next vector of starting periods $\mathbf{S}'$ to be evaluated.

In contrast to Graph G_2 the demand nodes asq in Graph G_3 need to be defined for each activity a, skill s, and executing period q. The demand of node asq is $b_{asq} = -r_{asq}$. For each $t = ES_a + q - 1, \ldots, LS_a + q - 1$ an arc (st, asq) connects the node with the internal supply. The unit costs $c_{st,asq}$ will be zero if $S_a = t - q + 1$ and sufficiently large otherwise.

To illustrate the network structure consider the example depicted in Figure 3.4 where a single activity a_1 with $d_{a_1} = 2$, $ES_{a_1} = 1$, $LS_{a_1} = 2$, and requiring two skills needs to be staffed. If a solution with $S_{a_1} = 1$ needs to be evaluated then the costs of the arcs with $t = q$, i.e. $(s_1\mathbf{t_1}, a_1s_1\mathbf{q_1})$, $(s_2\mathbf{t_1}, a_1s_2\mathbf{q_1})$, $(s_1\mathbf{t_2}, a_1s_1\mathbf{q_2})$, $(s_1\mathbf{t_2}, a_1s_1\mathbf{q_2})$, will be set to zero and the costs of the arcs with $t \neq q$, i.e. $(s_1\mathbf{t_2}, a_1s_1\mathbf{q_1})$, $(s_2\mathbf{t_2}, a_1s_2\mathbf{q_1})$, $(s_1\mathbf{t_3}, a_1s_1\mathbf{q_2})$, $(s_1\mathbf{t_3}, a_1s_1\mathbf{q_2})$, are set to a sufficiently high value. If $S_{a_1} = 2$ then the cost parameters are set vice versa.

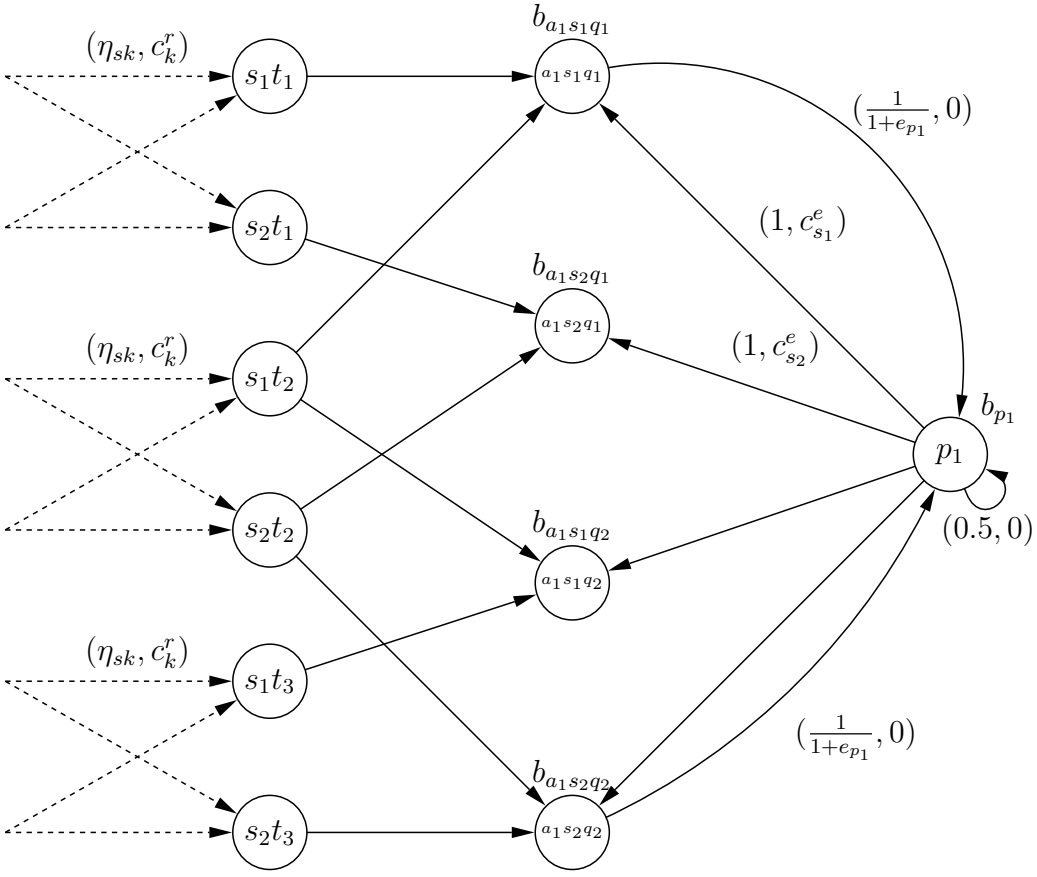


Figure 3.4: Relevant part of Graph G_3 for LP6 with $\mathcal{S} = \{s_1, s_2\}$, $\mathcal{P} = \{p_1\}$, $\mathcal{A}_{p_1} = \{a_1\}$, $d_a = 2$, $ES_{a_1} = t_1$, $LS_{a_1} = t_2$, and $t = \{t_1, t_2, t_3\}$

The Graph G_3 contains

$$|\mathcal{P}| + T \cdot (2|\mathcal{R}^i| + |\mathcal{S}|) + |\mathcal{S}| \cdot \sum_{a \in \mathcal{A}} d_a \tag{3.31}$$

nodes and

$$|\mathcal{P}| + T \cdot \left(3|\mathcal{R}^i| + \sum_{s \in S} |\mathcal{R}^i_s|\right) + |\mathcal{S}| \cdot \sum_{a \in \mathcal{A}} (d_a \cdot (3 + LS_a - ES_a)) \tag{3.32}$$

arcs. Again, the size of the network can be reduced for certain parameter constellations. Despite the fact that Graph G_3 has an increased number of nodes and arcs compared to Graph G_2, the generalized network simplex algorithm can find the optimal solution faster when repeatedly evaluating similar solutions. Results regarding this point will be reported in Section 3.5.

3.3.3 Generalized network simplex algorithm

To solve LP6 the (primal) generalized network simplex algorithm (cf. e.g. Ahuja et al. [2], Bazaraa et al. [11], Kennington and Helgason [52]) is applied on the networks G_2 and G_3. The generalized network simplex algorithm is closely related to the network simplex algorithm. The network simplex algorithm assumes gains $\mu_{ij} = 1$ and generates integer solutions (flows) if all input data is integer. The primal version maintains feasible solutions in a spanning tree structure and improves the solutions by moving from one feasible spanning tree structure to the next.

The (primal) generalized network simplex algorithm maintains feasible solutions in an augmented forest structure which is a set of augmented trees. An augmented tree (or 1–tree) is a connected subgraph of a network containing only a subset of nodes and arcs. An augmented tree is basically a tree but contains one additional arc forming a cycle. Hence, each augmented tree contains as many nodes as arcs. The (primal) generalized network simplex algorithm improves the solutions by moving from one feasible augmented forest structure to the next.

The initial feasible augmented forest structure can be obtained by adding artificial loops (i, i) (if not already existing) with high unit costs c_{ii} for each node i with $b_i \neq 0$. The artificial loop is defined to be gainy (lossy), i.e. $\mu_{ii} = 2$ ($\mu_{ii} = 0.5$), if the node has non–zero demand (supply), i.e. $b_i < 0$ ($b_i > 0$). The initial augmented forest then consists of augmented trees each with one node and its artificial loop. The initial flow x_{ii} on the artificial loop is $x_{ii} = \frac{b_i}{1-\mu_{ii}}$. When an artificial loop is removed from the augmented forest structure during the algorithm it is no longer needed and permanently removed from the network.

The network simplex algorithm as well as the generalized network simplex algorithm both rely on efficient representations and updates of the spanning tree and augmented forest, respectively, and on an efficient pivoting strategy. For the augmented forest the representation of Bazaraa et al. [11] and Kennington and Helgason [52] with five indices per node (predecessor, thread, final node, number of subnodes, and arc orientation) was chosen. Furthermore, the efficient candidate list pivot rule described by Ahuja et al. [2] is used. Preliminary tests for the given graphs revealed that the algorithm

performs best when the maximum length of the candidate list is 6% of the number of eligible arcs and that the candidate list should be recreated after at most 10% of the length of the candidate list.

3.4 Hybrid metaheuristic

Since the proposed mixed–binary linear program MIP3 is NP–hard (cf. Section 3.2) an exact solution of the problem might no longer be possible even for small problem instances due to the vast amount of binary variables. Therefore, a hybrid metaheuristic is proposed in order to solve the problem in reasonable time. The hybrid metaheuristic will first try to find the best solution using a genetic algorithm (cf. Section 3.4.1). Afterwards a tabu search based post–optimization will be applied on the best solution found by the genetic algorithm (cf. Section 3.4.2).

3.4.1 Genetic algorithm

The proposed hybrid metaheuristic will first try to find a good solution using a genetic algorithm (GA). The GA is a well–known metaheuristic based on concepts of evolution theory. The GA maintains a population *Pop* consisting of individuals (or chromosomes) $i \in Pop$. Each individual is coded by its genotype and represents a solution. A new generation is created by reproduction (also known as crossover), mutation, and selection. Crossover is the primary operator that creates children based on the parents. Mutation randomly changes individuals in order to obtain diversity. Selection ensures that fitter (i.e. better) solutions survive with higher probability. The standard genetic algorithm scheme presented in Glover and Kochenberger [36] is adapted (cf. Algorithm 3).

Algorithm 3 Genetic algorithm scheme

```
1: Choose and evaluate initial population Pop
2: while termination condition not satisfied do
3:     Do crossover on the population to obtain children
4:     Mutate children
5:     Evaluate children
6:     Select new population out of parents and children
7: end while
```

As solution representation integer vectors $\boldsymbol{\delta} = (\delta_a)$ are used. Each integer value $\delta_a = S_a - S_{pred(a)}$ of the vector depicts the realized start–to–start time–lag between activity a and its immediate predecessor $pred(a)$. The solutions

are always precedence feasible as long as $\delta_a^{min} \leq \delta_a \leq \delta_a^{max}$. Given $\boldsymbol{\delta}$ the starting periods S_a of all activities $a \in \mathcal{A}$ can be calculated recursively by setting $S_{a_0} = 1$ and employing

$$S_a = S_{pred(a)} + \delta_a \tag{3.33}$$

Note that the change of δ_a affects the starting period of activity a and all of its direct and indirect successors.

Given $\boldsymbol{\delta}$ the objective function value of a solution can be evaluated by applying the generalized network simplex algorithm on LP2. Of course, lower objective function values must correspond to higher fitness values. Furthermore, objective function values might include inevitable costs, since e.g. the cumulative demand exceeds the capacities of internal resources. To handle these two issues linear ranking of the solutions $i \in Pop$ is applied according to their objective function value in order to obtain fitness values (cf. Glover and Kochenberger [36]). If a solution is infeasible due to constraints (3.9) the solution of the network flow problem will have positive flows on artificial loops with high unit costs. Thus, the level of infeasibility is implicitly included in the objective function value, such that infeasible solutions will automatically be ranked lower than feasible solutions and amongst infeasible solutions the ones with a greater extend of infeasibility will be ranked lower.

For selection the standard roulette–wheel selection procedure is applied based on the rank of a solution. To keep promising solutions within the population (elitism), a new population is not only selected from the children but also from the parent generation. A single–point crossover operator is applied. Mutation of a gene occurs with probability p and sets δ_a to a random integer value within the interval $[\delta_a^{min}, \delta_a^{max}]$.

The initial population is built with two types of individuals where equal individuals are prohibited. The time–lags δ_a of individuals of type 1 are completely random integer values within $[\delta_a^{min}, \delta_a^{max}]$. The time–lags δ_a of individuals of type 2 have random values δ_a for the first activity of each project (i.e. activities with $pred(a) = a_0$) while the time–lags of all other (succeeding) activities are set to $\delta_a = \delta_a^{max}$. Individuals of type 2 were included due to the fact that MIP3 basically resembles a resource–levelling problem (cf. e.g. Neumann and Zimmermann [60]) and that spreading activities as much as possible is an intuitive way of obtaining good solutions.

3.4.2 Tabu search

The proposed hybrid metaheuristic will apply tabu search (TS) as a post–processing optimization on the best solution found by the GA. TS is an

expansion of local search in order to find global optima. A steepest descent TS algorithm on a well–defined neighbourhood was chosen and the standard tabu search scheme presented in Glover and Kochenberger [36] was adapted (cf. Algorithm 4). A neighbourhood is a set of adjacent solutions reachable with simple moves (changes) from the current solution. The neighbourhood excludes solutions maintained in a tabu list of recently visited solutions or performed moves. These solutions may not be revisited for some iterations in order to avoid cycling.

A solution representation differing from the one employed by the GA was chosen for the TS. For the TS the integer vectors $\mathbf{S} = (S_a)$ of starting periods are used. Each integer value of the vector depicts the realized starting period of activity a. This solution representation was chosen for the TS because two arbitrary solutions within the subsequently described neighbourhood are very similar and hence can be evaluated very efficiently using Graph G_3 (cf. Section 3.3.2).

The construction of the neighbourhood $N(\mathbf{S})$ works according to Algorithm 5: For each neighbourhood solution $\mathbf{S}' \in N(\mathbf{S})$ the starting period S'_a of exactly one activity a is increased or decreased by one period relative to the current solution S_a, i.e. for one activity either $S'_a = S_a - 1$ (lines 2–11) or $S'_a = S_a + 1$ (lines 13–22) holds. However, each neighbourhood solution $\mathbf{S}' \in N(\mathbf{S})$ is required to be precedence–feasible, i.e. $\delta_a^{min} \leq S'_a - S'_{pred(a)} \leq \delta_a^{max}\ \forall\, a \in \mathcal{A}$. Hence, after modifying the starting period of an activity the starting period of the successor is checked and repaired if necessary. As long as a repair is necessary for activity a also the successor $succ(a)$ of activity a is checked and repaired if necessary (lines 5–9 and 16–20, respectively). The vector $\mathbf{S}$ and an arbitrary solution $\mathbf{S}' \in N(\mathbf{S})$ of the neighbourhood differ in at most $\max_{p\in\mathcal{P}} |\mathcal{A}_p|$ elements since the starting period of the modified activity and (in the worst case) of all succeeding activities might have changed. Two arbitrary vectors $\mathbf{S}', \mathbf{S}'' \in N(\mathbf{S})$ of a neighbourhood differ in at most $2 \max_{p\in\mathcal{P}} |\mathcal{A}_p|$ elements. Hence, two neighbourhood solutions can be efficiently evaluated using Graph G_3 which is suited for repeatedly evaluating similar solutions (cf. Section 3.3.2). The maximum size of the neighbourhood is $2|\mathcal{A}|$ since for each activity at most two modifications are obtained. Each solution of the neighbourhood is evaluated and the best one is selected as next current solution.

Algorithm 4 Tabu search scheme

Require: Initial solution $\mathbf{S}$

Create empty tabu list $TL \leftarrow \emptyset$
while termination condition is not satisfied **do**
    Construct and evaluate neighbourhood $N(\mathbf{S})$
    Ignore solutions in tabu list: $N(\mathbf{S}) \leftarrow N(\mathbf{S}) \backslash TL$
    Choose best neighbour $\mathbf{S}' \in N(\mathbf{S})$ to be the new current solution: $\mathbf{S} \leftarrow \mathbf{S}'$
    if $|TL| > maxsize$ **then**
        Remove oldest entry from the tabu list
    end if
    $TL \leftarrow TL \cup \mathbf{S}$
end while

Algorithm 5 Construction of the neighbourhood $N(\mathbf{S})$

for $a \in \mathcal{A}$ **do**
    if $S_a - S_{pred(a)} > \delta_a^{min}$ **then**
        $\mathbf{S}' \leftarrow \mathbf{S}$
        $S'_a \leftarrow S'_a - 1$
        $\hat{a} \leftarrow succ(a)$
        while $\hat{a} \neq \emptyset$ and $S_{\hat{a}} - S_{pred(\hat{a})} > \delta_{\hat{a}}^{max}$ **do**
            $S'_{\hat{a}} \leftarrow S'_{\hat{a}} - 1$
            $\hat{a} \leftarrow succ(a)$
        end while
        $N(\mathbf{S}) \leftarrow N(\mathbf{S}) \cup \mathbf{S}'$
    end if

    if $S_a - S_{pred(a)} < \delta_a^{max}$ **then**
        $\mathbf{S}' \leftarrow \mathbf{S}$
        $S'_a \leftarrow S'_a + 1$
        $\hat{a} \leftarrow succ(a)$
        while $\hat{a} \neq \emptyset$ and $S_{\hat{a}} - S_{pred(\hat{a})} < \delta_{\hat{a}}^{min}$ **do**
            $S'_{\hat{a}} \leftarrow S'_{\hat{a}} + 1$
            $\hat{a} \leftarrow succ(a)$
        end while
        $N(\mathbf{S}) \leftarrow N(\mathbf{S}) \cup \mathbf{S}'$
    end if
end for

3.5 Experimental investigation

3.5.1 Experimental setup

Test instances. Test instances were created according to Table 3.1. One set of instances is generated with 5 projects and 5 resources, and another set with 10 projects and 10 resources, respectively. The demands r_{asq} are normally distributed with a coefficient of variation $COV = 0.1$ such that the expected average workload

$$\rho = \frac{\sum_{a\in\mathcal{A}}\sum_{s\in\mathcal{S}}\sum_{q=1}^{d_a}\mathbb{E}(r^{asq})}{\sum_{t=\min_{a\in\mathcal{A}}ES_a}^{T}\sum_{r\in\mathcal{R}^i}R^r_{kt}} \tag{3.34}$$

is equal to 1.5. The expected average workload ρ is the ratio of the total expected demand of all work packages and the total availability of all internal resources over the planning horizon. The number of activities per project $|\mathcal{A}_p| = 6$ stems from the waterfall model (cf. Royce [64]) which is a very common software development process. The cost rates for external resources are drawn from a truncated normal distribution $TN_{a,b}(\mu,\sigma)$ with the expected value $\mu = 800$, a standard deviation of $\sigma = 100$, a minimum value of $a = 600$, and a maximum value of $b = 1{,}000$.

$(\lvert\mathcal{P}\rvert, \lvert\mathcal{R}^i\rvert)$	$\in$	$\{(5,5),$	d_a	$=$	4 (weeks)
		$(10,10)\}$	R^r_{kt}	$=$	5 (days)
$\lvert\mathcal{A}_p\rvert$	$=$	6	R^o_{kt}	$=$	1.5 (days)
$\lvert\mathcal{S}\rvert$	$=$	4	c^r_{kt}	$=$	0
$\lvert\mathcal{S}_r\rvert$	$=$	2	c^o_{kt}	$=$	600
$\lvert\mathcal{S}\rvert$ per period and activity	$=$	2	c^e_s	$\sim$	$TN_{600,1000}(800,100)$
$\lvert\mathcal{S}\rvert$ per activity	$\leq$	3	e_p	$=$	0.2
ρ	$=$	1.5			

Table 3.1: Parameters of the test instances

The minimum and maximum time–lags δ^{min}_a and δ^{max}_a are equal for all activities of one instance. For the instances with 5 projects all possible variations within the range $1 \leq \delta^{min}_a < \delta^{max}_a \leq 6$ were considered resulting in $\sum_{i=1}^{5} i = 15$ variations. For each combination of δ^{min}_a and δ^{max}_a five instances yielding a total of 75 instances were generated. For the instances with 10 projects $\delta^{min}_a = 1$ holds and δ^{max}_a is varied within the range $[2,6]$. By this 5 variations with 5 instances each were obtained and hence 25 instances in total.

Solution method	time limit [sec]
CPLEX (LP)	none
CPLEX (3,600s)	3,600
CPLEX (100s)	100
CPLEX (300s)	300
GA (80s)	80
HMH (100s)	80+20

Table 3.2: Solution methods applied

Solution methods. For the experimental investigation all instances were solved using an Intel Pentium 4 with 2.4 GHz and 1.5 GB RAM. The hybrid metaheuristic (HMH) and the generalized network simplex based evaluation function were implemented in Java 6.0. The solution methods given in Table 3.2 were applied. First of all CPLEX 10.1 with default parameters was employed to solve LP4 (LP–relaxation of MIP4) (CPLEX LP). Next CPLEX with an optimality gap of 0% and a time limit of 3,600 seconds (CPLEX 3,600s) was used to solve MIP4. With these limits optimal or near–optimal benchmark solutions can be found. Finally, MIP4 was solved with CPLEX and a time limit of 100 seconds (CPLEX 100s) for the 5 projects instances and 300 seconds (CPLEX 300s) for the 10 projects instances, respectively, to have a fair comparison with HMH described in Section 3.4. The computation time of HMH was set to 100 seconds where 80 seconds were allocated to the genetic algorithm and the remaining 20 seconds to the tabu search. The intermediate results after the GA part of HMH (GA 80s) were captured as well as the final results after 100 seconds (HMH 100s). For the GA the population size was set to 10 and was equally initialized with individuals of type 1 and 2. Four individuals of the parent generation and six individuals of the offsprings are selected for the next generation. The mutation probability p per gene was set to 0.05. For the TS a maximum tabu list length of 20 was chosen.

3.5.2 Computation times

Evaluation functions. In Section 3.3 two different graphs G_2 and G_3 and the generalized network simplex algorithm (GNS) were proposed in order to evaluate solutions with fixed starting periods. In order to identify the best suited network representation for each metaheuristic and to compare the proposed evaluation procedure with CPLEX a full–factorial comparison of the computation times required to evaluate a solution was conducted. The

Metaheuristic	Evaluation method	$\|\mathcal{P}\| = 5$	$\|\mathcal{P}\| = 10$
TS	GNS on G_2	58.4	147.1
	GNS on G_3	31.1	68.6
	CPLEX	108.0	518.7
GA	GNS on G_2	56.6	151.4
	GNS on G_3	90.2	277.0
	CPLEX	113.5	494.5

Table 3.3: Average evaluation times for instances with $TWS = 5$ in milliseconds

instances with 5 projects and 10 projects and with a time window size

$$TWS = \delta_a^{max} - \delta_a^{min} \tag{3.35}$$

of 5 were tested. The first factor of the full–factorial comparison was the size of the problem, i.e. the number of projects and resources. The second factor was the applied metaheuristic where the TS with a limit of 50 iterations is opposed to the GA with a limit of 100 generations. In order to compute the evaluation time per solution the number of evaluations as well as the total computation time were measured. The final factor was the evaluation function. The three evaluation functions considered were the generalized network simplex (GNS) algorithm using the network representations G_2 vs. G_3 and CPLEX 10.1 applied on LP6.

Table 3.3 provides the mean evaluation times per solution in milliseconds for this full–factorial comparison. The results show that CPLEX is clearly outperformed by GNS up to a factor of 7. Despite its larger network size Graph G_3 outperforms G_2 in the course of the TS where consecutively evaluated solutions are rather similar. However, in the course of the GA the evaluation times are considerably lower for Graph G_2. This is due to the fact that consecutively evaluated solutions are not similar enough to apply Graph G_3 efficiently. However, it was observed that for instances with smaller TWS the GA can benefit from Graph G_3, too.

The results confirm the efficiency of the proposed evaluation function and the proposed network representations. Which of the two network representations is better suited clearly depends on the sequence of evaluated solutions defined by the applied metaheuristic. Hence, for the subsequent results of HMH the GA with GNS on G_2 and TS with GNS on G_3 were used.

CPLEX. The following results provide the mean computation times when solving MIP4 and its LP–relaxation with CPLEX. Table 3.4 shows the mean

δ_a^{min}		δ_a^{max} 2	3	4	5	6	avg.
1	CPLEX (LP)	2.9	7.0	10.7	19.9	94.0	26.9
	CPLEX (100s)	6.6	74.2	100.1	100.2	100.1	76.3
	CPLEX (3,600s)	7.3	81.1	723.6	3,578.9	3,599.5	1,598.1
2	CPLEX (LP)		2.3	9.5	10.8	24.4	11.7
	CPLEX (100s)		11.7	100.1	100.1	100.1	78.0
	CPLEX (3,600s)		11.9	240.9	2,951.2	3,599.6	1,700.9
3	CPLEX (LP)			3.2	6.9	13.0	7.7
	CPLEX (100s)			16.3	100.3	100.1	72.2
	CPLEX (3,600s)			17.4	1,535.9	3,600.0	1,717.8
4	CPLEX (LP)				2.3	6.5	4.4
	CPLEX (100s)				69.0	100.1	84.6
	CPLEX (3,600s)				56.6	1,986.5	1,021.5
5	CPLEX (LP)					4.0	4.0
	CPLEX (100s)					83.8	83.8
	CPLEX (3,600s)					139.6	139.6
avg.	CPLEX (LP)	2.9	4.6	7.8	10.0	28.4	14.5
	CPLEX (100s)	6.6	42.9	72.2	92.4	96.9	77.5
	CPLEX (3,600s)	7.3	46.5	327.3	2,030.7	2,585.0	1,475.3

Table 3.4: Average computation times for combinations of δ_a^{min} and δ_a^{max} and $(|\mathcal{P}|, |\mathcal{R}^i|) = (5, 5)$ in seconds

computation times for all instance with 5 projects and all combinations of δ_a^{min} and δ_a^{max}. Table 3.5 aggregates the data to the time window size TWS. Instances with the same TWS correspond to the diagonals in Table 3.4. Note that the number of instances per TWS for the 5 projects instances is not equal due to the constraint $\delta_a^{max} > \delta_a^{min}$ applied during test data generation. Hence, only 5 instances have $TWS = 5$ while 25 instances have $TWS = 1$. Table 3.5 also contains results for the 10 projects instances.

For the 5 projects instances the Branch&Cut–algorithm of CPLEX requires the entire time span of 3,600 seconds searching for an optimal solution for the instances with $TWS = 5$. CPLEX hits the time limit of 100 seconds and cannot prove optimality for any instance with $TWS = 3$ or higher. The LP–relaxation is solved relatively fast. However, note that it takes almost 100 seconds on average to solve the LP–relaxation of instances with $TWS = 5$ and 5 projects.

An obvious observation is that the computation times are growing with TWS since TWS determines the number of binary variables. Furthermore,

		TWS				
$(\mathcal{P}, \mathcal{R}^i)$	method	1	2	3	4	5
(5,5)	CPLEX (LP)	3.0	7.5	11.5	22.1	94.0
	CPLEX (100s)	37.5	93.7	100.1	100.1	100.1
	CPLEX (3,600s)	46.6	961.1	2,425.0	3,589.2	3,599.5
(10,10) $(\delta_a^{min} = 1)$	CPLEX (LP)	30.9	137.7	322.8	567.6	1,152.0
	CPLEX (300s)	299.9	300.2	300.1	300.1	300.3
	CPLEX (3,600s)	3,600.1	3,600.1	3,600.1	3,600.3	3,600.4

Table 3.5: Average computation times for different time window sizes *TWS* in seconds

the computation times are increasing with δ_a^{max} even if considering the same value of TWS, i.e. in Table 3.4 the computation times increase from the upper left to the lower right. This is due to the fact that the planning horizon T depends on δ_a^{max} and this results in more constraints (e.g. Constraints (3.7)) and variables. Although not explicitly shown here this observation is also true for the hybrid metaheuristic due to the increased size of the generalized minimum cost flow problems (cf. Section 3.3).

For the 10 projects instances the time limit is always hit. Note that the LP–relaxation takes almost 20 minutes on average for the 10 projects instances with $\delta_a^{max} = 6$.

3.5.3 Solution gaps

Table 3.6 reports the mean solution gaps

$$\Delta = \frac{z - LP}{z} \tag{3.36}$$

of the four solution methods for the 5 projects instances with all combinations of δ_a^{min} and δ_a^{max}. The objective function value obtained by the corresponding solution method is denoted z while LP is the optimal value of the LP–relaxation of MIP4. For two instances marked with a plus sign ($^+$) CPLEX could not find a feasible solution within 100 seconds. In Table 3.7 solution gaps for instances with 5 and 10 projects are aggregated to the time window size TWS. Finally, Table 3.8 shows the percentage of instances with 5 projects proved to be optimal. A table for the 10 projects instances is omitted since optimality could not be proved for any of them.

CPLEX performs very well on instances with small time window sizes, especially on $TWS = 1$. However, for the 5 projects instances beginning with $TWS \geq 4$ the hybrid metaheuristic clearly outperforms CPLEX (100s).

δ_a^{min}		δ_a^{max} 2	3	4	5	6
1	CPLEX (3,600s)	0.70%	1.62%	4.03%	5.95%	8.35%
	CPLEX (100s)	0.70%	1.62%	4.51%	9.89%	$^{+}$18.76%
	GA (80s)	0.86%	2.26%	6.84%	8.64%	12.16%
	HMH (100s)	0.86%	2.20%	6.53%	7.63%	10.78%
2	CPLEX (3,600s)		2.32%	3.67%	7.71%	9.00%
	CPLEX (100s)		2.32%	4.00%	8.44%	$^{+}$17.30%
	GA (80s)		2.56%	4.53%	11.12%	14.01%
	HMH (100s)		2.56%	4.06%	9.55%	11.34%
3	CPLEX (3,600s)			2.95%	6.29%	10.57%
	CPLEX (100s)			2.95%	6.83%	11.36%
	GA (80s)			3.06%	8.64%	14.62%
	HMH (100s)			3.06%	7.62%	13.50%
4	CPLEX (3,600s)				4.81%	5.56%
	CPLEX (100s)				4.81%	6.00%
	GA (80s)				7.12%	8.34%
	HMH (100s)				6.53%	7.68%
5	CPLEX (3,600s)					6.19%
	CPLEX (100s)					6.21%
	GA (80s)					8.68%
	HMH (100s)					8.24%

Table 3.6: Average solution gaps Δ for combinations of δ_a^{min} and δ_a^{max} and $(|\mathcal{P}|, |\mathcal{R}^i|) = (5, 5)$ ($^{+}$ indicates one unsolved instance)

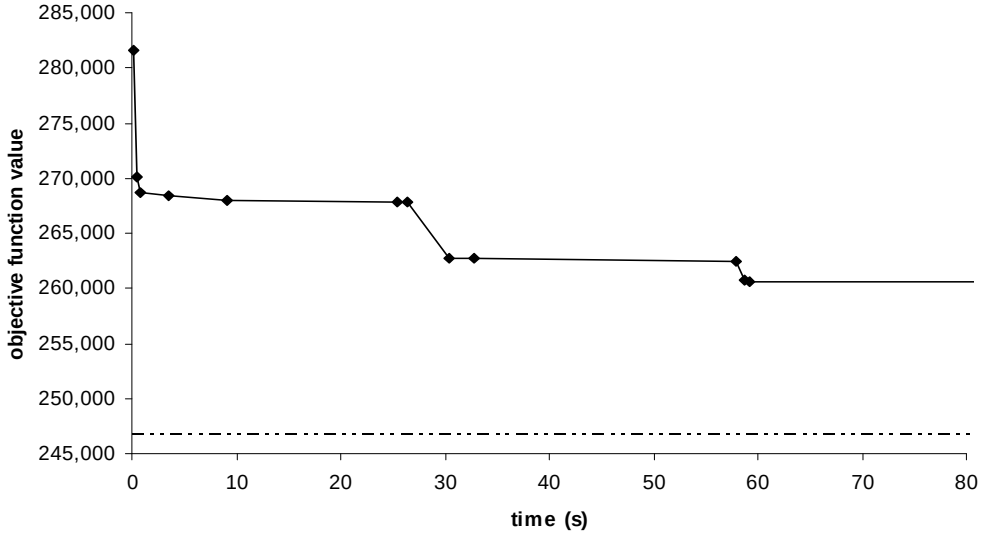


Figure 3.5: Typical evolution of the objective function value during the GA over time. The dotted line represents the optimal solution.

For one instance with 5 projects the solution found by HMH was even better than the one found by CPLEX (3,600s). The use of the tabu search post–optimization reduces the gap by about 0.5% for $TWS = 2$ and up to 2.7% for $TWS \geq 3$. In preliminary tests it could be seen that it is more beneficial to allot the final 20 seconds of the runtime to the TS post–optimization instead of using the entire 100 seconds for the GA. Figure 3.5 shows a typical example for the evolution of the objective function value over time when applying the GA to a 5 projects instance.

For the 10 projects instances and $TWS \leq 2$ (corresponding to $\delta_a^{max} \leq 3$) CPLEX (300s) provides better results than HMH. Note that the CPLEX time limit had to be increased to 300 seconds in order to obtain feasible solutions. However, even CPLEX (300s) fails to generate any feasible solution for $TWS \geq 3$. HMH performs very well for larger TWS as the gap to CPLEX (3,600s) decreases. For one instance with 10 projects the solution found by HMH was even better than the one found by CPLEX (3,600s).

As a conclusion standard solution procedures fail to solve small problem instances with 5 projects and 5 resources optimally and cannot derive feasible solutions for $TWS \geq 4$ in reasonable time. For the 10 projects instances and $TWS \geq 3$ they cannot even find feasible solutions within 5 minutes. On the contrary HMH provides feasible and good results throughout the tested

$(\mathcal{P}, \mathcal{R}^i)$		*TWS* 1	2	3	4	5
(5,5)	CPLEX (3,600s)	3.40%	4.29%	7.44%	7.47%	8.35%
	CPLEX (100s)	3.40%	4.61%	8.10%	$^{+}$13.18%	$^{+}$18.76%
	GA (80s)	4.46%	5.94%	10.86%	11.32%	12.16%
	HMH (100s)	4.25%	5.39%	9.86%	9.48%	10.78%
(10,10)	CPLEX (3,600s)	1.58%	3.03%	3.14%	4.22%	6.44%
	CPLEX (300s)	1.68%	4.01%	*3.19%	—	—
	GA (80s)	5.87%	9.39%	9.48%	11.85%	11.39%
	HMH (100s)	3.91%	6.70%	6.82%	8.94%	9.83%

Table 3.7: Average solution gaps compared to the LP–relaxation for different time windows sizes *TWS* ($^{+}$ indicates one unsolved instance, * indicates four unsolved instances)

problem instances. It is supposed that HMH is the more advantageous the larger the problem instances become.

3.6 Summary

In this chapter simultaneous scheduling and staffing multiple IT–projects with serial structures with a multi–skilled human workforce that has heterogeneous and static efficiencies has been addressed. The considered problem is typical for development and maintenance projects of IT–systems and IT–services as well as research and development projects. A mixed–binary linear program has been proposed and a hybrid metaheuristic consisting of a genetic algorithm and a tabu search has been developed in order to tackle the problem. The hybrid metaheuristic relies on a very efficient network representation of the staffing subproblem. In the experimental study it was demonstrated that the generalized network simplex algorithm can solve the network representation of the staffing subproblem up to 7 times faster than standard solvers like CPLEX. CPLEX often fails to solve the MIP–formulation of small size instances optimally within 100 seconds. Especially for larger instances, e.g. 10 projects, 10 resources and a time window size per activity of 3, CPLEX cannot even find any feasible solution within 5 minutes. Especially regarding these instances the proposed hybrid metaheuristic performs clearly better than CPLEX and finds feasible solutions within seconds.

This chapter has focused on a network representation of the staffing subproblem for resources with static efficiencies. However, in practice efficiencies are not static at all. Hence, the following chapter will concentrate on the

δ_a^{min}		δ_a^{max} 2	3	4	5	6
1	CPLEX (3,600s)	100%	100%	100%	20%	0%
	CPLEX (100s)	100%	80%	0%	0%	0%
	HMH (100s)	60%	20%	0%	0%	0%
2	CPLEX (3,600s)		100%	100%	40%	0%
	CPLEX (100s)		100%	0%	0%	0%
	HMH (100s)		40%	20%	0%	0%
3	CPLEX (3,600s)			100%	100%	0%
	CPLEX (100s)			100%	0%	0%
	HMH (100s)			60%	0%	0%
4	CPLEX (3,600s)				100%	80%
	CPLEX (100s)				100%	0%
	HMH (100s)				20%	0%
5	CPLEX (3,600s)					100%
	CPLEX (100s)					60%
	HMH (100s)					0%

Table 3.8: Percentage of optimally solved instances with $(|\mathcal{P}|, |\mathcal{R}^i|) = (5, 5)$

staffing subproblem w.r.t. learning, knowledge depreciation and skill development requirements.

Chapter 4

Qualification of human resources

4.1 Introduction

In practice the human resources' efficiencies are often not static but change over time due to e.g. learning and forgetting. The type of work the resources are assigned to strongly influences whether a skill improves or deteriorates and determines the resources' qualification profile. Thus, in this chapter the problem of assigning project work to resources considering learning and depreciation of knowledge is addressed (cf. Heimerl and Kolisch [44, 46]). It is assumed that the project schedules are given and that project work can be aggregated resulting in a resource demand of a certain skill in a period of e.g. one month length. Learning and depreciation of knowledge of resources will be taken into account explicitly. In this context two company–specific goals need to be achieved: The operative goal — also pursued in Chapters 2 and 3 — is to minimize costs for performing a given amount of project work during the planning horizon. On a strategic level, however, a decision has to be made who is applying and improving which skills. The strategic goal is to obtain given skill level targets for the organizational unit at the end of the planning horizon. The skill level of an organizational unit is defined as the aggregation of the efficiencies of all individual resources having that skill. Of course, assignment problems can be applied to arbitrary–sized organizational units, e.g. the whole company, departments, or groups. W.l.o.g. it will be referred to the organizational unit of a company for the remainder of this chapter.

This chapter is organized as follows: An overview on relevant literature is given in the upcoming Section 4.2. The optimization model will be in-

troduced in Section 4.3. Section 4.4 will show details on the solution implementation and the analysed test data. Results regarding computational performance and managerial insight will be presented in Section 4.5. Finally, the chapter will conclude with a summary in Section 4.6.

4.2 Literature Review

Effects causing time–dependent efficiencies can be split up in endogenous effects (caused and influenced by properties of the person) and exogenous effects (caused by the environment). Endogenous effects are e.g. learning (efficiency increasing) and forgetting (efficiency decreasing). There is a large amount of scientific work on the description of learning and forgetting in general (cf. e.g. Arzi and Shtub [6], Jaber and Bonney [50], Yelle [78]). However, only few assignment models consider learning or training of human resources (cf. Gutjahr et al. [40], Wu and Sun [76]). Efficiency decreasing effects are considered by even fewer assignment models (cf. Gutjahr et al. [40], Nembhard [56], Süer and Tummaluri [67]). Instead, most assignment models assume static and often homogeneous efficiencies for all resources. However, staffing decisions directly influence which type of skill is used and, thus, determine whether knowledge for a skill is used and built up or forgotten.

The process of gaining experience by accomplishing real tasks and learning from successes and failures is often also called training–on–the–job. Training is defined as "the systematic acquisition of skills, rules, concepts or attitudes that result in improved performance" (cf. Goldstein [37, p. 230]). Learning effects have been studied and observed extensively for repetitive physical tasks, often called "blue collar work" (cf. Hopp et al. [49]). However, they have also been measured for cognitive and knowledge–based work (cf. e.g. Arzi and Shtub [6], Nembhard and Uzumeri [59]). The empirical results of Boh et al. [17] show that learning effects are present even in special project environments with knowledge–based work. Their finding is based on software maintenance projects. This non–manual and often creative work ("white collar work", cf. Hopp et al. [49] for definitions) is getting more and more important. In 2006 already 72% of Germany's labour force worked in the service sector and this rising trend still continues (cf. Statistisches Bundesamt Deutschland [66]). A large amount of the employees in this sector have to do cognitive and knowledge–based work as "information technology (IT) has given virtually every job an element of knowledge work" (cf. Hopp et al. [49, p. 2]).

An elegant way to describe learning processes has been proposed by Wright [75]. He introduces learning curves which describe how unit pro-

duction time decreases with the amount of cumulative produced units. An extensive amount of literature deals with different types of learning curves for different types of work (cf. e.g. Nembhard and Uzumeri [58], Yelle [78] for an overview). Learning curves have been applied primarily to physical work (cf. Hopp et al. [49]). However, the aforementioned empirical results of Boh et al. [17] for knowledge based project work encourage to apply learning curves to project work, too. Furthermore, Amor [5] applies learning curves when scheduling programs with repetitive projects. Finally, Hopp et al. [49, p. 26] propose to "extend the learning curve approach [...] to knowledge–intensive work environments".

Forgetting as the counterpart to learning is usually caused by breaks (cf. Globerson et al. [35]). The degree of forgetting is based on "how recently an individual's practice was obtained" (cf. Nembhard [56, p. 1959]). Obviously, staffing decisions have direct impact on the future efficiency of a human resource due to learning and forgetting.

An example for an exogenous effect on efficiency is technological progress. It is known to be extremely rapid in the IT–sector and requires the adoption of new knowledge in order to stay competitive. Furthermore it renders fractions of old knowledge useless (cf. Chen and Edgington [24]) and often increases complexity (cf. Vanhoucke [72]). This can lead to a relative deterioration of efficiency. On the other hand after having adopted to new technologies resources' efficiency is usually higher. To illustrate the relative deterioration of efficiency imagine the example of computer programming skills. Over the past years and decades several new programming languages (e.g. C++, Java, and C#) have been developed and have now become state of the art. The importance of other older languages (FORTRAN, Pascal, or C) dropped rapidly because they are hardly able to support and take advantage of current technologies like e.g. the internet, 3D–graphics or service–oriented architectures. Hence, nowadays a human resource would not be considered a highly efficient programmer if he or she had only expert knowledge in FORTRAN and Pascal. It cannot be said that this human resource had no programming skills at all since basic concepts are similar in every programming language. However, this particular human resource would have had to gain knowledge in evolving programming languages in order to stay competitive. Hence, it would take much more time to perform today's work with his meanwhile obsolete knowledge. His efficiency has deteriorated compared to a state of the art benchmark programmer.

The approach presented in this chapter has only been scarcely discussed in the literature. Nembhard [56] solves the problem of assigning resources to skills with assignment policies but does not consider depreciation of knowledge or strategic goals. Wu and Sun [76] apply a meta–heuristic to the

scheduling and staffing problem of projects w.r.t. learning but they neglect depreciation of knowledge and strategic goals as well. Strategic goals and the topic of skill development have been treated by Gutjahr [38]. He derives analytical results concerning the optimal distribution of efforts from a mathematical optimization model. However, in the approach presented by Gutjahr [38] projects are aggregated to project classes and resources are not distinguished in detail.

The approach proposed in the following section has many similarities with Gutjahr et al. [40]. Both models use multi–skilled resources and apply the concept of training–on–the–job and an additive forgetting model. However, Gutjahr et al. [40] employ a multi–objective function maximizing strategic goals and project portfolio value while the model proposed in this chapter has a single–objective function which minimizes costs while the strategic goal of company skill level targets will be taken into account by constraints. Furthermore, neither project selection nor scheduling will be considered. Instead only staffing and outsourcing decisions are taken into account.

4.3 Model

A work package (s,t) is defined as the aggregated demand r_{st} of project work requiring skill $s \in \mathcal{S}$ in period $t = 1,\ldots,T$, where T denotes the planning horizon. The pool of resources consists of internal resources $\mathcal{R}^i$ and external resources $\mathcal{R}^e$, i.e. $\mathcal{R} = \mathcal{R}^i \cup \mathcal{R}^e$. The demand of work packages requiring skill s can be assigned to resources $k \in \mathcal{R}_s = \mathcal{R}^i_s \cup \mathcal{R}^e_s$ with cost rates c_{kt} in period t. $\mathcal{R}_s$ is the set of resources capable of performing project work which requires skill s. The set of skills that resource k possesses is denoted with $\mathcal{S}_k$. Working times of resources are limited by time–dependent availabilities R_{kt}.

The projects are broken up into skill–specific work packages (s,t), which — in contrast to the projects itself — are expected to have a repetitive character. Thus, the related learning process is well–suited to be described by learning curves (cf. Wright [75]).

Learning curves are usually monotonically decreasing and convex functions. An extensive amount of literature deals with different types of learning curves for different types of work (cf. e.g. Nembhard and Uzumeri [58], Yelle [78] for an overview). A learning curve $f_{ks}(z_{ks})$ describes the unit production time, i.e. the amount of time that resource k requires to produce one additional unit after having produced z_{ks} units using skill s. The number of units z_{ks} produced by resource k in skill s can be interpreted as the experience of the resource w.r.t. the particular skill. The function f_{ks} is indexed with

resource k and skill s since the learning process depends on the type of work s and the abilities of resource k to adapt new knowledge. Furthermore, the argument z_{ks} is also indexed with k and s, i.e. cross–skill or team learning effects are neglected.

The time $F_{ks}(z_{ks})$ that resource k requires to produce the first z_{ks} units using skill s can be expressed by the integral $F_{ks}(z_{ks}) = \int_{z'=0}^{z_{ks}} f_{ks}(z')dz'$. The time τ_{ks} required to produce x_{ks} units after having produced $z_{ks} - x_{ks}$ units can then be calculated by $\tau_{ks} = F_{ks}(z_{ks}) - F_{ks}(z_{ks} - x_{ks})$.

The experience level of resource k in skill s at the end of period t is defined by z_{kst}. Resource k's amount of project work performed using skill s in period t is represented by x_{kst}. Resource k's amount of depreciated knowledge β_{kst} in skill s is modelled as a loss of experience at the beginning of period t. Given an initial experience level z_{ks0} of resource k in skill s at the end of period $t = 0$, z_{kst} can be calculated by $z_{kst} = z_{ks(t-1)} - \beta_{kst} + x_{kst}$. As stated earlier β_{kst} can depict e.g. forgetting and/or technological progress: For example release dates of new technologies which are known in advance can be modelled due to the index t. Using index s the impact on experience can be modelled skill–dependent. E.g. the release of a new programming language might affect programming skills but not database skills. With index k resources can be discriminated w.r.t. their forgetting properties. Note that z_{kst} can become negative. However, w.l.o.g. $z_{kst} \geq 0$ can be assured by shifting the learning curve $\sum_{t=1}^{T} \beta_{kst}$ units to the right by and setting $z_{ks0} := z_{ks0} + \sum_{t=1}^{T} \beta_{kst}$.

This approach is similar to dynamic inventory models (cf. e.g. Zipkin [81]) with z_{kst} being the current inventory of knowledge, β_{kst} being the demand or loss of knowledge and x_{kst} being the ordered and instantaneously delivered (or produced) knowledge.

Strategic goals might require to keep or develop certain skills across the company. A company's skill level is defined as the aggregation of the efficiencies of all individual resources having that skill. To define strategic goals the production possibility frontier (PPF, also known as transformation curve) is employed. The PPF is a concave curve describing maximum production quantities for different skills given several (but limited) resources (cf. Varian [73]). Note that in the described assignment problem the PPF is not static or given in advance but depends on the actual assignments as it is transformed over time by learning and depreciation of knowledge of human resources. In the following model no requirements on the shape of the complete PPF are supposed but only on the values of its intercept points at the axes at the end of the planning horizon. The intercept point at axis s is required to be at least ϕ_s. Thus, the level ϕ_s is the guaranteed total amount of work per time unit requiring skill s that internal human resources could perform at the end of the planning horizon T if they all used that skill. Figure 4.1 gives

an example for the possible result of the PPF at the end of the planning horizon. A different assignment decision would yield a different PPF. W.r.t. the given values of ϕ_1 and ϕ_2 this solution is feasible, since production rates for both skills could attain level ϕ_s independently (but not concurrently).

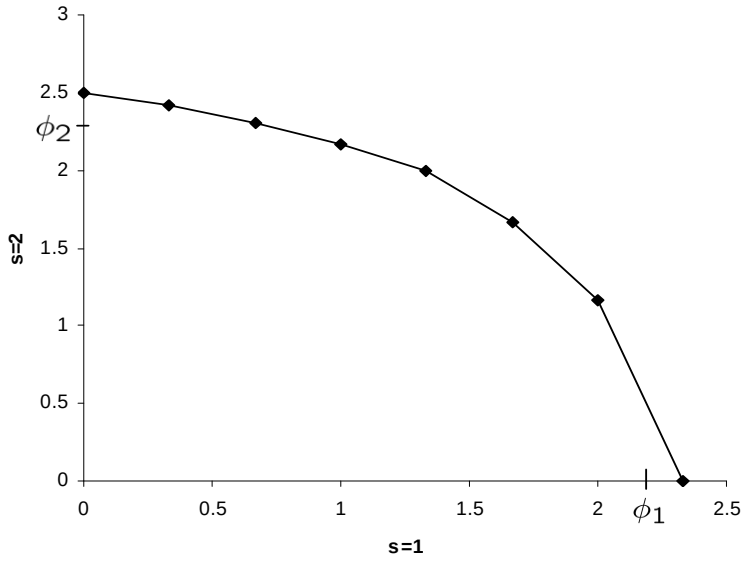


Figure 4.1: Production possibility frontier

The intercept points of the PPF can be calculated as the sum of production rates of the internal resources with skill s at the end of the planning horizon T defined by

$$\sum_{k \in \mathcal{R}_s^i} \frac{1}{f_{ks}(z_{ksT})}. \tag{4.1}$$

Note that the production rate is the reciprocal of the unit production time calculated by $f_{ks}(\cdot)$.

The outlined problem is modelled as a non–linear program by employing the following decision variables: The amount of work done by resource k with skill s in period t is denoted with x_{kst}. Resource k's experience in skill s cumulated up to period t considering depreciation of knowledge is depicted with z_{kst}. Finally, τ_{kst} is the time that resource k requires to process the amount of work x_{kst} of skill s in period t.

$$Z = \text{Min} \quad \sum_{s \in \mathcal{S}} \sum_{k \in \mathcal{R}_s} \sum_{t=1}^{T} c_{kt} \cdot \tau_{kst} \tag{4.2}$$

subject to

$$z_{kst} = z_{ks(t-1)} - \beta_{kst} + x_{kst} \qquad \begin{array}{l} k \in \mathcal{R}_s \\ s \in \mathcal{S} \\ t = 1, \ldots, T \end{array} \tag{4.3}$$

$$\tau_{kst} = F_{ks}\left(z_{kst}\right) - F_{ks}\left(z_{kst} - x_{kst}\right) \qquad \begin{array}{l} k \in \mathcal{R}_s \\ s \in \mathcal{S} \\ t = 1, \ldots, T \end{array} \tag{4.4}$$

$$\sum_{k \in \mathcal{R}_s^i} \frac{1}{f_{ks}(z_{ksT})} \geq \phi_s \qquad s \in \mathcal{S} \tag{4.5}$$

$$\sum_{k \in \mathcal{R}_s} x_{kst} \geq r_{st} \qquad \begin{array}{l} s \in \mathcal{S} \\ t = 1, \ldots, T \end{array} \tag{4.6}$$

$$\sum_{s \in \mathcal{S}} \tau_{kst} \leq R_{kt} \qquad \begin{array}{l} k \in \mathcal{R} \\ t = 1, \ldots, T \end{array} \tag{4.7}$$

$$x_{kst}, \tau_{kst} \geq 0 \qquad \begin{array}{l} k \in \mathcal{R}_s \\ s \in \mathcal{S} \\ t = 1, \ldots, T \end{array} \tag{4.8}$$

$$z_{kst} \in \mathbb{R} \qquad \begin{array}{l} k \in \mathcal{R}_s \\ s \in \mathcal{S} \\ t = 1, \ldots, T \end{array} \tag{4.9}$$

The objective function (4.2) minimizes the costs for performing the work packages (s, t). Constraints (4.3) are the dynamic experience level constraints, where the experience level z_{kst} of resource k in skill s at the end of period t is the experience level of the preceding period $(t-1)$ decreased by depreciation of knowledge β_{kst} and increased by the acquisition of knowledge x_{kst} due to work assignments in period t. Constraints (4.4) calculate the time τ_{kst} required to perform x_{kst} units of work package (s, t). Constraints (4.5) enforce the company skill level targets ϕ_s at the end of the planning horizon. Constraints (4.6) ensure that the demand r_{st} of work package (s, t) is assigned to the resources. Continuous fractional assignment of work packages to resources is allowed. Constraints (4.7) restrict the availability of the resources. Constraints (4.8) and (4.9) define the decision variables.

If there is no learning and no forgetting, model (4.2)–(4.9) becomes linear with the learning function $f_{ks}(z_{ks}) = \frac{1}{\eta_{ks}}$, where η_{ks} is the static efficiency of resource k w.r.t. skill s (cf. e.g. Heimerl and Kolisch [45]).

Although the learning functions $f_{ks}(z_{ks})$ are assumed to be decreasing and $F_{ks}(z_{ks})$ is therefore concave, the model is neither a convex nor a concave optimization problem since constraints (4.4) calculate the difference of two concave functions resulting in an unclassifiable function.

As stated in Section 4.2 this model is similar to the one of Gutjahr et al. [40], especially concerning constraints (4.3). However, in Gutjahr et al. [40] the experience (called "competence score" in [40]) of resources increases due to the amount of *time* invested into a skill. In contrast, the amount of *work* done by a resource is assumed to drive its experience in the proposed model. The strategic goal of the proposed model is formulated in constraints (4.5) enforcing minimum *cumulative production rates per skill* while Gutjahr et al. [40] maximize the *sum of final competence scores* in a multi–objective function. For the conversion of experience to production rates ("efficiency values" in [40]) in the experimental studies of the following section non–linear learning curves are used as opposed to their linear approximations.

4.4 Implementation and Test setup

For the experimental study an adaptation of the exponential learning function (cf. Nembhard and Uzumeri [58], Pendharkar and Subramanian [62]) is employed, i.e.

$$f_{ks}(z_{ks}) = a_{ks}e^{-\lambda_{ks}z_{ks}} + b_{ks} \tag{4.10}$$

and therefore

$$F_{ks}(z_{ks}) = \frac{a_{ks}}{\lambda_{ks}}\left(1 - e^{-\lambda_{ks}z_{ks}}\right) + b_{ks}z_{ks}. \tag{4.11}$$

$b_{ks} > 0$ represents the steady state unit production time, $\lambda_{ks} \geq 0$ is the learning rate and $a_{ks} \geq 0$ the learning potential of resource k in skill s. Note that knowledge depreciation is depicted by β_{kst} in constraints (4.3) and is therefore implicitly included in the learning curve. This type of learning function was chosen due to the ability to depict steady state unit production times and its mathematical tractability. However, the implementation can easily be adapted to other learning functions.

The model was implemented in C++ using Ipopt 3.3.2 of the COIN–OR library (cf. [26]). Ipopt uses a primal–dual interior point filter line search algorithm (cf. Wächter and Biegler [74]) to return an (at least locally) optimal solution. The model was solved 50 times using random starting points in order to partially overcome the lack of only local optimality.

Test instances were generated using the base case depicted in Table 4.1. The values are chosen to obtain realistic small size instances with period lengths of one month and efforts measured in man days. A set of external

resources $\mathcal{R}_s^e$ was taken into account by adding one additional resource with unlimited availabilities for each skill, i.e. $|\mathcal{R}_s^e| = 1\ \forall s$ and $R_{kt} = \infty\ \forall t, k \in \mathcal{R}_s^e$. External resources will not be subject to learning nor forgetting and their unit production time is 1, i.e. $a_{ks} = 0$ and $b_{ks} = 1\ \forall s, k \in \mathcal{R}_s^e$. These numbers are based on the assumption that there will always be enough qualified external resources with state–of–the–art knowledge. The costs of external resources depend on the requested skill, and it is referred to skill 1 as the cheapest and skill 4 as the most expensive skill.

All six internal resources are capable of performing two out of four skills and each resource has a unique combination of skills. A learning rate of $\lambda_{ks} = 0.012$ together with $a_{ks} = 0.2$ and $b_{ks} = 0.9$ leads to a unit production time of 1 after approximately 60 work units. The parameters of $|\mathcal{R}^i|$, $|\mathcal{S}_k|$, r_{st}, R_{kt} were chosen such that the utilization of internal resources is 150% if no external resources are employed and all internal resources had static unit production times of 1, i.e. $f_{ks}(\cdot) \equiv 1$. Furthermore, despite learning effects, the utilization will always be higher than 100%.

T	=	6	
$\lvert\mathcal{S}\rvert$	=	4	
$\lvert\mathcal{R}^i\rvert$	=	6	
R_{kt}	=	20	$\forall t, k \in \mathcal{R}^i$
r_{st}	=	45	$\forall s, t$
a_{ks}	=	0.2	$\forall s, k \in \mathcal{R}_s^i$
b_{ks}	=	0.9	$\forall s, k \in \mathcal{R}_s^i$
z_{ks0}	=	0	$\forall s, k \in \mathcal{R}_s^i$

		Skill			
		1	2	3	4
	c_s^e	400	500	600	700
Resource	1	x	x		
	2	x		x	
	3	x			x
	4		x	x	
	5		x		x
	6			x	x

Table 4.1: Parameters and skill matrix of the base case

Four parameters were varied to generate the 16 instances given in Table 4.2. The first group of instances (1–8) uses a flat learning curve ($\lambda_{ks} = 0.012$), the other group (9–16) uses a steeper learning curve ($\lambda_{ks} = 0.02$). The shape of the learning curve affects the speed of learning in the sense that resources with a steeper learning curve learn faster. Figure 4.2 depicts the plots of the learning curves. Within each group some instances (1–4, 9–12) neglect forgetting ($\beta_{kst} = 0$), the remaining instances impose forgetting ($\beta_{kst} = 10$). Some instances (1–2, 5–6, 9–10, 13–14) do not demand company skill level targets ($\phi_s = 0$), while all the other instances do ($\phi_s > 0$). Furthermore, cases with internal costs $c_k^i = 0$ (odd instance numbers) and $c_k^i = 500$ (even instance numbers) are compared. For the latter instances

internal resources are more expensive than external resources w.r.t. skill 1 ($c_k^i = 500$ vs. $c_1^e = 400$) in case both internal and external resources had equal and horizontal learning curves (i.e. $f_{ks}(\cdot) \equiv 1$).

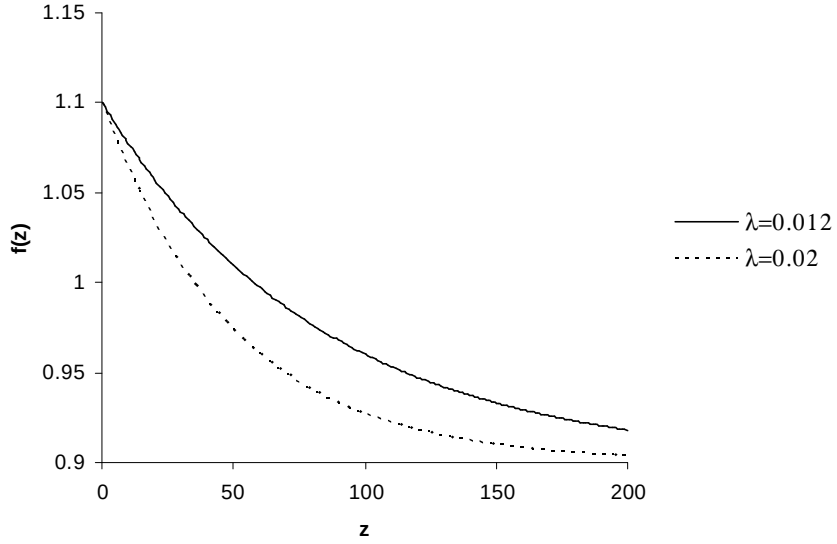


Figure 4.2: Plots of the flat ($\lambda = 0.012$) and the steep ($\lambda = 0.02$) learning curve $f(z)$

4.5 Results

4.5.1 Computational study

The test instances led to a model with 96 variables and 70 constraints. They were solved on a Pentium 4 with 2.4 GHz and 1.5 GB RAM. The required time to solve 50 random starting point problems was recorded (cf. Table 4.3). For the instances given in Table 4.2 the solution time is 21 seconds on average and always less than 33 seconds. The instances with internal costs $c_k^i = 500$ are harder to solve than those with $c_k^i = 0$ (24.6 vs. 17.0 seconds). This is probably due to the fact that the full utilization of internal resources is not always an optimal solution in cases where $c_k^i = 500$ and thus constraints (4.7) are not always binding.

The benefit of using random starting points is measured using the average and maximum deviation of different starting point solutions to the best solution found. For 5 out of 16 instances (31%) the use of random starting

#	λ_{ks}	β_{kst}	ϕ_s	c_k^i
1	0.012	0	0	0
2	0.012	0	0	500
3	0.012	0	2.8	0
4	0.012	0	2.8	500
5	0.012	10	0	0
6	0.012	10	0	500
7	0.012	10	2.5	0
8	0.012	10	2.5	500
9	0.02	0	0	0
10	0.02	0	0	500
11	0.02	0	3.0	0
12	0.02	0	3.0	500
13	0.02	10	0	0
14	0.02	10	0	500
15	0.02	10	2.25	0
16	0.02	10	2.25	500

Table 4.2: Variations of the base case

points led to no improvements. On the other hand, the average distance from the locally optimal solutions to the best solutions found was 0.24% and the maximum distance was 8.3%. Thus, for some instances choosing a good starting point matters and random starting points are a straight–forward strategy to find a good solution.

Additionally, in order to analyse the limits of Ipopt–algorithm some larger instances as given in Table 4.4 were tested. The instances are based on instance #1 given in Table 4.2, where the number of resources $|\mathcal{R}^i|$ and skills $|\mathcal{S}|$ has been increased and two random skills were assigned to each resource. For each combination of $|\mathcal{R}^i|$ and $|\mathcal{S}|$ five test instances were generated. The results are given in Table 4.4. Obviously, even large instances with 100 resources and 50 skills can be solved within minutes using the 50 random starting points approach. However, the instances did not benefit from the use of random starting points.

4.5.2 Managerial insights

4.5.2.1 Analysis methods

To gain managerial insights the results are analysed on the individual level of a resource and on the aggregate level of the company, in which the values

#	time [sec]	best	avg dev %	max dev %
1	13.5	159,164	0.00	0.00
2	21.8	516,994	0.06	0.18
3	18.1	163,787	0.00	0.00
4	24.9	523,407	0.00	0.02
5	17.9	177,638	0.06	1.38
6	32.7	531,690	0.00	0.00
7	18.7	191,577	0.35	2.07
8	23.6	551,577	0.11	0.55
9	14.8	147,619	0.13	0.14
10	18.1	506,487	0.18	0.26
11	14.0	168,480	0.00	0.00
12	17.4	528,480	0.00	0.00
13	18.5	173,990	0.60	8.30
14	30.6	527,117	0.72	1.85
15	20.3	182,632	1.45	3.52
16	27.7	542,631	0.23	0.91
mean	20.8	349,579	0.24	1.20

Table 4.3: Computational results

$\lvert\mathcal{R}^i\rvert$	$\lvert\mathcal{S}\rvert$	avg. time [sec]	max dev %
25	10	45.5	0.00
50	25	110.7	0.00
100	50	494.5	0.00

Table 4.4: Additional computational results

of all internal resources are summed up. Let

$$x_{ks}^{\mathcal{R}} = \sum_{t} x_{kst} \tag{4.12}$$

be the amount of work done by resource k for skill s during the entire planning horizon and let

$$x_{s}^{\mathcal{C}} = \sum_{k \in \mathcal{R}^i} x_{ks}^{\mathcal{R}} \tag{4.13}$$

be the amount of work done by internal (company) resources for skill s during the planning horizon, then

$$\rho_s = \frac{x_s^{\mathcal{C}}}{\sum_t r_{st}} \tag{4.14}$$

is the fraction of work ρ_s done by internal resources for skill s during the planning horizon. To analyse the *outsourcing decisions* the value of ρ_s is used. Further let

$$X_k^{\mathcal{R}} = \{x_{k1}^{\mathcal{R}}, x_{k2}^{\mathcal{R}}, x_{k3}^{\mathcal{R}}, \ldots\} \tag{4.15}$$

and

$$X^{\mathcal{C}} = \{x_1^{\mathcal{C}}, x_2^{\mathcal{C}}, x_3^{\mathcal{C}}, \ldots\} \tag{4.16}$$

be sets that combine the data points of the different skills. The coefficient of variation

$$\mathrm{CV}(X^{\mathcal{C}}) = \frac{\sqrt{\mathrm{VAR}\left(X^{\mathcal{C}}\right)}}{\mathrm{E}\left(X^{\mathcal{C}}\right)} \tag{4.17}$$

gives an indication about the *specialization of the company*: A high coefficient of variation indicates high specialization of the company, since the assignment of internal resources to work packages of different skills is very unequal, i.e. high for some skills, low for others. The values of ρ_s and $\mathrm{CV}(X^{\mathcal{C}})$ for the solutions of the instances are listed in Table 4.5.

The *specialization of individual resource* $k \in \mathcal{R}^i$ can be measured with

$$\overline{\sigma}_k = \frac{\sigma_k}{\sigma_k^{max}} \tag{4.18}$$

which uses the standard deviation

$$\sigma_k = \sqrt{\mathrm{VAR}(X_k^{\mathcal{R}})} = \sqrt{\mathrm{E}\left((X_k^{\mathcal{R}})^2\right) - \left(\mathrm{E}\left(X_k^{\mathcal{R}}\right)\right)^2} = \sqrt{\frac{\sum_s \left(x_{ks}^{\mathcal{R}}\right)^2}{|\mathcal{S}_k|} - \left(\frac{\sum_s x_{ks}^{\mathcal{R}}}{|\mathcal{S}_k|}\right)^2} \tag{4.19}$$

and is normalized with the maximum possible standard deviation σ_k^{max}. σ_k^{max} can be determined by calculating σ_k if only one skill is used as much as

#	ρ_1	ρ_2	ρ_3	ρ_4	$\mathrm{CV}(X^{\mathcal{C}})$	$\overline{\sigma}_1$	$\overline{\sigma}_2$	$\overline{\sigma}_3$	$\overline{\sigma}_4$	$\overline{\sigma}_5$	$\overline{\sigma}_6$	avg.
1	0.00	0.62	1.00	1.00	0.156	1.00	1.00	1.00	1.00	0.37	0.48	0.81
2	0.00	0.44	1.00	1.00	0.172	1.00	1.00	1.00	1.00	1.00	0.17	0.86
3	0.16	0.46	1.00	1.00	0.138	1.00	1.00	0.40	1.00	0.92	0.48	0.80
4	0.16	0.16	1.00	1.00	0.180	0.03	1.00	1.00	1.00	1.00	0.17	0.70
5	0.00	0.48	1.00	1.00	0.167	1.00	1.00	1.00	0.68	1.00	0.21	0.82
6	0.00	0.00	1.00	1.00	0.250	0.00	1.00	1.00	1.00	1.00	0.11	0.68
7	0.31	0.36	0.81	1.00	0.117	0.84	0.91	0.21	1.00	1.00	1.00	0.83
8	0.31	0.36	0.81	1.00	0.117	0.84	0.91	0.21	1.00	1.00	1.00	0.83
9	0.00	0.81	0.91	1.00	0.146	1.00	1.00	1.00	1.00	0.64	1.00	0.94
10	0.00	0.46	1.00	1.00	0.170	1.00	1.00	1.00	1.00	1.00	0.20	0.87
11	0.39	0.47	0.81	1.00	0.092	0.22	0.56	0.41	1.00	0.15	1.00	0.56
12	0.39	0.47	0.81	1.00	0.092	0.33	0.57	0.41	1.00	0.15	1.00	0.58
13	0.00	0.68	0.86	1.00	0.151	1.00	1.00	1.00	1.00	0.30	1.00	0.88
14	0.00	0.00	0.98	1.00	0.250	0.00	1.00	1.00	1.00	1.00	0.24	0.71
15	0.28	0.40	0.86	1.00	0.120	0.91	1.00	0.30	1.00	1.00	1.00	0.87
16	0.27	0.30	0.86	1.00	0.134	0.83	1.00	0.27	1.00	1.00	1.00	0.85

Table 4.5: Fraction ρ_s of work done internally, specialization of the company $\mathrm{CV}(X^{\mathcal{C}})$ and specialization $\overline{\sigma}_k$ of internal resources

possible. If all skills are used equally by resource k, i.e. $x^{\mathcal{R}}_{ks} = x^{\mathcal{R}}_{ks'}\ \forall s, s'$, a value of $\overline{\sigma}_k = 0$ is obtained. If only one skill is used, $\overline{\sigma}_k = 1$ holds. The values for the specialization $\overline{\sigma}_k$ are also listed in Table 4.5.

Additionally the assignment of work packages to internal resources over time on the company level are visualized as in Figure 4.3(a). The work done by all internal resources ($\sum_{k\in\mathcal{R}^i} x_{kst}$) is shown on the ordinate while the time line is on the abscissa. The four skills are represented by four different curves. For example in Figure 4.3(a) skill 1 is not done internally at all while the amount of work that is done internally for skill 2 increases with each period. Note that the curves for skill 3 and 4 are super–imposed at a maximum of $\sum_{k\in\mathcal{R}^i} x_{k3t} = \sum_{k\in\mathcal{R}^i} x_{k4t} = 45$.

Figure 4.3(b) depicts the assignments on an individual level. Each subdiagram shows the assignments of work requiring one skill to the resources. The work done by internal resources is shown as stacked bars on the ordinate while the time line is on the abscissa. For example in Figure 4.3(b) the top right subdiagram depicts skill 2. The work requiring that skill is done by resource 1 and furthermore by resource 5 in a growing amount. The height of the stacked bars correspond to the strictly monotone increasing curve for skill 2 in Figure 4.3(a).

4.5.2.2 Outsourcing decisions

As stated in Section 4.4, due to the chosen problem parameters, the average utilization of the internal resources is higher than 100%. Therefore, it is not possible to do all the work internally. From Table 4.5 it can be observed that, in general, those work packages are outsourced (low values of ρ_1 and ρ_2)

which require cheap skills, i.e. a skill s that can be outsourced at relatively low costs c^e_s. On the other hand work packages which require expensive skills are (almost) completely done internally (ρ_3 and ρ_4 close or equal to 1). As expected, the inequality $\rho_s \geq \rho_{s'}$ holds if $c^e_s > c^e_{s'}$.

4.5.2.3 Impact of the learning curve

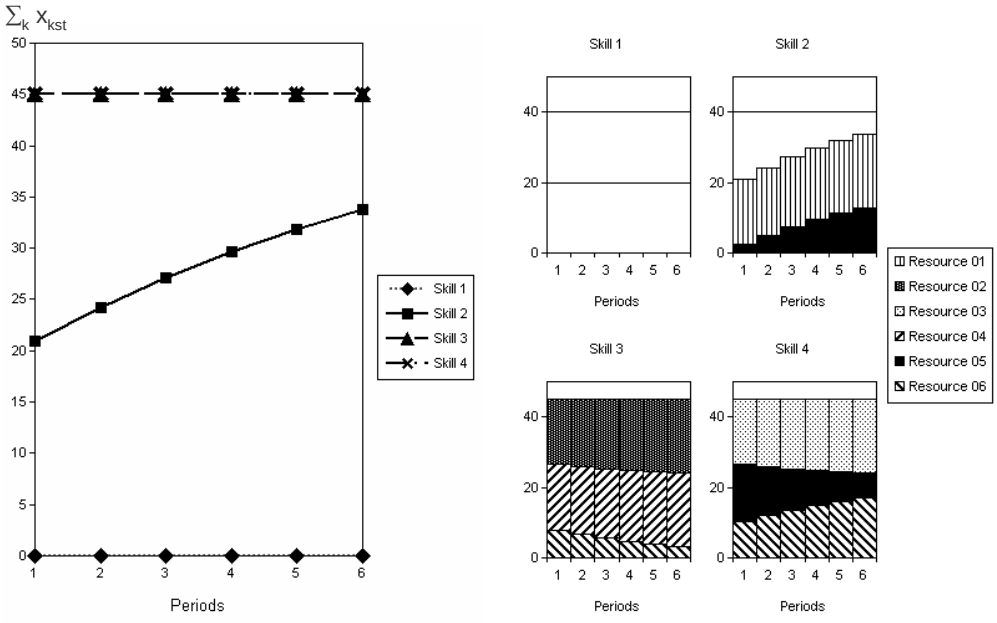
Impact on the company When comparing instances 1 and 5 with its counterparts 9 and 13 that have a steeper learning curve, it can be seen that a steeper learning curve leads to a *broader qualification of the company*, since $\mathrm{CV}(X^{\mathcal{C}})$ in Table 4.6 is lower for instances 9 and 13. This holds regardless of the depreciation of knowledge.

#	λ_{ks}	β_{kst}	ρ_1	ρ_2	ρ_3	ρ_4	$\mathrm{CV}(X^{\mathcal{C}})$
1	0.012	0	0.00	0.62	1.00	1.00	0.156
5	0.012	10	0.00	0.48	1.00	1.00	0.167
9	0.02	0	0.00	0.81	0.91	1.00	0.146
13	0.02	10	0.00	0.68	0.86	1.00	0.151

Table 4.6: Fraction ρ_s of work done internally and specialization of the company $\mathrm{CV}(X^{\mathcal{C}})$ for instances 1, 5, 9 and 13

Figure 4.3(a) which graphically shows the solution for instance 1 illustrates this finding. Skill 3 and 4 are completely done by internal resources ($\sum_{k\in\mathcal{R}^i} x_{kst} = r_{st} = 45$), skill 1 is completely outsourced ($\sum_{k\in\mathcal{R}^i} x_{kst} = 0$) and the amount of work that is done internally for skill 2 increases over time. The increase of internal work for skill 2 can be explained with the four diagrams shown in Figure 4.3(b). It stems from the efficiency gains of resources 3 and 6, that enable resource 5 to use more of its capacity performing skill 2. Furthermore, resources 5 and 1 are becoming more efficient in doing skill 2 as well.

At first sight it seems that the following greedy heuristic would provide optimal results: *i*) Order work packages in decreasing order of external skill costs. *ii*) Beginning with the work package requiring the most expensive external skill, assign work packages to available internal resources as long as possible. *iii*) Outsource the remaining work. However, Figure 4.4(a) gives a counterexample. The figure shows the solution for instance 9 which, compared to instance 1, has a steeper learning curve. The difference compared to instance 1 is that some of the work for the more expensive skill 3 has been outsourced ($\sum_{k\in\mathcal{R}^i} x_{kst} < r_{st} = 45$) in favour of skill 2 which is now done more intensively by internal resources and less outsourced. The reason for this phenomenon can be examined using Figure 4.4(b) which reveals that

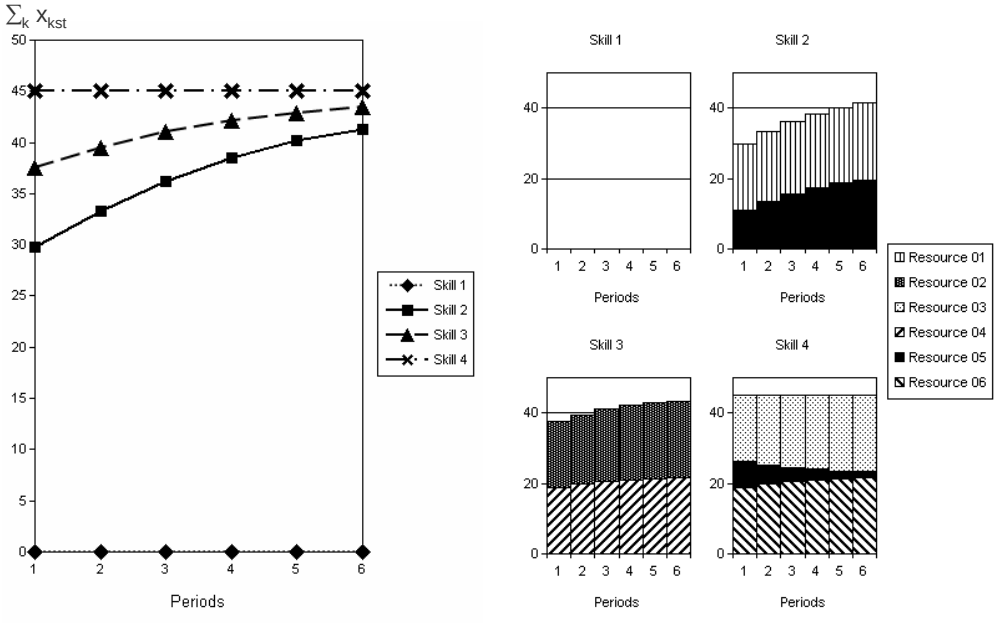


(a) company level (b) individual level

Figure 4.3: Solution of instance 1 with $\lambda_{ks} = 0.012$, $\phi_s = 0$, $\beta_{kst} = 0$, and $c_k^i = 0$

the full dedication of resources 2 and 4 to skill 3 and resource's 5 higher dedication to skill 2 leads to lower overall costs.

Impact on individual resources The impact of the learning curve's shape on the individual resources can be examined with a comparison of $\overline{\sigma}_k$ for instances 1 and 5 with its counterparts 9 and 13 in Table 4.7. Regarding these four instances a steeper learning curve seems to lead to *more specialized individual resources* since $\overline{\sigma}_k$ is higher on average. These numbers are supported by the fact that there are 4 fully specialized resources ($\overline{\sigma}_k = 1$) in instance 1 while there are 5 of them in instance 9. Resource 5 (with skills 2 and 4) and resource 6 (with skills 3 and 4) are both capable of performing skill 4, while skill 2 (the other skill of resource 5) is cheaper than skill 3 (the other skill of resource 6) (cf. Table 4.1). Therefore, one would expect that resource 5 (with the cheaper other skill) is devoted to perform skill 4 rather than resource 6 (with the more expensive other skill). However, in instance 9 resource 5 is qualified for both skills while resource 6 is fully specialized for skill 4. This contradicts the intuition that a higher difference of skill



(a) company level (b) individual level

Figure 4.4: Solution of instance 9 with $\lambda_{ks} = 0.02$, $\phi_s = 0$, $\beta_{kst} = 0$, and $c_k^i = 0$

values (as for resource 5) should lead to a stricter specialization for the more expensive skill.

#	λ_{ks}	β_{kst}	$\overline{\sigma}_1$	$\overline{\sigma}_2$	$\overline{\sigma}_3$	$\overline{\sigma}_4$	$\overline{\sigma}_5$	$\overline{\sigma}_6$	avg.
1	0.012	0	1.00	1.00	1.00	1.00	0.37	0.48	0.81
5	0.012	10	1.00	1.00	1.00	0.68	1.00	0.21	0.82
9	0.02	0	1.00	1.00	1.00	1.00	0.64	1.00	0.94
13	0.02	10	1.00	1.00	1.00	1.00	0.30	1.00	0.88

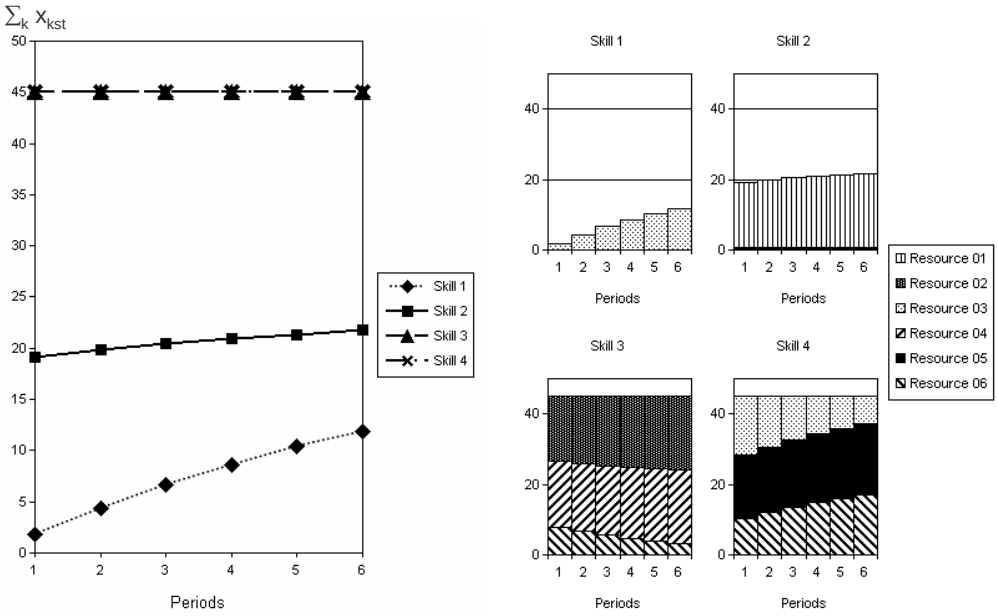
Table 4.7: Specialization $\overline{\sigma}_k$ of internal resources for instances 1, 5, 9 and 13

The numbers disclose that the specialization measures on the individual and company level diverge. A steeper learning curve leads to more specialized individual resources in terms of σ_k but a broader qualified company in terms of $\mathrm{CV}(X^{\mathcal{C}})$.

4.5.2.4 Impact of company skill level targets

Company skill level targets broaden the company's qualification. This can be seen by comparing the coefficients of variation $\mathrm{CV}(X^{\mathcal{C}})$ of instances 1 and

3 (0.156 vs. 0.138), 5 and 7 (0.167 vs. 0.117), 9 and 11 (0.146 vs. 0.092), as well as 13 and 15 (0.151 vs. 0.120) in Table 4.5, respectively. The proposition is also supported by comparison of Figures 4.3(a) and 4.5(a). It can clearly be seen that in the latter figure the cheapest skill 1 is used and trained to reach the required company skill level targets of $\phi_s = 2.8$ for all skills, while it is completely outsourced in Figure 4.3(a). In order to reach this goal less work of skill 2 is done by internal resources which in turn leads to a higher amount of outsourced work demanding this skill.



(a) company level (b) individual level

Figure 4.5: Solution of instance 3 with $\lambda_{ks} = 0.012$, $\phi_s = 2.5$, $\beta_{kst} = 0$, and $c_k^i = 0$

Of course, increasing company skill level targets ϕ_s lead to higher overall costs since the solution space is reduced due to constraints (4.5). Due to the different shapes of the learning curves the company skill level targets ϕ_s are hardly comparable in a quantitative manner. Therefore, Figure 4.6 depicts the dependence of the costs on the company skill level targets for further variations of instance 1 with a flat learning curve. For $\phi_s \geq 3$ no feasible solutions could be found.

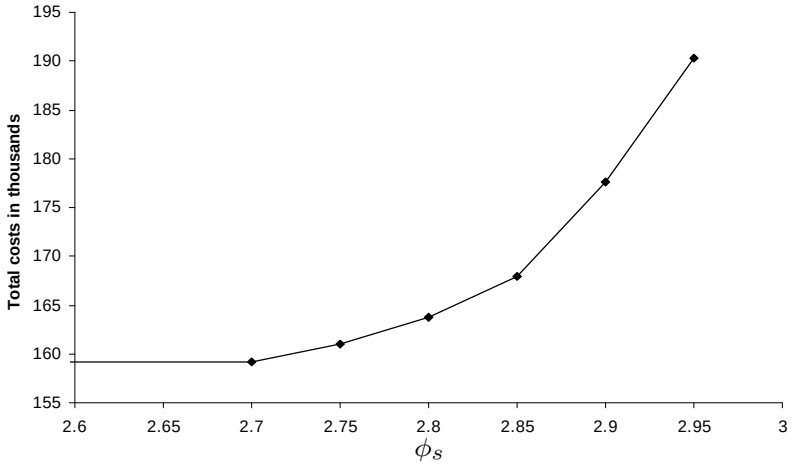


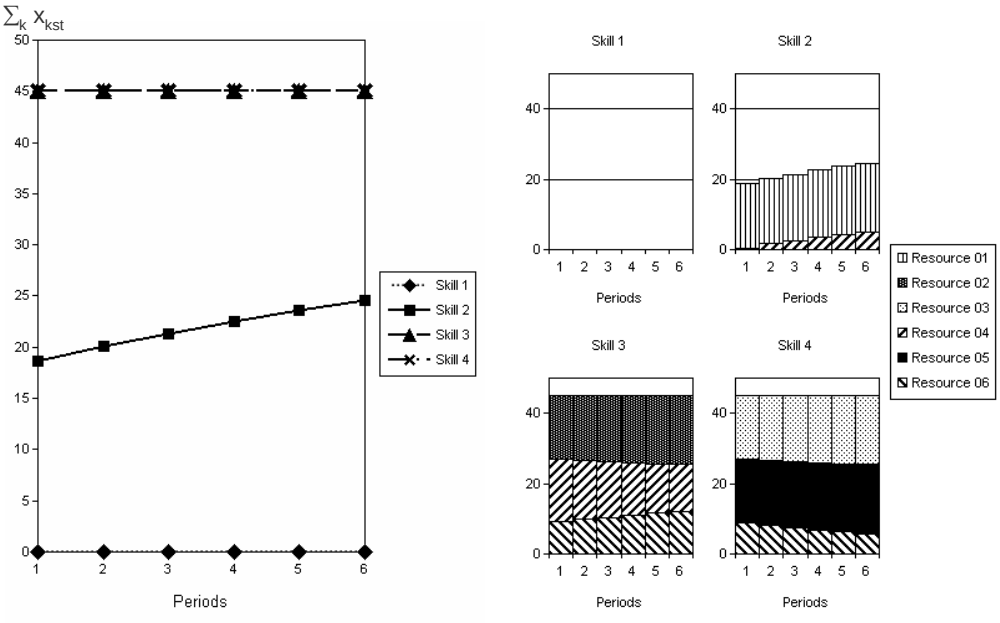
Figure 4.6: Influence of company skill level targets ϕ_s on costs

4.5.2.5 Impact of knowledge depreciation

Basically similar results were observed comparing instances 1 and 5 as well as 9 and 13. No structural difference on the company level between the solutions of different levels of knowledge depreciation can be identified for instances with company skill level targets $\phi_s = 0$: As can be seen in Figure 4.3(a) and Figure 4.7(a), expensive skills are done completely internally while cheap skills are outsourced. On the individual level, however, the assignments of work to resources shift. Of course, the fraction of work ρ_s that can be done internally decreases with the amount of knowledge depreciation. As a consequence costs increase by more than 10% (cf. Table 4.3). No particular impact of the influence of knowledge depreciation on specialization measures $\mathrm{CV}(X^{\mathcal{C}})$ and $\overline{\sigma}_k$ can be observed.

The combined effects of knowledge depreciation and synchronously imposed company skill level targets $\phi_s = 2.5$ are illustrated by Figure 4.8(a) which shows the results for instance 7. As before, assignment in decreasing order of external skill costs is basically still reasonable. But as it has been observed for the case of company skill level targets without knowledge depreciation, the amount of work for skill 1 done by internal resources must be non–zero.

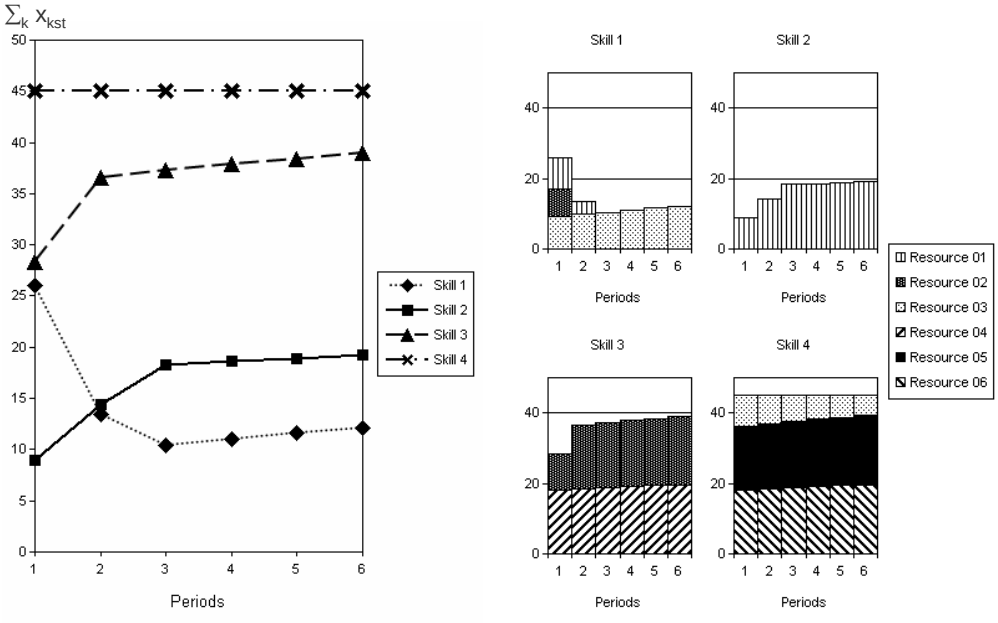
It is noteworthy that, in order to reach a company skill level target in constraints (4.5), only the cumulated amount of work done in that skill is important. This is due to constraints (4.3) in which depreciation of knowledge is a subtractive term. Thus, it is a constant at the end of planning horizon regardless of the assignment decisions. Therefore, scheduling of work pack-



(a) company level (b) individual level

Figure 4.7: Solution of instance 5 with $\lambda_{ks} = 0.012$, $\phi_s = 0$, $\beta_{kst} = 10$, and $c_k^i = 0$

age assignments is irrelevant for reaching company skill level targets ϕ_s. In fact, scheduling of the work package assignments is only important for the objective function (4.2). In case of knowledge depreciation and company skill level targets the solutions do not only depend on the assignment decisions (i.e. who does what) alone but on the time line of assignments (i.e. scheduling, who does what in which period) as well. The decision of doing a skill rather in the beginning or in the end of the planning horizon in order to reach company skill level targets at minimum costs is obviously important. If an amount r_s of work has to be done in one single period t by resource k subject to knowledge depreciation, it can be done in less time in period $t = 1$ (when $f_{ks}(\cdot) = f_{ks}(z_{ks0})$) than in period $t = T$ (when $f_{ks}(\cdot) \gg f_{ks}(z_{ks0})$ due to knowledge depreciation). In this context, it makes sense that in Figure 4.8(b) resource 3 is completely dedicated to skill 1 during the entire planning horizon while resources 1 and 2 perform skill 1 only in the first periods to reach the company skill level targets.



(a) company level (b) individual level

Figure 4.8: Solution of instance 7 with $\lambda_{ks} = 0.012$, $\phi_s = 2.5$, $\beta_{kst} = 10$, and $c_k^i = 0$

4.5.2.6 Impact of internal costs

With costs of $c_k^i = 500$ the allocation of internal resources to cheap skills ($c_1^e = 400$) is avoided, even if internal resources are not completely utilized. For the parameters of the relevant test instances internal resources cannot compete with external resources in terms of costs for skill 1 and 2 despite learning effects. This can be seen in the results when comparing ρ_s of odd instance numbers ($c_k^i = 0$) with the values of even instance numbers ($c_k^i = 500$). However, in order to reach company skill level targets $\phi_s > 0$ in instances 4, 8, 12 and 16 the assignment of relatively expensive internal resources for these skills is mandatory.

4.6 Summary

In this chapter a constrained non–linear continuous optimization model was presented in order to address the problem of cost–minimal assignment of project work to internal and external multi–skilled human resources. The effects of learning and depreciation of knowledge as well as the goal of reaching

company skill level targets at the end of the planning horizon are taken into account.

The results show that the heuristic implementation has a promising computational performance. The use of random starting points improved the quality of the solutions significantly for some instances. The solutions were analysed w.r.t. the influence of problem parameters. For selected parameter combinations the intuition is supported that work packages requiring externally expensive skills should be done by internal resources while work packages requiring cheaper skills should be outsourced. However, several observations showed that an implementation of this policy as a greedy assignment heuristic will not provide optimal results. Furthermore, it was shown that the shape of the learning curve leads to different qualification strategies which depend on the level of analysis: Faster learning makes human resources more specialized which paradoxically leads to a broader qualification of the company. Company skill level targets are an effective tool to broaden the qualification of the company but they can induce high costs. Due to the objective of cost–minimization, depreciation of knowledge together with company skill level targets cause assignment decisions to depend on the time line (i.e. scheduling): Work packages requiring less expensive skills when outsourced are done more intensively by internal resources in the beginning than in the end of the planning horizon in order to minimize costs.

Chapter 5

Conclusion and Outlook

The problems treated in this thesis arose in the IT–departments of a large semiconductor manufacturer and of a large telecommunication company. There, multiple projects such as the selection, development, installation, and configuration of new IT–systems for different departments have to be done on a recurrent basis. For the processing of the projects external and internal human resources with different skills and different unit costs have to be used. The question is how human resources are assigned to project work such that different requirements are met and costs are minimized.

Several properties of human resources and projects in an IT–environment that are different from other environments have been introduced. One characteristic property of human resources working in IT–projects is that they are multi–skilled. Hence, resources cannot be aggregated to disjoint groups and this adds further complexity to the decision problem. Furthermore, learning and depreciation of knowledge have considerable impact on the efficiencies of individual human resources in an IT environment. Due to these effects assignment decisions strongly determine the development of human resources and the whole company in terms of their skill portfolio. Thus, requirements on the required skill portfolio in the future should guide assignment decisions. Finally, outsourcing is a very common and popular option in IT–projects due to lack of either skill or availability of internal human resources. Hence, outsourcing needs to be considered in IT–specific optimization models. One requirement is that e.g. bounds on the degree of outsourcing must be imposed; otherwise the success of the projects is at risk. All these properties render the use of many existing project staffing models inappropriate.

In Chapter 2 of the thesis the focus was on the integrated multi–skill project staffing and scheduling problem. Projects were strongly aggregated and were only allowed to shifted en bloc, i.e. only the starting time of the whole project and not of single activities was subject to optimization. The

results point out that from a mathematical point of view scheduling and staffing IT–projects is a very complex and challenging task. The problem is NP–hard even if the projects are strongly aggregated. Hence, large instances can hardly be solved to optimality using standard solution methods implemented in commercial software. Nevertheless they have been applied to test data inspired by real data of a semiconductor manufacturer. A sophisticated problem formulation has been proposed which clearly outperforms the straightforward formulation. Using the sophisticated formulation even standard solution methods can find optimal solutions in a reasonable amount of time for problem sizes relevant in practice. Based on the test data the dependency of costs on different problem parameters has been shown. One of the findings is that marginal cost savings are high at a low qualification level but decrease rapidly with higher qualification levels. In this context qualification level is measured in terms of the number of skills per resource. Together with costs for training one can find optimal qualification levels for the resources. A similar effect has been observed regarding the temporal flexibility of project start times. Allowing a temporal flexibility of one period reduced costs by about 3% compared to no flexibility at all. However, additional flexibility could hardly contribute to further cost savings.

Regarding the staffing problem it has been demonstrated that assignment heuristics that are often applied in practice perform clearly worse than an optimization based approach. When considering outsourcing constraints the heuristics have significant problems to find feasible solutions at all. Furthermore, it has been shown to which extent a centralised staff assignment approach is advantageous compared to decentralised and separate planning. Depending on the heterogeneity of the human resources' skills cost savings have been measured up to 14%.

In a subsequent step the project scheduling problem was generalized in Chapter 3. There, projects were defined as a stream of project phases. These phases may either overlap or succeed each other with no or some lag. The generalized problem is NP–hard as well. For problem sizes relevant in practice standard solution methods cannot find optimal or even any feasible solution in reasonable time at all; even if applying the sophisticated problem formulation. Thus, a generalized minimum cost flow based hybrid metaheuristic has been proposed. Starting times of the project phases are chosen and optimized using a hybrid genetic and tabu search algorithm. For the evaluation of intermediate solutions the staffing subproblem has to be solved. This subproblem can be formulated as a generalized minimum cost flow problem and can then be solved by the generalized network simplex algorithm. It has been shown that the subproblem can be solved much faster by this network flow approach than by standard solution methods. The solutions of large problem

instances of the staffing and scheduling problem that have been found by the metaheuristic are better than those found by standard solution methods. Especially when imposing time limits the metaheuristic outperforms standard solution methods also for smaller problem instances.

Finally, in Chapter 4 the focus was on the staffing subproblem considering learning and depreciation of knowledge. Furthermore, the goal of reaching company target skill levels at the end of the planning horizon has been introduced. To tackle the problem a non–linear optimization model has been developed. It can be solved by a primal–dual interior point filter line search algorithm. A public library implementing this algorithm has been adapted to the given problem. Some test data was generated with different parameters regarding the problem size, learning and knowledge depreciation rate (technological progress) and target skill levels. It has been shown that small problem instances can be solved within seconds. Larger problems with 100 human resources can still be solved within five minutes. One of the key findings was that the learning rate has opposite impacts on the specialization of the company and of individual human resources. The higher the learning rate the more specialized the human resources tend to get. In contrast the company's skill portfolio will not be specialized but broadened. Furthermore, the costs of broadening a company's skill portfolio have been analysed. For the chosen test instances they can increase costs up to 20%. Finally, it has been demonstrated that depreciation of knowledge do have structural impact on the results especially when imposing company skill level targets. In that case the temporal assignment of resources to project work, i.e. the order in which skills should be used, is of particular importance.

This thesis demonstrated the benefit of quantitative approaches to the management of human resources in IT–projects. Of course, due to the complexity of the proposed optimization models some of the aspects treated in other publications or relevant in practice needed to be simplified and neglected. E.g. aspects of fairness, satisfaction and adequacy of project staffing decisions had to be neglected in favour of a single goal of cost minimization. General project network structures or the prolongation (stretching) of projects were beyond the scope of this thesis, too. Finally, all values and data were assumed to be deterministic.

Based on the results of this thesis an interesting field for further research is the stepwise consideration of stochastics within the optimization models. The highest uncertainty in IT–projects is imposed by the size of the work packages as scope changes in the course of a project are quite common and technologies change rapidly. Furthermore, the size of the work packages for implementation is often determined only after some preliminary analysis

phases. Hence, it is of particular importance to embed the optimization models in a dynamic and repeated planning process.

On the topic of learning and depreciation of knowledge efforts should be spent on the combination of the proposed project staffing model and the metaheuristic framework used for project scheduling. On the field of algorithms a more sophisticated selection of starting points for the nonlinear optimization problem could provide better results. Furthermore, other global search techniques like e.g. genetic algorithms might also be well–suited to solve the nonlinear problem.

Appendix A

Notation

$a \in \mathcal{A}$	=	set of activities
$a_0 \notin \mathcal{A}$	=	dummy start activity
$\mathcal{A}_p$	=	set of activities of project p
B_p	=	budget of project p
c_s^e	=	cost rate for external resources performing skill s
c_k^o	=	cost rate for internal resource k during overtime
c_k^r	=	cost rate for internal resource k during regular work time
d_p, d_a	=	duration of project p / activity a
δ_a^{max}	=	maximum start–to–start time–lag between activity $pred(a)$ and a
δ_a^{min}	=	minimum start–to–start time–lag between activity $pred(a)$ and a
e_p	=	minimum relative amount of project p's work to be performed by internal resources
ES_p, ES_a	=	earliest start period of project p / activity a
η_{sk}	=	efficiency of resource k when working in skill s
$\mathcal{R} = \mathcal{R}^i \cup \mathcal{R}^e$	=	set of (internal and external) human resources
LF_p, LF_a	=	latest finish period of project p / activity a
LS_p, LS_a	=	latest start period of project p / activity a
$p \in \mathcal{P}$	=	set of projects
$pred(a)$	=	predecessor of activity a
$q = 1, \ldots, d$	=	index of the executing period of project p (project periods) / activity a
r_{psq}	=	amount of work that is required for skill s by project p in period t

Table A.1: Notation for the MIP (continued on following page)

R^e_{ts}	=	availability of external resources with skill s in period t
R^o_{kt}	=	availability of resource k in period t during overtime
R^r_{kt}	=	availability of resource k in period t during regular work time
$\mathcal{R}_s = \mathcal{R}^i_s \cup \mathcal{R}^e_s$	=	set of resources, capable to work in skill s
$s \in \mathcal{S}$	=	set of skills
$\mathcal{S}_k$	=	set of skills, which can be performed by resource k
$succ(a)$	=	successor of activity a
$t = 1, \ldots, T$	=	index of the (calendar) periods
T	=	index of the last period of the planning horizon
$\mathcal{T}_{pt}$, $\mathcal{T}_{at}$	=	set of tupels (τ, q) of project/activity start period τ and executing period q which lead to a demand of project p / activity a in calendar period t
u	=	size of work units
x^o_{ptsk}, x^o_{atsk}	=	amount of work done by internal resource k for project p / activity a using skill s in period t during overtime
x^r_{ptsk}, x^r_{atsk}	=	amount of work done by internal resource k for project p / activity a using skill s in period t during regular work time
y_{pts}, y_{ats}	=	amount of work done by external resources for project p / activity a using skill s in period t
y^i_{pts}	=	discrete number of work units done by external resources for project p using skill s in period t
z_{pt}, z_{at}	=	1, if project p / activity a is started at the beginning of period t, 0 otherwise

Table A.2: Notation for the MIP (continued)

Bibliography

[1] N. Abboud, M. Inuiguchi, M. Sakawa, and Y. Uemura. Manpower allocation using genetic annealing. *European Journal of Operational Research*, 111(2):405–420, Dec. 1998.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows, Theory, Algorithms, and Applications*. Prentice Hall, New Jersey, 1993.

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

[4] H. K. Alfares and J. E. Bailey. Integrated project task and manpower scheduling. *IIE Transactions*, 29(9):711–717, Sept. 1997.

[5] J.-P. Amor. Scheduling programs with repetitive projects using composite learning curve approximations. *Project Management Journal*, 33 (3):16–29, Sept. 2002.

[6] Y. Arzi and A. Shtub. Learning and forgetting in mental and mechanical tasks: A comparative study. *IIE Transactions*, 29(9):759–768, 1997.

[7] M. Asay. 62 percent of it projects fail. Why? World Wide Web electronic publication, Mar. 2008. http://news.cnet.com/8301-13505_3-9900455-16.html.

[8] D. P. Ballou and G. K. Tayi. A decision aid for the selection and scheduling of software maintenance projects. *IEEE Transactions on Systems, Man, and Cybernetics — Part A: Systems and Humans*, 26(2):203–212, 1996.

[9] A. Barreto, M. d. O. Barros, and C. M. Werner. Staffing a software project: A constraint satisfaction and optimization–based approach. *Computers & Operations Research*, 35(10):3073–3089, Oct. 2008.

[10] M. Bassett. Assigning projects to optimize the utilization of employees' time and expertise. *Computers & Chemical Engineering*, 24(2–7):1013–1021, July 2000.

[11] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali. *Linear programming and network flows*, volume 2. Wiley, New York, 1990.

[12] O. Bellenguez and E. Néron. Methods for the multi-skill project scheduling problem. In *9th International Workshop on Project Management and Scheduling (PMS'2004), Nancy*, pages 66–69, 2004.

[13] O. Bellenguez and E. Néron. Exact method for a fixed job problem. In *Proceedings der International Conference on Industrial Engineering and Systems Management (IESM05), Marrakech, Morocco*, pages 203–205, 2005.

[14] O. Bellenguez and E. Néron. Lower bounds for the multi–skill project scheduling problem with hierarchical levels of skills. *Lecture Notes in Computer Sciences*, 3616:229–243, 2005.

[15] O. Bellenguez-Morineau and E. Néron. A branch-and-bound method for solving multi-skill project scheduling problem. *RAIRO - Operations Research*, 41(2):155–170, 2007.

[16] R. Bhatnagar, V. Saddikutti, and A. Rajgopalan. Contingent manpower planning in a high clock speed industry. *International Journal of Production Research*, 45(9):2051–2072, May 2007.

[17] W. F. Boh, S. A. Slaughter, and J. A. Espinosa. Learning from experience in software development: A multilevel analysis. *Management Science*, 53(8):1315–1331, Aug. 2007.

[18] P. Brucker, A. Drexl, R. Möhring, K. Neumann, and E. Pesch. Resource–constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1):3–41, 1998.

[19] J. O. Brunner, J. Bard, and R. Kolisch. Flexible shift scheduling of physicians. *Health Care Management Science*, 3(3):285–305, Sept. 2009.

[20] M. J. Brusco. An exact algorithm for a workforce allocation problem with application to an analysis of cross–training policies. *IIE Transactions*, 40(5):495–508, May 2008.

[21] X. Cai and K. Li. A genetic algorithm for scheduling staff of mixed skills under multi–criteria. *European Journal of Operational Research*, 125(2): 359–369, 2000.

[22] G. M. Campbell. Cross–utilization of workers whose capabilities differ. *Management Science*, 45(5):722–732, 1999.

[23] G. M. Campbell and M. Diaby. Development and evaluation of an assignment heuristic for allocating cross–trained workers. *European Journal of Operational Research*, 138(1):9–20, Apr. 2002.

[24] A. N. K. Chen and T. M. Edgington. Assessing value in organizational knowledge creation: Considerations for knowledge workers. *MIS Quarterly*, 29(2):279–309, June 2005.

[25] S. C. Chu and C. K. Lin. A manpower allocation model of job specialization. *Journal of the Operational Research Society*, 44(10):983–989, Oct. 1993.

[26] Coin-OR. *http://www.coin-or.org/projects/Ipopt.xml*.

[27] A. Corominas, J. Ojeda, and R. Pastor. Multi–objective allocation of multi–function workers with lower bounded capacity. *Journal of the Operational Research Society*, 56(6):738–743, Nov. 2005.

[28] A. Corominas, R. Pastor, and E. Rodriguez. Rotational allocation of tasks to multifunctional workers in a service industry. *International Journal of Production Economics*, 103(1):3–9, Sept. 2006.

[29] E. Davis. Project network summary measures constrained–resource scheduling. *AIIE Transactions*, 7:132–142, 1975.

[30] B. Dodin and A. A. Elimam. Audit scheduling with overlapping activities and sequence-dependent setup costs. *European Journal of Operational Research*, 97(1):22–33, Feb. 1997.

[31] A. Drexl. Scheduling of project networks by job assignment. *Management Science*, 37(12):1590–1602, 1991.

[32] H. Eiselt and V. Marianov. Employee positioning and workload allocation. *Computers & Operations Research*, 35(2):513–524, Feb. 2008.

[33] A. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27, Feb. 2004.

[34] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to NP–Completeness*. W.H. Freeman and Company, New York, 1979.

[35] S. Globerson, N. Levin, and A. Shtub. The impact of breaks on forgetting when performing repetitive tasks. *IIE Transactions*, 21:376–381, 1989.

[36] F. Glover and G. A. Kochenberger. *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston et al., 2003.

[37] I. L. Goldstein. Training in work organizations. *Annual Review of Psychology*, 31:229–272, 1980.

[38] W. J. Gutjahr. Optimal dynamic portfolio selection for projects under a competence development model. *OR Spectrum*, accepted for publication, 2009.

[39] W. J. Gutjahr and P. Reiter. Bi–objective project portfolio selection and staff assignment under uncertainty. Technical report, Department of Statistics and Decision Support Systems, Universität Wien, 2008.

[40] W. J. Gutjahr, S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk. Competence–driven project portfolio selection, scheduling and staff assignment. *Central European Journal of Operations Research*, 16(3):281–306, 2008.

[41] W. J. Gutjahr, S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk. Multi–objective decision analysis for competence–oriented project portfolio selection. Technical report, Department of Statistics and Decision Support Systems, Universität Wien, 2008.

[42] C. Heimerl. Ressourcenoptimierung in der IT. Master's thesis, Diplomarbeit, Lehrstuhl für Technische Dienstleistungen und Operations Management, Technische Universität München, Nov. 2005.

[43] C. Heimerl and R. Kolisch. Integrated manpower allocation and multi–project scheduling in an IT–environment. In J. Jozefowska and J. Weglarz, editors, *Abstracts of the Tenth International Workshop on Project Management and Scheduling*, Poznan, Poland, Apr. 2006.

[44] C. Heimerl and R. Kolisch. Qualification of multi–skilled human resources considering learning and forgetting. In *Abstracts of the Eleventh International Workshop on Project Management and Scheduling*, Istanbul, Turkey, Apr. 2008.

[45] C. Heimerl and R. Kolisch. Scheduling and staffing multiple projects with a multi–skilled workforce. *OR Spectrum*, online first, 2009.

[46] C. Heimerl and R. Kolisch. Work assignment to and qualification of multi–skilled human resources under knowledge depreciation and company skill level targets. *International Journal of Production Research*, online first, June 2009.

[47] C. Heimerl and R. Kolisch. An efficient hybrid metaheuristic for integrated scheduling and staffing IT–projects based on a generalized minimum cost flow network. Technical report, Technische Universität München, Mar. 2009.

[48] M. H. A. Hendriks, B. Voeten, and L. H. Kroep. Human resource allocation in a multi–project R&D environment: Resource capacity allocation and project portfolio planning in practice. *International Journal of Project Management*, 17(3):181–188, June 1999.

[49] W. J. Hopp, S. M. R. Iravani, and F. Liu. Managing white–collar work: An operations–oriented survey. *Production and Operations Management*, 18(1):1–32, 2009.

[50] M. Jaber and M. Bonney. A comparative study of learning curves with forgetting. *Applied Mathematical Modelling*, 21(8):523–531, 1997.

[51] G. Karthikeyan and K. Krishnaswamy. Assembly manpower allocation under proportionality constraints. *European Journal of Operational Research*, 44(1):39–46, Jan. 1990.

[52] J. L. Kennington and R. V. Helgason. *Algorithms for network programming*. Wiley, New York et al., 1980.

[53] P. Kløvstad. The technological network: Macro and micro structure. In H. Lombaers, editor, *Project planning by network analysis*, pages 441–450. North–Holland, Amsterdam, 1969.

[54] D. Krüger and A. Scholl. Managing and modelling general resource transfers in (multi–)project scheduling. *OR Spectrum*, online first, 2008.

[55] W. Kwak, Y. Shi, and K. Jung. Human resource allocation in a CPA firm: A fuzzy set approach. *Review of Quantitative Finance & Accounting*, 20(3):277–290, May 2003.

[56] D. A. Nembhard. Heuristic approach for assigning workers to tasks based on individual learning rates. *International Journal of Production Research*, 39(9):1955–1968, 2001.

[57] D. A. Nembhard and N. Osothsilp. Learning and forgetting–based worker selection for tasks of varying complexity. *Journal of the Operational Research Society*, 56(5):576–587, 2005.

[58] D. A. Nembhard and M. V. Uzumeri. An individual–based description of learning within an organization. *IEEE Transactions on Engineering Management*, 47(3):370–378, 2000.

[59] D. A. Nembhard and M. V. Uzumeri. Experiential learning and forgetting for manual and cognitive tasks. *International Journal of Industrial Ergonomics*, 25(4):315–326, 2000.

[60] K. Neumann and J. Zimmermann. Resource levelling for projects with schedule–dependent time windows. *European Journal of Operational Research*, 117(3):591–605, Sept. 1999.

[61] K. Neumann, C. Schwindt, and J. Zimmermann. *Project scheduling with time windows and scare resources*. Springer, Berlin, Heidelberg, 2nd edition, 2003.

[62] P. C. Pendharkar and G. H. Subramanian. An empirical study of ICASE learning curves and probability bounds for software development effort. *European Journal of Operational Research*, 183(3):1086–1096, Dec. 2007.

[63] A. Pritsker, L. Watters, and P. Wolfe. Multiproject scheduling with limited resources : A zero–one programming approach. *Management Science*, 16(1):93–108, Sept. 1969.

[64] W. W. Royce. Managing the development of large software systems: Concepts and techniques. In *Technical Papers of Western Electronic Show and Convention (WesCon)*, Los Angeles, USA, Aug. 1970.

[65] S. Sayin and S. Karabati. Assigning cross–trained workers to departments: A two–stage optimization model to maximize utility and skill improvement. *European Journal of Operational Research*, 176(3):1643–1658, Feb. 2007.

[66] Statistisches Bundesamt Deutschland. Dienstleistungen und Finanzdienstleistungen — Strukturwandel in Deutschland. World Wide Web electronic publication, June 2009. http://www.destatis.de/.

[67] G. A. Süer and R. R. Tummaluri. Multi–period operator assignment considering skills, learning and forgetting in labour–intensive cells. *International Journal of Production Research*, 46(2):469–493, Jan. 2008.

[68] B. W. I. Taylor, L. J. Moore, and E. R. Clayton. R&D project selection and manpower allocation with integer nonlinear goal programming. *Management Science*, 28(10):1149–1158, Oct. 1982.

[69] G. L. Vairaktarakis. The value of resource flexibility in the resource–constrained job assignment problem. *Management Science*, 49(6):718–732, June 2003.

[70] V. Valls, A. Pérez, and S. Quintanilla. A graph colouring model for assigning a heterogeneous workforce to a given schedule. *European Journal of Operational Research*, 90(2):285–302, 1996.

[71] V. Valls, A. Pérez, and S. Quintanilla. Skilled workforce scheduling in service centres. *European Journal of Operational Research*, 193(3): 791–804, Mar. 2009.

[72] M. Vanhoucke. The effect of project schedule adherence and rework on the duration forecast accuracy of earned value metrics. Technical report, Ghent University, Mar. 2009.

[73] H. R. Varian. *Intermediate Microeconomics: A Modern Approach*. Norton, New York, 7th edition, 2006.

[74] A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[75] T. Wright. Factors affecting the cost of airplanes. *Journal of Aeronautical Science*, 3(4):122–128, 1936.

[76] M.-C. Wu and S.-H. Sun. A project scheduling and staff assignment model considering learning effect. *The International Journal of Advanced Manufacturing Technology*, 28(11–12):1190–1195, May 2006.

[77] Y.-K. Wu. On the manpower allocation within matrix organization: A fuzzy linear programming approach. *European Journal of Operational Research*, 183(1):384–393, Nov. 2007.

[78] L. E. Yelle. The learning curve: Historical review and comprehensive survey. *Decision Sciences*, 10(2):302–328, 1979.

[79] M. Yoshimura, Y. Fujimi, K. Izui, and S. Nishiwaki. Decision–making support system for human resource allocation in product development projects. *International Journal of Production Research*, 44(5):831–848, Mar. 2006.

[80] G. Zhu, J. F. Bard, and G. Yu. A branch–and–cut procedure for the multimode resource–constrained project–scheduling problem. *Informs Journal on Computing*, 18(3):377–390, Jan. 2006.

[81] P. H. Zipkin. *Foundations of Inventory Management*. McGraw-Hill, Boston, 2000.