Competence-Driven Project Portfolio Selection, Scheduling and Staff Assignment

Walter J. Gutjahr(1)[*], Stefan Katzensteiner(1), Peter Reiter(1), Christian Stummer(2), Michaela Denk(3)

(1) Department of Statistics and Decision Support Systems, University of Vienna, Universitaetsstr. 5/9, 1010 Vienna, Austria
(2) Department of Business Administration, University of Vienna, Bruenner Str. 72, 1210 Vienna, Austria
(3) E-Commerce Competence Center, Donau-City Str. 1, 1220 Vienna, Austria

Abstract: This paper presents a new model for project portfolio selection, paying specific attention to competence development. The model seeks to maximize a weighted average of economic gains from projects and strategic gains from the increment of desirable competencies. As a sub-problem, scheduling and staff assignment for a candidate set of selected projects must also be optimized. We provide a nonlinear mixed-integer program formulation for the overall problem, and then propose heuristic solution techniques composed of (i) a greedy heuristic for the scheduling and staff assignment part, and (ii) two (alternative) metaheuristics for the project selection part. The paper outlines experimental results on a real-world application provided by the E-Commerce Competence Center Austria and, for a slightly simplified instance, presents comparisons with the exact solution computed by CPLEX.

1 Introduction

Project portfolio planning (i. e., the selection, prioritization and scheduling of project proposals, as well as the proper staff assignment) is a challenging decision making problem in various management fields and more often than not is of high practical relevance, as it must ensure an effective and efficient use of substantial resources. Research and development (R&D) investment planning may serve as an illustrative application in which huge amounts of resources are at stake, as evidenced in the fact that in 2005 (the most recent year for which the relevant data is available) governments and industries in the EU-25 countries spent 1.77 % of the gross domestic product on R&D and, thus, employed researchers to an extent of more than 1.2 million of

[*]Corresponding author

full-time equivalents (FTEs), while even in a comparatively small country like Austria R&D activities totalled more than €5.8 billion and involved more than 28,000 researcher FTEs [19].

From a resource-based point of view, superior firm performance is linked to the resources and capabilities "possessed" by a particular firm [21]. Even though conceptualizing and/or measuring these capabilities is not straightforward (for a discussion cf. [7]), an in-depth analysis of employees' competences and their development is inevitable because they form a key source for competitive advantage in enterprises [11]. This holds particularly true for industrial branches facing so-called "hypercompetition" [4], which denotes a competitive situation where the key success factor is the ability to constantly develop new products, processes or services providing the customer with increased functionality and performance. Institutions relying on the competencies of their employees therefore are not first and foremost concerned with the money to be distributed amongst a set of (R&D) project opportunities, but rather with the allocation of human capital. The reason for this lies in the fact that for these cases, available expertise (i. e., competence) mainly decides whether a research or innovation endeavor may turn into something like a success or if it is doomed to fail because of a (not appropriately anticipated) lack of critical intellectual capabilities of some sort. However, due to its dual nature, human capital is both a resource indispensable for conducting research and innovation as well as the eventual result of these activities.

From an economic modeling point of view, allocating available resources amongst a set of project opportunities poses a decision making problem of intriguing complexity. The question to be answered involves addressing how the goals of generating (innovation) value and strengthening "innovation capacity" can best be accomplished. In a sense, these goals are partially conflicting, because output orientation stresses the shorter term, whereas capacity building focuses on assuring the longer-term existence of the institution and thus reflects strategic objectives rather than immediate performance under varying environmental demands. Hence, a need to steer allocation procedures arises, as the complexity of the decision problem already defies "intuitive" decision making for moderately-sized project portfolios.

While portfolio selection already constitutes a challenging problem, the problem at hand comes with two additional aspects that add considerably to its complexity, namely, (i) the need to develop schedules for the selected projects with (financial, as well as competence-related) resource constraints, and (ii) the concurrent staff assignment that particularly influences the scheduling in the short run and the competence development in the long run. As even "simple" scheduling problems are usually addressed by means of heuristic procedures (cf. [14, 20] for general applications and [3, 13, 23] for R&D-related ones), it is obvious that (meta-) heuristic procedures ought also to be used when combining portfolio selection, project scheduling and staff assignment.

The remainder of this paper therefore proceeds as follows: Section 2 provides the formulation of the mathematical model, as well as a discussion of an asymptotic approximation and two special cases. In section 3, we outline a heuristic procedure for the scheduling and the staff assignment and provide a detailed description of both an Ant Colony Optimization procedure and a Genetic Algorithm procedure for the portfolio selection problem. Section 4 is dedicated to our case study and also provides results from numerical experiments. Finally, the paper concludes in section 5 with a summary and an outlook on further research.

2 The Model

2.1 Mathematical Formulation

In the following, the *Project Selection, Scheduling and Staffing with Learning* problem ("PSSSL problem") will be described in formal terms.

First of all, let us consider a set of *candidate projects*, indexed by $i = 1, \ldots, n$, from which a subset has to be selected, a so-called *project portfolio*. We represent the decision which candidate project is selected by decision variables y_i $(i = 1, \ldots, n)$, where $y_i = 1$ if project i is to be included in the portfolio, and $y_i = 0$ otherwise.

The planning horizon is given by a time interval consisting of T *periods*, indexed by $t = 1, \ldots, T$. (In our case study, the length of a period is one month.) Period t starts at time $t - 1$ and ends at time t. The decision on project selection, scheduling and staff assignment has to be made at time $t = 0$, which is the start time of period 1, and is conceived as the solution of a static optimization problem; of course, this does not exclude that later re-scheduling in a rolling-horizon fashion might be undertaken. However, the last aspect is not subject of this investigation.

To each project i, a real number $w_i > 0$ denoting the (economic) *benefit* the company draws from project i is assigned $(i = 1, \ldots, n)$. The value w_i can refer to profit, turnover, market share or to any other economic measure of gain, or to a combination of several measures.

Each project consists of one or several *tasks*, indexed by $k = 1, \ldots, K$. The assignment of tasks to projects is given by binary constants c_{ik}, where, for each project i and task k, the value of c_{ik} is 1 if project i contains task k, and 0 otherwise. It is always supposed that $\sum_{i=1}^{n} c_{ik} = 1$ for all k, i.e., that each task belongs to exactly one project. Moreover, for each task k, the following information is given: (i) its *ready time* $\rho_k \in \{1, \ldots, T\}$, and (ii) its *due date* $\delta_k \in \{1, \ldots, T\}$. Ready time and due date refer to periods with periods ρ_k and δ_k being the first and the last period, respectively, where work in task k is possible. (In other words: Work on task k can begin at time point $\rho_k - 1$ and must end at time point δ_k.)

Furthermore, we consider a set of *employees*, indexed by $j = 1, \ldots, m$, which form the staff, which is assumed to be fixed during the entire planning horizon. We neither take new hires nor employment terminations into account. Moreover, the outsourcing of work is not taken into consideration as well.

Employees are assumed to possess different knowledge, education, skills, abilities, etc. in different fields. We refer to these fields by the term *competencies* and index competencies by $r = 1, \ldots, R$. In classical terms of the project scheduling literature, competencies can also be conceived as (human) *resources*. Not all competencies are of the same value for the company; it is assumed that based on long-term strategic considerations, the management can assign a weight v_r to each competency r that quantifies the relative importance of competency r in comparison to the other competencies. We further assume that the weights v_r are scaled in a specific way (to be explained below) in relation to the economic benefits w_i.

A basic assumption of our approach is that the degree to which an employee j possesses a certain competency r can be quantified in the form of a real (not necessarily positive) value. We call this value the *competence score* and denote it by z_{jrt}. The third index t indicates that the competence score of employee j in competency r can evolve over time: by learning effects, the value z_{jrt} increases if employee j works in a task requiring competency r; on the other hand, the so-called knowledge depreciation effect reduces z_{jrt} in periods where employee j is not active in competency r. Initial values z_{jr1} of the competency scores in period 1 are assumed as known. Based on qualification information, on tests, on subjective estimates or on a combination of these information sources, competence scores can be measured by established methods of labor psychology in a way respecting classical quality criteria, such as validity or reliability.

From the competency score z_{jrt}, we derive an *efficiency* value γ_{jrt} of employee j in competency r during period t by applying some (in general nonlinear) monotonous transformation function φ. The function φ maps the set of reals into the interval $[0, 1]$. By the efficiency of employee j in competency r, one understands the share of work performed in one time unit by employee r on a task requiring only competency r, if the entire task takes one time unit for an employee with "perfect skills" in competency r (cf., e.g., [24]). The specification of an appropriate transformation function φ is an empirical problem. It is convenient to restrict the consideration to a parameterized class of functions of the desired type and to estimate the parameters from empirical data. The class of *logistic functions*, which has been frequently used for modeling organizational learning (see, e.g., [2, 18]), represents a promising choice for this purpose. If this class is used, the function φ is given by

$$\varphi(z) = \frac{1}{1 + a\ \exp(-b\, z)} \tag{1}$$

with real parameters $a > 0$ and $b > 0$.

We assume that task k requires an overall *ideal effort* of d_{kr} in competency r ($k = 1, \ldots, K$; $r = 1, \ldots, R$). The ideal effort d_{kr} is the time needed by an employee with efficiency $\gamma_{jrt} \equiv 1$ for completing the part of the task related to competency r. As the unit for working times, we always take the overall maximum possible working time in one period. The real numbers d_{kr} are assumed to be both known and deterministic.

That part of a task k that requires a particular competency r will be called the *work package* with index (k, r). Thus, d_{kr} measures the effort needed for work package (k, r).

In period t, employee j has a free *capacity* of $a_{jt} \in [0, 1]$ $(j = 1, \ldots, m; t = 1, \ldots, T)$, expressed in working time units; this free capacity is also assumed as known.

We allow the formulation of constraints of the type that for each period, the ideal effort invested in competency r of task k must not exceed a value b_{kr} $(k = 1, \ldots, K;\ r = 1, \ldots R)$. For example, if it is required to distribute the workload of task k related to competency r equally over the time window between period ρ_k and period δ_k, this can be enforced by setting $b_{kr} = d_{kr}/(\delta_k - \rho_k + 1)$.

In addition to the binary decision variables y_i that describe the project portfolio selection decision, we also require a second set of decision variables specifying the decision both on (i) the scheduling of the selected projects over time with respect to their required efforts, ready times and due dates, as well as on (ii) the assignment of staff to the tasks of the selected projects with special attention paid to the required competencies. The decisions of types (i) and (ii) are captured simultaneously by real decision variables $x_{kjrt} \in [0, 1]$, where x_{kjrt} denotes the time employee j works within period t in competence r of task k $(k = 1, \ldots, K;\ j = 1, \ldots, m;\ r = 1, \ldots R;\ t = 1, \ldots, T)$. As in the case of efforts and capacities, time is again measured in multiples of the overall maximum possible working time in one period.

We assume that the competency score of an employee j in competency r increases in each period where employee j has worked during an amount x of time in competency r by an increment of size $\eta_r \cdot x$, where the proportionality factor η_r is a constant that can depend on r. Similarly, we assume that the competency score of an employee j in competency r is reduced by the amount β_r in each time period by knowledge depreciation. (This loss can be over-compensated by the gain achieved by activity in competency r, as described above.) The parameters η_r and β_r will be called the *learning rate* and the *depreciation rate* of competency r, respectively. We always assume $\eta_r > \beta_r$.

Given the notation above, we are now in the position to formulate our optimization problem PSSSL as a nonlinear mixed-integer program in the following way. We allow t to take also the value $T + 1$ in order to be able to refer to the time point T (end of the planning horizon, i.e., beginning of period $T + 1$).

$$\sum_{i=1}^{n} w_i y_i + \sum_{r=1}^{R} v_r \sum_{j=1}^{m} (\gamma_{j,r,T+1} - \gamma_{jr1}) \to \max \tag{2}$$

$$\gamma_{jrt} = \varphi(z_{jrt}) \quad \forall j, r, t \tag{3}$$

$$z_{jrt} = z_{jr1} - \beta_r\,(t-1) + \eta_r \sum_{k=1}^{K}\sum_{s=1}^{t-1} x_{kjrs} \quad \forall j, r, t \tag{4}$$

$$\sum_{k=1}^{K}\sum_{r=1}^{R} x_{kjrt} \le a_{jt} \quad \forall j, t \tag{5}$$

$$\sum_{t=\rho_k}^{\delta_k}\sum_{j=1}^{m} \gamma_{jrt}\, x_{kjrt} = d_{kr} \sum_{i=1}^{n} c_{ik}\, y_i \quad \forall k, r \tag{6}$$

$$\sum_{j=1}^{m} \gamma_{jrt}\, x_{kjrt} \le b_{kr} \quad \forall k, r, t \tag{7}$$

$$x_{kjrt} = 0 \text{ if } (t < \rho_k \text{ or } t > \delta_k) \quad \forall k, j, r, t \tag{8}$$

$$x_{kjrt} \ge 0 \quad \forall k, j, r, t \tag{9}$$

$$y_i \in \{0, 1\} \quad \forall i \tag{10}$$

The objective function (2) is a weighted average of (i) the economic benefits w_i gained from the completion of the selected projects i, and (ii) the strategic benefits accrued from the increments of the efficiency values γ_{jrt}, aggregated over all employees j, over the planning horizon. The numbers v_r are used as weights for the strategic importance of the competencies. Note that in order to make the entire objective function meaningful, the values v_r have to be scaled in such a way that their relative size compared to the economic benefits w_i is appropriate. In practice, this usually implies that the gains v_r from competence development must also be expressed in monetary units.

Constraints (3) specify the dependence of the efficiency values on the competence scores. Constraints (4) describe the evolution of the competence scores by knowledge depreciation and by learning. Constraints (5) bound the invested working times of each employee by her or his capacity limits. Constraints (6) ensure that the real working time of each employee in a competency r within a given task k, multiplied by her or his efficiency (which gives the ideal working time), and cumulated over all employees and over the runtime of the task, must sum up to the overall required ideal effort d_{kr} for task k in competency r, if the project to which task k belongs is selected in the portfolio, and to zero otherwise. (Note that $\sum_{i=1}^{n} c_{ik} y_i$ is equal to $y_{i(k)}$, where $i(k)$ is the index of the project to which task k belongs.) Constraints (7) bound the ideal working time in each competency of a given task by the maximum allowed amount per period. Constraints (8) require that no work is carried out on this task before the ready time or after a task's due date. Constraints (9) are non-negativity constraints for the real-valued decision variables x_{kjrt}, and, finally, constraints (10) require that the decision variables y_i for the portfolio selection are binary.

We observe that even in the special case where the function φ is linear, the PSSSL problem is a nonlinear problem, since the variables γ_{jrt}, which depend on the decision variables x_{kjrt} by equations (3) and (4), are multiplied with the variables x_{kjrt} in equation (6). For linear φ, the problem becomes obviously a quadratic mixed-integer program.

2.2 Asymptotic Approximation and Special Cases

In this subsection, we deal with an asymptotic approximation obtained by assuming small learning and depreciation rates, as well as with the situation of a (piecewise) linear transformation function φ. Both special contexts allow the reduction of the nonlinear PSSSL problem to either a linear or at least to a quadratic mixed-integer problem.

2.2.1 Asymptotic Approximation

The assumption of small learning rates η_r and small depreciation rates β_r can be represented mathematically by setting

$$\eta_r = \bar{\eta}_r \cdot \epsilon \text{ and } \beta_r = \bar{\beta}_r \cdot \epsilon, \tag{11}$$

where $\bar{\eta}_r$ and $\bar{\beta}_r$ are constants, and $\epsilon \ll 1$. Letting ϵ become small without changing the weights v_r automatically reduces the importance of the second, "strategic" term in the objective function (2), such that in the limit $\epsilon \to 0$, this term does not play a role anymore. In R&D projects under competitive circumstances, this is usually not the situation of practical interest: here, even comparably small increments of the competencies of the personnel may have eminent positive consequences, as they may be critical for the question whether it is possible to enter into innovative business fields. For this reason, we compensate for the decreasing importance of the competency gain as $\epsilon \to 0$ by simultaneously increasing the weights v_r, i.e., we set

$$v_r = \bar{v}_r / \epsilon. \tag{12}$$

Combining (3) and (4) and inserting (11) yields

$$\gamma_{jrt} = \gamma_{jrt}(\epsilon) = \varphi\left(z_{jr1} - \bar{\beta}_r \epsilon (t-1) + \bar{\eta}_r \epsilon \sum_{k=1}^{K} \sum_{s=1}^{t-1} x_{kjrs}\right) = \varphi\left(z_{jr1} + \epsilon h_{jrt}\right)$$

with

$$h_{jrt} = -\bar{\beta}_r (t-1) + \bar{\eta}_r \sum_{k=1}^{K} \sum_{s=1}^{t-1} x_{kjrs}.$$

By Taylor expansion at $\epsilon = 0$, we get

$$\gamma_{jrt}(\epsilon) = \varphi(z_{jr1}) + h_{jrt} \cdot \varphi'(z_{jr1}) \cdot \epsilon + \frac{h_{jrt}^2}{2} \cdot \varphi''(z_{jr1}) \cdot \epsilon^2 + O(\epsilon^3). \tag{13}$$

In a first-order approximation, we neglect already terms of order $O(\epsilon^2)$, such that

$$\gamma_{jrt}(\epsilon) \sim \varphi(z_{jr1}) + h_{jrt} \cdot \varphi'(z_{jr1}) \cdot \epsilon \tag{14}$$

with the consequence that the objective function (2) becomes

$$\sum_{i=1}^{n} w_i y_i + \sum_{r=1}^{R} \frac{\bar{v}_r}{\epsilon} \sum_{j=1}^{m} (h_{j,r,T+1} - h_{jr1}) \cdot \varphi'(z_{jr1}) \cdot \epsilon$$

$$= \sum_{i=1}^{n} w_i y_i + \sum_{r=1}^{R} \bar{v}_r \sum_{j=1}^{m} \varphi'(z_{jr1}) \cdot \left\{ -\bar{\beta}_r T + \bar{\eta}_r \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \right\}.$$

Because $-\bar{\beta}_r T$ is a constant, instead of maximizing the expression above, solving

$$\sum_{i=1}^{n} w_i y_i + \sum_{r=1}^{R} \bar{v}_r \bar{\eta}_r \sum_{j=1}^{m} \varphi'(z_{jr1}) \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \to \max \tag{15}$$

gives the same result. This objective function is linear in the decision variables x_{kjrt} (and of course also in the decision variables y_i).

Now let us consider the constraints. Apart from (3) which we have already substituted, only constraints (6) – (7) contain the efficiencies γ_{jrt}. Applying (14), we see that a first order-approximation for $\sum_{j=1}^{m} \gamma_{jrt}\, x_{kjrt}$ is given by

$$\sum_{j=1}^{m} \varphi(z_{jr1})\, x_{kjrt} = \sum_{j=1}^{m} \gamma_{jr1}\, x_{kjrt}.$$

Hence, also the (approximated) constraints are linear in the decision variables x_{kjrt}.

It is interesting to look at the special case of the linear transformation function

$$\varphi(z) = \begin{cases} 0, & z < 0 \\ z, & 0 \le z \le 1, \\ 1, & z > 0. \end{cases}$$

Observe that if $0 < z_{jr1} < 1$ for all j, r, and if ϵ is sufficiently small such that all competence scores z_{jrt} remain within the open interval $]0,1[$ during all periods, application of the identity function $id(z) = z$ yields the same result as applying $\varphi(z)$. Since $id'(z) \equiv 1$, (15) then reduces to

$$\sum_{i=1}^{n} w_i y_i + \sum_{r=1}^{R} \tilde{v}_r \sum_{j=1}^{m} \sum_{k=1}^{K} \sum_{s=1}^{T} x_{kjrs} \to \max \tag{16}$$

with $\tilde{v}_r = \bar{v}_r \bar{\eta}_r$. This function is a weighted average of economic benefits and the overall amounts of work invested within competencies $r = 1, \ldots, R$.

If we include the $O(\epsilon^2)$ term in the Taylor expansion (13) and only neglect terms of order $O(\epsilon^3)$, we get a *second-order* approximation for our problem. In this case, the approximation of the objective function becomes quadratic in the decision variables x_{kjrt}, since the expressions h_{jrt} depending linearly on the x_{kjrt} then also occur in squared form. In addition, the constraints (6) – (7) become quadratic in the variables x_{kjrt} in this refined approximation: the efficiencies γ_{jrt} must be approximated here by (14), which leads to a multiplication of the expressions h_{jrt} by the variables x_{kjrt}.

2.2.2 A Special Transformation Function

It is also possible to obtain a *quadratic* mixed-integer program from (2) – (10) as a *special case* without performing an asymptotic approximation. For this purpose, we start with the observation that for $t = 1, \ldots, T+1$, lower and upper bounds z_{min} and z_{max}, respectively, for the competence scores z_{jrt} can be derived. First, note that

$$z_{jrt} \geq z_{jr1} - \beta_r(t-1) \geq z_{jr1} - \beta_r T.$$

Furthermore, because of $\sum_{k=1}^{K} x_{kjrs} \leq a_{jt} \leq 1$,

$$z_{jrt} = z_{jr1} + \sum_{s=1}^{t-1} \left(-\beta_r + \eta_r \sum_{k=1}^{K} x_{kjrs} \right)$$

$$\leq \; z_{jr1} + \sum_{s=1}^{t-1} (\eta_r - \beta_r) = z_{jr1} + (\eta_r - \beta_r)(t-1) \leq z_{jr1} + (\eta_r - \beta_r)T.$$

Therefore,

$$z_{min} = \min_{j,r} z_{jr1} - T \max_r \beta_r$$

and

$$z_{max} = \max_{j,r} z_{jr1} + T \max_r (\eta_r - \beta_r)$$

yield the desired bounds. z_{min} can be positive, zero or negative. With the exception of the case in which all initial values z_{jr1} are smaller or equal to zero, z_{max} is always positive, because of $\beta_r < \eta_r$ for all r.

Now, let us define the piecewise linear transformation function

$$\varphi(z) = \begin{cases} 0, & z < z_{min} \\ \frac{z - z_{min}}{z_{max} - z_{min}}, & z_{min} \leq z \leq z_{max}, \\ 1, & z > z_{max}. \end{cases}$$

Since the competence scores z_{jrt} never leave the interval $[z_{min}, z_{max}]$, the behavior of the process is the same as if we would apply the linear function $\bar{\varphi}(z) = (z - z_{min})/(z_{max} - z_{min})$ instead of $\varphi(z)$. In this case, it is easy to see that the objective function becomes linear and the constraints become quadratic in the variables x_{kjrt}, since the efficiencies γ_{jrt} now depend linearly on the x_{kjrt}.

2.3 Additional Constraints

It may prove necessary to introduce additional constraints to adapt the model to reality even better. We outline five types of such constraints and show how they can be expressed in the model.

2.3.1 Maximum Number of Employees per Task

For each task k, a maximum number σ_k of employees to be engaged in this task may be defined. The purpose of such a constraint is to avoid the team being assigned a task is too large, in which case the work might be paralyzed by communication overhead. (In software project planning, the counter-productive effect of increasing team size in order to meet tight due dates is known under the term "Brooks' Law", a term referring to the insights in [1].)

The constraint can be formulated by the introduction of additional variables ξ_{kj}, where ξ_{kj} is used as an indicator variable for the event that employee j works on task k. The constraint on the maximum number of employees per task can then be expressed as

$$\sum_{j=1}^{m} \xi_{kj} \le \sigma_k \quad \forall k.$$

To serve their purpose, the variables ξ_{kj} have to satisfy the constraints

$$\sum_{r=1}^{R} \sum_{t=1}^{T} x_{kjrt} \le M\, \xi_{kj} \quad \forall k, j$$

and

$$\xi_{kj} \in \{0, 1\} \quad \forall k, j,$$

where M is a large number.

2.3.2 "Expert" Constraint

Each team assigned to a competency r of a task k can be required to contain an employee who contributes an ideal amount of work of a certain minimum size α_{kr} to competency r of task k. The purpose of this rule is to avoid that a required competency is covered numerically by cumulating small contributions from a large number of different employees with comparably small efficiency. Although the required level of ideal work might be reached mathematically by such an approach, the team would presumably fail unless it contains at least one "expert" guiding the members with low competency scores. We identify an "expert" by a sufficiently large *ideal* work contribution, where the lower bound α_{kr} on the work contribution can be specified for each task and each competency separately. The constraint can be expressed by the introduction of additional variables ζ_{kjr}, where ζ_{kjr} is an indicator variable for the event that employee j serves as an expert for competency r of task k in the sense defined above. The constraint is then the following:

$$\sum_{j=1}^{m} \zeta_{kjr} = 1 \quad \forall k, r$$

where the variables ζ_{kjr} have to satisfy

$$-\sum_{t=1}^{T} \gamma_{jrt}\, x_{kjrt} + \alpha_{kr} \leq M\,(1 - \zeta_{kjr}) \quad \forall k, j, r \tag{17}$$

and

$$\zeta_{kjr} \in \{0, 1\} \quad \forall k, j, r,$$

where M is a large number. Condition (17) expresses that the variable ζ_{kjr} can only take the value 1 if $\sum_t \gamma_{jrt}\, x_{kjrt} \geq \alpha_{kj}$.

2.3.3 Minimum and Maximum Number of Selected Projects from Project Sets

One can require that a minimum number $\underline{n}_\ell$ and a maximum number $\bar{n}_\ell$ of projects must be selected for a subset U_ℓ of the project set $\{1, \ldots, n\}$. The special cases $\underline{n}_\ell = 0$ and $\bar{n}_\ell = n$ represent the cases in which only a maximum or only a minimum number, respectively, is defined. Let $\{U_\ell \,|\, \ell = 1, \ldots, L\}$ be the family of all sets for which constraints of this type are given. The sets U_ℓ are allowed to overlap. In that event, the constraints can be defined as follows:

$$\underline{n}_\ell \leq \sum_{i \in U_\ell} y_i \leq \bar{n}_\ell \quad \forall \ell.$$

2.3.4 Precedence Relations Between Tasks

Sometimes, precedence relations between different tasks of a project are given. For treating such relations, we introduce $2 \cdot K \cdot T$ auxiliary variables ψ_{kt} and ψ'_{kt} ($k = 1, \ldots, K$; $t = 1, \ldots, T$) and subject them to the linear constraints

$$\begin{aligned}
\psi_{kt} \;&\leq\; M \sum_{s=1}^{t} \sum_{j=1}^{m} \sum_{r=1}^{R} x_{kjrs} \quad \forall k, t, \\
M\psi_{kt} \;&\geq\; \sum_{s=1}^{t} \sum_{j=1}^{m} \sum_{r=1}^{R} x_{kjrs} \quad \forall k, t, \\
&\psi_{kt} \in \{0, 1\} \quad \forall k, t, \\
\psi'_{kt} \;&\leq\; M \sum_{s=t}^{T} \sum_{j=1}^{m} \sum_{r=1}^{R} x_{kjrs} \quad \forall k, t,
\end{aligned}$$

$$M\psi'_{kt} \;\geq\; \sum_{s=t}^{T}\sum_{j=1}^{m}\sum_{r=1}^{R} x_{kjrs} \quad \forall k,t,$$

$$\psi'_{kt} \in \{0,1\} \quad \forall k,t,$$

where M is a large number. It is easily seen that by these constraints, ψ_{kt} becomes the indicator variable for the event that task k has already been started in period t or before, and ψ'_{kt} becomes the indicator variable for the event that task k is not yet terminated at the beginning of period t. Now, for each precedence relation $k_1 \prec k_2$ between two tasks k_1 and k_2, we add the linear constraints

$$\psi_{k_2t} \leq 1 - \psi'_{k_1t} \quad \forall t,$$

which ensure that there is no period t where task k_2 is already started, although task k_1 is not yet terminated.

2.3.5 Avoiding Project Interruption

Basically, our scheduling model is *preemptive*, i.e., we allow that work in a project is interrupted by work in another project and reassumed later, provided that the given ready times and due dates are not violated. In some cases, the management might wish to ensure that there is a continuous stream of work in a project once it has been started that does not end before the project is terminated. This can be modelled by defining, for each project i, a lower bond h_i for the (real) work time invested into project i in each period between its start and its termination. To handle these conditions, we introduce $2 \cdot n \cdot T$ auxiliary variables χ_{it} and χ'_{it} similar to the variables ψ_{kt} and ψ'_{kt} in subsection 2.3.4, but now referring to projects instead of tasks, and subject them to the linear constraints

$$\chi_{it} \;\leq\; M\sum_{s=1}^{t}\sum_{k=1}^{K}\sum_{j=1}^{m}\sum_{r=1}^{R} c_{ik}\, x_{kjrs} \quad \forall i,t,$$

$$M\chi_{it} \;\geq\; \sum_{s=1}^{t}\sum_{k=1}^{K}\sum_{j=1}^{m}\sum_{r=1}^{R} c_{ik}\, x_{kjrs} \quad \forall i,t,$$

$$\chi_{it} \in \{0,1\} \quad \forall i,t,$$

$$\chi'_{it} \;\leq\; M\sum_{s=t}^{T}\sum_{k=1}^{K}\sum_{j=1}^{m}\sum_{r=1}^{R} c_{ik}\, x_{kjrs} \quad \forall i,t,$$

$$M\chi'_{it} \;\geq\; \sum_{s=t}^{T}\sum_{k=1}^{K}\sum_{j=1}^{m}\sum_{r=1}^{R} c_{ik}\, x_{kjrs} \quad \forall i,t,$$

$$\chi'_{it} \in \{0,1\} \quad \forall i,t,$$

where M is a large number again. Thus, χ_{it} and χ'_{it} become the indicator variables for the event that project i has already been started in period t, resp. that it is not yet terminated at the beginning of period t. The additional linear constraints

$$\sum_{k=1}^{K}\sum_{j=1}^{m}\sum_{r=1}^{R} c_{ik}x_{kjrt} \;\geq\; h_i - M(1-\chi_{it}) - M(1-\chi'_{it}) \quad \forall i,t$$

enforce then the desired minimum work times per period for each project i.

3 Heuristic Solution Algorithms

In this section, we describe our approach to solving the PSSSL problem heuristically for those cases in which an exact solution by means of an ILP solver is no longer possible, either on account of nonlinearity or because of an excessively large number of integer variables. The overall heuristic solution approach relies on a greedy algorithm for solving the scheduling-and staff-assignment part of the problem (lower decision level). This procedure is described in subsection 3.1. It is called repeatedly as a subroutine by a master procedure optimizing the portfolio decision. For the master procedure, we implemented two metaheuristic solution approaches, one relying on the Ant Colony Optimization (ACO) paradigm, the other applying a Genetic Algorithm (GA). These two procedures are outlined in subsections 3.2 and 3.3, respectively.

3.1 Heuristic Scheduling and Staff Assignment

The scheduling-and-staff-assignment procedure (SSAP) takes the problem instance and a special project portfolio y as input and attempts to compute a feasible scheduling-and-staff-assignment plan, described by the array $x = (x_{kjrt})$, to the given portfolio y. The computation of such a plan can fail, either because the portfolio y under consideration is infeasible, or because the (only heuristic) greedy procedure does not recognize that a feasible solution exists. In this event, the SSAP returns the result "failure" to the master procedure, which causes the latter to search for an alternative portfolio y.

For the description of SSAP, we use the following additional notation: The set $S \subseteq \{1,\dots,n\}$ consists of the candidate projects i currently under consideration, i.e., those for which $y_i = 1$. The ready time ρ'_i of project i is equal to the earliest ready time ρ_k of all tasks k associated with it, i.e., $\rho'_i = \min\{\rho_k \,|\, c_{ik} = 1,\, 1 \leq k \leq K\}$. The due date δ'_i of project i is equal to the latest due date δ_k of all tasks k associated with it, i.e., $\delta'_i = \max\{\delta_k \,|\, c_{ik} = 1,\, 1 \leq k \leq K\}$. The total required effort d'_{ir} of a project i in competency r is the sum of the required efforts d_{kr} of all tasks k associated with it, i.e., $d'_{ir} = \sum_{k=1}^{K} c_{ik}\, d_{kr}$.

SSAP works *priority-based* (cf. [17] for an example) in five nested loops. First, the projects i are sorted according to their due dates, with projects with earlier due date getting a higher priority of being scheduled and being assigned staff (i.e., we follow an "earliest-due-date rule"). Second, for each project, we sort the competencies it requires on the basis of the efforts needed and give competencies with higher needed effort a higher priority. Third, for each competency, employees are sorted according to their efficiency in the required competency; employees with higher efficiency are assigned first. Fourth, the tasks contained in the project are sorted according to their due dates, with an earlier due date leading to higher priority. The innermost loop goes over the periods of the time window for the respective task. The pseudo-code for the overall procedure is shown in Figure 1.

```
Procedure Scheduling-and-Staff-Assignment
for all projects $i \in S$ in ascending order of $\delta'_i$ {
  for all competencies $r$ in descending order of $d'_{ir}$ {
    for all employees $j$ in descending order of $\gamma_{jr\rho'_i}$
      for all tasks $k$ with $c_{ik} = 1$ in ascending order of $\delta_k$ {
        for period $t = \rho_k$ to $\delta_k$ {
          assign to employee $j$ a maximum share of the remaining work
            in work package $(k, r)$, respecting the current free capacity
            of employee $j$ and the bound $b_{kr}$;
          given $x$ additional time units have been assigned to employee $j$
            during period $t$ in work package $(k, r)$, reduce the remaining
            ideal effort for work package $(k, r)$ by $\gamma_{jrt}\, x$;
  } } } }
  if (needed effort for some work package in project $i$ not fully covered)
    return("infeasible");
}
```

Figure 1: Greedy procedure for scheduling and staff assignment.

3.2 Portfolio Selection by Ant Colony Optimization

The first approach we apply on the upper decision level of portfolio selection is based on Ant Colony Optimization (see [5]). ACO is a population-based metaheuristic technique combining stochastic search with a learning mechanism. There are several variants of ACO; we applied the MAX-MIN Ant System [22]. The main ideas will be outlined for the special situation of a search space $S = \{0, 1\}^n$, as it occurs in the portfolio selection part of the PSSSL problem. Solutions are encoded as walks in a so-called *construction graph* (CG); for the problem at hand, we took a very simple CG, the *chain* graph introduced in [9]. An example for the case $n = 4$ is shown in Figure 2. A conceptual agent starts a random walk in node $\underline{0}$ of the CG and traverses directed arcs until no move is possible anymore. An up-move in the chain graph from node $\underline{i-1}$ to node i corresponds to a selection of item i (in our case: project i), a down-move from node $\underline{i-1}$ to node $-i$ corresponds to rejection of item i.

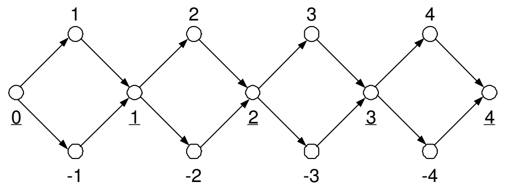

Figure 2: "Chain" construction graph for a subset selection problem with $n = 4$.

One iteration of the procedure consists of the mutually independent construction of N walks (by N agents); several iterations are performed.

To the arcs of the graph, so-called *pheromone values* are assigned which govern the learning process. As the last action in each iteration, the walks of all agents are de-coded as solutions y, and their objective function values are determined. We applied an iteration-best pheromone update mechanism (for details, see [5]). First, all pheromone values are multiplied by a factor $1 - \rho$, where $\rho \in]0, 1[$ is called the *evaporation rate*. Then, in the iteration-best update, pheromone values along the best walk constructed in the current iteration are increased by a value proportional to the fitness of this walk.

The effect of an increased pheromone value on an arc is that this arc is given a higher probability of being chosen by the agents in the next iteration. We computed the transition probabilities as proportional to the pheromone values and did not use so-called *visibility values*, which are sometimes applied to influence the transition probabilities in a problem-specific way.

To avoid stagnation situations that can arise from the chosen pheromone update strategy, pheromone limits have been used, as proposed by the MAX-MIN Ant System [22].

A final remark concerns the feasibility of the obtained solutions. If one of the two possible moves in node $\underline{i-1}$ turns out as infeasible (in our case, this can happen when the choice of a specific additional project leads to a portfolio exceeding the capacities of the staff), the agent is simply forced to make the other move. In our tests, we restricted ourselves to cases where for the sets U_ℓ, only maximum numbers $\bar{n}_\ell$ of projects to be selected are defined, but no minimum numbers $\underline{n}_\ell > 0$ (cf. subsection 2.3.3), and we did not include the constraints presented in subsections 2.3.4 – 2.3.5. In this situation, the described simple procedure suffices to cope with the problem of infeasible solutions, since portfolios can always be made feasible by omitting projects. In the case of minimum numbers $\underline{n}_\ell$, more involved techniques such as repair mechanisms or penalty functions would have to be applied.

The ACO parameters were set to the following values: $N = n$ and $\rho = 0.02$.

3.3 Portfolio Selection by a Genetic Algorithm

As an alternative approach for the upper decision level of portfolio selection, we implemented a genetic algorithm GA in a very "classical" fashion, following the standard GA scheme presented in [15]. As GAs are very well-known, we can restrict the description to the presentation of some implementation details. The binary string structure of the portfolio selection part of the problem lends itself very well to the application of a GA. (This is in contrast to ACO, which exhibits its strengths usually rather in problems with routing or sequencing structure).

In generation 0, an initial population of N chromosomes is generated, with each chromosome y consisting of n bits that are chosen uniformly at random in the initialization phase. The fitness functions of all elements of the population are evaluated, where we set fitness equal to the objective function value to be maximized. Then, by a standard roulette-wheel selection procedure, a generation 1 is created, again consisting of N chromosomes. The genetic operators mutation and crossover, both with certain rates R_m and R_c, respectively, are applied to this generation. Mutation is implemented bit-wise by an independent random flip of each bit in each of the chromosomes with probability R_m. For crossover, we use a standard one-point crossover, which is applied to a fraction of R_c of the population; the two generated offsprings replace their parents in the new population. This procedure is repeated until a termination criterion is met.

As in the case of the application of ACO, we must take care that feasible portfolios y are obtained. Whereas in the ACO case, in the absence of minimum numbers $\underline{n}_\ell$, feasibility of the overall portfolio can be ensured directly by the construction mechanism outlined in the previous subsection, this is no longer true for the GA, where the crossover operator can easily lead to infeasible solutions. Several repair mechanisms have been proposed in the relevant literature to deal with the occasional infeasibility of solutions in knapsack-type problems. In [16], a greedy repair is reported to provide the best results. In our case, the complex constraints make it impossible to compute an analogue to the "weight" of an item in a knapsack problem; therefore, benefit/weight ratios, on which a greedy repair relies, are also not applicable. For this reason, we implement a simpler repair mechanism instead, removing randomly selected projects from the portfolio in the event of an infeasible portfolio and continuing to do so as long as feasibility is not yet achieved. Evidently, in the case of minimum numbers $\underline{n}_\ell$, this is not sufficient, but – as stated in the previous subsection – this case was excluded from tests and could in principle be dealt with by a more refined repair mechanism or by the application of the penalty function method.

The GA parameter were set to the following values: $N = 20$, $R_m = 0.01$, and $R_c = 0.9$.

4 Case Study

4.1 Test Data

We tested our approach in a real-world setting provided by the Electronic Commerce Competence Center (EC3) Austria. The EC3 is a public-private partnership institution that is funded by the Austrian Federal Ministry of Economic Affairs and the City of Vienna, as well as by twelve private enterprises (e.g., T-Mobile, SAP, Tiscover, Swarovski Crystal Online, etc.). By embedding innovation practices into a collaborative network consisting of both the three major universities in Vienna (i.e., the University of Vienna, the Vienna University of Technology and the Vienna University of Economics and Business Administration) and the company partners, EC3 strives to implement a fast and problem-tailored transfer of knowledge into its business partners' realm of production and value generation. To this end, some 15 FTEs of permanent research staff are assigned to four working groups dealing with (i) structuring and representation of information corpora, including methods of information access and information visualization, (ii) logical models, designs, and mechanisms of inter-operable Web-based systems, (iii) empirical business analyses using formal quantitative methodologies and modeling techniques, and (iv) the evaluation of business ideas and models, including empirical analysis of customer needs and further methods of market research.

As a foundation for the data collection process, a catalogue of $R = 80$ professional and methodological competencies [8] relevant at the EC3 and 56 competence indicators (or evidences, cf. [12]), including objective evidences in terms of formal qualifications and professional experience, as well as subjective evidences, viz. competence ratings by peers, the scientific director, and the researcher him-/herself, was devised. A score matrix was specified that provides the contribution of each objective evidence to each competence, essentially based on background information such as curricula or journal citation indices. Resorting to Dreyfus' skill acquisition model [6], the subjective evidences were measured on a six-item ordinal scale discerning no competence, novice, advanced beginner, competent performer, expert, and mentor.

A total number of $m = 28$ employees – including the heads of the research groups, the scientific director, and several freelancers, in addition to the 15 permanent researchers – were surveyed via e-mail to collect the objective and subjective evidences. The competence score z_{jrt} was then computed as the sum of the contributions of all objective evidences a researcher holds plus a score built from the subjective competence ratings as an adjusted, weighted average and was constrained to the interval $[0, 100]$. Learning and depreciation rates, η_r and β_r, were defined assuming that learning by experience is faster and more sustainable than depreciation. Bearing in mind the score contributions specified for objective evidences, the rationale behind the setting of the learning rate was that the score contribution of a master's degree should approximate the score contribution of three to four years of research experience in the same competence. The rate of oblivion (competence depreciation) was fixed at a rather 'optimistic' level. For the time being, no differences were made between the competencies, except for several methodological competencies that were supposed to grow and diminish more slowly.

The logistic function from equation (1) was chosen to transform the competence score z_{jrt} to an efficiency value γ_{jrt}. The parameters a and b were set based on the specification of the input value domain and the conception of a relatively high increase of efficiency for medial competence scores in contrast to relatively small gains for rather low and rather high competence scores. Thereby, "rather low" indicates a competence score below the score that is assigned to graduation (approximately 30 to 40, subject to the competency), i.e., competencies trained at university level are located at the lower bound of the range of competence scores with high gains in efficiency. On the other hand, experts with a long record of formal qualifications and/or research experience that have already reached a high level of efficiency do not gain much in efficiency any more. The actual parameter values of a and b were obtained via ad-hoc "educated guesses" satisfying these basic ideas and plausibility considerations. Although first experiments with varying parameters have already been carried out, a comprehensive sensitivity analysis has yet to be done.

Data on $n = 18$ potential projects with two projects composed of two tasks, the other projects of only one task (i.e., $K = 20$), was gathered from project plans and assumptions on the distribution of the scheduled efforts with respect to the competence catalogue. Nine different competencies were required per task and the time period between ready time ρ_k and due date δ_k was 12 months on average. The amount of third-party funding was provided as a measure of economic gain w_i. In order to enable a comparison of the decision support obtained from the results of the optimization problem and the decisions actually made, the data was collected ex-post for a previous research period of two years (i.e., $T = 24$). Projects actually carried out as well as project opportunities that had not been seized were included. Those projects for whom it had already been decided that they would be carried out at the beginning of the selected research period were not described, but rather used to estimate the disposable capacities a_{jt} of researchers. The parameters for the additional constraints, such as the expert rule, are usually not subject of project plans and had to be specified ad-hoc. Finally, the weights of relative importance of the competencies v_r were fixed in line with EC3's overall strategy.

4.2 Testing Scheme

Our goal in this work is not to give an extensive experimental evaluation of the implemented heuristic algorithms, but rather to illustrate some results obtained in our case study. Two different problem instances are used for the tests presented here: a real-life instance (18 candidate projects, 24 planning periods, 28 employees and 80 competencies), and a simplified instance (14 candidate projects, 24 planning periods, 28 employees and 40 competencies), with the latter constructed in order to provide comparisons with exact solutions. For the real-life problem instance, we were also interested in knowing how the behavior of the algorithms changes with the introduction of additional constraints, as described in section 2.3. Furthermore, we performed tests for two scenarios: in the first scenario, all competencies were given the same weight in the objective function. In the second scenario, a small subset of six competencies was selected and provided with nonnegative weights, whereas the weights of the remaining competencies were set to zero. This represents a case where the decision maker intends to pursue a rather focused strategic goal in competence development.

Each heuristic was allowed to consume a previously specified runtime budget, which was set to the value $RT_{CE} * 0.5/counter$, where RT_{CE} is the runtime required by complete enumeration over all project portfolios y (combined with the heuristic scheduling-and-staff-assignment procedure described in subsection 3.1), and $counter = 1, \ldots, 30$. For each instance, constraint configuration and runtime, 100 runs of each heuristic with different seeds for the random numbers were carried out, with the mean values over the 100 runs being used for the comparisons.

Table 1 provides an overview of the test cases.

	Problem Size	Addit. Constraints	Compet. Weights
Test case 1	small	no	equal
Test case 2	big	no	equal
Test case 3	big	yes	equal
Test case 4	small	no	unequal
Test case 5	big	no	unequal
Test case 6	big	yes	unequal

Table 1: Test case survey.

4.3 Equal Competence Weights

4.3.1 Simplified Instance

In the case of the simplified instance, it was possible to compute the *exact* solution of the problem (2) – (10) in its linear approximation given in subsection 2.2.1 by means of the MILP solver of CPLEX. (In addition the heuristic approaches were then provided with this linearization to make the results comparable.) Thus, the results of these tests make it possible to evaluate not only the performance of the metaheuristics ACO and GA applied to the portfolio decision problem, but also to gather information on the performance of the *overall* heuristic approach, including the heuristic scheduling-and-staff-assignment procedure (SSAP). It should be kept in mind that even applying complete enumeration (CE) to the project portfolios while scheduling each single portfolio by means of SSAP will usually not produce the exact solution. The result is depicted in Figure 3. For convenience, the solution quality values achieved by the MILP solver and by CE have been represented by horizontal bars starting already at time 0; note, however, that by construction (see subsection 4.2), the computation time required even for CE exceeds the time scale of the figure: The MILP solver and the CE approach required about one hour and about 34 seconds of computation time, respectively.

As Figure 3 shows, there is a comparably large gap between the solution quality of the exact optimum (denoted by "MILP solution", since it has been determined by means of the MILP solver) and that of the solution delivered by CE plus SSAP, whereas the further gap between CE plus SSAP on the one hand, ACO plus SSAP or GA plus SSAP on the other hand is distinctly smaller.

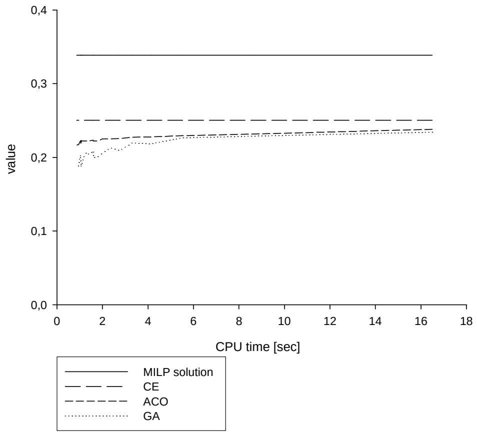

Figure 3: Solution quality development test case 1

Figure 3 demonstrates that for the simplified instance, ACO shows a slightly better performance than GA, and that the CE solution quality lies about 25 % below the optimum, i.e., about one quarter of the theoretically possible objective function value is given away by the (only heuristic) SSAP procedure. The further loss by the gap between CE and ACO is only about 6 % for the (small) runtime being considered. This indicates that future investigations should focus on improving the SSAP for the scheduling-and-staffing part rather than on the metaheuristics for the portfolio optimization part.

Nevertheless, when judging the gap between CE and exact solution, it should also be noted that portfolio selection based on the SSAP tends to be conservative with respect to the number of selected portfolios, a bias

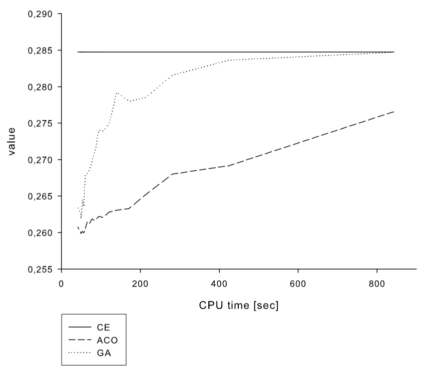

Figure 4: Solution quality development test case 2

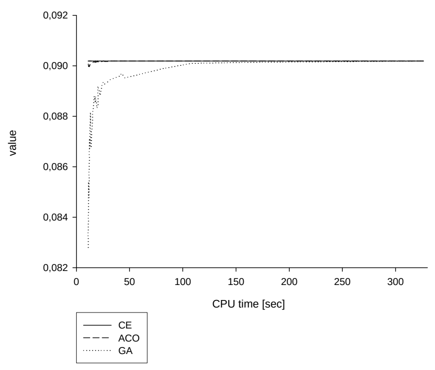

Figure 5: Solution quality development test case 3

that can be advantageous in a context of work times that are not precisely known in advance: the exact approach packs the projects very densely, exploiting every opportunity to schedule them. If there is uncertainty on actual work times, exact deterministic optimization can lead to schedules lacking robustness. The looser way of packing projects provided by the SSAP may deliver more realistic plans in this context.

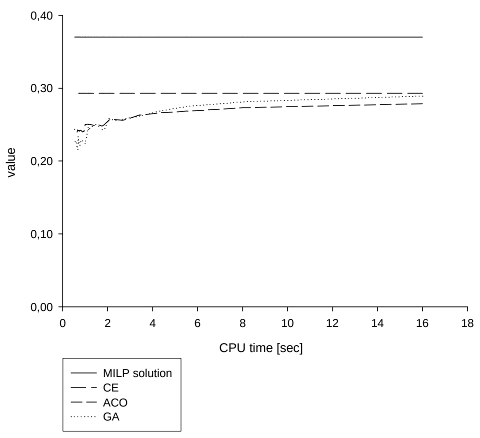

Figure 6: Solution quality development test case 4

4.3.2 Real-Life Instance

Figures 4 and 5 show the results for the original real-life test cases without and with additional constraints, respectively. The exact solution is no

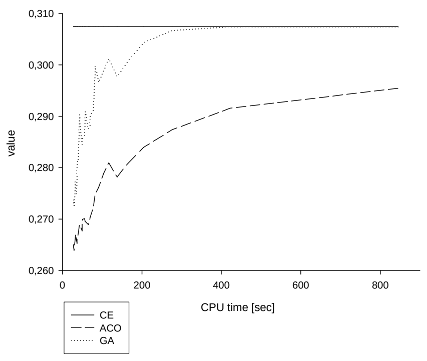

Figure 7: Solution quality development test case 5

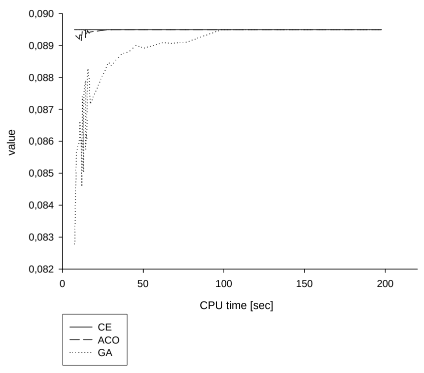

Figure 8: Solution quality development test case 6

longer known for these two instances, but the CE solution based on the SSAP results can still be determined. We see that if the solution space is not very restricted (Fig. 4), GA performs now much better than ACO and approaches the solution quality of CE within reasonable time. This seems to indicate that GA can best unfold its strengths for larger problem instances. In the situation where additional constraints delimit the solution space (Fig. 5), both approaches reach the solution quality of CE rather fast. In this instance, ACO clearly outperforms GA. An intuitive explanation may be that the implemented ACO algorithm uses an incremental solution construction procedure so that the generation of unfeasible solutions can already be avoided during the construction process. GA, on the other hand, constructs a complete solution and then uses a repair function if the constructed solution is not feasible. This may be very time-consuming in the presence of restrictive constraints.

4.4 Unequal Competence Weights

To test the robustness of the obtained results with respect to a broad resp. narrow choice of the strategic aim, we repeated the experiments with unequal competence weights as described before.

4.4.1 Simplified Instance

By comparing Figures 3 and 6, one can see that the general behavior of the algorithms basically remains the same. The main part of the gap to the MILP solution is not caused by the metaheuristics, but by the SSAP. One difference can be observed: in Figure 6, ACO is not consistently better than GA, but looses its dominant position when computation times are increased.

4.4.2 Real-Life Instance

Again, the trends in overall performance essentially stay the same as in test cases 2 and 3. However, the plots in Figures 7 and 8 show that now, for short computation times, the obtained solution quality value behaves in a rather erratic way. For larger computation times, the results stabilize, with GA once again outperforming ACO if there are no additional constraints, and vice versa for the opposite case.

4.5 Increasing the Number of Tasks per Project

In our real-world application instance, there were only two projects that consisted of more than one task, and also in these two projects, only two tasks occurred. In order to study the influence of the number of tasks per project, we performed some additional tests: First, we took a special test instance from the EC3 application (with a specific objective function emphasizing economic benefits) and joined, in each of the two projects with two tasks, the two tasks to a single one, which produced a baseline test instance T1 with 18 projects, where each project contained one task. Then, we gradually increased the number of tasks per project by splitting projects into tasks according to the following scheme:

- Test instance T2: Each project of T1 was split into two tasks. The time windows of the two tasks were shortened to 2/3 of the time window length of the corresponding project, such that they partially overlapped: The time window of the first task of each project was chosen to cover the first two thirds of the project's time window, the time window of the second task was chosen to cover the last two thirds.
- Test instance T3: Each project of T1 was split now into three tasks, with time windows shortened to the half of the time window lengths of the corresponding projects, and partially overlapping in an analogous manner as in T2.
- Test instance T4: Analogously as for T3, each project was now split into four tasks.
- Test instance T5: Starting from T4, we selected two projects, increased the numbers of tasks contained in them from 4 to 10, and shortened the time windows accordingly.

We performed 20 runs with the GA variant of our optimization program for each of these test instances. The results are shown in Table 2. In the second column, the mean objective function value (benefit) of the best found portfolio is indicated (averaged over the 20 runs). The third column contains the mean number of projects contained in the portfolio. As it can be seen, increasing the number of tasks per project from one to two tasks considerably worsened the optimal objective function value, and also reduced the number of projects contained in the portfolio. Obviously, this is a consequence of the loss of flexibility caused by the division of projects into tasks with specific competency requirements and time windows. Interestingly, for an increment from two to three tasks per project and then from three to four tasks, this trend turned out as much weaker. Further increasing the number of tasks for two projects from 4 to 10 did not change the picture anymore, although, as we verified, one of the projects with 10 tasks was eventually included in the portfolios.

test instance	mean benefit	mean number of projects
T1	218.8	7.6
T2	160.0	6.1
T3	157.4	6.0
T4	155.8	5.9
T5	156.2	5.9

Table 2: Results for instances with increasing number of tasks per project.

5 Conclusion

We have developed a model for project portfolio selection that pays attention to *competencies* which, on the one hand, act as resources for the efficient execution of projects, and, on the other hand, are increased as a result of individual learning processes during the projects that require them. The model is able to simultaneously consider both the economic benefits from projects and the achievement of strategic aims connected with competence development in desirable directions. The relative importance of economic and strategic aims can be controlled by formulating the overall objective function as a weighted mean. For the execution of the projects contained in a selected portfolio, the scheduling-and-staff-assignment problem has been taken explicitly into account. Further, work times, capacities and the competencies of employees are considered on an individual level. We have shown that the model allows a nonlinear mixed-integer programming (MIP) formulation that can be approximated by a linear MIP formulation in certain cases.

For solving the problem, we proposed two metaheuristic techniques, one based on Ant Colony Optimization (ACO), the other based on Genetic Algorithms (GA), combined with a problem-specific greedy heuristic which is called as a sub-procedure for doing the scheduling and staff assignment. We evaluated the approach by means of a real-world test application provided by the E-Commerce Competence Center Austria. For a reduced instance without additional constraints, comparisons with exact solutions obtained by CPLEX were possible. Here, the performance of the scheduling-and-staff-assignment procedure proved to be sufficiently good, and that of the metaheuristics used in the "master procedure" for portfolio optimization turned out to be very satisfactory. The real-life instance itself was solved by both metaheuristic algorithms within a few minutes; here, only a comparison with a complete enumeration approach also using the scheduling-and-staff-assignment procedure was possible to evaluate the results. In general, the GA approach seems to be slightly superior, except in those cases where the solutions space is highly constrained, in which case ACO yielded the better results.

Two directions deserve particular attention as topics for future research: first, the mathematics-based multiple objective programming approach chosen in this paper required the user to provide a-priori weights for the single (economic and strategic) goals. This approach should be extended to a multiple criteria decision analysis (MCDA) approach in which weights do not have to be defined in advance; instead, the Pareto front of the multi-objective problem could be determined and explored by an interactive technique. Second, it is highly advisable that future research include the consideration of *uncertainty* (e.g., on benefits and/or on work times) into the problem description by giving stochastic (multi-objective) optimization problem formulations, and that suitable techniques for solving such models be designed. Both extensions are works in progress; some first results concerning the second extension are already available [10].

Acknowledgment. Financial support from the Austrian Science Fund (FWF) by grant # L264-N13 is gratefully acknowledged. – We are indebted to Karl Fröschl for valuable ideas and fruitful discussions by which he contributed to this work already in an early stage. Furthermore, we would like to thank Markus Günther and Mariusz Malinowski for stimulating hints.

References

[1] Brooks, F., *The Mythical Man-Month*, Addison-Wesley, New York (1975).

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

[3] Coffin, M.A., Taylor, B.W., "Multiple criteria R&D project selection and scheduling using fuzzy logic", *Computers & Operations Research* 23, pp. 207–220 (1996).

[4] D'Aveni, R.A., *Hypercompetition: Managing the Dynamics of Strategic Maneuvering*, The Free Press, New York (1994).

[5] Dorigo, M., Stützle, T., *Ant Colony Optimization*, MIT Press, Cambridge, MA (2004).

[6] Dreyfus, H., Dreyfus, S., *Mind over Machine: the Power of Human Intuition and Expertise in the Era of the Computer*, The Free Press, New York (1986).

[7] Dutta, S., Narasimhan, O., Rajiv, S., "Conceptualizing and measuring capabilities: methodology and empirical application", *Strategic Management Journal* 26, pp. 277–285 (1999).

[8] Erpenbeck, J., Heyse, V., *Kompetenzbiographie – Kompetenzmilieu – Kompetenztransfer*, QUEM-Report 62, Berlin (1999). In German.

[9] Gutjahr, W.J., "On the finite-time dynamics of ant colony optimization", *Methodology and Computing in Applied Probability* 8, pp. 105–133 (2006).

[10] Gutjahr, W.J., Katzensteiner, S., Reiter, P., "A VNS algorithm for noisy problems and its application to project portfolio analysis", *Proc. SAGA 2007* (Stochastic Algorithms: Foundations and Applications), eds.: J. Hromkovic et al., Springer Lecture Notes in Computer Science 4665, pp. 93–104 (2007).

[11] Haesli, A., Boxall, P. "When knowledge management meets HR strategy: an exploration of personalization-retention and codification-recruitment configurations", *International Journal of Human Resource Management* 16, pp. 1955–1975 (2005).

[12] HR-XML Consortium, *Competencies (Measurable Characteristics) Recommendation 2006-02-28*, available at http://ns.hr-xml.org/2_4/HR-XML-2_4/CPO/Competencies.html, last visited: March 30, 2007.

[13] Kolisch, R., Meyer, K., Mohr, R., "Maximizing R&D portfolio value", *Research Technology Management* 48, pp. 33–39 (2005).

[14] Kolisch, R., Hartmann, S., "Experimental investigation of heuristics for resource-constrained project scheduling: an update", *European Journal of Operational Research* 174, pp. 23–37 (2006).

[15] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, London (1996).

[16] Michalewicz, Z., Arabas, J., "Genetic algorithms for the 0/1 knapsack problem", *Proc. 8th Int. Symposium on Methodologies for Intelligent Systems*, LNCS 869, Springer, Berlin, pp. 134–143 (1994).

[17] Möhring, R.H., Stork, F., "Linear preselective policies for stochastic project scheduling", Mathematical Methods of Operations Research 52, pp. 501–515 (2000).

[18] Ngwenyama, O., Guergachi, A., McLaren, T. "Using the learing curve to maximize IT productivity: A decision analysis model for timing software upgrades", *Int. J. Production Economics* 105, pp. 524–535 (2007).

[19] OECD, *Main Science and Technology Indicators 2006/2*, OECD, Paris (2006).

[20] Padman, R., Zhu, D., "Knowledge integration using problem spaces: a study in resource-constrained project scheduling", *Journal of Scheduling* 9, pp. 133–152 (2006).

[21] Peteraf, M.A., "The cornerstones of competitive advantage: a resource-based view", *Strategic Management Journal* 1, pp. 179–191 (2001).

[22] Stützle, T., Hoos, H.H., "MAX-MIN Ant System", *Future Generation Computer Systems* 16, pp. 889–914 (2000).

[23] Venkatraman, R., Venkatraman, S., "R&D project selection and scheduling for organizations facing product obsolescence", *R&D Management* 25, pp. 57–70 (1995).

[24] Wu, M.-C., Sun, S.-H., "A project scheduling and staff assignment model considering learning effect", *International Journal of Advanced Manufacturing Technology* 28, pp. 1190–1195 (2006).