Enriched Workflow Modelling and Stochastic Branch-and-Bound

Karl Doerner[(1)], Walter J. Gutjahr[(2)], Gabriele Kotsis[(3)],

Martin Polaschek[(4)], Christine Strauss[(1)*]

(1) Department of	(2) Department of	(3) Department of	(4) Department of
Business Studies	Statistics and DSS	Telecooperation	Business Information Systems
Univ. of Vienna	Univ. of Vienna	Univ. of Linz	Univ. of Vienna
Bruenner Str. 72	Universitaetsstr. 5	Altenberger Str. 69	Rathausstr. 19
A-1210 Vienna	A-1010 Vienna	A-4040 Linz	A-1080 Vienna

[karl.doerner,walter.gutjahr,martin.polaschek,christine.strauss]@univie.ac.at, gabriele.kotsis@jku.ac.at

* corresponding author, Tel +43 1 4277 38112, Fax +43 1 4277 38115

Abstract

Workflow Systems provide means and techniques for modelling, designing, performing and controlling repetitive (business) processes. The quality of commercial workflow systems is usually determined to a large extent by their versatility and multi-purpose application. One of the current trends in improving workflow systems lies in enriching modelling methods and techniques in order to enlarge design alternatives.

The need for such advanced methods is particularly apparent in those fields in which the process duration can be determined only vaguely, but whose completion schedules are at the same time strictly enforced by a highly competitive market by means of fines and penalties. The risk of an overrun has to be weighed against the expected costs and benefits of certain measures reducing turnaround time and their combinations. Because they can help to avoid such penalties — or, at least, keep any potential losses low by identifying critical subprocesses and evaluate appropriate measures — modelling and evaluation techniques are becoming essential features of workflow systems.

Methodologically, we use Stochastic Branch-and-Bound as a technique for finding "optimal" bundles of measures. A numerical study shows the benefits of this meta-approach

by means of five stepwise-developed decision scenarios requiring rich modelling. Petri nets as a modelling tool and Stochastic Branch-and-Bound as an optimization technique determine for multi-mode resource constrained workflows of varying complexity an optimal workforce strategy with respect to the number of workers and their qualification.

Keywords: discrete optimization, Stochastic Branch-and-Bound, simulation, workflow optimization, Petri nets, stochastic timing, resource constrained modelling

1 Introduction

Following the notation of the Workflow Management Coalition [25], "workflow" is defined as the facilitation and automation of business processes, i. e., procedures for exchange of tasks, documents, or information among collaborating entities (humans or agents), by information and communication technology. Workflow Management systems provide tools and methods in order to support the modelling (so called build-time functions) and execution (or run-time functions) of workflows. In this work, we are focusing on the first aspect, namely the modelling of workflows.

To use a Workflow Management system, enterprise-relevant knowledge and data needs to be collected: What is done? How? By whom? Using what means? Workflow Management Systems are the basis of business process management systems and are capable of delegating business tasks to the right people at the right time using the right information resources (cf. Karagiannis [12]). The business process management system paradigm defines central tasks which are required to plan, design and optimize business processes involving the evaluation of alternative restructuring measures by analytical methods and/or simulation. The ability to create and evaluate ample what-if scenarios is a basic feature and a crucial quality-determining criterion for high-standard commercial business process management systems. Both effectiveness (i. e., doing the right things) and efficiency (i. e., doing things right) can be increased by performing workflow optimization which provides the basis for further decisions by generating correct and useful data if the appropriate methods and tools are applied.

Modelling formalisms need to be able to represent both functional aspects of the workflow, e. g., the precedence relations among activities, as well as quantitative aspects, e. g., duration of tasks. A variety of modelling formalisms have been discussed in the literature (see e. g., Ferscha

et al. [7] for a summarizing report), including information control nets [17], speech-act-based models [4], or modelling frameworks such as ARIS (http://www.ids-scheer.de/) or ADONIS (http://www.boc-eu.com/).

During the past decade, a wide body of literature has discussed and demonstrated the suitability of Petri nets for workflow modelling [2, 3, 23, 24]. Petri nets have been successfully applied to model the functional aspects of workflows (see e. g., Oberweis [18]), as well as time aspects (see e. g., Dehnert et al. [5]).

The advantage of using time-enhanced Petri nets for modelling workflows lies in their high degree of expressiveness allowing an arbitrarily detailed model of workflows. Whereas their disadvantage lies in the limited practical usage on account of the high complexity involved in solving the model using either analytic or simulation-based approaches. Research has focused on parallelization techniques to assure that such complex models can be solved by simulation (see e. g., Ferscha [6]).

In this paper, we advocate an alternative approach that combines the modelling power of time-enhanced Petri nets with an optimization heuristic called Stochastic Branch-and-Bound (SBB) [15, 16]. The basic idea behind this approach lies in "wrapping" the Petri net model into an optimization framework, so that solving the time-enhanced Petri net model in any node in the search space is reduced to "playing the token game" in a Petri net with deterministic timing, for which simulation is both fast and straightforward. Instead of explicitly modelling and simulating all possible what-if scenarios, our approach allows the analyst to specify the set of options and measures that can be applied in the workflow under study and the set of quality criteria in an objective function. SBB will evaluate possible, promising measure combinations in order to come up with a solution that is "good" according to the chosen objective function.

Our models and solution techniques are based on previously published work [8, 9]. In Gutjahr et al. [8] simulated annealing was combined with Importance Sampling, a rare event simulation technique, to solve a stochastic discrete time-cost problem (SDTCP) with binary alternatives. In Gutjahr et al. [9] SBB was applied to a variety of instances of SDTCPs using a heuristic instead of an exact method to solve the deterministic subproblem. This paper seeks to enrich the aforementioned model with some common problem features in the context of workflow systems. The enrichments of the basic application model are as follows: (1) constrained resources are introduced in place of unlimited resources (2) supplementary criteria (in addition

to project duration) are added to the objective function, and (3) some of the binary decisions are substituted by integer ones. When dealing with resource constraints, we evaluate decisions on the number of resources assigned to activities. Currently, we assume a first-come-first-serve strategy for assigning resources to activities in the numerical case studies as scheduling aspects are not the focus of this paper. With this approach, we introduce SBB in order to generate reliable results for stochastic workflows by controlling the sampling process, which helps to make expensive scenario techniques affordable in terms of computational effort. At the same time, we show that Petri nets serve as adequate tools for modelling resource constrained workflows.

Our numerical study includes a workflow in an R&D environment and is based on real world data from an electronic module development project. This model, presented by Moder et al. in [13], is frequently used in workflow and project management as a reference or benchmark example and will allow for a comparison of our results. The most basic setting sketches out a decision that is to be made on an operational level: alternative qualifications of the workforce will reduce (the risk of) delays on selected activities in the workflow at some cost (e. g., training costs). The qualifications can be interpreted as measures to avoid a delay of the project and — as a consequence — to avoid penalties; the underlying problem can be modelled as a SDTCP that selects those measures which minimize the expected overall costs.

The remainder of the paper is organized as follows: Section 2 presents the SDTCP as a general case of the deterministic discrete time-cost problem (DDTCP) and provides a description of both a basic model and the enriched model. Section 3 discusses our approach to model workflow with time-enhanced Petri nets and presents SBB as a solution technique for workflow optimization problems. To illustrate the approach, SBB is applied to a set of examples in a numerical study (Section 4). Our paper concludes with a summary of results and an outlook on future work in Section 5.

2 The Basic Optimization Problem

The basic problem structure that a workflow manager faces often takes the form of a time–cost tradeoff: On the one hand, meeting due dates necessitates a sufficiently large and qualified workforce (cf. e. g., [21, 26]). On the other hand, additional employees and workforce training increase the costs associated with the workflow. As very often the durations of tasks are not

known in detail in advance, but they follow some *probability distributions*, it cannot be predicted with certainty, whether certain due dates can be met or not; one can only estimate a *probability* for the total completion time of a workflow.

Activity planning methods dealing with time–cost problems (e. g., the well-established CPM- or PERT-based approaches) very often assume that resource/duration alternatives for activities can be expressed as a continuous time–cost function. However, this assumption is unrealistic because of the discrete nature of most resources in workflows (see e. g., Hindelang and Muth [11] or Panagiotakopoulos [19]). In many situations, workflow managers need to make zero/one decisions (e. g., provide training) and/or decide upon the assignment of an (integer) number of resources (e. g., workers). Determining an efficient workforce size and selecting adequate skills involves two intertwined problems: (1) estimating workflow completion time, taking into account the stochastic durations of tasks, and (2) solving a hard combinatorial optimization problem by determining the efficient training measures.

Therefore, we make the following assumptions on the workflow model:

First, it is assumed that there is a finite number of well-defined performance criteria C_1,...,C_k of the workflow system. These may represent such quantities as process runtimes, queue lengths, idle times of resources, etc. To each performance criterion C_i, a performance cost function p_i has to be assigned, indicating the loss incurred with respect to this performance criterion. For example, the cost function assigned to the process runtime indicates the loss caused by violation of termination dates.

The overall performance of the system can be measured by an overall performance cost function p which we assume to depend linearly on the single performance cost functions p_i, i. e., the overall performance cost is given as a weighted average with known weights w_i:

$$p = w_1 * p_1 + \ldots + w_k * p_k.$$

Secondly, it is assumed that there is a finite number of *measures* $M_1, \ldots, M_m$ that may improve the performance of the workflow system. For example, the measure "recruit additional personnel" may reduce the process runtime and by doing so improve the value of the performance cost function assigned to this performance criterion. Each measure M_j is connected with certain measure costs c_j. The decision maker has to choose a subset of measures which he wants to apply. Such a subset is specified by a binary vector $x = (x_1, \ldots, x_m)$, where $x_j = 1$, if measure M_j is chosen, and $x_j = 0$ else. Furthermore, we denote a subset of chosen measures

(specified by a particular x) as a measure combination. For a fixed chosen measure combination, the total costs result as the sum of the overall performance costs and the measure costs of the selected measures.

Besides the chosen measures, the performance of the workflow system may depend on random influences. We represent these random influences formally by using the symbol ω. This aspect of the model reflects the fact that, in practice, the performance parameters of a workflow system often can not be predicted with certainty. Moreover, different executions of one and the same process will usually lead to different values for the performance parameters. The classical statistical paradigm can be used for a formal representation of uncertainty on a specific execution of the process being considered. Following this paradigm, it is assumed that ω obeys a well–defined (probability) distribution.

In total, each performance cost function p_i (and therefore also the overall performance cost function p) has two arguments:

- the vector x, representing the chosen measure combination, and
- the random influence ω.

Using the notation above, we may represent the total costs as follows:

$$f(x, \omega) = p(x, \omega) + \sum_{j=1}^{m} c_j x_j,$$

where $p(x, \omega)$ represents the overall performance costs, also called indirect costs, and $\sum_{j=1}^{m} c_j x_j$ represents the costs of the selected measures (the direct costs).

The objective we adopt is to minimize the *expected total costs*: Denoting the mathematical expectation by E (note that E "acts" on the parameter ω), we aim to solve the problem

$$\begin{aligned} &\text{Minimize } E(f(x, \omega)) \\ &\text{subject to } x \in \mathcal{X}, \end{aligned} \tag{1}$$

where $\mathcal{X}$ can be

- either the set $\{0, 1\}^m$ of *all* possible measure combinations x, or
- a proper subset of $\{0, 1\}^m$, i. e., some measures cannot be combined with certain other measures. A typical situation leading to a restricted set $\mathcal{X}$ can be discerned in the

following: There are pools of resources $R_1, \ldots, R_s$, and each measure requires resources from pool R_i to an extent of a_i. Now, if the available resources in each pool R_i are limited by some b_i, measure combinations x with required resources that exceed some pool R_i have to be excluded from the set $\mathcal{X}$.

From the viewpoint of optimization theory, (1) is a so-called *stochastic optimization problem*. In Gutjahr et al. ([9], p. 126) it is shown that the described problem is a stochastic discrete time-cost problem (SDTCP) and that it reduces to the *deterministic discrete time–cost problem* (DDTCP), which in turn reduces to a *knapsack problem* (as to knapsack problems, see [20]). Moreover, *each* knapsack problem instance can be represented as a special case of the DDTCP. Therefore, since the knapsack problem is known to be an NP-hard problem ([20], pp. 374-375), the DDTCP is also NP-hard (cf. [8], pp. 69).

For the application of our approach, the following additional conditions must be satisfied:

- It must be possible to obtain estimates of the performance cost function values to each fixed given measure combination x by means of *(Monte Carlo) simulation*: Doing N simulation runs, one must be able to produce values

$$p_i(x, \omega_1), \ldots, p_i(x, \omega_N)$$

for $i = 1, \ldots, k$, where $\omega_1, \ldots, \omega_N$ are sampled according to the given probability distribution.

- Suppose that the first r components $x_1, \ldots, x_r$ of the vector x have already be fixed (i. e., values 0 or 1 have been assigned to these components), while the remaining components are still unspecified. Let $\mathcal{X}^*$ denote the subset of $\mathcal{X}$ containing all those vectors x for which the first r components have the indicated values. Then it must be possible to solve, for any fixed ω, the *deterministic* optimization problem

$$\text{Minimize } f(x, \omega)$$

$$\text{subject to } x \in \mathcal{X}^*, \tag{2}$$

in an acceptable computation time.

In principle, this can be done in one of two ways:

– In some cases, the problem structure is of a type where (2) can be solved by a particular combinatorial optimization technique (integer programming, ordinary branch-and-bound, dynamic programming, or a problem-specific optimization algorithm). Using such a technique as a subcomponent may improve the runtime of our overall optimization system considerably.

– In all other cases, complete enumeration must be applied: For this purpose, a procedure to generate all feasible elements x of $\mathcal{X}^*$ is needed. Evaluating the cost function $f(x, \omega)$ for each of these x, the best obtained x may be returned as the solution of (2).

In this paper we follow the second approach; details on solving the deterministic subproblem are given in Section 3.3.

3 Modelling Formalism and Solution Techniques

3.1 Workflow Modelling Using Generalised Stochastic Petri Nets

Petri nets provide a well-known formalism for modelling dynamic systems (see [14] for a survey and formal definitions). A Petri net (PN) is usually denoted by a tuple $(P, T, F, W, \mu^{(0)})$ where P is the set of places $(p_1, p_2, \ldots, p_{|P|})$, T is the set of transitions $(t_1, t_2, \ldots, t_{|T|})$, and $F \subseteq (P \times T) \cup (T \times P)$ defines an input/output relation to and from transitions. $\mu^{(0)}$ is a marking vector which gives for every place p the number of tokens $\mu^{(0)}(p)$ initially assigned to it. A transition t is said to be "enabled", if all its input places contain a sufficient number of tokens as given by W. An enabled transition can "fire" by removing a certain number of tokens from its input places and adding tokens to its output places.

This formalism can be used to model the flow of work, where activities or tasks are modelled as transitions and places denote the states or conditions before/after the execution of a task. Tokens can model both the flow of control as well as the flow of work. They can also be used to model the availability or the need for resources for an activity (see e. g., Ferscha [6] or van der Aalst [22] for the seminal work on using PNs as workflow models).

While structural properties of the Petri net can be used to derive qualitative workflow performance indicators (e. g., deadlocks in the flow of work), quantitative indicators can also be

derived by using a time enhanced PN model, e. g., generalized stochastic Petri nets as defined in Ajmone Marsan et al. [1]. In such a net, exponentially distributed or deterministic timings are added to transitions, representing the firing time of a transition or, in the workflow context, the duration of a task.

In this paper, we consider semi-structured workflows, i. e., systems where the specific tasks and their relationships are either fully known or where alternatives can be modelled explicitly. The duration of tasks, as well as the probabilities for choosing alternative paths in the workflow, are given as random variables with known distribution. In the following sections, we refer to this class of Petri nets as time-enhanced PNs. Resource places can be connected to activities (transitions) and the number of tokens in this resource place corresponds to the number of available resources. Resources can be either consumable, which are deleted after the completion of the activity, or persistent, i. e., the resource token is put back into the resource place after completion of the activity (after the transition has fired).

For convenience of modelling, we also assume the existence of a single transition that only fires once (this can be assured by connecting this transition with an input place containing only one consumable resource) to start a workflow process, as well as a defined end place, where the arrival of a token models the completion of this workflow. Note that the approach is formally not restricted to Petri nets following this specific structure, but the definition of performance characteristics, as well as the stopping criterion, in simulating an instance of the Petri net need to be adapted for other structures.

For time-enhanced Petri nets, we can derive for example the following basic performance characteristics (additional criteria can be derived, e. g., calculation of average values based on probability distributions) relevant for the analysis of workflows: (1) the total turn-around time defined as the time between firing the first transition in the net and the arrival of a token in the end place, (2) the time needed to execute subprocesses in the workflow, given by the time between the firing of two transitions, (3) the number and frequency of firing a transition, indicating how many times a certain activity was performed, and (4) the usage of resources given by the distribution of tokens in resource places.

In other approaches using time-enhanced Petri nets as their modelling formalism, these values can be obtained either by analytical techniques or by simulation, both being computationally intensive due to the stochastic nature of the net.

3.2 Stochastic Branch-and-Bound

We apply *Stochastic Branch-and-Bound* as the numerical optimization technique of our approach. This subsection gives a short description of SBB, which is a technique applicable to arbitrary combinatorial stochastic optimization problems (see e. g., Gutjahr et al. [10]). In the following subsection, we shall provide necessary details for the application of SBB to our concrete optimization problem (1).

A general combinatorial stochastic optimization problem can be formulated as follows:

$$\text{Minimize } F(x) = E(f(x,\omega)) \quad \text{for } x \in \mathcal{X}, \tag{3}$$

where $\mathcal{X}$ is a finite set of possible decisions or actions and $\omega \in \Omega$ denotes the influence of randomness, formally described by a probability space (Ω, Σ, P). The stochastic branch-and-bound method for solving (3), as developed in Norkin et al. [15], consists in

- partitioning the feasible set $\mathcal{X}$ into smaller subsets, and
- estimating lower and upper bounds of the objective function $F(x)$ within the subsets.

A key idea is to apply the well-known *lowest-bound rule* from deterministic branch-and-bound: At each step of the algorithm, the subset with minimum estimated lower bound is selected for a further partition into smaller subsets. By following this approach, "promising" subsets are investigated in more detail. Contrary to deterministic branch-and-bound, there is no definite step when the algorithm terminates with the exact solution. Instead, the computation can be aborted according to some stopping criterion selected by the user, yielding an approximate solution for (3).

For a more explicit description, let us denote by $\mathcal{X}^p$ $(p = 1, 2, \ldots)$ the current subsets into which the original set $\mathcal{X}$ has been divided. In total, the sets $\mathcal{X}^p$ form a partition $\mathcal{P}$ of $\mathcal{X}$. Correspondingly, the original problem (3) is divided into subproblems

$$\text{Minimize } F(x) = E(f(x,\omega)) \quad \text{for } x \in \mathcal{X}^p,$$

where $\mathcal{X}^p \in \mathcal{P}$. Let us set $F^*(\mathcal{X}^p) = \min_{x \in \mathcal{X}^p} F(x)$.

The following assumptions are made:

1. There is a *lower bound function* L and an *upper bound function* U, both of them mapping the set of subsets of $\mathcal{X}$ into the set $I\!R$ of real numbers, such that for all $\mathcal{X}^p \in \mathcal{P}$,

$$L(\mathcal{X}^p) \leq F^*(\mathcal{X}^p) \leq U(\mathcal{X}^p)$$

with $U(\mathcal{X}^p) = F(x')$ for some $x' \in \mathcal{X}^p$, and

$$L(\mathcal{X}^p) = F^*(\mathcal{X}^p) = U(\mathcal{X}^p)$$

if $\mathcal{X}^p$ is a singleton set.

2. There exist sequences $\xi^l(\mathcal{X}^p)$, $l = 1, 2, \ldots$, and $\eta^m(\mathcal{X}^p)$, $m = 1, 2, \ldots$, of *random estimates* of $L(\mathcal{X}^p)$ resp. $U(\mathcal{X}^p)$, such that

$$\xi^l(\mathcal{X}^p) \to L(\mathcal{X}^p) \text{ with probability one as } l \to \infty, \text{ and}$$

$$\eta^m(\mathcal{X}^p) \to U(\mathcal{X}^p) \text{ with probability one as } m \to \infty.$$

Note that $\xi^l(\mathcal{X}^p)$ and $\eta^m(\mathcal{X}^p)$ are random variables, while $L(\mathcal{X}^p)$ and $U(\mathcal{X}^p)$ are deterministic quantities.

Suppose that for each set $\mathcal{X}^p$, subsequences $(l_r(\mathcal{X}^p))$ and $(m_r(\mathcal{X}^p))$ of the index sequence $1, 2, \ldots$ are defined. We shall add some explanations later. With these assumptions and notations, we are now in the position to formulate the SBB algorithm:

Stochastic Branch-and-Bound:

$\mathcal{P}_0 := \{\mathcal{X}\}$; $\xi_0(\mathcal{X}) := \xi^{l_0(\mathcal{X})}(\mathcal{X})$; $\eta_0(\mathcal{X}) := \eta^{m_0(\mathcal{X})}(\mathcal{X})$;

for $r = 0, 1, \ldots$ **until** stopping criterion satisfied

{ select a set $\mathcal{Y}^r \in \mathcal{P}_r$ that minimizes the lower bound estimate $\xi_r(\mathcal{X}^p)$ $(\mathcal{X}^p \in \mathcal{P}_r)$;

if (a current approximate solution is desired) select an arbitrary element $x^r \in \mathcal{X}^r$, where $\mathcal{X}^r$ is a set from $\mathcal{P}_r$ that minimizes the upper bound estimate $\eta_r(\mathcal{X}^p)$ $(\mathcal{X}^p \in \mathcal{P}_r)$;

if ($\mathcal{Y}^r$ is a singleton) set $\mathcal{P}_{r+1} := \mathcal{P}_r$;

else { construct a partition $\mathcal{P'}_r(\mathcal{Y}^r)$ of $\mathcal{Y}^r$ consisting of the n_r disjoint sets

$\mathcal{Y}^r_i \subseteq \mathcal{Y}^r$ $(i = 1, \ldots, n_r)$ };

construct the new full partition:

$\mathcal{P}_{r+1} = (\mathcal{P}_r \setminus \{\mathcal{Y}^r\}) \cup \mathcal{P}'_r(\mathcal{Y}^r)$,

where the symbol "\" denotes the setminus operation; }

/* elements of $\mathcal{P}_{r+1}$ will be denoted by $\mathcal{X}^p$ again */

for all subsets $\mathcal{X}^p \in \mathcal{P}_{r+1}$

determine estimates $\xi_r(\mathcal{X}^p) = \xi^{l_r(\mathcal{X}^p)}(\mathcal{X}^p)$ and $\eta_r(\mathcal{X}^p) = \eta^{m_r(\mathcal{X}^p)}(\mathcal{X}^p)$;

}

As in the common branch-and-bound, the solution process can be visualized by the expansion of a tree, where in our case subsets correspond to nodes and partitioning a subset corresponds to appending successor nodes to a given node.

The algorithm requires for its full specification the description of a *bound step* (how to compute the lower and upper bound estimates $\xi^{(l)}(\mathcal{X}^p)$ resp. $\eta^{(m)}(\mathcal{X}^p)$), and that of a *branch step* (how to construct a partition $\mathcal{P}'_r(\mathcal{Y}^r)$ of $\mathcal{Y}^r$).

1. *Bound Step*

   As proposed in [15], we used the following lower bound:

$$L(\mathcal{X}^p) = E(\min_{x\in\mathcal{X}^p} f(x,\omega)) \leq \min_{x\in\mathcal{X}^p} E(f(x,\omega)) = \min_{x\in\mathcal{X}^p} F(x) = F^*(\mathcal{X}^p).$$

   It is immediately seen that in the case where $\mathcal{X}^p$ is a singleton, the formula above is satisfied with equality, as required. For the random estimates $\xi^l(\mathcal{X}^p)$, we take *sample averages* obtained by simulation over l random scenarios $\omega_1, \ldots, \omega_l$:

$$\xi^l(\mathcal{X}^p) = \frac{1}{l}\sum_{\nu=1}^{l} \min_{x\in\mathcal{X}^p} f(x,\omega_\nu) \to L(\mathcal{X}^p) \quad (l\to\infty). \tag{4}$$

   (Notation has been abbreviated, since in fact the random scenarios ω_ν also depend on l and on $\mathcal{X}^p$, i.e., $\omega_\nu = \omega_\nu(l,\mathcal{X}^p)$). For general remarks on the sample function values $f(x,\omega_\nu)$ in the case of our problem and on strategies for doing the required (deterministic) minimization in (4), we refer the reader to Section 2 above.

   The upper bound $U(\mathcal{X}^p)$ is obtained by selecting an element $x' = \phi(\mathcal{X}^p) \in \mathcal{X}^p$ (according to an arbitrary rule ϕ) and by setting $U(\mathcal{X}^p) = F(x') = F(\phi(\mathcal{X}^p))$. The random estimates $\eta^m(\mathcal{X}^p)$ are computed as sample averages with sample size m:

$$\eta^m(\mathcal{X}^p) = \frac{1}{m}\sum_{\nu=1}^{m} f(\phi(\mathcal{X}^p),\omega_\nu). \tag{5}$$

   As mentioned, $(l_r(\mathcal{X}^p))$ and $(m_r(\mathcal{X}^p))$ can be arbitrary subsequences of $1,2,\ldots$. The simplest choice for these subsequences is $l_r(\mathcal{X}^p) = m_r(\mathcal{X}^p) = r+1$ $(r = 0,1,\ldots)$. In this case the sample size for the sample estimator in iteration r is just $r+1$. In fact, this was the sampling scheme we implemented for our experiments. However, the general formulation of the algorithm admits also more evolved schemes for the growth of the sample size, possibly also schemes depending on the current subset $\mathcal{X}^p$.

2. *Branch Step*

In our special case, the solution space is the set of possible measure combinations

$$\mathcal{X} = \{0,1\}^m = \{x = (x_1, \ldots, x_m) \,|\, x_i \in \{0,1\}\}.$$

In our partitioning strategy, each subset $\mathcal{Y}^r$ is of the form

$$\mathcal{Y}^r = \{x = (x_1, \ldots, x_k, x_{k+1}, \ldots, x_m) \,|\, x_i \in \{0,1\}\ (i = k+1, \ldots, m)\} \tag{6}$$

with some fixed prefix string $(x_1, \ldots, x_k) \in \{0,1\}^k$ $(k \in \{0, \ldots, m\})$. We choose $n_r = 2$ for all r. The set $\mathcal{Y}^r$, as given by (6), is partitioned into two subsets $\mathcal{Y}_1^r$ and $\mathcal{Y}_1^r$ as follows:

$$\mathcal{Y}_1^r = \{x = (x_1, \ldots, x_k, 0, x_{k+2}, \ldots, x_m) \,|\, x_i \in \{0,1\}\ (i = k+2, \ldots, m)\},$$

$$\mathcal{Y}_2^r = \{x = (x_1, \ldots, x_k, 1, x_{k+2}, \ldots, x_m) \,|\, x_i \in \{0,1\}\ (i = k+2, \ldots, m)\}.$$

The solution x^r is the currently proposed overall solution in iteration r. In Norkin et al. [15], conditions for convergence of x^r $(r \to \infty)$ to an optimal solution are given. In this convergence result, the set $\mathcal{X}^r$ is shown to contract to a singleton, so that finally there remains no ambiguity on which x^r to select from $\mathcal{X}^r$. Of course, for a practical application of the algorithm, it suffices to compute x^r only in the *last* iteration. For the numerical results reported in the following section of this paper, the stopping criterion was defined as follows: (i) At least one leaf node of the search tree has been reached, and (ii) at least 30 iterations of the algorithm have been performed. This seemed to be adequate for finding *good* solutions within reasonable time. It is clear that due to the stochastic nature of the estimator, there is no guarantee that the found solution must be an exact optimizer, nor that the next iteration needs to produce a solution x^{r+1} identical to x^r.

3.3 Simulation Techniques for Solving the Deterministic Subproblem

In each iteration of the SBB it is necessary to solve the Petri net, which models the workflow to obtain the lower bounds for the objective function. A set of samples is selected for each node v in the tree (corresponding to a specific choice of measures). A sample is an instance of the deterministic subproblem (i. e., a deterministic Petri net). In order to compute the desired

performance indicators as needed in the objective function (token distribution in places, time between firing of transitions, and number of firings of a transition), all random variables are evaluated and the resulting deterministic Petri net is simply executed by playing the token game. The stochastic nature of the workflow is handled by the SBB procedure.

The following algorithm describes the "simulation" of a deterministic Petri net. Note that for all "decisions" the appropriate values of the random variables have already been defined.

repeat
    determine all enabled transitions
    select transition to fire
    update marking
    advance virtual simulation time
    update performance indicators
until (simulation end time) **or** (final marking)

Following a standard Petri net convention, we select the transition with the shortest execution time if multiple transitions are simultaneously enabled for firing. In the event that multiple transitions are enabled with identical execution times, one is selected at random. This is just a choice of implementation; other strategies could be implemented as well.

Note that there are two options for determining the end of the simulation: either a bound can be given for the virtual time up to which simulation should progress or a marking can be defined. In the following numerical examples we have chosen the latter option, as there exists a place which corresponds to a system state in which all the activities in the workflow have been completed; we define the arrival of one token in this place as the end of simulation. The structure of the nets in the examples guarantees that this is a reachable marking. If this condition cannot be guaranteed, then it is advisable to additionally specify a maximum value for virtual simulation time to guarantee the termination of the algorithm.

In order to obtain the performance characteristics defined in the previous sections, the deterministic simulator basically generates an event log file, in which each event in the simulation (i. e., a change in the marking caused by the firing of a transition) is recorded with the virtual time stamp. This log file contains all the information needed to derive the desired basic performance characteristics. The deterministic simulator provides an interface that can be accessed

by the SBB algorithm in order to obtain the corresponding values. For example, the token distribution can be obtained by the function $prob(n, p_k)$, which will return the percentage of time during which there were n tokens in place p_k.

The execution time for solving one instance of a deterministic Petri net on a Pentium IV with 2.4 GHz is in the range of milliseconds and grows linearly with the number of transition firings in the Net.

4 From Simple to Enriched Application Modelling — A Numerical Study

Starting with a real world example from literature, we provide a stepwise development of five different decision scenarios (see Table 1) and the relevant stochastic problem instances by modifying the resource constraints and objective functions. We conclude with a problem instance considering multiple workflows to be executed in parallel. The decision complexity increases as new restrictions and varying workforce conditions are sketched out. A basic workflow structure together with a set of measures given in the well-known PERT syntax gives an overview of the complexity of the precedence structure. Setting a measure will reduce the duration of activities corresponding, for example, to effective training of (human) resources. In this first introductory case, the resources are assumed to be unconstrained and some predefined activities may be executed with or without a speed up measure, a process which is also called "crashing". Our first approach supports the selection of the appropriate mix of crashing measures.

The first enrichment introduces resource limitations on one single type of renewable resource and we consider the decision on the number of resources to be assigned to the workflow process in addition to qualification measures. In order to model a balanced utilization of the workforce, we introduce an additional cost factor arising from boundaries that avoid idle times, but also support personal allowances. The decision scenarios are two numerical stochastic instances where the size of the workforce (i. e., the number of employees) that should be assigned to the stochastic workflow has to be determined and the selection of desirable qualifications (crashing measures) for a homogeneously-skilled workforce has to be supported. Formally, one loss function describes the penalties that occur if due dates are not met and another loss function imposes minimum busy times expected from management and the minimum personal

Case		Resources	Resource Type	Workflow Type
Section 4.1	(see Fig.1)	unconstrained	n.a.	single
Section 4.2.1	(see Fig.3)	constrained	homogeneous	single
Section 4.2.2	(see Fig.5)	mixed mode	homogeneous	single
Section 4.3	(see Fig.7)	constrained	heterogeneous	single
Section 4.4	(see Fig.9)	constrained	heterogeneous	multiple

Table 1: Survey of Workflow Examples

allowances imposed, for example, by labor unions. In addition to the binary variables modelling the choice of crashing measures, we introduce integer variables to model the assignment of a certain number of workers.

A further enhanced scenario assumes a heterogeneous workforce consisting of two types of employees at different fixed costs; our approach suggests a selection of efficient qualification measures on the basis of a stochastic workflow, due dates and busy times.

Finally, our stochastic branch-and-bound approach is applied to three parallel stochastic workflows, each of which requires two types of skilled staff. The decision regarding the size of a workforce that can be assigned globally is supported by our proposed approach.

The tests described were performed on a Pentium IV with 2.4 GHz.

4.1 A Simple Workflow

Gutjahr et al. [8] introduce a rare event simulation technique combined with Simulated Annealing by means of a real world example involving an electronic module development project. The application to that SDTCP instance required some additional information (e. g., loss function, cost of measures, effects of measures), which was added into the basic structure given by the activities, their precedence relations and beta-distributed durations described in Moder et al. ([13], p. 294). The resources in this basic core workflow are assumed to be unconstrained; some selected activities may be executed in two different modes, depending if a crashing measure is set or not. The decision being considered involves the selection of crashing measures, which in an application could consist of assigning an additional worker to some process or replacing a worker by an advanced one to speed up one or several of the selected processes.

As network planning techniques are well-known tools used mainly in project management for

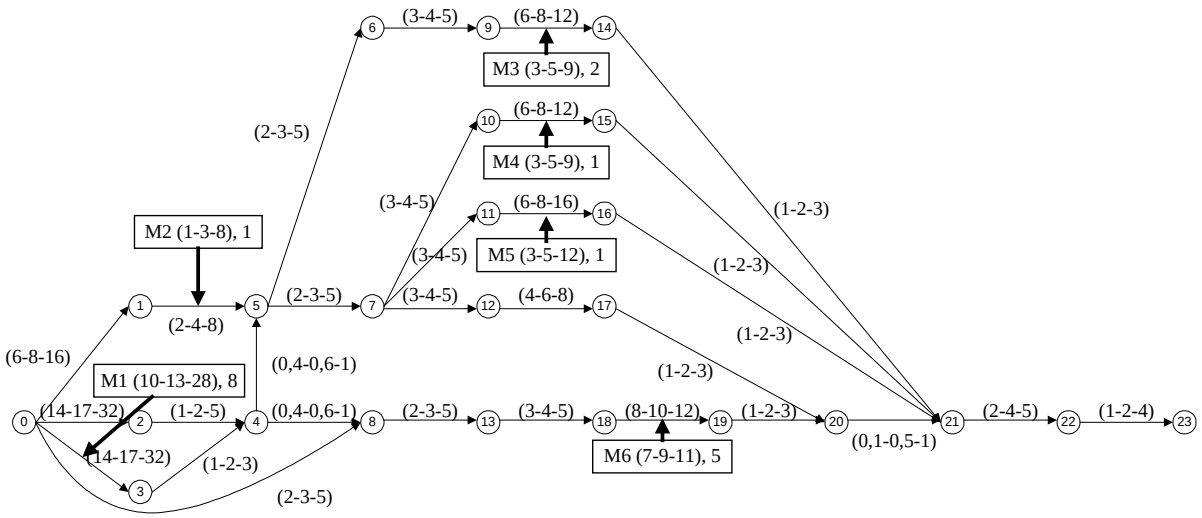

Figure 1: A PERT modelling a simple flow of work

planning and monitoring the fulfillment of (task) deadlines within a project, we apply PERT to graphically represent the workflow under consideration. PERT is a network planning technique in which an event is represented by a node in a graph and arcs correspond to the activities in the project. For each activity, the optimistic, pessimistic, and the most likely duration are given. In the context of workflow modelling, the arcs represent the tasks in the flow of work and the order of events defines the precedence relation among tasks.

Figure 1 depicts the workflow structure and the data of the example investigated. The graph is an activity-on-arc network and shows the precedence structure of the processes described in Moder et al. ([13], p. 294). The values on the arcs in brackets give the beta-distributed duration of an activity by the optimistic, the most likely and the pessimistic value. Rectangles are positioned at some processes and indicate a measure Mi that may reduce the duration or the variance of the process it acts on. It contains an identifier of the measure, new distribution values in the sequence optimistic value, most likely value and pessimistic value, as well as the cost of the measure given in monetary units after the comma.

The cost factors consist of the penalty applied for the late termination of the workflow, which is an estimated value, and the costs for the measures, which are deterministic values.

$$\text{Minimize } F(x) = E(f(x,\omega)) + \sum_{j=1}^{m} c_j \cdot x_j \quad \text{for } x_j \in \{0,1\}, j = 1,..,m$$

where f is a loss function non-decreasing in x for each fixed ω, which models the penalty in case of an overrun. The cost function maps a penalty of 500 monetary units if the work turnaround

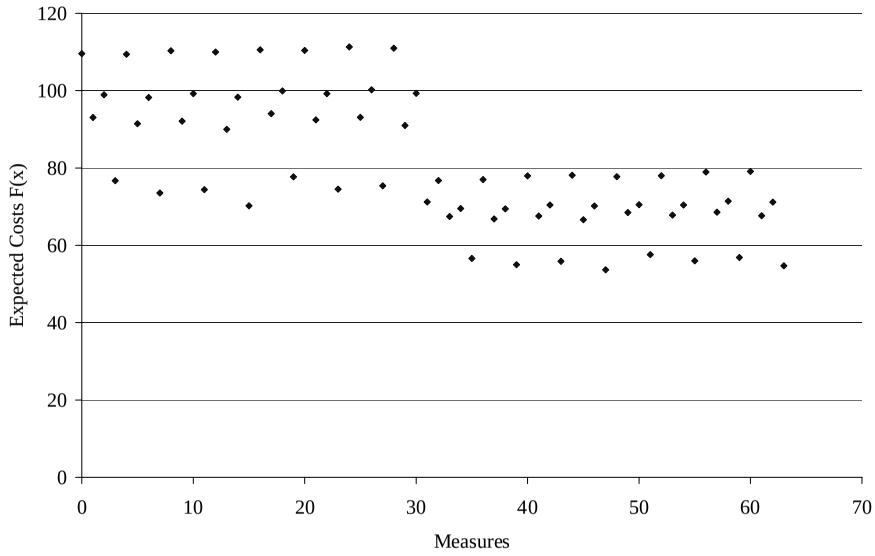

Figure 2: Solution space for all possible measure combinations in a simple workflow

time exceeds 52 time units.

Applying a traditional PERT approach, measure $M6$ is to be chosen, as this is the only measure that affects a process on the critical path. Figure 2 visualizes the solution space; for each of the 64 combinations of measures (on the x-axis denoted by the decimal representation of the binary vector of each of the six measures being set ($x_j = 1$) or not set ($x_j = 0$), with $M1$ at the most significant bit and Mn at the least significant bit) the objective function value $F(x)$ is given. A brute force simulation with $N = 10^6$ — which usually is computationally much too expensive — reveals the effects of the different measures (see Figure 2). The left half of the picture has $M1 = 0$, with much higher costs than setting $M1 = 1$, as in the right half. There is an obvious period of length 4 corresponding to the effects of the two least significant bits (measures $M5$ and $M6$), and it is clearly advantageous to set both equal to 1. Concentrating on only the lower envelope one then sees the effects of $M3$ and $M4$, which should also both be set equal to 1. The least important effect is that of measure $M2$, but it is still apparent that $M2 = 0$ is better. A comparison with such a brute force simulation further reveals that the solution proposed by the PERT approach (select measure $M6$) only has rank 46 due to the well-known weakness of classic PERT: underestimating the risk of overrun as a result of only focusing on processes on the critical path. Whereas the optimal solution proposes to set all measures except measure $M2$ (at costs of 53.68 monetary units), our SBB approach suggests (after simulating few scenarios, within a few minutes of CPU time) the choice of the entire set of measures except measure $M3$, which leads to overall costs of 55.99 monetary units. The

solution proposed by SBB achieved rank 5 in a ranking order according to the brute force simulation results.

Since also the brute force simulation results are subject to random errors (in spite of the high sample size), we performed a statistical significance test for the differences in the mean cost values of the single solutions. We obtained the following results: (a) The solution on rank 1 (all measures except $M2$) is highly significantly ($\alpha = 0.001$) better than all other solutions. (b) Whereas the solutions on ranks 2 and 3 and those on ranks 4 and 5 show no significant differences, rank 3 is highly significantly better than rank 4, and rank 5 (our SBB solution) is highly significantly better than rank 6 ($\alpha = 0.001$ in both cases). Note that the eight best solutions lie very closely in an interval of only 7% above the optimum; this relativizes the proposal of the fifth best solution.

4.2 Enriched Workflow: Homogeneous Resources

The first major enrichment introduces resource constraints: in this first step, we assume one single type of renewable resources that can be interpreted as a homogenous workforce, in which all employees have identical skills. Initially, we consider the assignment of resources to each process in the workflow, whereas a slightly modified example posits that only a predefined subset of tasks requires the limited resources. Such conditions can be found, for example, in manufacturing environments, where some of the tasks are performed by workers while others are performed by machines.

4.2.1 Assignment of Constrained Resources to Each Task

The initial scenario assumes a multi-skilled workforce in which everyone has the qualification to perform any process and a decision must be made with regard to what (if any) additional, process-specific know-how should be acquired. Given upper and lower bounds for an average degree of busy times, decision support is provided to the manager upon both the number of employees that should be assigned to the workflow and the specific extra qualification these employees should provide. Due to the structure of the workflow (see Figure 1), which shows a maximum of 5 processes in parallel, we analyze workforce sizes between 1 and 5 worker(s). For this type of decision, additional information about costs per worker and a loss function for violating the bounds for average busy times is added to the aforementioned core workflow: we

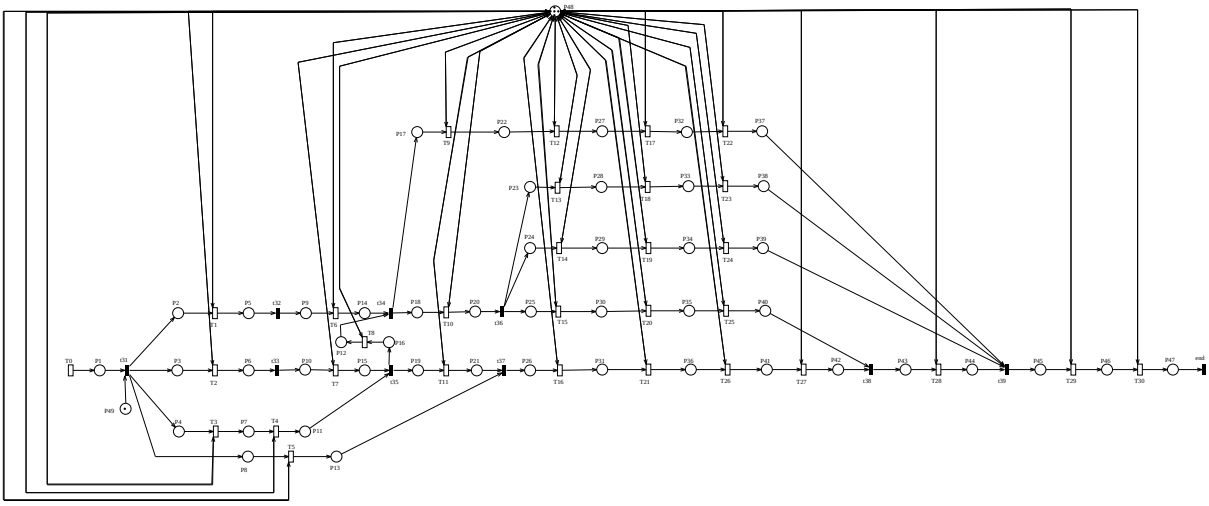

Figure 3: A Petri net modelling the workflow with limited resource for the entire set of tasks

assume that the costs for each worker are 240 monetary units and the costs for utilizing the workforce on average less than 75% causes costs of 1000 monetary units. The same costs apply if the average minimum personal allowance of 15% (equals an average utilization of 85%) is violated.

The outlined scenario is a NP-hard problem with $2^m \cdot \prod_{r=1}^{R} k_r$ possibilities with m binary measures and R different resources each in multiplicity k_r. The cost function consists of four terms: the expected costs of a late termination of the whole process ($E(f(x,\omega))$), a penalty if workforce utilisation u is outside certain boundaries ($p(1-I(0.75 < u(x) < 0.85))$), the training cost (setting binary measures x_j) and costs of the workforce ($\sum_{r=1}^{R} c_r \cdot x_r$). This leads to the following mathematical formulation of the problem.

$$\begin{aligned} \text{Minimize } F(x) \quad &= \quad E(f(x,\omega)) + p(1 - I(0.75 < u(x) < 0.85)) + \sum_{j=1}^{m} c_j \cdot x_j + \sum_{r=1}^{R} c_r \cdot x_r \\ &\text{for } x_j \in \{0,1\}, j = 1,..,m; \quad \text{and } x_r \in \{1,..,k_r\}, r = 1,..,R \end{aligned}$$

Figure 3 shows the basic stochastic workflow using Petri net syntax. The resource place in the upper center is connected with edges to each process and holds the varying numbers of resource tokens which represent 1, 2, 3, 4, or 5 worker(s).

Figure 4 shows the results for all possible combinations of measures as obtained from a brute force search (using 10,000 simulation runs for each measure combination and listing the

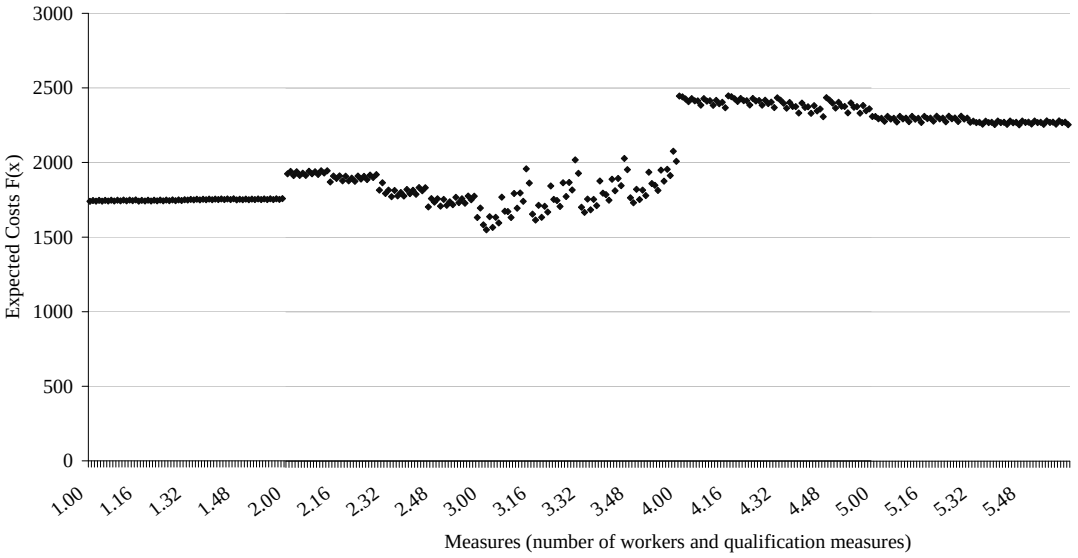

Figure 4: Solution space for all possible measure combinations in a resource constrained net

average over those runs as $F(x)$). The measure combinations are denoted by N.D, where N stands for the number of workers assigned to activities and D is the decimal representation of the binary vector of the six crashing activities being set ($x_j = 1$) or not set ($x_j = 0$).

The optimum value for expected costs is obtained at 1550.20 monetary units by chosing a workforce size of three workers and chosing training measure $M6$ (value 3.01 on the x-axis in Figure 4). Although the 20 global best solutions lie very closely in a narrow deviation range of only 10%, SBB proposes the second best solution of 1565.60 monetary units which deviates only 1% from this optimum solution. The SBB solution determines also three workers and suggests to acquire qualification measures $M5$ and $M6$ (measure combination 3.03). The execution time of SBB is in a range of only a few minutes, compared to 6 hours 40 minutes of execution time for a brute force search.

4.2.2 Assignment of Constrained and Unconstrained Resources to Subsets of Tasks

In a modified scenario, a distinction is made between those activities that are resource constrained and those which are resource unconstrained. Only a subset of the stochastic tasks in the workflow consumes a scarce resource. Figure 5 depicts a Petri net modelling the basic workflow structure with one limited resource type which is required only from those processes for which measures can be chosen. The number of resources is varied from 1 to 4 worker(s) (at most four of the activities with measures can be performed in parallel as can be seen in

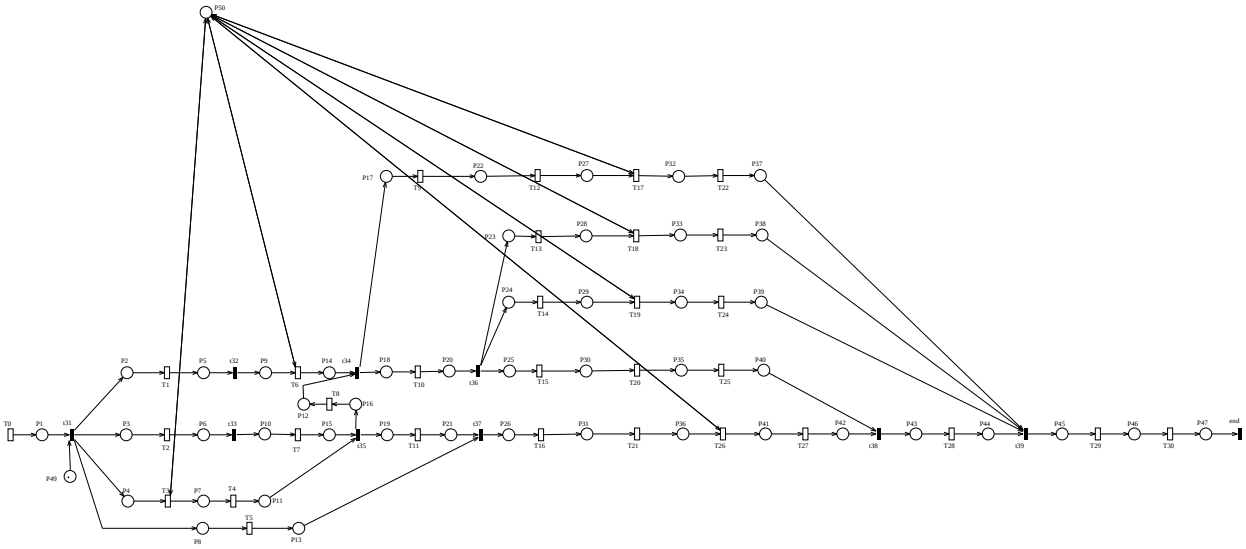

Figure 5: A Petri net modelling the workflow with limited resources for a subset of tasks

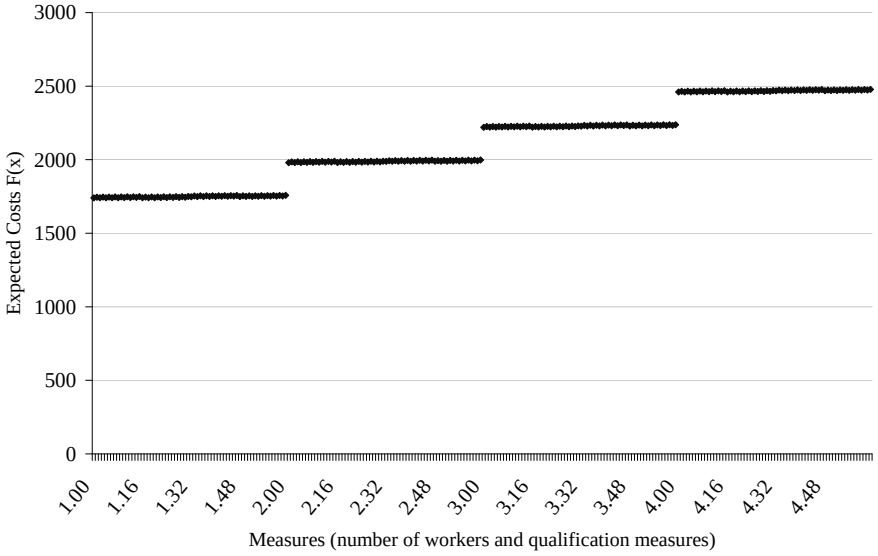

Figure 6: Solution space for all possible measure combinations in a net with resource constrained and unconstrained activities

Fig. 1). The cost function is the same as in Section 4.2.1. This model can represent a scenario, where tasks with measures are performed by workers that can reach better performance levels by means of training measures, while other tasks are performed without consuming any critical resource (e. g., transportation on conveyor belts in a manufacturing workflow).

Figure 6 visualizes the solution space generated by a brute force approach ($N = 10,000$). The optimum value for expected costs is obtained at some 1740.00 monetary units by the assignment of one worker to the workflow under consideration and setting no measures on any activity (1.00). SBB proposes this optimum solution within a few minutes of CPU time, compared to nearly 5 hours of execution time for a brute force search.

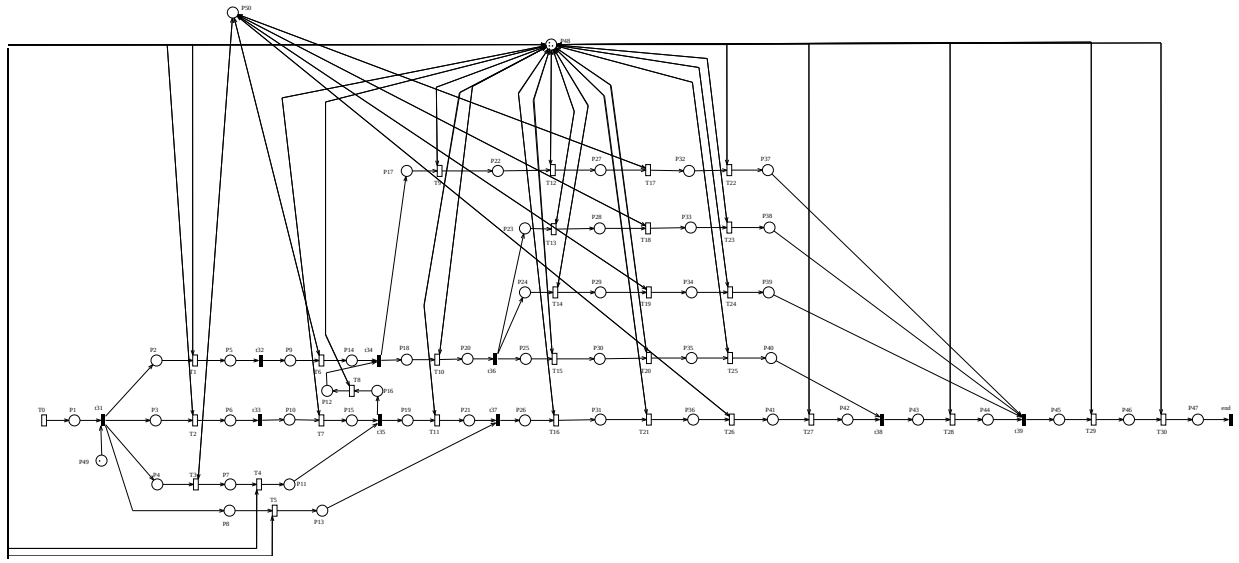

Figure 7: A Petri net modelling the workflow with heterogeneous constrained resource types

4.3 Enriched Workflow: Heterogeneous Resources

A second major step of enrichment introduces different types of constrained, renewable resources: one resource type may be assigned exclusively to those processes that have an option to speed up selected, predefined processes, while the second type can only be assigned to all other processes. This approach can be used to represent, for example, a pool of standby workers. Again, a decision support is provided upon the choice of qualifications the employees of the first resource type should acquire at some determined cost. The objective function, as well as the reference instance, is kept the same. This decision scenario assumes four workforce settings, with one or two employees being considered for each resource type. Therefore it is necessary to provide additional data about personnel cost: the workers of the resource category which may obtain extra skills result in fixed costs of 240 monetary units per worker, whereas the remaining category incur only costs of 200 monetary units per worker. In Figure 7 the two resource places hold one or two resource tokens; the resource place representing the resource with higher costs is connected with edges to those processes where qualification measures may be selected, the resource place representing the resource with lower costs controls the resources for all remaining processes.

Figure 8 shows the results for all possible combinations of measures as obtained from a brute force search (using 10,000 simulation runs for each measure combination and listing the average over those runs as $F(x)$). We use a similar notation for measures, MN.D, where M denotes the

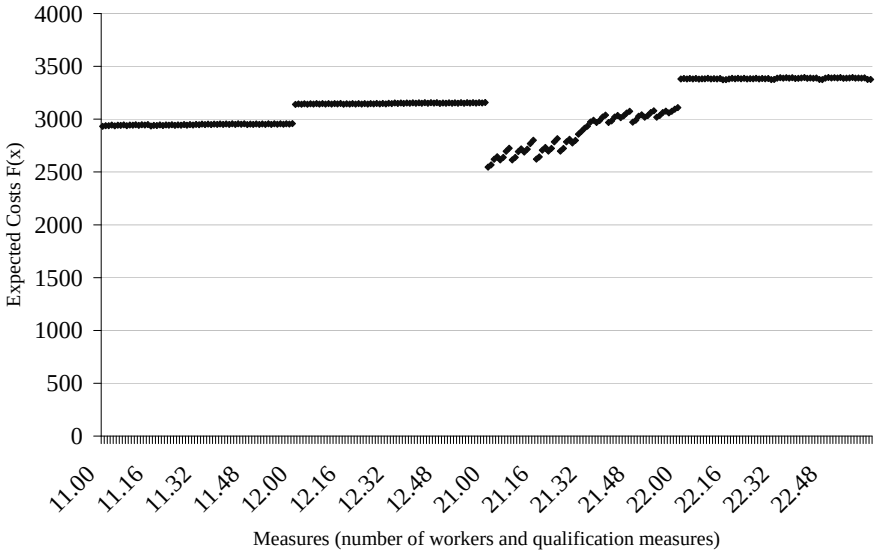

Figure 8: Solution space for all possible measure combinations in a net with heterogeneous resources

number of standby resources (to be assigned to activities without qualification measures), N the number of qualified resources (to be assigned to activities where crashing can be applied), and D being the decimal representation of the binary vector of the six crashing activities being set ($x_m = 1$) or not set ($x_m = 0$).

The optimum value for costs is obtained at 2545.80 monetary units by using a pool of two standby workers and one worker assigned to those tasks where qualification measures may reduce the processing time. The optimal solution further suggests setting no measures on any activity (21.00). The SBB approach finds this optimum solution within few minutes, compared to more than five hours of execution time for a brute force search.

4.4 An Enriched Integrated Workflow Approach

The last major step of enrichment involves the integration of three single workflows in parallel into one major workflow. For the sake of simplicity, we assume a triple workflow of the process structure used so far; we assume that each subworkflow involves the assignment of one employee of the resource type with higher cost, i. e., 240 monetary units (cf. Section 4.2.2) and that the number of standbys at some costs of 200 monetary units per worker for the triple of the workflows must be optimized. Again, the cost functions used so far remain unchanged.

Figure 9 shows a Petri net representation of the triple workflow with two different resource

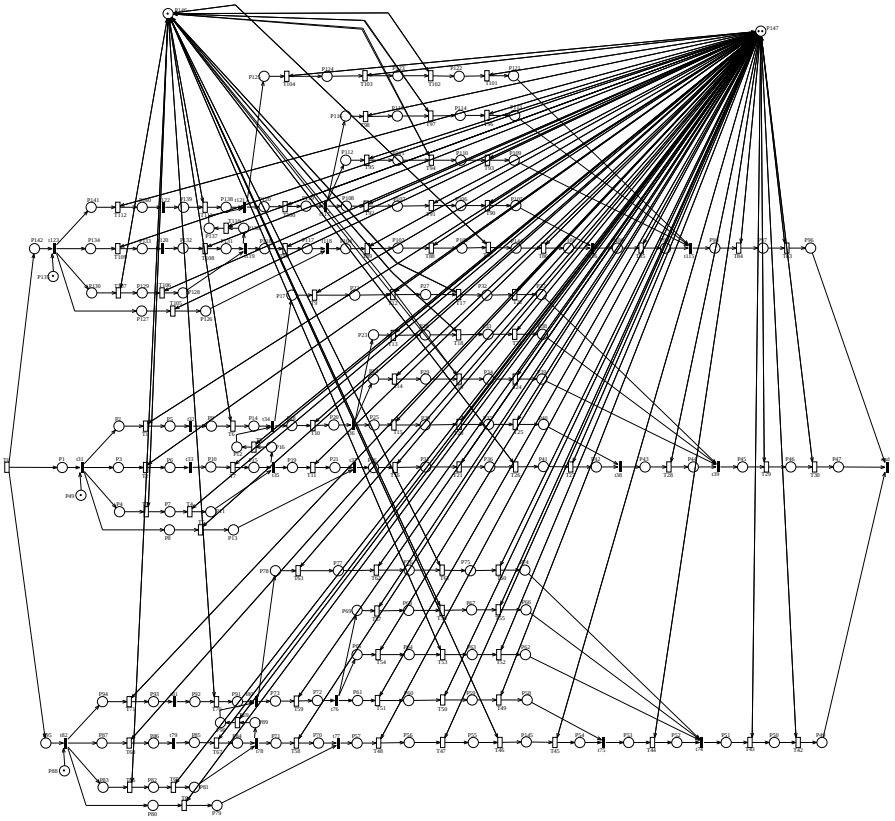

Figure 9: A Petri net modelling a large enriched integrated workflow

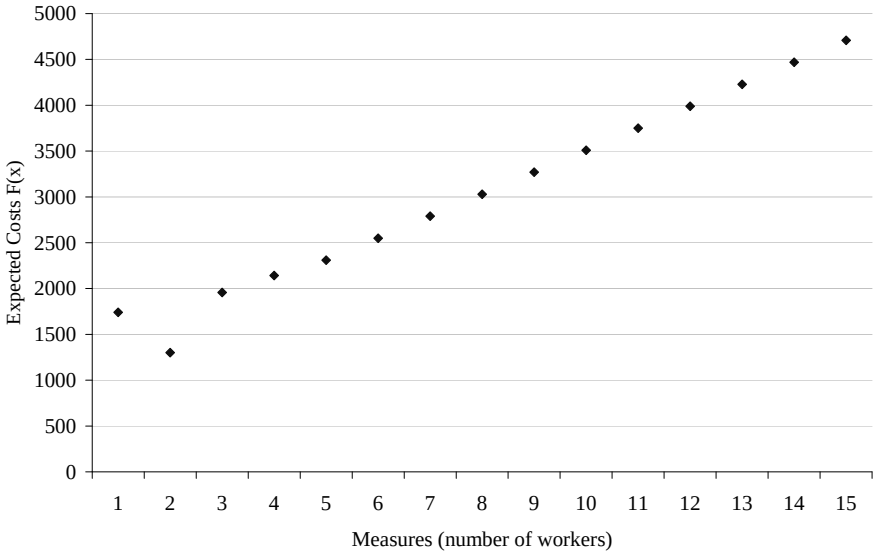

Figure 10: Solution space for all possible measure combinations in a large resource constrained net

types, whereas the standbys may vary from 1 to 15 employee(s). As only one worker is assigned to one process, it can be concluded from the structure of each single workflow that no more than 5 standbys are busy at the same time and as a consequence a maximum of 15 workers is considered for the intergrated workflow.

Figure 10 shows graphically the solution space and the optimal solution with two standbys for the whole integrated workflow that is proposed by the SBB approach within 8 minutes of CPU time.

5 Conclusions

The contribution presents an advanced approach supporting the modelling and optimization process of complex real world workflows where the duration of the tasks may vary and the resources are constrained. While it proposes Petri nets as a general design syntax for the modelling of discrete, stochastic optimization problems, our approach furthermore applies a novel method, Stochastic Branch-and-Bound, for the optimization part to evaluate workflow scenarios and to support decisions upon alternative resource assignments. The complexity of the described problem class is given by two interconnected problems: (1) providing an unbiased estimation of the completion time of a stochastic workflow and (2) solving an NP-hard combinatorial optimization problem to select an adequate workforce size and an efficient bundle

of measures (e. g., acquiring specific qualification). Therefore, the formal method developed for rich models intertwines simulation and combinatorial optimization. The benefits of the advanced approach are shown by means of five stepwise developed scenarios that are based on real world data. Constrained resources of one type are introduced in a first step, subsequently multi-mode resources are optimized in a more fully developed scenario. Finally, the workforce size of two resource types is determined for a large triple workflow.

Some further extensions may be analyzed with regard to the modelling issues: In order to refine control of workload on subgroups of the workforce, a disaggregation of resources is desirable (in the extreme case, on the basis of a single worker). Petri nets with "colored tokens" appear to be appropriate for modelling such systems. A second field to experiment in the modelling part includes testing various priority rules as alternative scheduling mechanisms within the simulation process.

Finally, future methodological work will include experiments with SBB variants that use a local search or meta-heuristic instead of the complete enumeration technique to solve the deterministic subproblem within the SBB. It may lead to even shorter run times and make it possible to solve even larger, more complex workflows.

Acknowlegdements The authors would like express their appreciation to the three anonymous referees; their valuable feedback and constructive comments strengthened this paper.

References

[1] M. Ajmone Marsan, G. Balbo, G. Chiola, G. Conte, Generalized stochastic Petri nets revisited: random switches and priorities, in: Proc. of the International Workshop on Petri Nets and Performance Models, August 24–26, IEEE Computer Society Press, Madison, Wisconsin, 1987, pp. 44–53.

[2] A. Alessandra, G. De Michelis, K. Petruni, Keeping workflow models as simple as possible, in: Proc. of the Workshop on Computer-Supported Cooperative Work, Petri nets and related formalisms, within the 15th International Conference on Application and Theory of Petri Nets (ATPN '94), 1994, pp. 11–23.

[3] V. Atluri, W. Huang, An authorization model for workflows, in: Proc. of the Fourth European Symposium on Research in Computer Security, Rome, Italy, September, 1996, pp. 25-27.

[4] E. Auramäki, E. Lehtinen, K. Lyytinen, A Speech-Act-Based office modelling approach, ACM Transactions on Office Information Systems 6 (2)(1988) 126–152.

[5] J. Dehnert, H. Freiheit, A. Zimmermann, Modelling and evaluation of time aspects in business processes, Journal of the Operational Research Society 53 (9)(2002) 1038–1047.

[6] A. Ferscha, Qualitative and quantitative analysis of business workflows using generalized stochastic Petri nets, in: G. Chroust, A. Benczur (Eds.), Connectivity 94: Workflow Management, Oldenbourg, Vienna, 1994, pp. 222–234.

[7] A. Ferscha, G. Haring, A. Hofmann, G. Kotsis, T. Nemec, Methods for the representation and analysis of workflows, University of Vienna, Internal Report, Vienna, 1994.

[8] W.J. Gutjahr, C. Strauss, M. Toth, Crashing of stochastic processes by sampling and optimisation, Business Process Management Journal 6 (1)(2000) 65–83.

[9] W.J. Gutjahr, C. Strauss, E. Wagner, A Stochastic Branch-and-Bound approach to activity crashing in project management, Journal on Computing 12 (2)(2000) 125–135.

[10] W.J. Gutjahr, A. Hellmayr, G.C. Pflug, Optimal stochastic single-machine tardiness scheduling by Stochastic Branch-and-Bound, European Journal of Operational Research 117 (2)(1999) 396–413.

[11] T.J. Hindelang, J.F. Muth, A Dynamic Programming algorithm for decision CPM networks, Operations Research 27 (2)(1979) 225–241.

[12] D. Karagiannis, BPMS: Business Process Management Systems, ACM SIGOIS Bulletin 16 (1)(1995) 10–13.

[13] J.J Moder, C.R. Phillips, E.W. Davis, Project management with CPM, PERT and precedence diagramming, 3rd ed. Van Nostrand, New York, 1983.

[14] T. Murata, Petri nets: properties, analysis and applications, Proceedings of the IEEE 77 (4)(1989) 541–580.

[15] V.I. Norkin, Y.M. Ermoliev, A. Ruszczyński, On optimal allocation of indivisibles under uncertainty, Operations Research 46 (3)(1998) 381–395.

[16] V.I. Norkin, G.C. Pflug, A. Ruszczyński, A branch and bound method for stochastic global optimization, Mathematical Programming 83 (3)(1998) 425–450.

[17] G.J. Nutt, An experimental distributed modelling system, ACM Transactions on Office Information Systems 1 (2)(1983) 117–142.

[18] A. Oberweis, Modellierung und Ausfuehrung von Workflows mit Petrinetzen, Teubner-Reihe Wirtschaftsinformatik, B.G. Teubner, Stuttgart, Leipzig, 1996.

[19] D. Panagiotakopoulos, A CPM time–cost computational algorithm for arbitrary activity cost functions, INFOR 15 (2)(1977) 183–195.

[20] C.H. Papadimitriou, K. Steiglitz, "Combinatorial optimization: algorithms and complexity". Prentice-Hall, Englewood Cliffs, NJ (1982).

[21] L. Sunde, S. Lichtenberg, Net-present-value cost/time tradeoff, International Journal of Project Management 13 (1)(1995) 45–49.

[22] W.M.P. van der Aalst, Verification of workflow nets, in: P. Azema, G. Balbo (Eds.), Application and Theory of Petri Nets, Lecture Notes in Computer Science 1248, Springer, Berlin, 1997, pp. 62–81.

[23] W.M.P. van der Aalst, The application of Petri nets to workflow management, The Journal of Circuits, Systems and Computers 8 (1)(1998) 21–66.

[24] M. Voorhoeve, Compositional modelling and verification of Workflow processes, in: Business Process Management: Models, Techniques, and Empirical Studies, Lecture Notes in Computer Science 1806, Springer, Berlin, 2000, pp. 184–200.

[25] Workflow Management Coalition, Interface 1 — process definition interchange. tc0020 (draft 5.0), Brussels, 1996.

[26] C. Yau, E. Ritchie, Project compression: A method for speeding up resource constrained projects which preserve the activity schedule, European Journal of Operational Research 49 (1)(1990) 140–152.