Cost-optimizing compositions of services – analysis and synthesis

Jan Sürmeli and Marvin Triebel

Humboldt-Universität zu Berlin, Institut für Informatik,
Unter den Linden 6, D-10099 Berlin, Germany
{suermeli|triebel}@informatik.hu-berlin.de

Abstract A *complex service* is a *stateful* implementation of a complex task, involving *interaction* with partner services via a well-defined interface. In this setting, several analysis and verification problems arise, such as discovery, substitutability, configuration, and adaptation. *Partner synthesis* is a solution strategy for these problems. Here, the task is to construct an *optimal partner* for a given service, based on a set of requirements and preferences. We solve partner synthesis for *non-functional* requirements and preferences, such as *cost-boundedness* and *cost-minimality.*

Keywords: partner synthesis, controller synthesis, non-functional properties, formal methods

1 Introduction

A *complex service* [1] is a *stateful* implementation of a complex task, involving interaction with other services via an interface. We study *asynchronously* interacting complex services and their composition. In this setting, several analysis and verification problems arise, for instance *discovery*, *substitutability*, *configuration*, and *adaptation*. *Service discovery* is the task to find an *optimal* partner service Q in a repository for a given service N. *Service substitutability* is the decision problem concerned with the preservation of all optimal partners after an update of N. *Service configuration* and *adaptation* resolve incompatibilities between services and ensure optimality.

In each of the problems, the concept of optimality bases on *requirements* and *preferences* on the *partnership*; that is, the composition $N \oplus Q$ of two services N and Q. Requirements and preferences may be defined on different levels [2]. For instance, consider two partnerships $N \oplus Q$ and $N \oplus Q'$. Assume that both $N \oplus Q$ and $N \oplus Q'$ match a certain behavioral requirement, however, $N \oplus Q$ causes lower execution costs than $N \oplus Q'$. Consequently, one would prefer Q' over Q.

Partner synthesis is a core technique to approach each of the aforementioned problems [3,4,5,6]. In a nutshell, partner synthesis is the task to construct a partner service Q for a given service N based on requirements and preferences.

To this end, we assume N to be given as a *formal model.* There are approaches to compile [7] formal models from practically feasible models such as BPEL [8].

There exist solutions for partner synthesis for both behavioral [9] and non-functional requirements [10]. We tackle partner synthesis for a class of *non-functional preferences* involving the *costs* of a partnership. Thereby, costs are an abstract concept to describe non-functional aspects such as execution costs, energy consumption, or reliability. We concentrate on solving partner synthesis for the cost model of *total costs.* In short, we consider non-negative costs for transitions. We determine the worst case total costs of a partnership by totaling the costs along each run, and taking the supremum of the costs of all runs. According to this cost model, we *prefer* a partnership $N \oplus Q$ over a partnership $N \oplus Q'$, if $N \oplus Q$ causes lower total costs in the worst case.

Additionally, we introduce a cost model based on *use cases.* Intuitively, a use case U describes a part of N with a distinct start and end, such as the booking of a flight, or the processing of an order. The motivation is that the concrete execution of U strongly depends on the partner. According to this cost model, we prefer a partnership $N \oplus Q$ over $N \oplus Q'$, if Q chooses a less expensive execution of U in the worst case than Q' does.

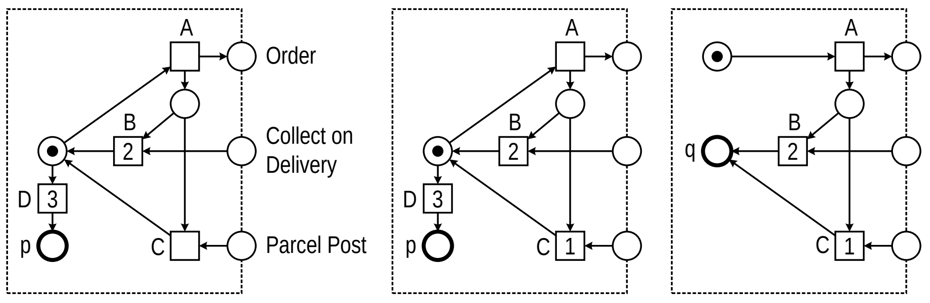
We structure our work as follows. Section 2 introduces a running example. Section 3 describes our formal model of services. Section 4 formalizes the cost models and the related non-functional preferences. Sections 5 and 6 prepare our synthesis approach. Sections 7 and 8 propose the actual synthesis procedure and discuss our prototypical implementation, respectively. Sections 9 and 10 evaluate related work and summarize the paper, respectively.

2 Running example

As running example we consider an online shop customer N and four different online shops $\mathsf{Q}_1, \dots, \mathsf{Q}_4$. From its initial state, N may either send an Order or terminate. After sending an Order, N waits for a shipping method – either Parcel Post or Collect on Delivery. Upon receiving the shipping method, N returns to its initial state. This behavior is modeled in Fig. 1 as a Petri net with an interface: Transition A sends the Order, transition B receives Collect on Delivery, transition C receives Parcel Post and transition D is internal.

In our running example, we assume that the Collect on Delivery shipping method causes costs of 2 units and terminating causes costs of 3 units to N. Therefore, we assign costs of 2 to transition B and 3 to D. For simplicity, we consider all other transitions to have costs of 0. Figure 2 shows partners $\mathsf{Q}_1, \dots, \mathsf{Q}_4$ of N. Partner Q_1 non-deterministically chooses the shipping method for each received Order. Q_2 and Q_3 each always choose the same shipping method, and Q_3 chooses Collect on Delivery at most once, and Parcel Post for all other orders. In the remainder of this section, we sketch the two cost models introduced in Sect. 4. Both cost models are based on the worst case.

The total costs of $\mathsf{N} \oplus \mathsf{Q}_2$ are unbounded, because transition B fires in each "round", and the number of rounds is unbounded. For similar reasons, the total



(a) Online shop customer N (b) Variant N' of N (c) Use case U of N'

Figure 1: Weighted nets N, N' and U with final markings $M^f_{\mathsf{N}} = M^f_{\mathsf{N}'} = \{[p]\}$, $M^f_{\mathsf{U}} = \{[q]\}$, and pairwise equal interfaces.

costs of $\mathsf{N} \oplus \mathsf{Q}_1$ are unbounded. Here, B does not fire in every round; however, the number of rounds with B is still unbounded. In contrast to that, the total costs of $\mathsf{N} \oplus \mathsf{Q}_3$ are trivially bounded, because B never fires in $\mathsf{N} \oplus \mathsf{Q}_3$, causing total costs of 3. A less trivial example of bounded total costs is $\mathsf{N} \oplus \mathsf{Q}_4$: Here, B may fire at most once, causing total costs of $2 + 3 = 5$. We classify the partners $\mathsf{Q}_1, \ldots, \mathsf{Q}_4$ of N based on the requirement of *cost-boundedness*. Only Q_3 and Q_4 match the requirement of *cost-boundedness*, because $\mathsf{N} \oplus \mathsf{Q}_3$ and $\mathsf{N} \oplus \mathsf{Q}_4$ have bounded total costs. Next, we compare the partners based on the preference of *total cost-optimality*. We prefer Q_3 w.r.t. total costs over Q_4, and in turn, Q_4 over Q_1 and Q_2, because $\mathsf{N} \oplus \mathsf{Q}_3$ causes lower total costs than $\mathsf{N} \oplus \mathsf{Q}_4$, and so on. We observe that Q_3 is a cost-optimal partner of N w.r.t. total costs, because no partner could cause less costs than 3 units.

Figure 1 shows a variant N' of N: The only difference is that the transition C causes costs of 1. We observe that each of the partnerships $\mathsf{N}' \oplus \mathsf{Q}_1 \ldots \mathsf{N}' \oplus \mathsf{Q}_4$ causes unbounded total costs. Thus, using the preference of total costs, we do not prefer one over the other. We overcome this problem by applying the cost model of *use case costs*. Figure 1 shows a *use case* U of N', which models a part of N'. Intuitively, U resembles one processing of an order by the partner: U begins with N' sending an order, and then ends with N' receiving one of the shipping methods. Studying partnerships of N' and a partner, we observe that U may occur more than once. The use case costs of $\mathsf{N}' \oplus \mathsf{Q}_2$ w.r.t. U are 2, because Q_2 always sends Collect on Delivery. One can see that U may be executed arbitrarily often, each execution causing costs of 2. The costs of D are not considered, as D does not belong to U. $\mathsf{N}' \oplus \mathsf{Q}_1$ and $\mathsf{N}' \oplus \mathsf{Q}_4$ are also examples for use case costs of 2. In contrast to $\mathsf{N}' \oplus \mathsf{Q}_2$, it is also possible to execute U with costs of 1 in those partnerships. However, we study the worst case, and therefore the use case costs amount to 2. The partnership $\mathsf{N}' \oplus \mathsf{Q}_3$ causes use case costs of 1: Each completion of U causes costs of 1, as B never fires in $\mathsf{N}' \oplus \mathsf{Q}_3$. In terms of *use*

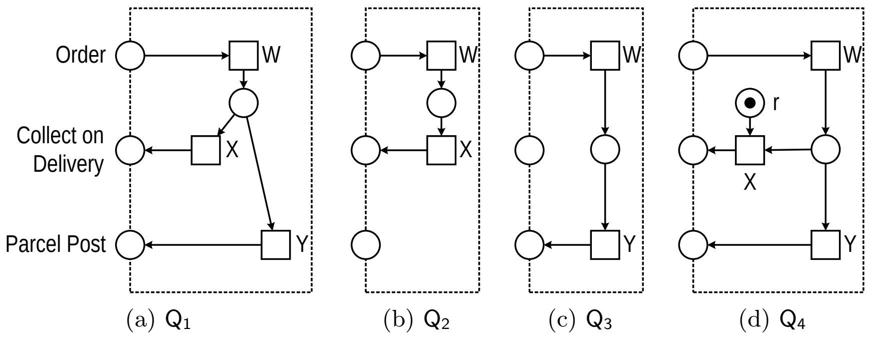


Figure 2: Weighted nets Q_1, Q_2, Q_3 and Q_4 with final markings $M^f_{\mathsf{Q}_1} = M^f_{\mathsf{Q}_2} = M^f_{\mathsf{Q}_3} = \{[\,]\}$, $M^f_{\mathsf{Q}_4} = \{[\,], [r]\}$, and pairwise equal interfaces.

case cost-optimality, we do not differ between Q_1, Q_2 and Q_4, and identify Q_3 as a cost-optimal partner w.r.t. use case costs.

3 Weighted nets, requirements, controllability

Let $\mathbb{N}_0$ and $\mathbb{N}$ denote the sets of natural numbers starting from 0 and 1, respectively. We write $\leq, +$ for the total order and the addition on $\mathbb{N}_0$, respectively. We use ∞ to denote infinity, and expand the notion $\leq$, such that for all $x \in \mathbb{N}_0 \cup \{\infty\}$, it holds (1) $x \leq \infty$ and (2) $\infty \leq x$ implies $x = \infty$. Let $X \subseteq \mathbb{N}_0 \cup \{\infty\}$. We define $\min(X) \in \mathbb{N}_0 \cup \{\infty\}$ as the greatest $y \in \mathbb{N}_0 \cup \{\infty\}$ with $y \leq x$ for all $x \in X$. We define $\sup(X) \in \mathbb{N}_0 \cup \{\infty\}$ as the least element of $y \in \mathbb{N}_0 \cup \{\infty\}$ with $y \geq x$ for all $x \in X$.

A *bag* b over a set U is a mapping $b : U \to \mathbb{N}$. A Bag b is *finite* if its *support* $\mathrm{supp}(b) := U$ is finite. We enumerate the content of the bag in square brackets, for instance, $x = [a, a, b]$ is the bag with $\mathrm{supp}(x) = \{a, b\}$, $x(a) = 2$ and $x(b) = 1$. We write $\mathrm{Bags}(U)$ for the set of all bags b with $\mathrm{supp}(b) \subseteq U$. For each bag $b \in \mathrm{Bags}(U)$ and each set V, we define the *V-mapping* $b_V : V \to \mathbb{N}_0$ of b by $b_V(v) := b(v)$, if $v \in V \cap U$ and $b_V(v) = 0$, otherwise. In the case of $\mathrm{supp}(b) \subseteq V$, we write b instead of b_V whenever it is clear from the context. We define the *sum* $b + b' \in \mathrm{Bags}(U \cup U')$ of two bags $b \in \mathrm{Bags}(U), b' \in \mathrm{Bags}(U')$ by $\mathrm{supp}(b + b') = \mathrm{supp}(b) \cup \mathrm{supp}(b')$ and $(b + b')(u) := b_{U \cup U'}(u) + b'_{U \cup U'}(u)$.

3.1 Petri nets and open nets

A *Petri net* [11] $N = [P, T, F, m^0]$ describes the functional aspects of behavior by means of a set of *places* P, a set of *transitions* T with $P \cap T = \emptyset$, a *flow function* $F : (P \times T) \cup (T \times P) \to \mathbb{N}_0$, and an *initial marking* $m^0 \in \mathrm{Bags}(P)$. Intuitively, a place $p \in P$ models a store for *tokens.* A *marking* $m \in \mathrm{Bags}(P)$

formalizes a state of N by determining the number $m(p)$ of tokens on each place p. A transition $t \in T$ is *enabled* in a marking m, iff for each place $p \in P$, it holds $m(p) \geq F(p,t)$. An enabled transition $t \in T$ may *fire*, written

$$m \xrightarrow{t}_N m', \tag{1}$$

consuming and producing tokens stored in places as determined by F, resulting in the marking m' defined by

$$m'(p) := (m(p) - F(p,t)) + F(t,p). \tag{2}$$

A finite sequence

$$m^0 \xrightarrow{t_1}_N m^1 \xrightarrow{t_2}_N \ldots \xrightarrow{t_n}_N m^n \tag{3}$$

of consecutive transition firings induces the *run* $w = t_1 \ldots t_n$ of N resulting in m^n, written

$$m^0 \xrightarrow{w} m^n. \tag{4}$$

The run w is *simple* in N if for each $0 \leq i < j \leq n$, it holds $m^i \neq m^j$. That is, no marking is visited twice. If W is a set of runs, then we write $W|_{\mathcal{S}}$ for the maximal subset of W containing only simple runs.

We write $\mathcal{R}(N,m)$ for the set of all runs of N resulting in m and $\mathcal{R}(N)$ for the set of all runs of N.

An *open net* [12] $N = [N', M^f, I, O]$, also known as service net, extends a Petri net N' by a set of *final markings* M^f and an *interface* $I \cup O$ consisting of two disjoint sets $I, O \subseteq P$ of places, such that for each transition $t \in T$ and place $p \in P$, it holds: If $p \in I$, then $F(t,p) = 0$, and if $p \in O$, then $F(p,t) = 0$. In words, a transition neither produces tokens on input places nor consumes tokens from output places. A run of an open net N is *terminating* if it results in a final marking of N. We write $\mathcal{T}(N)$ for the set of all terminating runs of N.

We *compose* two open nets N_1 and N_2 to a new open net

$$N_1 \oplus N_2 := [P_1 \cup P_2, T_1 \cup T_2, F_1 + F_2, m_1^0 + m_2^0, M^f, I, O], \tag{5}$$

where $M^f := \{m_1^f + m_2^f \mid m_i^f \in M_i^f\}$, $I := (I_1 \cup I_2) \setminus (O_1 \cup O_2)$, and $O := (O_1 \cup O_2) \setminus (I_1 \cup I_2)$.

Writing $N_1 \oplus N_2$, we assume:

$$(P_1 \setminus (I_1 \cup O_1) \cup T_1) \cap (P_2 \setminus (I_2 \cup O_2) \cup T_2) = \emptyset. \tag{6}$$

That is, each shared element of N_1 and N_2 is an interface place of either open net. Further, we assume:

$$I_1 \cap I_2 = O_1 \cap O_2 = \emptyset. \tag{7}$$

That is, no input (output) place of N_1 is an input (output) place of N_2.

We call $N_1 \oplus N_2$ a *partnership* if $I = O = \emptyset$; that is, the interface of $N_1 \oplus N_2$ is empty. We call N_1 and N_2 *partners* if $N_1 \oplus N_2$ is a partnership.

3.2 Weighted nets

A *weighted net* [10] $N = [N', c]$ extends an open net N' with transitions T by a cost function $c : T \to \mathbb{N}_0$. We write $\langle t \rangle_N := c(t)$ for the *costs* of t in N.

We extend the cost function c from transitions to sequences of transitions by totaling the costs along the sequence. For technical reasons, we formally extend c to words over arbitrary alphabets: Let w be a word over alphabet A with $w|_{T \cap A} = t_1 \dots t_n$, where $w|_{T \cap A}$ denotes the restriction of w to alphabet $T \cap A$. Then, we define the *costs* $\langle w \rangle_N$ of w in N by

$$\langle w \rangle_N := \langle t_1 \rangle_N + \ldots + \langle t_n \rangle_N. \tag{8}$$

We observe that for two words v, w, it holds $\langle vw \rangle_N = \langle v \rangle_N + \langle w \rangle_N$.

We further extend the cost function c to languages by choosing the supremum as the costs. Let L be a language, then we define the *costs* $\langle L \rangle_N$ of L in N by

$$\langle L \rangle_N := \sup(\{\langle w \rangle_N \mid w \in L\}). \tag{9}$$

We observe that for two languages L, M, it holds $\langle L \cup M \rangle_N = \sup(\{\langle L \rangle_N, \langle M \rangle_N\})$.

Whereas the costs of a word are always a natural number, the costs of a language may be ∞. A language L is *cost-bounded* in N iff $\langle L \rangle_N \in \mathbb{N}_0$.

We define the *costs* $\langle N \rangle$ of N as the costs of the set of terminating runs of N:

$$\langle N \rangle := \langle \mathcal{T}(N) \rangle_N. \tag{10}$$

We canonically extend the notion of *composition* from open nets to weighted nets: Let $N_1 = [N_1', c_1]$ and $N_2 = [N_2', c_2]$ be weighted nets. Then, we define the *composition* $N_1 \oplus N_2$ of N_1 and N_2 by $N_1 \oplus N_2 := [N_1' \oplus N_2', c_1 \cup c_2]$, where $c_1 \cup c_2$ is the union of the two functions c_1 and c_2. This notion is feasible due to assumption (6). We observe that for any word w, it holds $\langle w \rangle_{N_1 \oplus N_2} = \langle w \rangle_{N_1} + \langle w \rangle_{N_2}$.

3.3 Requirements and controllability

We consider a *requirement* Φ to be a set of partnerships. If $N_1 \oplus N_2 \in \Phi$, then N_2 is a *Φ-partner* of N_1. We call a weighted net N *Φ-controllable*, iff there exists a partnership $N \oplus X$ in Φ. In the remainder of this section, we define some important requirements.

Boundedness. A partnership $\mathcal{N}$ is *b-bounded* for $b \in \mathbb{N}_0$, if for each run $w \in \mathcal{R}(\mathcal{N}, m)$ and place p of $\mathcal{N}$, it holds $m(p) \leq b$. Furthermore, $\mathcal{N}$ is *bounded*, iff there exists some $b \in \mathbb{N}_0$, such that $\mathcal{N}$ is b-bounded. We write $\mathcal{P}_b$ for the set of all b-bounded partnerships. We further define $\mathcal{P} := \bigcup_{b \in \mathbb{N}_0} \mathcal{P}_b$. It is well-known that $\mathcal{N} \in \mathcal{P}_b$ and $\mathcal{N} \in \mathcal{P}$ are decidable [13].

Weak termination. A partnership $\mathcal{N}$ is *weakly terminating* if for each run $w \in \mathcal{R}(\mathcal{N})$, there exists a transition sequence x, such that $wx \in \mathcal{T}(\mathcal{N})$. We write $\mathcal{W}$ for the set of all weakly-terminating partnerships. For $\mathcal{N} \in \mathcal{P}$, $\mathcal{N} \in \mathcal{W}$ is decidable by applying model checking [14]. We further define $\mathcal{W}_b := \mathcal{W} \cap \mathcal{P}_b$. The problem $\mathcal{N} \in \mathcal{W}_b$ is decidable by first deciding $\mathcal{N} \in \mathcal{P}_b$ and if yes, deciding $\mathcal{N} \in \mathcal{W}$. It has been shown that $\mathcal{W}_b$-controllability is decidable [9].

Cost boundedness. A partnership $\mathcal{N}$ is *k-cost-bounded* for $k \in \mathbb{N}_0$, iff $\langle \mathcal{N} \rangle \leq k$. Furthermore, $\mathcal{N}$ is *cost-bounded*, iff there exists some $k \in \mathbb{N}_0$, such that $\mathcal{N}$ is k-cost-bounded. We write $\mathcal{C}_k$ for the set of all k-cost-bounded partnerships. We further define $\mathcal{C} := \bigcup_{k \in \mathbb{N}_0} \mathcal{C}_k$, $\mathcal{W}_b\mathcal{C} := \mathcal{W}_b \cap \mathcal{C}$ and $\mathcal{W}_b\mathcal{C}_k := \mathcal{W}_b \cap \mathcal{C}_k$. We have shown in previous work [10] that deciding $\mathcal{N} \in \mathcal{C}_k$ may be reduced to $\mathcal{N} \in \mathcal{P}_b$, and thus $\mathcal{N} \in \mathcal{C}_k$ is decidable. Using the same argument we can show that $\mathcal{N} \in \mathcal{C}$ and $\mathcal{N} \in \mathcal{W}_b\mathcal{C}$ are decidable. Additionally, we have shown that $\mathcal{W}_b\mathcal{C}_k$-controllability is decidable. In this paper, we will show that $\mathcal{W}_b\mathcal{C}$-controllability is decidable.

4 Preferences and optimal partners

We consider a *preference relation* $\succsim$ to be a total, reflexive and transitive relation on the set of all partnerships. Let $\mathcal{N} = N \oplus Q$ and $\mathcal{N}' = N' \oplus Q'$ be partnerships. Then, we *$\succsim$-prefer* $\mathcal{N}$ at least as much as $\mathcal{N}'$ iff $\mathcal{N} \succsim \mathcal{N}'$. A weighted net N *$\succsim$-prefers* a partner Q at least as much as a partner Q' iff $N \oplus Q \succsim N \oplus Q'$. Let Φ be a requirement. We call a Φ-partner Q of N a *$\succsim$-optimal partner* of N iff for each Φ-partner Q' of N, it holds $N \oplus Q \succsim N \oplus Q'$. In words, Q is a $\succsim$-optimal Φ-partner if N does not prefer any Φ-partner Q' over Q.

4.1 Total costs

In this section, we introduce the preference relation of *total cost-optimality* $\succsim$ studied in this paper. Intuitively, we compare the costs of two partnerships and prefer those partnerships with lower costs. Because the costs of a partnership depend on the set of all of its terminating runs, this cost model describes the worst case from a non-functional point of view. We define the preference relation *total cost-optimality* $\succsim$ on partnerships by $\mathcal{N} \succsim \mathcal{N}'$ iff $\langle \mathcal{N} \rangle \leq \langle \mathcal{N}' \rangle$. As explained in Sect. 2, Q_3 (Fig. 2) is a $\succsim$-optimal $\mathcal{W}_b$-partner of N (Fig. 1).

4.2 Use case costs

Intuitively, the preference of *use case cost-optimality* only considers the costs to complete a given *use case* of a service. A use case describes a partial behavior of a service. For example, think of "booking a flight" in an airline ticket store, which provides an unlimited number of bookings. Figure 1 shows a use case U for the running example N'. We observe that U contains behavioral alternatives with different costs.

The *use case cost* in a partnership is the maximal cost of all complete occurrences of a use case U along all runs. The use case may occur arbitrarily often but for this preference the costs are not accumulated. Instead, the most expensive occurrence determines the costs of interest. We ignore all costs outside the use case. Additionally, costs inside the use case only count if the use case is completely executed. Ordering the partnerships by their use case costs w.r.t. a use case U results in the preference for use case costs. If a use case is acyclic, each partner causes bounded use case costs. This allows to compare partners that result in unbounded total costs.

4.3 Comparison

We compare the two cost models of total costs and use case costs. Consider again our running example (Fig. 1) and its partners (Fig. 2): $\mathsf{N} \oplus \mathsf{Q}_3$ has lower total costs than $\mathsf{N} \oplus \mathsf{Q}_4$, so Q_3 is preferred over Q_4 regarding total costs. Imagine a use case for N, similar to U, with costs zero for C. Then $\mathsf{N} \oplus \mathsf{Q}_3$ and $\mathsf{N} \oplus \mathsf{Q}_4$ would yield the same use case costs. In contrast to that $\mathsf{N}' \oplus \mathsf{Q}_3$ and $\mathsf{N}' \oplus \mathsf{Q}_4$ cause unbounded total costs. Using the use case cost preference with U, Q_3 is preferred over Q_4. Thus, the two preferences prefer different partnerships. Though, for the same partnership, the use case costs are always less then the total costs.

For acyclic use cases we may reduce the problem of analyzing the use case costs to the problem of analyzing the total costs. We only sketch the reduction. We construct a new service N'_U from the given service N' and the use case U, with the following property: Every $\mathcal{W}_b$-Partner X of N' is a $\mathcal{W}_b$-Partner of N'_U and the use case costs of $\mathsf{N}' \oplus \mathsf{X}$ coincide with the total costs of $\mathsf{N}'_\mathsf{U} \oplus \mathsf{X}$. Intuitively, the new service may switch once from the given service to the use case and back. This switching behavior is not connected to the interface. In fact, switching may neither be enforced, excluded, nor observed by the partner. We plan to study cyclic use cases in future work.

The two approaches give two different opportunities to model cost preferences. The following results assume the cost model of total costs. As explained in the previous paragraph, we may transfer these results to use case cost-optimality with acyclic use cases.

5 The minimal cost bound

In this section, we introduce the *minimal cost bound* $\mathrm{mb}(N) \in \mathbb{N}_0 \cup \{\infty\}$ of a given $\mathcal{W}_b$-controllable weighted net N. Intuitively, $\mathrm{mb}(N)$ describes the costs incurring during the interaction with a cost-optimal partner regarding total costs. For the running example N (Fig. 1) the minimal cost bound is 0, as Q_3 (Fig. 2) is the cost-minimal partner with $\langle \mathsf{N} \oplus \mathsf{Q}_3 \rangle = 0$. Formally, we define the minimal cost bound $\mathrm{mb}(N)$ of a $\mathcal{W}_b$-controllable weighted net N by

$$\mathrm{mb}(N) := \min(\{\langle N \oplus Q \rangle \mid N \oplus Q \in \mathcal{W}_b\}) \qquad . \tag{11}$$

From the definition of $\precsim$-optimality, we may conclude that $\mathrm{mb}(N)$ indeed models the costs of the composition with a cost-minimal partner:

Corollary 1. *A weighted net Q is a $\precsim$-optimal $\mathcal{W}_b$-partner of a weighted net N iff $\langle N \oplus Q \rangle = \mathrm{mb}(N)$.*

Thus, we may decide $\precsim$-optimality of a $\mathcal{W}_b$-partner Q by comparing $\langle N \oplus Q \rangle$ and $\mathrm{mb}(N)$. Additionally, $\mathrm{mb}(N)$ is the core ingredient of our synthesis approach as described in Sect. 7. In general, $\mathrm{mb}(N)$ may either be a natural number or ∞. We observe the following relation between $\mathrm{mb}(N)$ and $\mathcal{W}_b\mathcal{C}_k$-controllability:

Corollary 2. *Let N be a $\mathcal{W}_b$-controllable weighted net. If $\mathrm{mb}(N) \neq \infty$, then N is $\mathcal{W}_b\mathcal{C}_{\mathrm{mb}(N)}$-controllable.*

Assuming $\mathrm{mb}(N) \in \mathbb{N}_0$, we may thus compute $\mathrm{mb}(N)$ in an iterative approach, checking $\mathcal{W}_b\mathcal{C}_k$-controllability for $k = 0, 1, \ldots$. In the case of $\mathrm{mb}(N) = \infty$, this procedure would not terminate and thus not be feasible. Therefore, the challenge in computing $\mathrm{mb}(N)$ is deciding the finiteness of $\mathrm{mb}(N)$. To this end, we connect the notion of $\mathrm{mb}(N)$ with the notion of $\mathcal{W}_b\mathcal{C}$-controllability. First, we observe that N is not $\mathcal{W}_b\mathcal{C}$-controllable if $\mathrm{mb}(N) = \infty$. From that and Corollary 2, we conclude:

Corollary 3. *Finiteness of* $\mathrm{mb}(N)$ *coincides with* $\mathcal{W}_b\mathcal{C}$*-controllability of* N.

From this, we conclude that finiteness of $\mathrm{mb}(N)$ is decidable iff $\mathcal{W}_b\mathcal{C}$-controllability of N is decidable. Therefore, we study the problem of deciding $\mathcal{W}_b\mathcal{C}$-controllability in the following section.

6 Deciding cost-bounded controllability

We have shown in Sect. 5 that $\mathrm{mb}(N)$ is computable iff $\mathcal{W}_b\mathcal{C}$-controllability is decidable. In this section, we reduce this decision problem to deciding $\mathcal{W}_b\mathcal{C}_k$-controllability by specifying a canonical k. More formally, given a weighted net N, we (1) define the *cost discriminant* $\mathrm{dis}(N)$ of N, and (2) show that N is $\mathcal{W}_b\mathcal{C}$-controllable iff N is $\mathcal{W}_b\mathcal{C}_{\mathrm{dis}(N)}$-controllable. Then, Corollary 3 implies that $\mathrm{mb}(N)$ is computable.

We define the *cost discriminant* $\mathrm{dis}(N)$ of N based on the partnership of N and its *most-permissive* $\mathcal{W}_b$*-partner* [9] $\mathrm{mp}(N)$. For the example service N (Fig. 1), the most-permissive $\mathcal{W}_b$-partner is Q_1 (Fig. 2). Intuitively, $N \oplus \mathrm{mp}(N)$ is the richest partnership of N in $\mathcal{W}_b$ regarding simulation of behavior: The partnership $N \oplus \mathrm{mp}(N)$ simulates each partnership $N \oplus Q \in \mathcal{W}_b$. It has been shown that for a given $\mathcal{W}_b$-controllable weighted net N, such a partner $\mathrm{mp}(N)$ exists and may be computed. Because the simulation relation implies finite trace inclusion [15], we observe the following: If $N \oplus Q \in \mathcal{W}_b$ and $v = t_1 \ldots t_n \in \mathcal{T}(N \oplus Q)$, then there exists $w = r_1 \ldots r_n \in \mathcal{T}(N \oplus \mathrm{mp}(N))$ such that

$$v|_{T_N} = w|_{T_N} \qquad . \tag{12}$$

Thus, the set $\mathcal{T}(N \oplus \mathrm{mp}(N))$ over-approximates the set of terminating runs in any partnership of N in $\mathcal{W}_b$. Consequently, we define $\mathrm{dis}(N)$ based on this set. To this end, we consider a most-permissive partner with a cost function mapping each transition to zero. More precisely, we define

$$\mathrm{dis}(N) := \langle \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}} \rangle_N \qquad . \tag{13}$$

That is, $\mathrm{dis}(N)$ equals the cost of the most expensive simple terminating run of $N \oplus \mathrm{mp}(N)$. We may use index N instead of $N \oplus \mathrm{mp}(N)$ in $\langle \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}} \rangle_N$, because we assume that the transitions of $\mathrm{mp}(N)$ have zero costs. In the example N (Fig. 1) with $\mathrm{mp}(\mathsf{N}) = \mathsf{Q}_1$ (Fig. 2), the set of simple terminating runs is: $\{\mathsf{D}\}$. Thus the cost discriminant for the example is $\mathrm{dis}(\mathsf{N}) = \langle \mathsf{D} \rangle_{\mathsf{N}} = 3$.

Computing $\mathrm{dis}(N)$ thus boils down to computing $\mathrm{mp}(N)$ and solving the longest path problem for the state space of $N \oplus \mathrm{mp}(N)$. While computationally expensive [16,17], this is possible because $\mathrm{mp}(N)$ is finite.

Corollary 4. *Let N be a weighted net. Then, $\mathrm{dis}(N)$ is computable.*

We observe that a weighted net N is not necessarily $\mathcal{W}_b\mathcal{C}_{\mathrm{dis}(N)}$-controllable: For instance, the variant of the running example N' is not controllable for any finite cost bound.

In the remainder of this section, we prove that $\mathcal{W}_b\mathcal{C}$-controllability reduces to $\mathcal{W}_b\mathcal{C}_{\mathrm{dis}(N)}$-controllability. The main ingredient for the proof is a relation of the cost discriminant $\mathrm{dis}(N)$ and the minimal cost bound $\mathrm{mb}(N)$. We develop this relation in the following lemmas.

First, we observe that if Q is a $\mathcal{W}_b\mathcal{C}$-partner of N, then the most expensive simple terminating run equals the most expensive terminating run.

Lemma 1. *Let $N \oplus Q \in \mathcal{W}_b\mathcal{C}$. Then, $\langle N \oplus Q\rangle = \langle \mathcal{T}(N \oplus Q)|_{\mathcal{S}}\rangle_{N\oplus Q}$.*

Proof. We show that for all $w \in \mathcal{T}(N \oplus Q)$, there exists $v \in \mathcal{T}(N \oplus Q)|_{\mathcal{S}}$ with $\langle w\rangle_{N\oplus Q} = \langle v\rangle_{N\oplus Q}$. The opposite direction trivially holds. If w is simple, w is a proof. Otherwise, there exists a marking m which is visited more than once along w. Hence, w has the form $m^0 \xrightarrow{w_1} m \xrightarrow{w_2} m \xrightarrow{w_3} m^f$, and $w_1w_2w_2w_3 \in \mathcal{T}(N\oplus Q)$. Because $N \oplus Q$ is cost-bounded by assumption, we conclude $\langle w_2\rangle_{N\oplus Q} = 0$. Hence, $\langle w\rangle_{N\oplus Q} = \langle w_1w_3\rangle_{N\oplus Q}$. Therefore, subsequent application of this argument leads to $v \in \mathcal{T}(N \oplus Q)|_{\mathcal{S}}$ with $\langle w\rangle_{N\oplus Q} = \langle v\rangle_{N\oplus Q}$. $\square$

If Q is a $\mathcal{W}_b\mathcal{C}$-partner of N, there always exists a $\mathcal{W}_b\mathcal{C}$-partner Q' of N, such that Q assigns zero costs to each of its transitions. Therefore, the $\precsim$-optimal $\mathcal{W}_b$-partners of N each assign costs of zero to their transitions. Hence, the costs of N with a $\precsim$-optimal $\mathcal{W}_b$-partner only depend on transitions of N.

Corollary 5. *Let Q be a $\precsim$-optimal $\mathcal{W}_b$-partner of a $\mathcal{W}_b\mathcal{C}$-controllable open net N. Then, $\langle N \oplus Q\rangle = \langle \mathcal{T}(N \oplus Q)|_{\mathcal{S}}\rangle_{N}$.*

Combining Def. (12) and Corollary 5, we find the following relationship between terminating runs of $N \oplus Q$ and the simple runs of $N \oplus \mathrm{mp}(N)$.

Lemma 2. *Let N be a $\mathcal{W}_b\mathcal{C}$-controllable weighted net. Let Q be a $\precsim$-optimal $\mathcal{W}_b$-partner of N. If $w \in \mathcal{T}(N \oplus Q)|_{\mathcal{S}}$, then there exists a run $\overline{w} \in \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}}$, such that $\langle w\rangle_N = \langle \overline{w}\rangle_N$.*

Proof. Let $w = t_1 \ldots t_n \in \mathcal{T}(N \oplus Q)|_{\mathcal{S}}$. Then, there exists $v = r_1 \ldots r_n \in \mathcal{T}(N \oplus \mathrm{mp}(N))$, such that for all $1 \leq i \leq n$:

1. $t_i \in T_N$ iff $r_i \in T_N$,
2. If $t_i \in T_N$, then $t_i = r_i$,
3. If $t_i \notin T_N$, then t_i sends (receives) a iff r_i sends (receives) a,
4. If $m^0_{N\oplus Q} \xrightarrow{t_1\ldots t_i}_{N\oplus Q} m$ and $m^0_{N\oplus \mathrm{mp}(N)} \xrightarrow{t_1\ldots t_i}_{N\oplus \mathrm{mp}(N)} m'$, then $m_{P_N} = m'_{P_N}$.

From 1)-2), we conclude $\langle w\rangle_N = \langle v\rangle_N$. If $v \in \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}}$, we are done. Consider the case where $v \notin \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}}$.

Then, there exist i, j with $0 \leq i < j \leq n$, such that $m^0 \xrightarrow{r_1 \ldots r_i}_{N\oplus\mathrm{mp}(N)} m^i \xrightarrow{r_{i+1} \ldots r_j}_{N\oplus\mathrm{mp}(N)} m^j \xrightarrow{r_{j+1} \ldots r_n} m_n$ and $m_i = m_j$. We define $z := r_{i+1} \ldots r_j$ and show $\langle z\rangle_N = 0$. By assumption, $\langle z\rangle_N = \langle z|_{T_N}\rangle_N$. Consider the cases $C_1 := z|_{T_N} = \epsilon$, $C_2 := z|_{T_N} = z$ and $C_3 :=$ otherwise.

In case C_1, we are done, all transitions in z have costs of 0. In case C_2, we apply a "pumping argument": If $z = z|_{T_N}$, then $t_1 \ldots t_i z z^k t_{j+1} \ldots t_n \in \mathcal{T}(N \oplus Q)$ for each $k \in \mathbb{N}_0$. Because $N \oplus Q$ is cost-bounded, the costs of z are 0. In case C_3, we may canonically split z into $z_1 z_2$, such that z_1 only consists of transitions of $\mathrm{mp}(N)$ and z_2 starts with a transition of N. Then, z_1 has costs of 0. We show $\langle z_2\rangle_N = 0$. If $z_1 = \epsilon$, we may apply the same argument as in case C_2. Consider now $z_1 \neq \epsilon$. If $t_1 \ldots t_i z_1 z_2 z_1 \in \mathcal{T}(N \oplus Q)$, then we may again use the pumping argument. Otherwise, firing z_1 is not required from this marking. Consequently, the costs of z_2 are 0, because Q is a cost-optimal partner.

If $v \in \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}}$, we are done. Otherwise, we apply the same argument as above, until $v \in \mathcal{T}(N \oplus \mathrm{mp}(N))|_{\mathcal{S}}$. Because z has costs of 0, the costs of w and v are equal. □

Using this lemma, we may prove the following decisive relationship between $\mathrm{mb}(N)$ and $\mathrm{dis}(N)$.

Theorem 1. *Let N be a $\mathcal{W}_b\mathcal{C}$-controllable weighted net. Then,* $\mathrm{mb}(N) \leq \mathrm{dis}(N)$.

Proof. If N is $\mathcal{W}_b\mathcal{C}$-controllable, $\mathrm{mb}(N) \in \mathbb{N}_0$. Let Q be a $\precsim$-optimal $\mathcal{W}_b$-partner of N. By Lemma 2, $\langle w\rangle_{N\oplus Q} \leq \mathrm{dis}(N)$ for each $w \in \mathcal{T}(N \oplus Q)|_{\mathcal{S}}$. Thus, by Def. 10 and Lemma 1, $\langle N \oplus Q\rangle \leq \mathrm{dis}(N)$. Applying Corollary 1, we conclude $\mathrm{mb}(N) \leq \mathrm{dis}(N)$. □

As a $\mathcal{W}_b\mathcal{C}$-controllable weighted net is $\mathcal{W}_b\mathcal{C}_{\mathrm{mb}(N)}$-controllable, we conclude:

Corollary 6. *$\mathcal{W}_b\mathcal{C}_{\mathrm{dis}(N)}$-controllability and $\mathcal{W}_b\mathcal{C}$-controllability of a weighted net N coincide.*

As shown in previous work [10], $\mathcal{W}_b\mathcal{C}_k$-controllability is decidable for any $k \in \mathbb{N}_0$. We have shown that $\mathrm{dis}(N)$ is computable. Therefore, we may conclude:

Corollary 7. *$\mathcal{W}_b\mathcal{C}$-controllability of a weighted net is decidable.*

Furthermore, we conclude the computability of the minimal cost bound:

Corollary 8. $\mathrm{mb}(N)$ *is computable.*

7 Synthesis

Our approach to synthesize cost-minimal partners consists of two steps: First, compute the minimal cost bound. Second, select the fitting synthesis procedure.

Computing the minimal cost bound. If $\mathrm{dis}(N)$ is given, it is possible to compute the minimal bound $\mathrm{mb}(N)$. First, decide $\mathcal{W}_b\mathcal{C}_{\mathrm{dis}(N)}$-controllability. If not, then $\mathrm{mb}(N) = \infty$: By Corollary 2 N is not $\mathcal{W}_b\mathcal{C}$-controllable. Otherwise, we know from Thm. 1 that $\mathrm{mb}(N) \in [0, \mathrm{dis}(N)]$. For every $i \in [0, \mathrm{dis}(N)]$ we can decide [10], whether N is $\mathcal{W}_b\mathcal{C}_i$-controllable and thus, if $i \geq \mathrm{mb}(N)$. Testing every i or using a more efficient binary search thus solves the problem. In fact, it suffices to know a natural number $K \geq \mathrm{dis}(N)$ to apply this search. Over-approximating $\mathrm{dis}(N)$ with K may reduce the computational effort, but increases the search interval. This leads to a trade-off situation regarding the computational effort.

Selecting the synthesis procedure. We propose a solution to the synthesis of $\succsim$-optimal $\mathcal{W}_b$-partners of a given weighted net N. Thereby, we reduce the problem to other problems which are known to be solvable. In the case of $\mathrm{mb}(N) = \infty$, synthesis of $\succsim$-optimal $\mathcal{W}_b$-partners reduces to synthesis of $\mathcal{W}_b$-partners [9]. In the case of $\mathrm{mb}(N) \in \mathbb{N}_0$, synthesis of a $\succsim$-optimal partners reduces to synthesis of $\mathcal{W}_b\mathcal{C}_{\mathrm{mb}(N)}$-partners [10].

8 Implementation and experimental results

Our tool Tara[1] synthesizes a cost-optimal partner for a given weighted net, applying the existing tools LoLA [18] and Wendy [19] for state space computation and $\mathcal{W}_b$-partner synthesis, respectively. We solve the problem of computing the cost discriminant $\mathrm{dis}(N)$ by first computing the most-permissive $\mathcal{W}_b$-partner $\mathrm{mp}(N)$ and then applying a simple heuristic instead of actually solving the longest path problem. In particular, for each state q, we note the costs $\overline{q}$ of the most-expensive outgoing edge. Then, we sum $\overline{q}$ for all states q, yielding $\sum \overline{q} \geq \mathrm{dis}(N)$.

The implementation is prototypical and lacks an elaborate evaluation. We present some experimental results in Table 1. To this end, we started from real world services and academic examples: We took Loan approval, Purchase order and Travel service 1 from the BPEL specification [8]. Travel service 2 is a variation of Travel service 1. We found Olive Oil Ordering [20], Beverage machine [3], Online shop 1 and Online shop 2 [7] in literature. SMTP models the SMTP protocol. 3 Philosophers and 5 Philosophers are weighted nets modeling 3 and 5 dining philosophers, respectively. Registration was provided by a consultant company. The original models were unweighted. Therefore, we added randomly determined cost functions. Some of the original models were given in BPEL, we obtained the Petri net models with BPEL2oWFN [7]. Column $|I \cup O|$ shows the size of the interface, that is, the number of input and output places. Column |state space| lists the number of reachable markings of the composition of the service with its most-permissive partner. Finally, the last column shows the computation time for synthesizing a cost-optimal partner. The rows are ordered ascending by the values in column |state space|.

[1] In order to try Tara, please visit: `http://service-technology.org/tara`

Table 1: Experimental results of TARA

Weighted net	$\lvert I \cup O \rvert$	$\lvert$state space$\rvert$	time (sec)
Beverage machine	7	37	< 0.01
Loan approval	6	43	0.04
Olive oil ordering	6	50	< 0.01
Online shop 2	8	77	0.12
Online shop 1	7	137	0.10
Travel service 1	8	192	0.06
3 Philosophers	6	499	0.06
Purchase order	10	1032	0.31
SMTP	15	1042	0.84
Travel service 2	12	1440	0.90
Registration (abstract)	6	2239	2.37
Registration	6	27372	10.20
5 Philosophers	10	43848	24.07

9 Related work

In this paper, we extend our previous work [10]. Weighted nets are inspired by *weighted automata* [21,22] over arbitrary semirings or similar algebraic structures. In this paper, we restrict ourselves to a cost model similar to the semiring known as max plus algebra. In general, we believe that our techniques can be applied for any semiring which is isomorphic to the max plus algebra. *Weighted timed automata* extend weighted automata by clocks and costs for staying inside one state. Weighted timed automata are a very complex model class (see for instance [23]) and we are not aware of partner synthesis approaches for this formalism. *Q-Automata* [24] constitute another model to capture non-functional requirements in behavioral models. Here, the focus is on composition of component models, making the setting very similar to ours. The approaches deviate in the communication model: Interaction of Q-Automata means synchronization of concurrent actions. Asynchronous communication between Q-automata may be realized by a buffer system in between. We compose open systems by means of asynchronous message exchange without a buffer system. The aim of Q-Automata is enabling analysis of compositions of open systems. To our knowledge, there does not exist a partner synthesis approach for Q-Automata.

In the area of service-oriented architectures, Oster et al. [25] present a framework to synthesize service compositions regarding non-functional preferences by applying model checking. In contrast to our approach, the composite is built from existing services. We consider the case where only one service is known beforehand. Zeng et al. [26] find an optimal composition of non-interacting atomic tasks each implemented by a web service. We consider interacting systems. In [27], the authors extend timed Petri nets with a cost model. The authors study the issue of minimal cost reachability and coverability. The formalism considers closed systems in contrast to our research of open systems.

10 Summary

In this paper, we introduced two preferences to distinguish between partners of a service based on costs, namely total cost-optimality and use case cost-optimality. We developed an approach to solve partner synthesis for total-cost optimality and the requirement of weak termination. Our approach is a reduction to partner synthesis for the requirements of k-cost-boundedness and weak termination. The difficulty lies in deciding cost-bounded controllability and the computation of the minimal cost bound. We have sketched that partner synthesis for the preference of use case cost-optimality may be reduced to the approach for total cost-optimality, if we restrict ourselves to acyclic use cases. We have presented a prototypical implementation and some first experimental results.

The cost models in this paper cover the worst case from a non-functional point of view. For future work, we plan to investigate cost models concerned with average costs. To this end, we plan to adopt algorithms from the field of mean-payoff games [28,29]. Another promising direction is to combine our techniques with timed or probabilistic models. Additionally, we would like to evaluate our approach with more realistic examples. In particular, we aim at checking the feasibility of our approach in the field of adaptation and substitutability. We believe that the runtime of the synthesis approach could be improved by developing a new synthesis algorithm for our problem class, instead of reducing our problem to another problem class.

References

1. Papazoglou, M.P.: Web Services: Principles and Technology. Pearson - Prentice Hall, Essex (2007)
2. Papazoglou, M.M.: What's in a Service? Software Architecture **4758** (2007) 11–28
3. Lohmann, N., Massuthe, P., Wolf, K.: Operating guidelines for finite-state services. In: Proceedings of the 28th international conference on Applications and theory of Petri nets and other models of concurrency. ICATPN 07, Berlin, Heidelberg, Springer-Verlag (2007) 321–341
4. Stahl, C., Massuthe, P., Bretschneider, J.: Deciding substitutability of services with operating guidelines. In: ToPNoC II. LNCS 5460, Springer (2009) 172–191
5. van der Aalst, W.M.P., Lohmann, N., Rosa, M.L.: Ensuring correctness during process configuration via partner synthesis. Inf. Syst. **37**(6) (2012) 574–592
6. Gierds, C., Mooij, A.J., Wolf, K.: Reducing adapter synthesis to controller synthesis. IEEE T. Services Computing **5**(1) (2012) 72–85
7. Lohmann, N., Massuthe, P., Stahl, C., Weinberg, D.: Analyzing interacting ws-bpel processes using flexible model generation. Data Knowl. Eng. **64**(1) (January 2008) 38–54
8. Alves, A., et al.: Web services business process execution language version 2.0 (2007)
9. Wolf, K.: Does my service have partners? LNCS ToPNoC **5460**(II) (2009) 152–171
10. Sürmeli, J.: Service discovery with cost thresholds. In ter Beek, M.H., Lohmann, N., eds.: WS-FM. Volume 7843 of Lecture Notes in Computer Science., Springer (2012) 30–48

11. Reisig, W.: Petri Nets: An Introduction. Volume 4 of Monographs in Theoretical Computer Science. An EATCS Series. Springer (1985)
12. Massuthe, P., Reisig, W., Schmidt, K.: An Operating Guideline Approach to the SOA. ANNALS OF MATHEMATICS, COMPUTING & TELEINFORMATICS **1** (2005) 35–43
13. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE **77**(4) (apr 1989) 541 –580
14. Clarke, E.M., Grumberg, O., Peled, D.: Model checking. MIT Press (2001)
15. van Glabbeek, R.: The linear time-branching time spectrum i - the semantics of concrete, sequential processes. In: Handbook of Process Algebra, chapter 1, Elsevier 3–99
16. Karp, R.: Reducibility among combinatorial problems. In Miller, R., Thatcher, J., eds.: Complexity of Computer Computations. Plenum Press, New York (1972)
17. Björklund, A., Husfeldt, T.: Finding a path of superlogarithmic length. SIAM J. Comput. **32**(6) (June 2003) 1395–1402
18. Wolf, K.: Generating petri net state spaces. In: Proceedings of the 28th international conference on Applications and theory of Petri nets and other models of concurrency. ICATPN07, Berlin, Heidelberg, Springer-Verlag (2007) 29–42
19. Lohmann, N., Weinberg, D.: Wendy: A tool to synthesize partners for services. Fundam. Inform. **113**(3-4) (2011) 295–311
20. Fisteus, J.A., Fernández, L.S., Kloos, C.D.: Applying model checking to bpel4ws business collaborations. In: Proceedings of the 2005 ACM symposium on Applied computing. SAC '05, New York, NY, USA, ACM (2005) 826–830
21. Droste, M., Kuich, W., Vogler, H.: Handbook of Weighted Automata. 1st edn. Springer Publishing Company, Incorporated (2009)
22. Buchholz, P., Kemper, P.: Model checking for a class of weighted automata. Discrete Event Dynamic Systems **20** (2010) 103–137
23. Bouyer, P., Brihaye, T., Markey, N.: Improved undecidability results on weighted timed automata. Information Processing Letters **98**(5) (2006) 188 – 194
24. Chothia, T., Kleijn, J.: Q-automata: Modelling the resource usage of concurrent components. Electronic Notes in Theoretical Computer Science **175**(2) (2007) 153 – 167 Proceedings of the Fifth International Workshop on the Foundations of Coordination Languages and Software Architectures (FOCLASA 2006).
25. Oster, Z., Ali, S., Santhanam, G., Basu, S., Roop, P.: A service composition framework based on goal-oriented requirements engineering, model checking, and qualitative preference analysis. In Liu, C., Ludwig, H., Toumani, F., Yu, Q., eds.: Service-Oriented Computing. Volume 7636 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2012) 283–297
26. Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., Chang, H.: QoS-Aware Middleware for Web Services Composition. IEEE Trans. Software Eng. **30**(5) (2004) 311–327
27. Abdulla, P., Mayr, R.: Minimal Cost Reachability/Coverability in Priced Timed Petri Nets. In de Alfaro, L., ed.: Foundations of Software Science and Computational Structures. Volume 5504 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2009) 348–363
28. Zwick, U., Paterson, M.: The complexity of mean payoff games on graphs. Theoretical Computer Science **158** (1996) 343–359
29. Brim, L., Chaloupka, J., Doyen, L., Gentilini, R., Raskin, J.F.: Faster algorithms for mean-payoff games. Form. Methods Syst. Des. **38**(2) (April 2011) 97–118