Compositional Discovery of Workflow Nets from Event Logs Using Morphisms⋆

Luca Bernardinello[2], Irina Lomazova[1], Roman Nesterov[1,2], and Lucia Pomello[2]

[1] National Research University Higher School of Economics,
20 Myasnitskaya Ulitsa, 101000 Moscow, Russia
[2] Dipartimento di Informatica, Sistemistica e Comunicazione,
Università degli Studi di Milano-Bicocca,
Viale Sarca 336 - Edificio U14, I-20126 Milano, Italia

Abstract. This paper presents a modular approach to discover process models for multi-agent systems from event logs. System event logs are filtered according to individual agent behavior. We discover workflow nets for each agent using existing process discovery algorithms. We consider asynchronous interactions among agents. Given a specification of an interaction protocol, we propose a general scheme of workflow net composition. By using morphisms, we prove that this composition preserves soundness of components. A quality evaluation shows the increase in the precision of models discovered by the proposed approach.

Keywords: Petri nets, workflow nets, multi-agent systems, morphisms, composition, process discovery

1 Introduction

Process discovery focuses on the synthesis of process models from event logs containing the observed record of an information system behavior. Process models are usually developed at the design stage of the information system life-cycle. However, the real observed behavior of the information system can eventually differ from the designed one. In some cases, designers cannot develop precise models describing all possible scenarios. That is why, process discovery is a topic of great interest at the moment.

Many process discovery algorithms have been proposed over recent years. They include Genetic algorithms, HeuristicMiner, Fuzzy miner, Inductive miner, the algorithms based on integer linear programming (ILP), and on the theory of regions (see [3] for a comprehensive review). They can be applied to solve typical problems of event logs, e.g. incompleteness and noise.

Within multi-agent systems (MAS), models obtained by existing process discovery tools can be incomprehensible since concurrent interacting agents produce rather sophisticated behavior. Process models of MAS, obtained by the

---

algorithms mentioned above, are not structured in such a way that it is possible to identify agent models as components as shown by the following example.

In order to give an intuition behind our approach, consider the system model shown in Fig. 1(a). Two interacting agents can be clearly identified as well as the way they interact by looking at the small grey places and arcs. Now take the event log produced by this system as the input to a process discovery algorithm. If we try to discover a model from this log directly, we can obtain, for instance, the results shown in Fig. 1(b) by *inductive miner* and in Fig. 1(c) by *ILP miner*. Although equivalent to the original one in their ability to reproduce the same event sequences, their structure hides the fact the original system is made of two agents, communicating through channels. More technically, the two agents correspond to two S-invariants in the original model, while they are not "separable" by means of S-invariants in the discovered models. It is true that we can improve the overall structure of models by configuring algorithm parameters. However, they still will not reflect the underlying MAS organization.

(a) an intitial system model

(b) inductive miner

(c) ILP miner
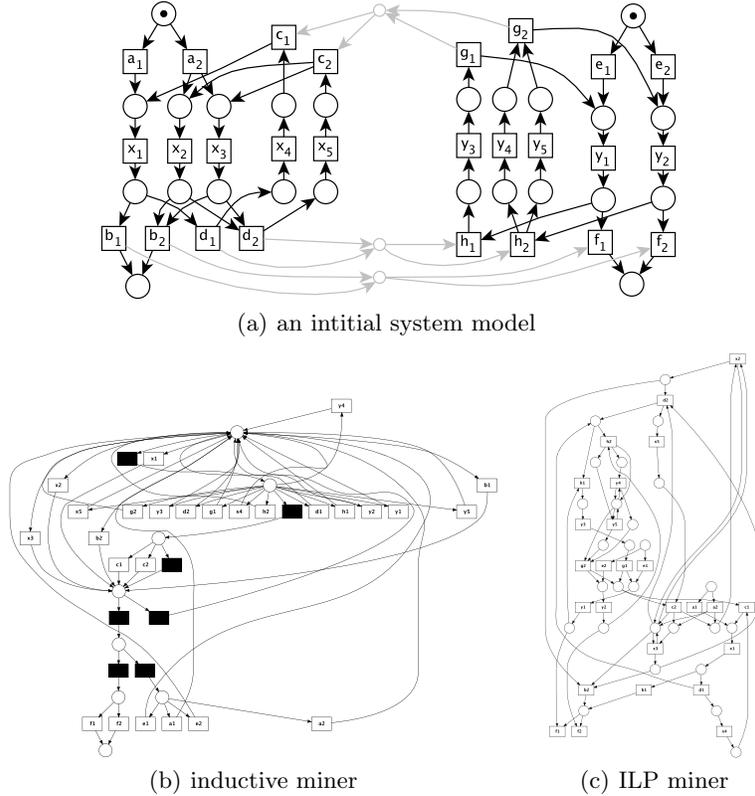
Fig. 1. Discovering process models of MAS from event logs

We propose a compositional approach to address the problem of discovering process models of MAS from event logs clearly expressing agents as components and their interactions. We assume that agents interact asynchronously

via message passing. System event logs are projected on each agent to discover component models in terms of workflow nets by using existing process discovery algorithms. By means of morphisms, we can construct abstract models of the agents and compose them by adding channels for message passing. This composition models the protocol of agent interaction. In this way, it is possible to check soundness of this simplified model and to compose the discovered agent models on the basis of this protocol. We prove that this composition preserves soundness of the protocol and components by construction. In this paper, we start constructing a *formal background* to the general compositional approach to discovering process models of multi-agent systems from event logs.

The problem of discovering structured models from logs is not new and has been studied in several works based on composition. In [18] the authors have designed a technique to discover readable models by decomposing transition systems. A special method to deal with process cancellations has been studied in [17]. Regular process behavior is composed with cancellations using reset arcs.

A rather large amount of literature has been devoted to Petri net composition (see, for example, [9, 13, 25]). In particular, regarding asynchronous communication, several approaches have been proposed. Among the others, in [14] asynchronous composition of Petri nets through channels has been studied considering preservation of various channel properties. A general approach to asynchronous composition has also been discussed in [4], where open nets have been introduced. The core problem of composition lies in preserving component properties. Open nets have been used by many researches especially focused on modeling composite services (see, for example, [15, 16]). They have proposed both structural and behavioral techniques assuring the correctness of composition.

Using morphisms is another possible way to achieve inheritance of component properties. Composition of Petri nets via morphisms has been studied in several works [4, 5, 8, 12, 22, 23, 27]. We will use α-morphisms and the related composition presented in [7]. In general, α-morphisms allow one to preserve/reflect properties checking structural and only *local* behavioral constraints. In the particular case considered here, α-morphisms preserve/reflect reachable markings and transition firings as well as preserve soundness as it will be shown in Section 3.

From a more practical point of view, many researchers have considered workflow net composition. Workflow nets (WF-nets) form a class of Petri nets used to model processes and services. Among the others, composition of WF-nets via shared resources has been studied in [21, 26] with a concern about soundness preservation. In addition, other approaches have been used for modeling and composing interacting workflow nets [20, 24]. Many works have concerned the web service composition (see the survey [11]), where the authors have stressed that there is a lack of execution engines based on Petri nets.

The paper is organized as follows. Section 2 gives preliminary definitions used in this paper as well as composition by channels and α-morphisms. Section 3 discusses properties preserved and reflected by α-morphisms, relevant to WF-nets. Section 4 introduces a modular approach to construct models of MAS from

event logs, using the same *illustrative* example shown in Fig. 1. In Section 5, we summarize the paper by discussing results and possible continuations.

Detailed proofs of some propositions stated in Section 3 are given in the extended version of this paper [6], whereas here we provide sketches of proofs.

2 Preliminaries

$\mathbb{N}$ denotes the set of non-negative integers. A^+ denotes the set of all finite non-empty sequences over A, and $A^* = A^+ \cup \{\epsilon\}$, ϵ is the empty sequence. A *multiset* m over a set S is a function $m \colon S \to \mathbb{N}$. Let m_1, m_2 be two multisets over S. Then $m_1 \subseteq m_2 \Leftrightarrow m_1(s) \leq m_2(s)$ for all $s \in S$. Also, $m' = m_1 + m_2 \Leftrightarrow m'(s) = m_1(s) + m_2(s)$, $m'' = m_1 - m_2 \Leftrightarrow m''(s) = \max(m_1(s) - m_2(s), 0)$ for all $s \in S$.

A *Petri net* is a triple $N = (P, T, F)$, where P and T are two disjoint sets of places and transitions, i.e. $P \cap T = \varnothing$, and $F \subseteq (P \times T) \cup (T \times P)$ is a flow relation, where $dom(F) \cup cod(F) = P \cup T$. We consider nets, s.t. $\forall t \in T \colon |{}^\bullet t| \geq 1$ and $|t^\bullet| \geq 1$.

Let $N = (P, T, F)$ be a Petri net, and $X = P \cup T$. The set ${}^\bullet x = \{y \in X | (y, x) \in F\}$ denotes the *preset* of $x \in X$. The set $x^\bullet = \{y \in X | (x, y) \in F\}$ denotes the *postset* of $x \in X$. The set ${}^\bullet x^\bullet = {}^\bullet x \cup x^\bullet$ denotes the *neighborhood* of $x \in X$. Let $A \subseteq X$, then ${}^\bullet A = \bigcup_{x \in A} {}^\bullet x$, $A^\bullet = \bigcup_{x \in A} x^\bullet$, ${}^\bullet A^\bullet = {}^\bullet A \cup A^\bullet$.

By $N(A)$ we denote a *subnet* of N *generated by* A, i.e. $N(A) = (P \cap A, T \cap A, F \cap (A \times A))$. Let $N(A)$ be a subnet of N generated by $A \subseteq X$. The set $^\bigcirc N(A) = \{y \in A | \exists z \in X \setminus A \colon (z, y) \in F \text{ or } {}^\bullet y = \varnothing\}$ denotes the *input* elements. The set $N(A)^\bigcirc = \{y \in A | \exists z \in X \setminus A \colon (y, z) \in F \text{ or } y^\bullet = \varnothing\}$ denotes the *output* elements.

A *marking* (state) of a Petri net $N = (P, T, F)$ is a function $m \colon P \to \mathbb{N}$. It is a multiset over a set of places P. A *marked* Petri net (N, m_0) is a Petri net together with its initial marking m_0. A marking m *enables* a transition $t \in T$, denoted $m[t\rangle$, if ${}^\bullet t \subseteq m$. The *firing* of t at m leads to a new marking $m' = m - {}^\bullet t + t^\bullet$, denoted $m[t\rangle m'$.

A sequence $w \in T^*$ is a *firing sequence* of $N = (P, T, F, m_0)$ if $w = t_1 t_2 \ldots t_n$ and $m_0[t_1\rangle m_1[t_2\rangle \ldots m_{n-1}[t_n\rangle m_n$. Then we can write $m_0[w\rangle m_n$. The set of all firing sequences of N is denoted by $FS(N)$.

A marking m of $N = (P, T, F, m_0)$ is *reachable* if $\exists w \in FS(N) \colon m_0[w\rangle m$. The set of all markings of N reachable from m is denoted by $[m\rangle$. A reachable marking is *dead* if it does not enable any transition. N is *deadlock-free* if no reachable marking is dead. N is *safe* if $\forall p \in P \, \forall m \in [m_0\rangle \colon m(p) \leq 1$. Then we will specify reachable markings as subsets of places.

A *state machine* is a connected Petri net $N = (P, T, F)$, s.t. $\forall t \in T \colon |{}^\bullet t| = |t^\bullet| = 1$. A subnet of a marked Petri net $N = (P, T, F, m_0)$ identified by a subset of places $A \subseteq P$ and its neighborhood, i.e. $N(A \cup {}^\bullet A^\bullet)$, is a *sequential component* of N if it is a state machine and has a single token in the initial marking. N is *covered* by sequential components if every place of N belongs to at least one sequential component. Then N is said to be *state machine decomposable* (SMD).

Semantics of a marked Petri net is given by its *unfolding* as defined below.

Let $N = (P, T, F)$ be a Petri net, and F^* be the reflexive transitive closure of F. Then $\forall x, y \in P \cup T$: (a) x and y are in *causal* relation, denoted $x < y$, if $(x, y) \in F^*$; (b) x and y are in *conflict* relation, denoted $x \# y$, if $\exists t_x, t_y \in T$, s.t. $t_x \neq t_y$, ${}^\bullet t_x \cap {}^\bullet t_y \neq \varnothing$, and $t_x < x$, $t_y < y$.

Definition 1. *A Petri net $O = (B, E, F)$ is an occurrence net if:*

1. $\forall b \in B \colon |{}^\bullet b| \leq 1$.
2. F^* *is a partial order.*
3. $\forall x \in B \cup E \colon \{y \in B \cup E | y < x\}$ *is finite.*
4. $\forall x, y \in B \cup E \colon x \# y \Rightarrow x \neq y$.

By definition, O is acyclic. $Min(O)$ denotes the set of minimal nodes of O w.r.t. F^*, i.e. the elements having the empty preset. We only consider nets having transitions with non-empty presets and postsets, then $Min(O) \subseteq B$.

Definition 2. *Let $N = (P, T, F, m_0)$ be a marked safe Petri net, $O = (B, E, F)$ be an occurrence net, and $\pi : B \cup E \to P \cup T$ be a map. (O, π) is a branching process of N if:*

1. $\pi(B) \subseteq P$ *and* $\pi(E) \subseteq T$.
2. π *restricted to $Min(O)$ is a bijection between $Min(O)$ and m_0.*
3. $\forall t \in T \colon \pi$ *restricted to ${}^\bullet t$ is a bijection between ${}^\bullet t$ and ${}^\bullet \pi(t)$, and similarly for $t^\bullet$ and $\pi(t)^\bullet$.*
4. $\forall t_1, t_2 \in T \colon$ *if ${}^\bullet t_1 = {}^\bullet t_2$ and $\pi(t_1) = \pi(t_2)$, then $t_1 = t_2$.*

The *unfolding* of N, denoted $\mathcal{U}(N)$, is the maximal branching process of N, s.t. any other branching process of N is isomorphic to a subnet of $\mathcal{U}(N)$ with the map π restricted to the elements of this subnet. The map associated with the unfolding is denoted u and called *folding*.

Workflow nets form a subclass of Petri nets used for process modeling. We define generalized workflow nets having an initial state m_0 and a final state m_f.

Definition 3. *A marked Petri net $N = (P, T, F, m_0, m_f)$ is a generalized workflow net (GWF-net) if and only if:*

1. $m_0 = \{s \in P \,|\, {}^\bullet s = \varnothing\}$ *and* $m_0 \neq \varnothing$.
2. $m_f = \{f \in P \,|\, f^\bullet = \varnothing\}$ *and* $m_f \neq \varnothing$.
3. $\forall x \in P \cup T \,\exists s \in m_0 \,\exists f \in m_f \colon (s, x) \in F^*$ *and* $(x, f) \in F^*$.

If $|m_0| = |m_f| = 1$, then a generalized workflow net is called just a workflow net (WF-net, for short). State machine decomposable GWF-nets are safe. The important correctness property of GWF-nets is *soundness* [2].

Definition 4. *A GWF-net $N = (P, T, F, m_0, m_f)$ is sound if and only if:*

1. $\forall m \in [m_0\rangle \colon m_f \in [m\rangle$.
2. $\forall m \in [m_0\rangle \colon m_f \subseteq m \Rightarrow m = m_f$.
3. $\forall t \in T \,\exists m \in [m_0\rangle \colon m[t\rangle$.

We consider composition of two GWF-nets by adding *channels*. This approach is similar to the one presented in [14]. A set of channel places (channels, for short) is denoted by P_c. They model asynchronous communication via message passing. Some transitions of two nets can either *send* a message by an incoming arc to a channel or *receive* a message by an outgoing arc from a channel. We assume to know exactly which transitions send/receive messages to/from which channel places. In order to simplify the notation, we will not introduce special transition labels indicating sending/receiving.

Two GWF-nets can be composed via a set of channels P_c iff any channel receiving a message from one GWF-net send it only to the other GWF-net.

Definition 5. *Let $N_i = (P_i, T_i, F_i, m_0^i, m_f^i)$ be a GWF-net for $i = 1, 2$, s.t. N_1 and N_2 are disjoint, where $P_1 \cap P_2 = \varnothing$ and $T_1 \cap T_2 = \varnothing$. Let P_c be a set of channels. A channel-composition of N_1 and N_2, denoted $N_1 \oplus_{P_c} N_2$, is a Petri net $N = (P, T, F, m_0, m_f)$, where:*

1. *$P = P_1 \cup P_2 \cup P_c$, where $P_c \cap (P_1 \cup P_2) = \varnothing$.*
2. *$T = T_1 \cup T_2$.*
3. *$F = F_1 \cup F_2 \cup F_c$, where $F_c \subseteq (P_c \times (T_1 \cup T_2)) \cup ((T_1 \cup T_2) \times P_c)$.*
4. *$m_0 = m_0^1 \cup m_0^2$ and $m_f = m_f^1 \cup m_f^2$.*
5. *$\forall p \in P_c$:*
    (a) *$({}^\bullet p \subseteq T_1$ or ${}^\bullet p \subseteq T_2)$ and $(p^\bullet \subseteq T_1$ or $p^\bullet \subseteq T_2)$,*
    (b) *${}^\bullet p \subseteq T_i \Leftrightarrow p^\bullet \subseteq T_{(i+1) \bmod 2}$,*
    (c) *${}^\bullet p \neq \varnothing$ and $p^\bullet \neq \varnothing$.*

Remark 1. The operation $\oplus_{P_c}$ is *commutative*, i.e. $N_1 \oplus_{P_c} N_2 = N_2 \oplus_{P_c} N_1$.

By the following proposition, the class of GWF-nets is closed under the channel-composition. Figure 2(a) provides an example of channel-composition of N_1 and N_2, where channels are indicated by small gray places.
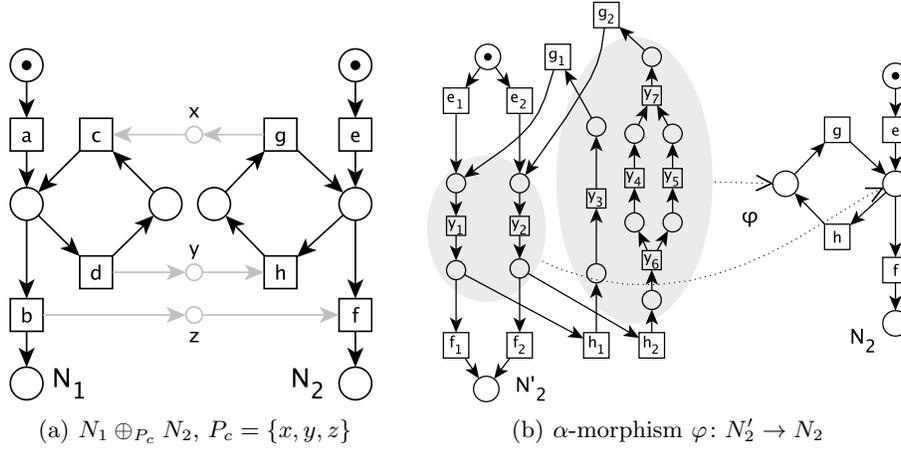
Proposition 1. *If N_1 and N_2 are GWF-nets, then $N_1 \oplus_{P_c} N_2$ is a GWF-net.*

Proof. We show that each channel lies on a path from a place in m_0^i to a place in m_f^j, where $j = (i+1) \bmod 2$ and $i = 1, 2$. Take $p \in P_c$. By Def. 5.5, ${}^\bullet p \subseteq T_i \Leftrightarrow p^\bullet \subseteq T_j$. Take $t_i \in {}^\bullet p$ and $t_j \in p^\bullet$. By Def. 3.3, $\exists s \in m_0^i$, s.t. $(s, t_i) \in F_i^*$. Then $(s, p) \in F^*$. By Def. 3.3, $\exists f \in m_f^j$, s.t. $(t_j, f) \in F_j^*$. Then $(p, f) \in F^*$. □

Further, we recall the definition of α-morphisms (see Definition 6 and Definition 7 in [7]) supporting abstraction and refinement for Petri nets. An example of α-morphism is shown in Fig. 2(b), where the refinement is given by the shaded ovals and by the transition labeling explicitly. Refinement can also require splitting transitions of the abstract model. After giving the formal definition of α-morphisms, we will provide an intuition behind them.

Definition 6. *Let $N_i = (P_i, T_i, F_i, m_0^i)$ be a marked SMD safe Petri net, $X_i = P_i \cup T_i$, $i = 1, 2$. An α-morphism from N_1 to N_2 is a total surjective map $\varphi \colon X_1 \to X_2$, also denoted $\varphi \colon N_1 \to N_2$, such that:*

1. $\varphi(P_1) = P_2$.
2. $\varphi(m_0^1) = m_0^2$.
3. $\forall t \in T_1$: *if* $\varphi(t) \in T_2$, *then* $\varphi(^{\bullet}t) = {}^{\bullet}\varphi(t)$ *and* $\varphi(t^{\bullet}) = \varphi(t)^{\bullet}$.
4. $\forall t \in T_1$: *if* $\varphi(t) \in P_2$, *then* $\varphi(^{\bullet}t^{\bullet}) = \{\varphi(t)\}$.
5. $\forall p_2 \in P_2$:
    (a) $N_1(\varphi^{-1}(p_2))$ *is an acyclic net;*
    (b) $\forall p \in {}^{\bigcirc}N_1(\varphi^{-1}(p_2))$: $\varphi(^{\bullet}p) \subseteq {}^{\bullet}p_2$, *and if* ${}^{\bullet}p_2 \neq \varnothing$, *then* ${}^{\bullet}p \neq \varnothing$;
    (c) $\forall p \in N_1(\varphi^{-1}(p_2))^{\bigcirc}$: $\varphi(p^{\bullet}) = p_2{}^{\bullet}$;
    (d) $\forall p \in P_1 \cap \varphi^{-1}(p_2)$: $p \notin {}^{\bigcirc}N_1(\varphi^{-1}(p_2)) \Rightarrow \varphi(^{\bullet}p) = p_2$ *and* $p \notin N_1(\varphi^{-1}(p_2))^{\bigcirc} \Rightarrow \varphi(p^{\bullet}) = p_2$;
    (e) $\forall p \in P_1 \cap \varphi^{-1}(p_2)$: *there is a sequential component* $N' = (P', T', F')$ *of* N_1, *s.t.* $p \in P'$ *and* $\varphi^{-1}(^{\bullet}p_2{}^{\bullet}) \subseteq T'$.

(a) $N_1 \oplus_{P_c} N_2$, $P_c = \{x, y, z\}$ (b) α-morphism $\varphi \colon N_2' \to N_2$

Fig. 2. Introductory examples

By definition, α-morphisms allow us to refine places by replacing them with subnets. Thus, if a transition is mapped on a place, then its neighborhood should be mapped on the same place (4). If a transition is mapped on a transition, it should have the corresponding neighborhood (3).

Indeed, α-morphisms are motivated by the attempt to ensure that properties of an abstract model hold in its refinement. Each output place of a subnet should have the same choices as its abstraction does (5c). However, input places do not need this constraint (5b), because the choice between them is done before, since there are no concurrent events in the neighborhood of the subnet (5e). Moreover, 5d guarantees that presets and postsets of places, internal to the subnet, are mapped to the same place as the subnet. Conditions 5a-5e together ensure the intuition behind α-morphisms. If a subnet of N_1 refines a place in N_2, then it behaves "in the same way". More precisely, by Lemma 1 of [7], (a) no input transition of the subnet is enabled whenever a token is inside the subnet; and (b) firing an output transition of the subnet empties it.

3 Properties Preserved and Reflected by α-Morphisms

In this section, we study properties preserved and reflected by α-morphisms. In [7] several properties of α-morphisms have already been studied. Here we will mention some of them and consider properties of α-morphisms for GWF-nets.

In the following propositions, we assume $N_i = (P_i, T_i, F_i, m_0^i)$ to be a marked SMD safe Petri net for $i = 1, 2$, s.t. there is an α-morphism $\varphi \colon X_1 \to X_2$, where $X_i = P_i \cup T_i$.

To begin with, α-morphisms preserve the structure of GWF-nets.

Proposition 2. *If N_1 is a GWF-net, then N_2 is a GWF-net.*

Proof. We prove that N_2 satisfies the three structural requirements of Def. 3. The conditions 1 and 2 on the initial and final state are assured by conditions 5b and 5c of α-morphisms (see Def. 6) respectively. The preservation of connectedness, i.e. condition 3, is proven by using the fact that α-morphisms are surjective maps preserving neighborhood of nodes (conditions 3 and 4 of Def. 6). $\qquad\square$

Remark 2. It follows from Proposition 2 that $\varphi(m_f^1) = m_f^2$. In the general case the converse of Proposition 2 is not true. In fact, α-morphisms do not reflect the initial state of GWF-nets properly (see Fig. 3(a)).
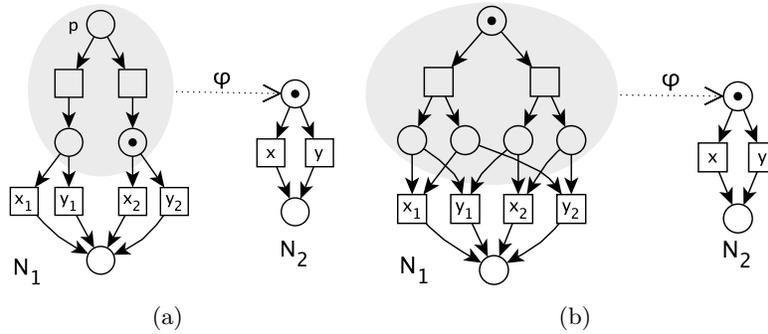
Fig. 3. An α-morphism: two examples

Recall from [7] that N_1 is *well marked* w.r.t. φ if each input place of a subnet in N_1, refining a marked place in N_2, is marked. Consider the α-morphism shown in Fig. 3(a), the token of the shaded subnet must be placed into p to make N_1 well marked w.r.t. to φ. The structure of GWF-nets is reflected under the well-markedness of N_1 (see the following Proposition). However, if N_1 is a GWF-net related to N_2 by an α-morphism, then N_1 is well marked w.r.t. φ.

Proposition 3. *If N_2 is a GWF-net and N_1 is well marked w.r.t. φ, then N_1 is a GWF-net.*

Proof. We prove that N_1 satisfies the three structural conditions of Def. 3. The condition 1 on the initial state is given by the fact that N_1 is well marked w.r.t. φ. The condition 2 on the final state is proven by contradiction using condition 4 of α-morphisms (see Def. 6). The connectedness of N_1 is proven by contrandiction as well. If a node of N_1 is not connected to at least one initial place, then there are two cases. In the first case, there should be a backward path to a place p, s.t. ${}^\bullet p = \varnothing$ and $p \notin m_0^1$ which contradicts the well-markedness of N_1 (a forward path to place p', s.t. $p'^\bullet = \varnothing$ and by Def. 3.2, $p' \in m_f^1$). In the second case, we are in a cycle in N_1. By Def. 6.5a, we deduce that there is a corresponding cycle in N_2. Knowing that N_2 is connected (by Def. 3.3) and that α-morphisms preserve neighborhood of nodes, we get a contradiction in N_1, s.t. at least one node in this cycle is connected to p, s.t. $p \in m_0^1$ (to p', s.t. $p' \in m_f^1$). $\qquad\square$

We recall in the following proposition that α-morphisms preserve reachable markings and firing of transitions.

Proposition 4 ([7]). *Let $m_1 \in [m_0^1\rangle$. Then $\varphi(m_1) \in [m_0^2\rangle$. If $m_1[t\rangle m_1'$, where $t \in T_1$, then:*

1. *$\varphi(t) \in T_2 \Rightarrow \varphi(m_1)[\varphi(t)\rangle\varphi(m_1')$.*
2. *$\varphi(t) \in P_2 \Rightarrow \varphi(m_1) = \varphi(m_1')$.*

Remark 3 ([7]). In the general case α-morphisms do not reflect reachable markings, i.e. $m_2 \in [m_0^2\rangle$ and $m_2[t_2\rangle$ do not imply that there exists $m_1 = \varphi^{-1}(m_2) \in [m_0^1\rangle$, s.t. $\forall t \in \varphi^{-1}(t_2) \colon m_1[t\rangle$.

To reflect reachable markings, we need to check local conditions as shown in [7]. For any place $p_2 \in P_2$, refined by a subnet of N_1, we construct a "local" net, denoted $S_2(p_2)$, of N_2 by taking the neighborhood transitions of p_2 plus an artificial input and output place if needed. At the same time, we construct the corresponding "local" net, denoted $S_1(p_2)$, of N_1 by taking the subnet of N_1 refining p_2, i.e. $N_1(\varphi^{-1}(p_2))$, and the transitions $\varphi^{-1}({}^\bullet p_2)$, $\varphi^{-1}(p_2{}^\bullet)$ plus an artificial input and output place if needed. The details are given in [7] (see Definition 9 there).

There is an α-morphism φ^S from $S_1(p_2)$ to $S_2(p_2)$ which is a restriction of φ on the places and transitions of $S_1(p_2)$. In the following Lemma, taking the unfolding of $S_1(p_2)$, we assure that the final marking of the subnet enables the same set of transitions which are enabled by its image. By Definition 6.5a, since $S_1(p_2)$ is acyclic, its unfolding is finite.

Lemma 1. *Let $\mathcal{U}(S_1(p_2))$ be the unfolding of $S_1(p_2)$ with the folding function u, and φ^S be an α-morphism from $S_1(p_2)$ to $S_2(p_2)$. Let N_1 be a sound GWF-net. Then, the map from $\mathcal{U}(S_1(p_2))$ to $S_2(p_2)$ obtained as $\varphi^S \circ u$ is an α-morphism.*

Proof. Since N_1 is a GWF-net, $S_1(p_2)$ is a GWF-net. By [7] (see Lemma 1 there), when a transition in $\varphi^{-1}(p_2{}^\bullet)$ fires, it empties the subnet $N_1(\varphi^{-1}(p_2))$. Then $S_1(p_2)$ is sound, and by Def. 4.3 $\forall t \in T_1 \exists m \in [m_0^1\rangle \colon m[t\rangle$. So, each transition of $S_1(p_2)$ will occur at least once. Then, the folding u is a surjective function from $\mathcal{U}(S_1(p_2))$ to $S_1(p_2)$, and $\varphi^S \circ u$ is an α-morphism from $\mathcal{U}(S_1(p_2))$ to $S_2(p_2)$. $\quad\square$

As for the main results, we obtain that under the assumption of soundness of the GWF-net N_1, α-morphisms not only preserve, but also reflect reachable markings and transition firings (see Proposition 5). Moreover, α-morphisms preserve soundness as shown in Proposition 6.

Proposition 5. *If N_1 is a sound GWF-net, then $\forall m_2 \in [m_0^2\rangle \exists m_1 \in [m_0^1\rangle$: $\varphi(m_1) = m_2$, and if $\exists t_2 \in T_2$: $m_2[t_2\rangle$, then $\forall t_1 \in \varphi^{-1}(t_2)$: $m_1[t_1\rangle$.*

Proof. Follows from Lemma 1 and from [7] (see Proposition 5 there).

Proposition 6. *If N_1 is a sound GWF-net, then N_2 is a sound GWF-net.*

Proof. We prove that N_2 satisfies the three behavioral requirements of Def. 4. The conditions 1 and 2 are assured by knowing the fact that, under the assumption of soundness of N_1, α-morphisms not only preserve reachable markings, but also reflect them (by using Prop. 4 and Prop. 5). The condition 3 is proven by the surjectivity of α-morphisms together with marking preservation. □

Remark 4. In the general case the converse of Proposition 6 is not true. Consider the example shown in Fig. 3(b), where N_2 is sound and N_1 is not sound, since transitions y_1 and y_2 are dead. Thus, α-morphisms do not reflect soundness. Note that reachable markings are also not reflected in this example. However, in the next section we will provide conditions under which soundness is reflected.

4 From Event Logs to Structured and Sound Models of Multi-Agent Systems

In this section, we present our approach to process discovery by composing individual agent models through α-morphisms. An *event log L* is a finite multiset of finite non-empty sequences (*traces*) over a set of *observable* actions $\mathcal{A}$.

We assume to have an event log L of two interacting agents. For instance, we will further work with the same event log obtained from the MAS shown in Fig. 1(a) which we have used in Section 1. We assume to know what actions are executed by which agent, $\mathcal{A} = A_1 \cup A_2$, s.t. $A_1 \cap A_2 = \varnothing$. Also, we assume to know actions corresponding to their asynchronous "message-passing" interaction.

Instead of discovering the model directly from L, we propose to filter the log according to the agent actions A_1 and A_2 producing two new logs L_1 and L_2. Traces of L_1 and L_2 contain actions done only by a corresponding agent. By using, for example, *inductive miner* [19], from L_1 and L_2 we obtain two GWF-nets N_1' and N_2' modeling the two agents. By construction, N_1' and N_2' are *well-structured* and sound, which implies that they are state machine decomposable (see Corollary 4 in [1]). Well-structured models are recursively built from blocks representing basic control flow constructs, e.g. choice, concurrency or cycle.

It is possible to compose N_1' and N_2' using the channel-composition as in Definition 5 obtaining a new GWF-net $N_1' \oplus_{P_c} N_2'$, where channels are defined according to the specification of agent interaction we have assumed to know.

However, it is obvious that $N'_1 \oplus_{P_c} N'_2$ might not be sound. In order to avoid the verification of $N'_1 \oplus_{P_c} N'_2$, we apply the following approach to get its soundness *by construction*.

We can abstract the discovered nets N'_1 and N'_2 w.r.t. interacting actions, thus producing two GWF-nets N_1 and N_2, s.t. there is an α-morphism $\varphi_i \colon N'_i \to N_i$ for $i = 1, 2$. According to Proposition 6, N_i is sound. The abstract models can also be composed via the same channels obtaining $N_1 \oplus_{P_c} N_2$. This abstract model represents the interaction protocol between the agents. Obviously, $N_1 \oplus_{P_c} N_2$ is less complex than $N'_1 \oplus_{P_c} N'_2$, and its soundness can be much easier verified.

Given $N_1 \oplus_{P_c} N_2$ and the two α-morphisms $\varphi_i \colon N'_i \to N_i$ for $i = 1, 2$, we can construct two new GWF-nets: $N'_1 \oplus_{P_c} N_2$ and $N_1 \oplus_{P_c} N'_2$, which actually represent different abstractions of the same MAS. This construction is formally defined in the following definition. We also show in Remark 5 that there is an α-morphism from $N'_1 \oplus_{P_c} N_2$ (by symmetry, from $N_1 \oplus_{P_c} N'_2$) towards $N_1 \oplus_{P_c} N_2$.

Definition 7. *Let $N_i = (P_i, T_i, F_i, m_0^i, m_f^i)$ be a GWF-net for $i = 1, 2$, and P_c be a set of channels. Let $N_1 \oplus_{P_c} N_2 = (P, T, F, m_0, m_f)$ be a channel-composition of N_1 and N_2. Let $N'_1 = (P'_1, T'_1, F'_1, m'_{01}, m'_{f1})$ be a GWF-net, s.t. there is an α-morphism $\varphi_1 \colon N'_1 \to N_1$. Construct $N'_1 \oplus_{P_c} N_2 = (P', T', F', m'_0, m'_f)$, where:*

1. *$P' = P'_1 \cup P_2 \cup P_c$.*
2. *$T' = T'_1 \cup T_2$.*
3. *$F' = F'_1 \cup F^c_{N_2} \cup F^c_{N'_1}$, where:*
    (a) *$F^c_{N_2} = F \cap [((P_c \cup P_2) \times T_2) \cup (T_2 \times (P_c \cup P_2))]$;*
    (b) *$F^c_{N'_1} \subseteq (P_c \times T'_1) \cup (T'_1 \times P_c)$.*
4. *$m'_0 = m'_{01} \cup m_0^2$ and $m'_f = m'_{f1} \cup m_f^2$.*
5. *$\forall p \in P_c, \forall t \in T_1$:*
    $((t, p) \in F \Rightarrow \varphi^{-1}(t) \times \{p\} \subseteq F^c_{N'_1})$ and $((p, t) \in F \Rightarrow \{p\} \times \varphi^{-1}(t) \subseteq F^c_{N'_1})$.
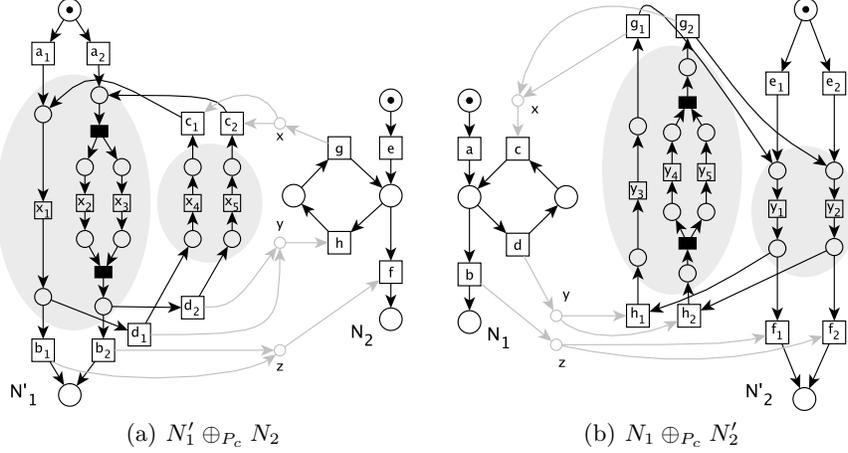
Remark 5. Let N_1, N'_1 and N_2 be GWF-nets, and $\varphi_1 \colon N'_1 \to N_1$ be an α-morphism. Then there is an α-morphism $\varphi'_1 \colon N'_1 \oplus_{P_c} N_2 \to N_1 \oplus_{P_c} N_2$. In fact, by construction, φ'_1 is given by φ_1 plus the identity mapping of places and transitions of N_2 together with the identity mapping of channel places.

Example 1. Here we consider as $N_1 \oplus_{P_c} N_2$ the GWF-net shown in Fig.2(a). We can refine it by models N'_1 or N'_2, discovered from filtered event logs L_1 and L_2 as shown in Fig. 4. The α-morphisms between the discovered models and the abstract ones are indicated by the shaded ovals and the transition labeling. The α-morphism between N'_2 and N_2 has also been shown in Fig. 2(b).

As for the main result, we will prove in Proposition 7 that an α-morphism from $N'_1 \oplus_{P_c} N_2$ (by symmetry, from $N_1 \oplus_{P_c} N'_2$) to $N_1 \oplus_{P_c} N_2$ reflects the soundness of $N_1 \oplus_{P_c} N_2$. To prove this fact, we will use the property of reachable markings of a channel-composition stated in the following Lemma.

Lemma 2. *Let $N_i = (P_i, T_i, F_i, m_0^i, m_f^i)$ a GWF-net for $i = 1, 2$, $N_1 \oplus_{P_c} N_2 = (P, T, F, m_0, m_f)$. Then, $\forall m \in [m_0\rangle \colon m = m_1 \cup m_c \cup m_2$, where $m_1 \in [m_0^1\rangle, m_2 \in [m_0^2\rangle$, and $m_c \subseteq P_c$.*

Proof. By Def. 5.4, $m_0 = m_0^1 \cup m_0^2$ and $m_f = m_f^1 \cup m_f^2$. Take $m \in [m_0\rangle$, then $\exists w \in FS(N_1 \oplus_{P_c} N_2)\colon m_0[w\rangle m$, where $w \in T^*$. By Def. 5.2, $T = T_1 \cup T_2$. Restricting w to T_1 and T_2 produces two firing sequences of N_1 and N_2 leading from m_0^1 and m_0^2 to the reachable markings m_1 and m_2 which constitute m. $\quad\square$

(a) $N_1' \oplus_{P_c} N_2$ (b) $N_1 \oplus_{P_c} N_2'$

Fig. 4. Refining $N_1 \oplus_{P_c} N_2$ with agent nets N_1' and N_2' discovered from filtered logs

Proposition 7. *Let N_1, N_1' and N_2 be sound GWF-nets, and $\varphi_1\colon N_1' \to N_1$ be an α-morphism. If $N_1 \oplus_{P_c} N_2$ is sound, then $N_1' \oplus_{P_c} N_2$ is sound.*

Proof. By Rem. 5, there is an α-morphism φ_1' from $N_1' \oplus_{P_c} N_2$ to $N_1 \oplus_{P_c} N_2$. Assume $N_1' \oplus_{P_c} N_2 = (P', T', F', m_0', m_f')$ and $N_1 \oplus_{P_c} N_2 = (P, T, F, m_0, m_f)$. We prove that $N_1' \oplus_{P_c} N_2$ satisfies the three conditions of soundness of Def. 4.

1. Take $m' \in [m_0'\rangle$. By Lemma 2, $m' = m_1' \cup m_2 \cup m_c$. By Prop. 4 for φ_1', $\varphi_1'(m') = m \in [m_0\rangle$. By Lemma 2, $m = m_1 \cup m_2 \cup m_c$, where m_2 and m_c are the same as in m' and $\varphi_1(m_1') = m_1$ (by Prop. 4 for φ_1). Since $N_1 \oplus_{P_c} N_2$ is sound, $\exists w \in FS(N_1 \oplus_{P_c} N_2)\colon m[w\rangle m_f$, where $w \in T^*$. It is possible to write $w = w_2^1 v$, where $v = \epsilon$ or $v = t_1^1 w_2^2 t_1^2 \ldots$ with $w_2^i \in T_2^*$ and $t_1^i \in T_1$, s.t. $i \geq 1$. Then w_2^i can be obviously executed on the component N_2 of $N_1' \oplus_{P_c} N_2$ as well, because φ' reflects the connection to channels (by Def. 7.5 and Rem. 5). Since N_1' is sound, φ_1 reflects reachable markings and firing of transitions (by Prop. 5) between N_1' and N_1. Thus, there is a reachable marking m_{1i}' in N_1', belonging to $\varphi_1^{-1}(m_1^i)$ in N_1, s.t. if $m_1^i[t_1^i\rangle$ in N_1, then any transition in $\varphi_1^{-1}(t_1^i)$ is enabled at m_{1i}' in N_1'. Moreover, these transitions are enabled in $N_1' \oplus_{P_c} N_2$, because φ_1' reflects connection to channels (by Def. 7.5 and Rem. 5). Hence, the sequence $w \in FS(N_1 \oplus_{P_c} N_2)$ is reflected in $N_1' \oplus_{P_c} N_2$ reaching its final marking m_f'.

2. Suppose by contradiction $\exists m' \in [m_0'\rangle\colon m' \supseteq m_f'$ and $m' \neq m_f'$. By Def. 7, $m_f' = m_{f1}' \cup m_f^2$. Then $m' = m_{f1}' \cup m_f^2 \cup m_3$. By Prop. 4 for φ_1', $\varphi_1'(m') \in [m_0\rangle$. In this way, $\varphi_1'(m') = \varphi_1'(m_{f1}') \cup \varphi_1'(m_f^2) \cup \varphi_1'(m_3) = \varphi_1(m_{f1}') \cup m_f^2 \cup m_3 = m_f^1 \cup m_f^2 \cup m_3 = m_f \cup m_3$ (by Rem. 5 and Def. 7.4). Thus, this marking strictly covers the final marking m_f of $N_1 \oplus_{P_c} N_2$ which contradicts its soundness.

3. We prove that $\forall t' \in T' \, \exists m' \in [m_0'\rangle \colon m'[t'\rangle$. By Lemma 2, $m' = m_1' \cup m_2 \cup m_c$. By Def. 7.2, $\forall t' \in T' \colon t' \in T_1'$ or $t' \in T_2$. If $t' \in T_2$, then since $N_1 \oplus_{P_c} N_2$ is sound, $\exists m \in [m_0\rangle$, s.t. $m[t'\rangle$. By Rem. 5 and by Def. 7.5, $m_2 \cup m_c$ enables t'. If $t' \in T_1'$, then there are two cases. If $\varphi_1'(t') \in P$, then t' is not connected to channels. Since N_1' is sound, $m_1' \subseteq m'$ enables t'. If $\varphi_1'(t') \in T$, then take $t \in T$, s.t. $\varphi_1'(t') = t$. Since $N_1 \oplus_{P_c} N_2$ is sound, $\exists m \in [m_0\rangle \colon m[t\rangle$. By Prop. 5, φ_1 reflects reachable markings and firings of transitions on N_1'. Moreover, φ_1' reflects connection to channels (by Def. 7.5 and Rem. 5). Then $m_1' \cup m_c$ enables t'. $\square$

Since there are two α-morphisms from $N_1 \oplus_{P_c} N_2'$ and $N_1' \oplus_{P_c} N_2$ towards $N_1 \oplus_{P_c} N_2$, we can compose them by using the composition defined in [7] (see Definition 12 there) obtaining as a result N with other two α-morphisms from N towards $N_1 \oplus_{P_c} N_2'$ and $N_1' \oplus_{P_c} N_2$, s.t. the diagram shown in Fig. 5(b) commutes.

Alternatively, by applying a similar construction as the one given in Definition 7, we can refine N_2 in $N_1' \oplus_{P_c} N_2$ by N_2' obtaining $N_1' \oplus_{P_c} N_2'$ which is isomorphic to the previously obtained composition N. Symmetrically, it is possible to refine N_1 in $N_1 \oplus_{P_c} N_2'$ by N_1' coming to the same result. According to Remark 5, there are α-morphisms from $N_1' \oplus_{P_c} N_2'$ to $N_1' \oplus_{P_c} N_2$ and to $N_1 \oplus_{P_c} N_2'$. According to Proposition 7, $N_1' \oplus_{P_c} N_2$ (by symmetry, $N_1 \oplus_{P_c} N_2'$) is sound, and then $N_1' \oplus_{P_c} N_2'$ is also sound.

Thus, we have shown that it is possible to even simultaneously refine N_1 and N_2 in the sound abstract model $N_1 \oplus_{P_c} N_2$ by sound models N_1' and N_2' obtained from filtered event logs, s.t. the result $N_1' \oplus_{P_c} N_2'$ is also sound.

Example 2. In Fig. 5(a), we show the result of composing, by means of α-morphisms, $N_1' \oplus_{P_c} N_2$ and $N_1 \oplus_{P_c} N_2'$ constructed in Example 1. This composition can also be obtained directly by the channel-composition of N_1' and N_2'. The obtained model meets the desired requirement stated in Section 1: we can identify the agents as subnets and explicitly see their asynchronous interaction.

(a) composition N isomorphic to $N_1' \oplus_{P_c} N_2'$ (b) diagram
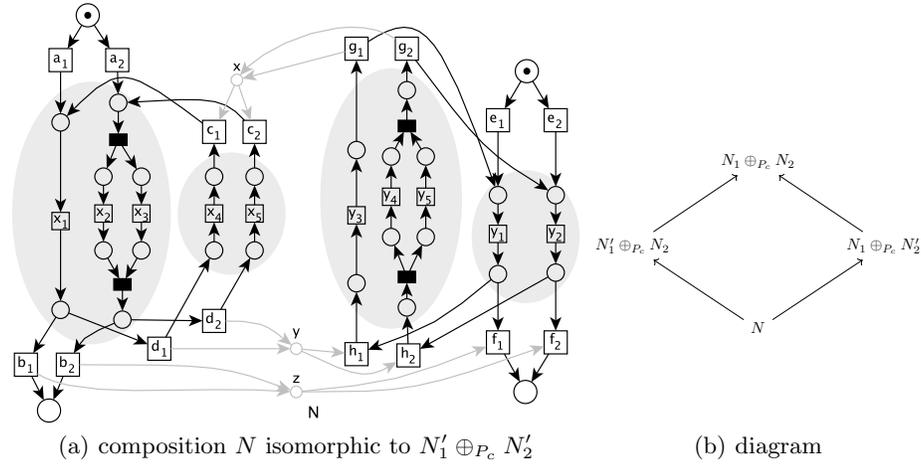
Fig. 5. Composition of $N_1' \oplus_{P_c} N_2$ and $N_1 \oplus_{P_c} N_2'$ based on α-morphisms

4.1 Comparing Quality of Presented MAS Process Models

Process models of MAS presented in Fig. 1(b) and in Fig. 1(c) are discovered directly from the log produced by simulating the model shown in Fig. 1(a). The composition shown in Fig. 5(a), in fact, is obtained from the same log. That is why, we have compared their quality using the standard process discovery metrics [10]. *Fitness* measures how accurately a model can replay traces of an initial event log. Intuitively, *precision* indicates a ratio between the behavior given by the log and the one allowed by the model. If it is low, then a model allows for too much additional behavior. Table 1 provides the result of this quality analysis.

Table 1. Fitness and precision of MAS models presented in the paper

Model	Algorithm	Discovery	Fitness	Precision
Figure 1(b)	Inductive miner	Direct	1,0000	0,1732
Figure 1(c)	ILP miner	Direct	1,0000	0,8516
Figure 5(a)	Inductive miner	Composed	1,0000	0,8690

The result of the compositional process discovery shows the increase in precision resulting from the separate analysis of agent behavior. However, we can see that the precision of the composed model is close to that of the model obtained by ILP miner. Thus, the model can be appropriate in terms of precision, but not in terms of the MAS structure.

5 Conclusion and Future Work

In this paper, we have proposed a compositional approach to discover process models of multi-agent systems from event logs. We have considered asynchronous agent interactions which are modeled by a channel-composition of two nets. We assume that events can be partitioned, s.t. system event logs are filtered according to agent behavior to discover their detailed models. In order to guarantee that their composition is sound, we have proposed to abstract agent models, by using α-morphisms, w.r.t. interacting actions thus constructing an abstract model of a communication protocol. The general algorithm of constructing α-morphisms is one of the open problems which is a subject for further research. We have proven that, when this abstract protocol is sound, the direct composition of the detailed models is also sound. In proving this fact, we have used two intermediate models corresponding to a detailed view of one agent composed with the abstract view of the other.

The obtained system model is structured in such a way that it is possible to identify agent models as components and their interactions are clearly expressed. We have compared the quality of directly obtained models with the quality of models obtained by the proposed approach. The quality of a composed model is seen to be at the appropriate level in terms of fitness and precision being the

general process discovery quality dimensions. In the future, we suggest introducing a structural indicator showing the extent to which it is easy to identify agents as parts of the model as well as their interaction.

We plan to explore more general asynchronous interactions, e.g. when the same channel can be used by several agents as well as the possibility to have both asynchronous and synchronous communication. Moreover, we would like to generalize the proposed approach to other classes of nets. Another line of research will be focused on dealing with dynamic changes. In particular, intermediate models, mentioned above, can be used to check if changes in the behavior of an agent affect the soundness of the whole system.

References

1. van der Aalst, W.M.P.: Workflow verification: Finding control-flow errors using petri-net-based techniques. In: Business Process Management: Models, Techniques, and Empirical Studies. LNCS, vol. 1806, pp. 161–183. Springer, Heidelberg (2000)
2. van der Aalst, W.M.P., van Hee, K.M., ter Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of workflow nets: classification, decidability, and analysis. Formal Aspects of Computing 23(3), 333–363 (2011)
3. Augusto, A., Conforti, R., Dumas, M., Rosa, M.L., Maggi, F.M., Marrella, A., Mecella, M., Soo, A.: Automated discovery of process models from event logs: Review and benchmark. CoRR abs/1705.02288 (2017)
4. Baldan, P., Corradini, A., Ehrig, H., Heckel, R.: Compositional modeling of reactive systems using open nets. In: Larsen, K.G., Nielsen, M. (eds.) CONCUR 2001. LNCS, vol. 2154, pp. 502–518. Springer, Heidelberg (2001)
5. Bednarczyk, M.A., Bernardinello, L., Caillaud, B., Pawłowski, W., Pomello, L.: Modular system development with pullbacks. In: ICATPN 2003. LNCS, vol. 2679, pp. 140–160. Springer, Heidelberg (2003)
6. Bernardinello, L., Lomazova, I., Nesterov, R., Pomello, L.: Compositional discovery of workflow nets from event logs using morphisms. Technical Report. Available: https://arxiv.org/pdf/1806.04254.pdf (2018)
7. Bernardinello, L., Mangioni, E., Pomello, L.: Local state refinement and composition of elementary net systems: An approach based on morphisms. In: Transactions on Petri Nets and Other Models of Concurrency VIII. LNCS, vol. 8100, pp. 48–70. Springer, Heidelberg (2013)
8. Bernardinello, L., Monticelli, E., Pomello, L.: On preserving structural and behavioural properties by composing net systems on interfaces. Fundamenta Informaticae 80(1-3), 31–47 (2007)
9. Best, E., Devillers, R., Hall, J.G.: The box calculus: A new causal algebra with multi-label communication. In: APN 1992. LNCS, vol. 609, pp. 21–69. Springer, Heidelberg (1992)
10. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: On the role of fitness, precision, generalization and simplicity in process discovery. In: OTM 2012. LNCS, vol. 7565, pp. 305–322. Springer, Heidelberg (2012)
11. Cardinale, Y., El Haddad, J., Manouvrier, M., Rukoz, M.: Web service composition based on petri nets: Review and contribution. In: RED 2012. LNCS, vol. 8194, pp. 83–122. Springer, Heidelberg (2013)

12. Fabre, E.: On the construction of pullbacks for safe petri nets. In: Donatelli, S., Thiagarajan, P.S. (eds.) ICATPN 2006. LNCS, vol. 4024, pp. 166–180. Springer, Heidelberg (2006)
13. Girault, C., Valk, R.: Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications. Springer, Heidelberg (2003)
14. Haddad, S., Hennicker, R., Møller, M.H.: Channel properties of asynchronously composed petri nets. In: ICATPN 2013. LNCS, vol. 7927, pp. 369–388. Springer, Heidelberg (2013)
15. van Hee, K.M., Mooij, A.J., Sidorova, N., van der Werf, J.M.: Soundness-preserving refinements of service compositions. In: Web Services and Formal Methods. LNCS, vol. 6551, pp. 131–145. Springer, Heidelberg (2011)
16. van Hee, K.M., Sidorova, N., van der Werf, J.M.: Construction of asynchronous communicating systems: Weak termination guaranteed! In: Software Composition. LNCS, vol. 6144, pp. 106–121. Springer, Heidelberg (2010)
17. Kalenkova, A.A., Lomazova, I.A.: Discovery of cancellation regions within process mining techniques. Fundamenta Informaticae 133(2-3), 192–209 (2014)
18. Kalenkova, A.A., Lomazova, I.A., van der Aalst, W.M.P.: Process model discovery: A method based on transition system decomposition. In: ICATPN 2014. LNCS, vol. 8489, pp. 71–90. Springer, Heidelberg (2014)
19. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs - a constructive approach. In: ICATPN 2013. LNCS, vol. 7927, pp. 311–329. Springer, Heidelberg (2013)
20. Lomazova, I.A.: Interacting workflow nets for workflow process re-engineering. Fundamenta Informaticae 101(1-2), 59–70 (2010)
21. Lomazova, I.A., Romanov, I.V.: Analyzing compatibility of services via resource conformance. Fundamenta Informaticae 128(1-2), 129–141 (2013)
22. Nielsen, M., Rozenberg, G., Thiagarajan, P.: Elementary transition systems. Theoretical Computer Science 96(1), 3–33 (1992)
23. Padberg, J., Urbášek, M.: Rule-based refinement of petri nets: A survey. In: Petri Net Technology for Communication-Based Systems: Advances in Petri Nets. LNCS, vol. 2472, pp. 161–196. Springer, Heidelberg (2003)
24. Pankratius, V., Stucky, W.: A formal foundation for workflow composition, workflow view definition, and workflow normalization based on petri nets. In: APCCM 2005, vol. 43. pp. 79–88. Australian Computer Society, Inc. (2005)
25. Reisig, W.: Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies. Springer, Heidelberg (2013)
26. Siegeris, J., Zimmermann, A.: Workflow model compositions preserving relaxed soundness. In: BPM 2006. LNCS, vol. 4102, pp. 177–192. Springer (2006)
27. Winskel, G.: Petri nets, morphisms and compositionality. In: APN 1985. LNCS, vol. 222, pp. 453–477. Springer, Heidelberg (1986)