Negotiating Inter-Organisational Processes
An approach based on Unfoldings and Workflow Nets

Michael Köhler-Bußmeier

University of Hamburg, Department for Informatics
Vogt-Kölln-Str. 30, D-22527 Hamburg
koehler@informatik.uni-hamburg.de

Abstract. In this paper we develop a negotiation and contracting framework for inter-organisational workflows. The overall aim is to compute a group-plan from a given set of individual plans, where plans are formulated in the context of a given inter-organisational workflow between the agents. A general problem is that the individual plans are not consistent, i.e. the intersection of all individual plans does not contain a complete process, which leads from the initial state of the workflow to its final state. Therefore, negotiation is needed to obtain a compromise. In this paper we develop a generic negotiation protocol and use branching processes as the elementary data structure.
The generic protocol is adapted within the specifics of our Sonar-framework. Sonar is a specification framework that defines the organisational structure of multi-agent systems. Sonar has a formal notion of teams and team-formation which is used here to instantiate the strategy parametres.
Keywords: branching processes, inter-organisational workflow, negotiation protocol, Petri net, Sonar, teamwork

1 Introduction

The paper studies the process of negotiation among self-interested agents in the context of business-to-business scenarios. These negotiation processes play a central role within distributed planning algorithms (cf. [1] for a survey). In this paper we define a generic negotiation protocol together with appropriate data types for handling partial plans.

In our context, here especially our Sonar-framework (Self-Organising Net Architecture) [2], we have several organisational constructs that frame the negotiation process. The Sonar-framework follows the *organisation centred design* paradigm [3].

One central construct are inter-organisational workflow nets [4]. The agent interaction plan – that is the object of the negotiation process – is defined by a workflow net, which generates the set of all partial plans.

A first attempt to describe the partial plans that are the data during the negotiation is to encode them as sets of action sequences (or more precisely: as computational trees). Despite this is a very simple and lean approach it also

has several drawbacks since the plans describe the interaction of two or more parties, so they are inherently concurrent. This has the consequence that there is no lean way to formalise the intersection of partial plans: The set theoretic intersection is not appropriate since due to concurrency different sequences may describe the same processes and a set theoretic intersection would remove too many elements. Analogously, also due to concurrency, there is no lean way of defining the distance of two plans.

We conclude that it is more natural to use *partially* ordered computations as they allow to define intersections of plans. Petri net unfoldings [5] (also known as branching processes [6]) are the natural candidate for partially ordered computational trees. Typically, unfoldings lead to a compact behaviour representation – when compared to sequential runs – since they abstract from any order of concurrent events. A compact representation is essential since we use unfoldings as the data format in the messages of our negotiation protocol.

Unfoldings have been used for planning before (e.g. [7]), but, to the best of our knowledge, they have not been used as the data structure in multi-agent negotiation and contracting.

The paper has the following structure: Section 2 recalls basic notions on Petri nets and branching processes. Section 3 defines a multi-party version of workflow nets. Section 4 defines partial plans as the intersection of branching processes and a negotiation protocol based on this formalisation. The work closes with a discussion of related work and an outlook.

2 Petri Nets and Branching Processes

In the following we give the basic definitions for branching processes. A branching process represents the causality as well as the conflicts between occurring events. The name *unfolding* is due to the fact that the process is unrolled by adding the transition pattern again and again at the end. An example of a branching process is given in Figure 1 ii), where the nodes are labelled with the original places and transitions from the Petri net given in Figure 1 i).

Petri Nets We recall basic definitions for Petri Nets [8]. A *Petri net* $N = (P, T, F)$ has a set of places P and a set of transitions T, which are disjoint, i.e. $P \cap T = \emptyset$, and a flow relation $F \subseteq (P \times T \cup T \times P)$. Some commonly used notations for Petri nets are ${}^\bullet y$ for the *preset* and $y^\bullet$ for the *postset* of a net element y. A Petri net is finitely branching whenever the preset and the postset are finite for each node. For notational convenience F also denotes the characteristic function of the relation F, i.e. $F : (P \times T \cup T \times P) \to \{0, 1\}$.

A P/T net $N = (P, T, F, M_0)$ consists of a finite Petri net (P, T, F) and the multiset $M_0 : P \to \mathbb{N}$ which is the initial marking.

A transition $t \in T$ of a P/T net is enabled in the marking M iff enough tokens are present: $M(p) \geq F(p, t)$ for all $p \in P$. The successor marking when firing t is $M'(p) = (M(p) - F(p, t)) + F(t, p)$. We denote the enablement of t in marking M by $M \xrightarrow{t}$. Firing of t is denoted by $M \xrightarrow{t} M'$. The notation extends to firing sequences $w \in T^*$.

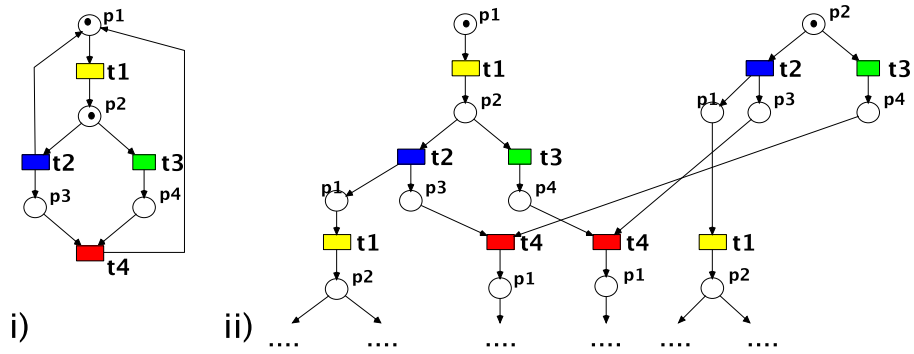

Fig. 1. i) A Petri Net; ii) An Unfolding of the Petri Net

Occurrence Nets Petri net processes [9] are a recognised alternative for describing the behaviour of Petri nets (instead of firing sequences).

For a partial order $\leq$ on the set X we define the downward closure $\downarrow x := \{y \in X \mid y \leq x\}$ and the upward closure $\uparrow x := \{y \in X \mid y \geq x\}$. The set of minimal elements of $Y \subseteq X$ wrt. $\leq$ is denoted by ${}^\circ Y$, the set of maximal elements by Y°.

Since occurrence nets $N = (B, E, F)$ allow for forward branching it is possible that two nodes x_1 and x_2 are in conflict which is the case whenever they are successors of two conflicting events $e_1, e_2 \in b^\bullet$. This is formalised by the *conflict relation* $\#$ which is a binary relation on $B \cup E$:

$$x_1 \# x_2 \iff \exists b \in B : \exists e_1, e_2 \in b^\bullet : e_1 \neq e_2 \wedge e_1 F^* x_1 \wedge e_2 F^* x_2$$

Definition 1. *A finitely branching Petri net $N = (B, E, F)$ is an* occurrence net *iff (i) the $< := F^+$ is acyclic, (ii) each node has only finitely many predecessors (i.e. $|\downarrow x| < \infty$ for all $x \in B \cup E$ holds), (iii) $\#$ is irreflexive, and (iv) each place $b \in B$ has at most one predecessor, i.e. $|{}^\bullet b| \leq 1$ holds.*

An occurrence net is a causal net *iff each place $b \in B$ has at most one successor, $|b^\bullet| \leq 1$ holds for all $b \in B$.*

It follows from $|b^\bullet| \leq 1$ that causal nets are conflict-free: $\# = \emptyset$.

The relations **li** (line) and **co** (concurrent) are defined as $\mathbf{li} := (\, < \cup <^{-1} \cup id_A)$ and $\mathbf{co} := ((B \cup E)^2 \setminus \mathbf{li} \cup id_A)$, where $id_A = \{(a, a) \mid a \in A\}$ is the identity relation. Note, that **li** and **co** are symmetric and reflexive relations. Since $\#$ is irreflexive and symmetric for occurrence nets, $\overline{\#}$ is reflexive and symmetric. Let $R \subseteq A \times A$ be a symmetric and reflexive relation. A set $K \subseteq A$ is a *clique* with respect to R iff all pairs of its elements are in the relation, i.e. for all $x, y \in K$ we have $x\ R\ y$. A maximal clique is called a *ken*. For an occurrence net a ken C with respect to $(\mathbf{li} \cap \overline{\#})$ is called a *line*, while a ken with respect to $(\mathbf{co} \cap \overline{\#})$ is called a *cut*. For causal nets lines are **li**-kens and cuts are **co**-kens (since $\# = \emptyset$). A cut C with $C \subseteq B$ is called a place-cut, a cut C with $C \subseteq E$ is called a transition-cut.

Branching Processes A *branching process* (or; unfolding) of a P/T net N is defined as an occurrence net R together with a pair of mappings $\phi = (\phi_P, \phi_T)$, where $\phi_P : B \to P$ and $\phi_T : E \to T$.

The mapping ϕ has to preserve the localities of transitions: Using the multiset extensions $\phi_P^{\oplus}$ and $\phi_T^{\oplus}$ this is expressed by the commutativity: $\phi_P^{\oplus}({}^\bullet e) = {}^\bullet\phi_T(e)$ and $\phi_P^{\oplus}(e^\bullet) = \phi_T(e)^\bullet$, which is equivalent to: ϕ_P is an isomorphism between ${}^\bullet e$ and ${}^\bullet\phi_T(e)$ as well as between $e^\bullet$ and $\phi_T(e)^\bullet$.

Definition 2. *Let $N = (P, T, F, M_0)$ be a P/T net, $R = (B, E, \lessdot)$ an occurrence net, and $\phi = (\phi_P : B \to P, \phi_T : E \to T)$ a pair of mappings. Then (R, ϕ) is a* branching process *of N if the following conditions hold:*

1. *Representation of the initial marking M_0 by the minimal elements ${}^\circ R$ of the run R: $\phi_P^{\oplus}({}^\circ R) = M_0$, which implies ${}^\circ R \subseteq B$.*
2. *Compatibility of ϕ with the localities: $\phi_P^{\oplus}({}^\bullet e) = {}^\bullet\phi_T(e)$ and $\phi_P^{\oplus}(e^\bullet) = \phi_T(e)^\bullet$.*

A branching process (R, ϕ) is a process *if R is a causal net.*

In the finite case, a process (R, ϕ) can be constructed from the possible firings, i.e. the enablement of transitions, of the net N. The construction is defined inductively for a process net, by adding transitions according to the enablement condition of the net N. The starting point is given by the initial marking, which defines a simple process without any transitions, but only a place for each token in the initial marking.

Full Branching Process/Unfolding There is one branching process, called the full branching process $\mathcal{U}(N)$, that contains all alternative events.

To simplify the notion of a maximal unfolding we assume that the initial marking M_0 is a set.

Definition 3. *Let $N = (P, T, F, M_0)$ be a P/T net. The full branching process $\mathcal{U}(N) = (R, \phi)$ is defined as follows:*

- *The occurrence net $R = (B, E, \lessdot)$ is given by the inductively defined sets $B := \bigcup_{k=0}^{\infty} B_k$ and $E := \bigcup_{k=0}^{\infty} E_k$ where*

$$\begin{aligned} B_0 &:= \{(p, \emptyset) \mid M_0(p) > 0\} \quad \textit{and} \quad E_0 := \emptyset \\ E_{n+1} &:= \{(t, U) \mid \textstyle\bigcup_{k=0}^{n-1} B_k \not\supseteq U \subseteq \bigcup_{k=0}^{n} B_k \wedge U \textit{ is conflict-free } \wedge \phi^{\oplus}(U) = {}^\bullet t\} \\ B_{n+1} &:= \{(p, \{e\}) \mid e = (t, U) \in E_{n+1} \wedge p \in t^\bullet\} \end{aligned}$$

  The flow $\lessdot$ is given in the obvious way.
- *The mappings $\phi = (\phi_P, \phi_T)$ are defined as the first projection: $\phi_P(p, X) = p$ and $\phi_T(t, U) = t$.*

Note, that the unfolding gives canonical names to places and transitions.

Note, that $\mathcal{U}(N)$ is in general infinite even for net with finite state spaces since cycles in the reachability graph of N lead to infinite long unfolding paths. Let $\mathcal{U}_n(N)$ denote the finite subnet of $\mathcal{U}(N)$ that contains the nodes $\bigcup_{k=0}^{n} B_k \cup E_k$.

For finite branching processes it is well known that if the set $C \subseteq B$ is maximal independent wrt. the relation $(\mathbf{co} \cap \overline{\#})$ (i.e. C is a place-cut) then the multiset $\phi_P^{\oplus}(C)$ is a reachable marking. Analogously, if set $C \subseteq E$ is maximal independent (i.e. a transition cut) then $\phi_T^{\oplus}(C)$ is a a maximal multiset of transitions that are concurrently enabled.

3 Inter-Organisational Workflows

In this section we define the notions for workflow nets and branching processes to define partial plans and group-plans.

In the following we define an extension of workflow nets to model multi-party interactions. Workflow Nets (WFN) [4, 10] are a well established framework for modelling business processes. A WFN N has a unique initial starting place i and a unique final place f and all nodes lie on paths between them (like in Fig. 2). The canonical initial marking M_0 of a WFN N has exactly one token on i, i.e. $M_0 := \{i\}$, the final marking is $M_f := \{f\}$.

A workflow net is *sound* [11] if (i) the final marking is reachable from each reachable marking; (ii) the final place is marked then all other places are unmarked; (iii) for each transition t there is a reachable marking enabling t.

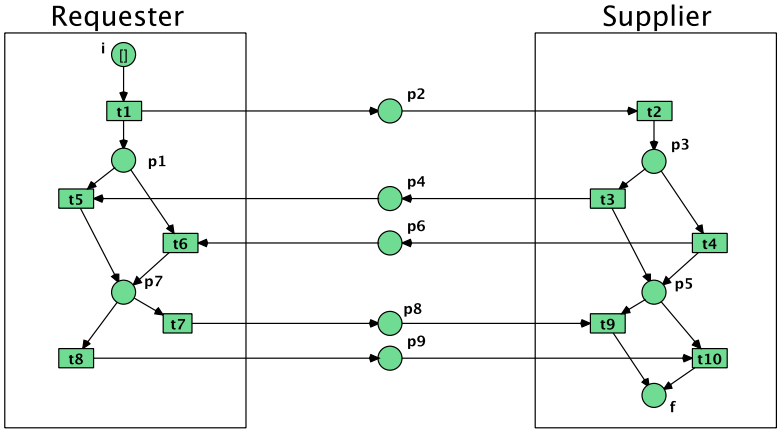

Fig. 2. An inter-organisational, multi-party Distributed Workflow Net

In a *Distributed Workflow Net* (DWFN) each transition is assigned to a role via a labelling $r : T \to Rol$, where Rol is a set of roles. For places we assume that whenever there are two transitions removing tokens from a place p then both transitions belong to the same role: $|r(p^\bullet)| \leq 1$, where we use r extended to sets of transitions. Analogously for transitions adding tokens to the same place. However, it is allowed that the role of a transition in the preset of p is different

from a transition in the postset (like e.g. p_2 in Fig. 2). In this case the place models a message exchange.

Definition 4. *A* Dwfn *is the tuple* $D = (P, T, F, r : T \to Rol)$*, where* (P, T, F) *is a WFN and for all* $p \in P$ *we have* $|r(p^\bullet)| \leq 1$ *and* $|r({}^\bullet p)| \leq 1$*.*

A Dwfn (P, T, F, r) *is sound whenever* (P, T, F) *is so.*

Figure 2 shows our running example of an Dwfn which describes the interaction between the two roles: *Requester* and *Supplier*. The boxes indicate the role labelling. The places in between the role parts model message channels.

4 Partial Plans as Branching Processes of Dwfn

Essentially a *partial plan* of a Dwfn N is an occurrence net π such that all runs lead to the final marking of the Dwfn N.

Definition 5. *The occurence net* $\pi = ((B_\pi, E_\pi, \lessdot_\pi), \phi_\pi)$ *is a* partial plan *for the* Dwfn $N = (P, T, F, r)$ *of depth* n *iff the following holds:*

1. π *is a branching process of* N*.*
2. π *is a sub-net of the unfolding* $\mathcal{U}_n(N) = ((B, E, \lessdot), \phi)$*, i.e. we have* $B_\pi \subseteq B$*,* $E_\pi \subseteq E$*,* $\lessdot_\pi = \lessdot \cap ((B_\pi \times E_\pi) \cup (E_\pi \times B_\pi))$*, and* $\phi_\pi = \phi | (B_\pi \cup E_\pi)$*.*
3. *Each maximal place-cut* C *denotes the final marking* $\{f\}$*:* $\forall C : C^\bullet = \emptyset \Longrightarrow \phi_P^{\oplus}(C) = \{f\}$

The set of all partial plans for the Dwfn N *of depth* n *is denoted* $\mathbb{PP}_n(N)$*. We define* $\mathbb{PP}(N) := \bigcup_{n \in \mathbb{N}} \mathbb{PP}_n(N)$*.*

If N is a *sound* workflow then we know that the final marking is always reachable and therefore there is some n such that $\mathbb{PP}_n(N)$ is non-empty.

The formalisation of partial plans as a partially ordered structure is an essential issue. If one has used sequential computation trees instead, then one obtains the problem that if two events a and b are concurrently enabled then one would obtain the two branches ab and ba and it would be possible for one agent to e.g. accept ab and reject ba during the local planning – despite the fact that both describe the same interaction. This problem does not arise with branching processes.

Mutual Commitments in Partial Plans The multi-party structure of the workflow net and its branching processes carries over to partial plans. Assume agent a implements a certain role r in a Dwfn. Note, that a partial plan of a does not includ events assigned to its role r, but also events assigned to other roles. This describes the circumstance, that local planning is only practical whenever agent a can rely on some assumptions about the future behaviour of other agents involved in the interaction and makes some commitments to others as well. Therefore, a team negotiation process has the function to establish *commitments*, i.e. promises some agent makes to allow the planning of others.

Commitments have a natural expression in our formalism: Assume that an agent a implements the role r_a in a Dwfn and the agent considers the partial plan π_a as relevant during his reasoning process.

– If the event e belongs not to a itself (i.e. $r(\phi(e)) \neq r_a$) then a desires a *commitment* from this other role (desired commitment).
– If the event e belongs to a itself (i.e. $r(\phi(e)) = r_a$) then a makes a *commitment* to the other roles (granted commitment).

We conclude, that branching processes of DWFNs, i.e. partial plans, are also useful to formalise the standard coordination concept: *commitment*.

4.1 Groups

In a multi-agent systems a task is implemented via teamwork, i.e. agents split the task into smaller subtasks, assign them to other agents etc. The origin of such approach lies in the contract net protocol [12]. The splitting/delegation/assigning process generates a group structure among the participating agents. We can identify a group with a labelled directed tree $(V, E, l : V \to A)$, where the nodes $v \in V$ are labelled with agents. Note, that only the agents at the leaves of this tree carry out the generated subtasks, while the inner nodes are managers, which have splitted and delegated tasks. Note, that the same agent can occur in different positions within a group, since it can delegate one task and implement another one.

Let $\mathbb{A}$ be the set of all agents and $A \subseteq \mathbb{A}$ the set of those agents that want to establish a common plan. We represents groups G as nested sets: A group has the form $G = (a, \{G_1, \ldots, G_n\})$, where a is the head of the group. Whenever $n = 0$ the group is an atomic group, i.e. a implemets a task and a is from the subset A; or $n > 0$ and the agent a delegates tasks to the groups $\{G_1, \ldots, G_n\}$ and a is from the whole set of agents $\mathbb{A}$. Note, that the head does not have to be in A, since it is not necessary that it participates in the team plan – whenever the agent delegates it has only administrative functions.

For a group G let $G^\star$ denote the set of immediate sub-groups $\{G_1, \ldots, G_n\}$. The set of all sub-groups of a group G is defined recursively:

$$\mathit{Subgroups}(G) := \{G\} \cup \bigcup\nolimits_{G' \in G^\star} \mathit{Subgroups}(G') \tag{1}$$

We denote the set of sub-groups with depth j as:

$$\mathit{Subgroups}_j(G) := \{G' \in \mathit{Subgroups}(G) \mid \mathit{depth}(G') = j\} \tag{2}$$

where $\mathit{depth}(G) := 1 + \max\{\mathit{depth}(G') \mid G' \in G^\star\}$.

4.2 Common Partial Plans among a Group of Actors

In the following, we define the *distance* of a branching process to a partial plan.

Assume we have a group G and the family $(\pi_g)_{g \in G^\star}$ of the group members' plans that are all of the same depth n. We define the common view of the group as the intersection $\bigcap \pi_g := \bigcap_{g \in G^\star} \pi_g$, where the intersection of the nets π_g is defined component-wise. In this case we encounter the problem that the intersection is not a partial plan in general. Therefore, we calculate the distance

between the intersection $(\pi_g)_{g\in G^\star}$ and a given partial plan π_0. Without loss of generality we assume that nodes of $\pi_0 = (B_0, E_0, \preccurlyeq_0)$ are given in canonical names of the unfolding $\mathcal{U}_n(N)$. In this case we calculate the distance as the number of nodes that are "missing" in the individual partial plans:

$$d((\pi_g)_{g\in G^\star}, \pi_0) := \sum\nolimits_{x\in B_0\cup E_0} |G^\star| - |\{g \in G^\star \mid x \in B_g \cup E_g\}| \tag{3}$$

We define the set of all partial plans that are at most d steps away from $(\pi_g)_{g\in G^\star}$:

$$\mathbb{PP}_n\Big(N, (\pi_g)_{g\in G^\star}, d\Big) := \Big\{\pi \in \mathbb{PP}_n(N) \mid d((\pi_g)_{g\in G^\star}, \pi) \leq d\Big\} \tag{4}$$

Obviously, these sets of partial plans are monotonous in d:

$$\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d) \quad \subseteq \quad \mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d+1) \tag{5}$$

Usually, $\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d) = \emptyset$ for small d. Especially, if the intersection $\bigcap_{g\in G^\star} \pi_g$ is not a partial plan, then $\mathbb{PP}_n(N, (\pi_g)_{g\in G}, d) = \emptyset$ for $d = 0$. If the intersection $\bigcap \pi_g$ is already a partial plan, then it is in $\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d)$ for $d = 0$ and – together with monotonicity – we obtain the following proposition.

Proposition 1. *If the intersection $\bigcap_{g\in G^\star} \pi_g$ is already a partial plan then $\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d) \neq \emptyset$ for all $d \geq 0$.*

A group-plan is a partial plan, that is very "close" to the intersection $\bigcap \pi_g$.

Definition 6. *The* minimal distance d_G *is the smallest d such that the set $\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d)$ is non-empty.*

A group-plan π_G *is a partial-plan with minimal distance, i.e. a partial plan from the set $\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d)$, where $d = d_G$.*

Note, that the definition of a group-plan π_G from $(\pi_g)_{g\in G^\star}$ does not require that all the roles of the Dwfn N are represented by agents within G, i.e. we allow negotiation within a subset of all agents. For example assume that N describes the interaction of the roles r_1, r_2, and r_3. Then we can formalise a negotiation between only two agents that implement r_1 and r_2. Their group-plan π_G is a partial-plan that is agreed only among r_1 and r_2 by mutual commitments, but not by r_3. Additionally, this group-plan formulates those commitments r_1 and r_2 desire from r_3.

When N is a sound Dwfn, we can be sure that the final marking is reachable, i.e. there exists an n such that $\mathbb{PP}_n(N) \neq \emptyset$. Since we always obtain a partial plan if we extend a sub-process till the final place f, we obtain the following existence property.

Proposition 2. *Let N be a sound* Dwfn. *There exists some d and some n such that $\mathbb{PP}_n(N, (\pi_g)_{g\in G^\star}, d) \neq \emptyset$.*

Example 1. The example in Figure 3 shows the group's view on the group's plans $(\pi_g)_{g\in G^\star}$, where $G = (H, \{(R, \emptyset), (S, \emptyset)\})$ that are constructed for the Dwfn in Figure 2. We abbreviate the roles *Requester* and *Supplier* as R and

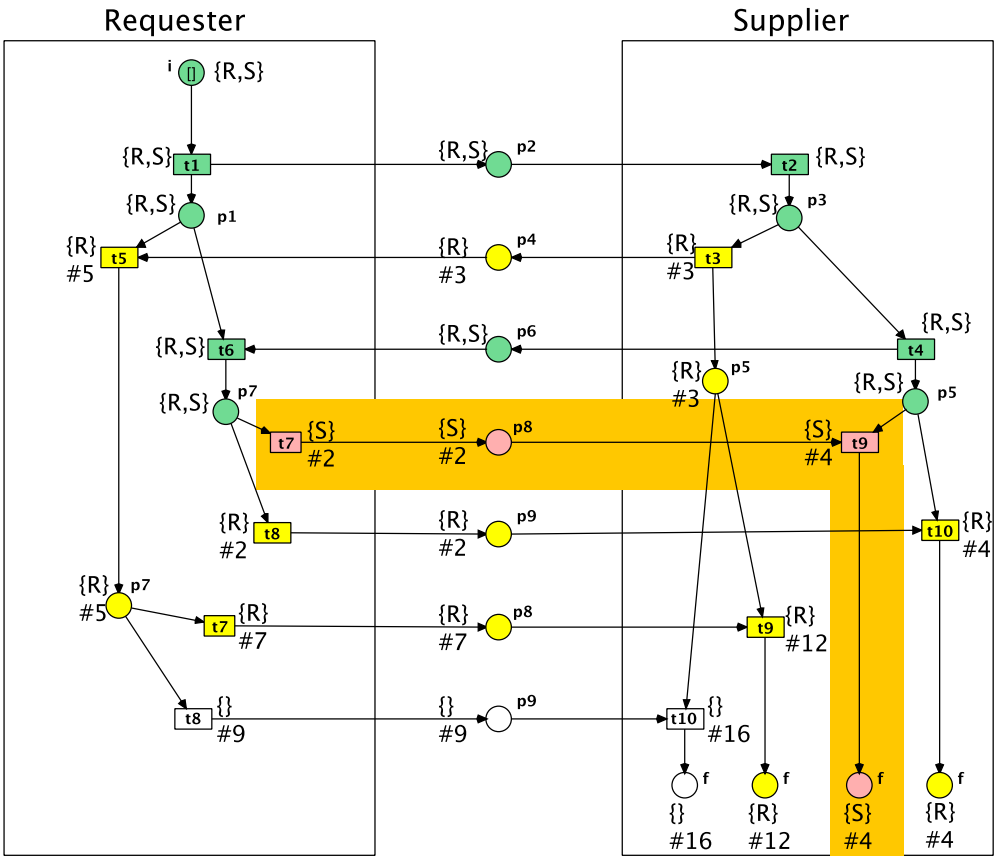

Fig. 3. The Group's Partial Plans

S, respectively. The set of agents executing tasks, i.e. implementing roles, is $\alpha_A(G) = \{R, S\}$. The agent H is an "inner" agent, that has delegated only.

To illustrate the family $(\pi_g)_{g \in G^\star}$ we label each node x in Fig. 3 with the set of agents/roles that have x in their partial plans (here: $\{\}$, $\{R\}$, $\{S\}$, or $\{R, S\}$). The inscriptions of the form $\#n$ describe the costs to add the node to the common plan.

Note, that if we restrict the plan to those nodes which are common for all agents, then we obtain the intersection $\bigcap_{g \in G^\star} \pi_g$. In this example the intersection is a branching process, but not a partial plan, since no final node f is labelled with $A = \{R, S\}$.

We oberserve that there is no partial plan within a distance less than 4. Within the distance $d = 4$ there is a partial plan which is obtained by adding the agent $R =$ *Requester* to the nodes in the shaded area. (There is another partial plan for $d = 4$ that we obtain if we add the agent *Supplier* to the nodes right/below the shaded area.)

4.3 Negotiation of a Group-Plan

The mathematical notion of the set of all partial plans within at most d steps, i.e. $\mathbb{PP}_n(N, (\pi_g)_{g \in G^\star}, d)$, is used in to define a distributed negotiation protocol. In the following we sketch our protocol underlying the negotiating process. The aim of negotiation is to construct a group-plan π_{G_0} for the whole group G_0.

Assume we have an initial value n for the the depth of the unfolding. The main idea is that we start with individual partial plans for each agent a, which is the smallest group of the form $G = (a, \emptyset)$, i.e. a sub-group of depth = 1. We assume that each individual agent a has a plan π_G^n for each planning depth n.

In the protocol we go through several rounds: The negotiation protocol enumerates all tuples (n, d) by a certain strategy σ_1, i.e. $\sigma_1 : \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ is a bijection. In each round (n, d) we increase the depth j of the sub-group G, until we generate a compromise for G_0 or there is no compromise for the current round. If there is no compromise, we step to the next round (n', d').

Since the parameter d denotes the freedom of the negotiation process to deviate from the local plans, it is likely that negotiation must fail for the first rounds, i.e. negotiation fails and the next (n, d) is enumerated. After some rounds we extend either extend the depth n of the unfolding or the distance d. Therefore, it is more likely that negotiation succeeds in this round.

Due to Prop. 2, we have $\mathbb{PP}_n(N, (\pi_g)_{g \in G^\star}, d) \neq \emptyset$ for some d and some n, which guarantees termination of the protocol.

5 Related Work and Outlook

In this paper we formalised a generic negotiation and contracting framework for inter-organisational workflows. The protocol is generic as it allows different strategies for generating sub-groups and generating compromises within a certain distance from the group's plans. As a very useful property we obtain the result that the protocol terminates for all possible choices of those strategies.

We have shown that branching processes are a very useful data structure to express the central concepts of negotiation, like *partiality of plans*, *intersection of partial plans*, and *distance of partial plans and compromises*. The protocol is used as a part of our Mulan4Sonar [14] system, which is the execution engine for our Sonar-framework [2, 15]. Mulan4Sonar is based on our multi-agent engine Mulan [16] which is based on Hornets [17] and Renew [18]. Sonar defines a formal organisation model and provides a generic infrastructure for team processes. Each Sonar-model defines the team-formation and introduces the set of possible sub-teams needed in the protocol in an lean way (cf. [15]).

Our context of negotiation in business-to-business scenarios has several aspects in common with standard algorithms in multi-agent systems, but is different at the same. For a survey on distributed problem solving and planning (cf. [19]). Oversimplifying things a little, one can say that standard planning algorithms in multi-agent systems, like the partial global planning (PGP) protocol, are based on the micro-perspective, i.e. the individual plans arise from the mental status of the agents. Contrary, business-oriented scenarios are based on the macro-perspective, i.e. we have organisational structures, like inter-organisational workflows, which underly the distributed planning. Both perspectives are closely related, but of course they give a certain bias to the approaches.

We give a short comparision of our approach and the PGP protocol [20]. The PGP protocol tries to identify individual goals that are partially consistent with the global goal, e.g. as sub-goals. Then it is tried to integrate those local plan that are designed for these sub-goals. PGP tries to remove redundant actions and reorders the plan steps. Then the communication is planned to coordinate the execution. This protocol is well suited for loosely coupled agents. On avery level, this makes PGP an a-posteriori approach, where the agents plan locally and the coordination primitives are added later on.

On the contrary, the protocol we propose here is designed for agents that act within an organisational setting following the *organisation centred design* metaphor. Here, we have a predefined organisational structure, which manifests e.g. in distributed workflow nets. At least at the abstract level of interaction the workflow clearly defines the possible interactions.

Therefore we have choosen an a-piori approach, where the coordination primitives as defined in the workflow are considered first and later on the individual preferences are added and aligned via negotiation. To the best of our knowledge, our negotiation approach in combination with unfoldings is novel in the literature.

For future work it is planned to implement another variant of the negation process, where we consider refinement of workflows also. During the teamwork it is allowed to refine the subtaks, i.e. replace a role component by a refined one. The generation generates a hierarch of workflow refinements. The negotiation should then start a negotiation process for the most abstract workflow. When a compromise is reached, it starts a negotiation process for the next workflow refinement with the constraint that the group-plan for the refined workflow has to be a refinement of the abstract group-plan. This process is iterated until an agreement is achived for the finest workflow. In this scenario we rely on on Dwfn

with an restricted set of refinement operations which carry over to the unfolding such that we can refine partial-plans during the negotiation.

References

1. Weiß, G., ed.: Multiagent systems: A modern approach to Distributed Artificial Intelligence. MIT Press (1999)
2. Köhler-Bußmeier, M., Wester-Ebbinghaus, M., Moldt, D.: A formal model for organisational structures behind process-aware information systems. Transactions on Petri Nets and Other Models of Concurrency. **5460** (2009) 98–114
3. Dignum, V., ed.: Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models. IGI Global(2009)
4. Aalst, W.v.d.: Interorganizational workflows: An approach based on message sequence charts and Petri nets. Systems Analysis - Modelling - Simulation **34** (1999) 335–367
5. Esparza, J., Heljanko, K.: Unfoldings - A Partial-Order Approach to Model Checking. Springer (2008)
6. Engelfriet, J.: Branching processes of Petri nets. Acta Informatica **28** (1991) 575–591
7. Hickmott, S.L., Rintanen, J., Thiébaux, S., White, L.B.: Planning via Petri net unfolding. In Veloso, M.M., ed.: IJCAI (2007) 1904–1911
8. Reisig, W., Rozenberg, G., eds.: Lectures on Petri Nets I: Basic Models. Volume 1491 of LNCS, Springer (1998)
9. Goltz, U., Reisig, W.: The non-sequential behaviour of Petri nets. Information and Control **57** (1983) 125–147
10. van der Aalst, W.M.P., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: Multiparty Contracts: Agreeing and Implementing Interorganizational Processes. The Computer Journal **53** (2010) 90–106
11. Aalst, W.v.d.: Verification of workflow nets. In Azeme, P., Balbo, G., eds.: ATPN'97. Volume 1248 of LNCS, Springer (1997) 407–426
12. Smith, R.G.: The contract net: A formalism for the control of distributed problem solving. IJCAI-77. (1977)
13. Grahlmann, B., Best, E.: Pep - more than a Petri net tool. In Margaria, T., Steffen, B., eds.: TACAS. Volume 1055 of LNCS, Springer (1996) 397–401
14. Köhler-Bußmeier, M., Wester-Ebbinghaus, M., Moldt, D.: Generating executable MAS-prototypes from Sonar specifications. In et al., M.D.V., ed.: COIN'10. Volume 6541 of LNAI. (2010) 21–38
15. Köhler, M.: A formal model of multi-agent organisations. Fundamenta Informaticae **79** (2007) 415 – 430
16. Cabac, L., Dörges, T., Rölke, H.: A monitoring toolset for Petri net-based agent-oriented software engineering. In Valk, R., van Hee, K.M., eds.: ATPN'08. Volume 5062 of LNCS, Springer (2008) 399–408
17. Köhler-Bußmeier, M.: Hornets: Nets within nets combined with net algebra. In Wolf, K., Franceschinis, G., eds.: ATPN'09. Volume 5606 of LNCS, Springer (2009) 243–262
18. Kummer, O., Wienberg, F., Duvigneau, et al. An extensible editor and simulation engine for Petri nets: Renew. In Cortadella, J., Reisig, W., eds.: ATPN'04. Volume 3099 of LNCS, Springer (2004) 484 – 493
19. Durfee, E.H.: Distributed problem solving and planning. [1] 425–458
20. Lesser, V. et al. Evolution of the GPGP/TAEMS Domain-Independent Coordination Framework. Autonomous Agents and Multi-Agent Systems **9** (2004) 87–143