On Negotiation as Concurrency Primitive

Javier Esparza[1] and Jörg Desel[2]

[1] Fakultät für Informatik, Technische Universität München, Germany
[2] Fakultät für Mathematik und Informatik, FernUniversität in Hagen, Germany

Abstract. We introduce negotiations, a model of concurrency close to Petri nets, with multiparty negotiation as primitive. We study the problems of soundness of negotiations and of, given a negotiation with possibly many steps, computing a *summary*, i.e., an equivalent one-step negotiation. We provide a complete set of reduction rules for sound, acyclic, weakly deterministic negotiations and show that, for deterministic negotiations, the rules compute the summary in polynomial time.

1 Introduction

Many modern distributed systems consist of components whose behavior is only partially known. Typical examples include open systems where programs (e.g. Java applets) can enter or leave, multi-agent systems, business processes, or protocols for conducting elections and auctions.

An interaction between a fixed set of components with not fully known behavior can be abstractly described as a *negotiation* in which several *parties* (the components involved in the negotiation) nondeterministically agree on an *outcome*, which results in a transformation of internal states of the parties. A more technical but less suggestive term would be a *synchronized nondeterministic choice* and, as the name suggests, these interactions can be modelled in any standard process algebra as a combination of parallel composition and nondeterministic choice, or as small Petri nets. We argue that much can be gained by studying formal models with *negotiation atoms* as concurrency primitive. In particular, we show that the negotiation point of view reveals new classes of systems with polynomial analysis algorithms.

Negotiation atoms can be combined into *distributed negotiations.* For instance, a distributed negotiation between a buyer, a seller, and a broker, consists of one or more rounds of atoms involving the buyer and the broker or the seller and the broker, followed by a final atom between the buyer and the seller. We introduce a formal model for distributed negotiations, close to a colored version of van der Aalst's *workflow nets* [1], and investigate two important analysis problems. First, just like workflow nets, distributed negotiations can be *unsound* because of deadlocks or livelocks (states from which no deadlock is reached, but the negotiation cannot be completed). The *soundness* problem consists of deciding if a given negotiation is sound. Second, a sound negotiation is equivalent to a negotiation with only one atom whose state transformation function determines the possible final internal states of all parties as a function of their initial internal states. We call this negotiation a *summary*. The *summarization problem* consists of computing a summary of a distributed negotiation. Both problems will shown to be PSPACE-hard for arbitrary negotiations, and NP-hard for acyclic ones. They can be solved by means of well-known algorithms based on the exhaustive exploration of the state space. However, this approach badly suffers from the state-explosion problem: even the analysis of distributed negotiations with a very simple structure requires exponential time.

In this paper we suggest *reduction* algorithms that avoid the construction of the state space but exhaustively apply syntactic reduction rules that simplify the system while preserving some aspects of the behavior, like absence of deadlocks. This approach has been extensively applied to Petri nets or workflow nets, but most of this work has been devoted to the liveness or soundness problems [5,16,17,12,23]. For these problems many reduction rules are known, and some sets of rules have been proved *complete* for certain classes of systems [15,10,11], meaning that they reduce all live or sound systems in the class, and only those, to a trivial system (in our case to a single negotiation atom). However, many of these rules, like the linear dependency rule of [11], cannot be applied to the summarization problem, because they preserve only the soundness property.

We present a complete set of reduction rules for the summarization problem of *acyclic* negotiations that are either *deterministic* or *weakly deterministic*. The rules are inspired by reduction rules used to transform finite automata into regular expressions by eliminating states [19]. In deterministic negotiations all involved agents are deterministic, meaning that they are never ready to engage in more than one negotiation atom. Intuitively, nondeterministic agents may be ready to engage in multiple atoms, and which one takes place is decided by the deterministic parties, which play thus the role of negotiation leaders. In weakly deterministic negotiations not every agent is deterministic, but some deterministic party is involved in every negotiation atom an agent can engage in next.

For *deterministic* negotiations we prove that a sound and acyclic negotiation can be summarized by means of a polynomial number of application of the rules, leading to a polynomial algorithm.

The paper is organized as follows. Section 2 introduces the syntax and semantics of the model. Section 3 introduces the soundness and summarization problems. Section 4 presents our reduction rules. Section 5 defines (weakly) deterministic negotiations. Section 6 proves the completeness and polynomial complexity results announced above. Finally, Section 7 presents some conclusions, open questions and related work.

This paper is an extended version of the conference paper [14].

2 Negotiations: Syntax and Semantics

We fix a finite set A of *agents* representing potential parties of negotiations. Each agent $a \in A$ has a (possibly infinite) nonempty set Q_a of *internal states*. We denote by Q_A the cartesian product $\prod_{a \in A} Q_a$. A *transformer* is a left-total relation $\tau \subseteq Q_A \times Q_A$, representing a nondeterministic state transforming function. Given $S \subseteq A$, we say that a transformer τ is an S-*transformer* if, for each $a_i \notin S$, $\Big((q_{a_1}, \ldots, q_{a_i}, \ldots, q_{a_{|A|}}), (q'_{a_1}, \ldots, q'_{a_i}, \ldots, q'_{a_{|A|}})\Big) \in \tau$ implies $q_{a_i} = q'_{a_i}$. So an S-transformer only transforms the internal states of agents in S.

Definition 1. *A* negotiation atom*, or just an* atom*, is a triple $n = (P_n, R_n, \delta_n)$, where $P_n \subseteq A$ is a nonempty set of* parties*, R_n is a finite, nonempty set of* outcomes*, and δ_n is a mapping assigning to each outcome r in R_n a P_n-transformer $\delta_n(r)$. We denote the transformer $\delta_n(r)$ by $\langle n, r\rangle$, and, if there is no confusion, by $\langle r\rangle$.*

Intuitively, if the states of the agents before a negotiation n are given by a tuple q and the outcome of the negotiation is r, then the agents change their states to q' for some $(q, q') \in \langle n, r\rangle$. Only the parties of n can change their internal states. Each outcome $r \in R_n$ is possible, independent from the previous internal states of the parties.

For a simple example, consider a negotiation atom $n_{\texttt{FD}}$ with parties `F` (Father) and `D` (teenage Daughter). The goal of the negotiation is to determine whether `D` can go to a party, and the time at which she must return home. The possible outcomes are $\{\texttt{yes}, \texttt{no}, \texttt{ask\_mother}\}$. Both sets $Q_{\texttt{F}}$ and $Q_{\texttt{D}}$ contain a state *angry* plus a state t for every time $T_1 \leq t \leq T_2$ in a given interval $[T_1, T_2]$. Initially, `F` is in state t_f and `D` in state t_d. The transformer $\delta_{n_{\texttt{FD}}}$ is given by

$$\begin{aligned}\langle \texttt{yes}\rangle &= \{((t_f, t_d), (t, t)) \;\mid\; t_f \leq t \leq t_d \vee t_d \leq t \leq t_f\}\\ \langle \texttt{no}\rangle &= \{((t_f, t_d), (\mathit{angry}, \mathit{angry}))\;\}\\ \langle \texttt{ask\_mother}\rangle &= \{((t_f, t_d), (t_f, t_d))\}\end{aligned}$$

That is, if the outcome is `yes`, then `F` and `D` agree on a time t which is not earlier and not later than both suggested times. If it is `no`, then there is a quarrel and both parties get angry. If the outcome is `ask_mother`, then the parties keep their previous times.

2.1 Combining negotiation atoms

If the outcome of the atom above is `ask_mother`, then $n_{\texttt{FD}}$ should be followed by a second atom $n_{\texttt{DM}}$ between `D` and `M` (Mother). The complete negotiation is the composition of $n_{\texttt{FD}}$ and $n_{\texttt{DM}}$, where the possible internal states of `M` are the same as those of `F` and `D`, and $n_{\texttt{DM}}$ is a copy of $n_{\texttt{FD}}$, but without the `ask_mother` outcome. In order to compose atoms, we add a *transition function* $\mathcal{X}$ that assigns to every triple (n, a, r) consisting of an atom n, a party a of n, and an outcome r of n a set $\mathcal{X}(n, a, r)$ of atoms. Intuitively, this is the set of atoms agent a is ready to engage in after the atom n, if the outcome of n is r.

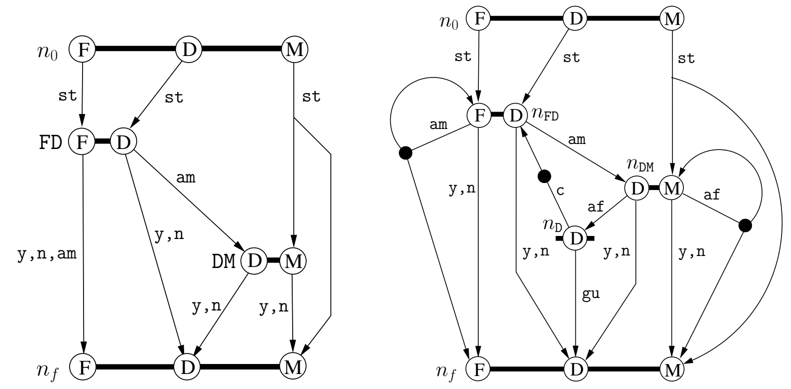

Fig. 1. An acyclic negotiation and the ping-pong negotiation.

Definition 2. *Given a finite set of atoms N, let $T(N)$ denote the set of triples (n, a, r) such that $n \in N$, $a \in P_n$, and $r \in R_n$. A* negotiation *is a tuple $\mathcal{N} = (N, n_0, n_f, \mathcal{X})$, where $n_0, n_f \in N$ are the* initial *and* final *atoms, and $\mathcal{X} \colon T(N) \to 2^N$ is the* transition function. *Further, $\mathcal{N}$ satisfies the following properties:*

(1) every agent of A participates in both n_0 and n_f;

(2) for every $(n, a, r) \in T(N)$: $\mathcal{X}(n, a, r) = \emptyset$ iff $n = n_f$.

We may have $n_0 = n_f$. Notice that n_f has, as all other atoms, at least one outcome $r \in R_{n_f}$.

Negotiations are graphically represented as shown in Figure 1. For each atom $n \in N$ we draw a black bar; for each party a of P_n we draw a white circle on the bar, called a *port*. For each $(n, a, r) \in T(N)$, we draw a hyperarc leading from the port of a in n to all the ports of a in the atoms of $\mathcal{X}(n, a, r)$, and label it by r. Figure 1 shows on the left the graphical representation of the Father-Daughter-Mother negotiation sketched above. Instead of multiple (hyper)arcs connecting the same input port to the same output ports we draw a single (hyper)arc with multiple labels. In the figure, we write `y` for `yes`, `n` for `no`, and `am` for `ask_mother`. `st` stands for `start`, the only outcome of n_0. Since n_f has no outgoing arc, the outcomes of n_f do not appear in the graphical representation.

Definition 3. *The* graph associated to a negotiation $\mathcal{N} = (N, n_0, n_f, \mathcal{X})$ *is the directed graph with vertices N and edges $\{(n, n') \in N \times N \mid \exists (n, a, r) \in T(N) \colon n' \in \mathcal{X}(n, a, r)\}$. The negotiation $\mathcal{N}$ is* acyclic *if its graph has no cycles.*

The negotiation on the left of Figure 1 is acyclic. The negotiation on the right (ignore the black dots on the arcs for the moment) is the ping-pong negotiation, well-known in every family. The $n_{\mathtt{DM}}$ atom has now an extra outcome `ask_father` (`af`), and Daughter `D` can be sent back and forth between Mother and Father. After each round, `D` “negotiates with herself” (atom $n_{\mathtt{D}}$) with possible outcomes `c` (`continue`) and `gu` (`give up`). This negotiation is cyclic because, for instance, we have $\mathcal{X}(n_{\mathtt{FD}}, \mathtt{D}, \mathtt{am}) = \{n_{\mathtt{DM}}\}$, $\mathcal{X}(n_{\mathtt{DM}}, \mathtt{D}, \mathtt{af}) = \{n_{\mathtt{D}}\}$, and $\mathcal{X}(n_{\mathtt{D}}, \mathtt{D}, \mathtt{c}) = \{n_{\mathtt{FD}}\}$.

2.2 Semantics

A *marking* of a negotiation $\mathcal{N} = (N, n_0, n_f, \mathcal{X})$ is a mapping $\boldsymbol{x}\colon A \to 2^N$. Intuitively, $\boldsymbol{x}(a)$ is the set of atoms that agent a is currently ready to engage in next. The *initial* and *final* markings, denoted by $\boldsymbol{x}_0$ and $\boldsymbol{x}_f$ respectively, are given by $\boldsymbol{x}_0(a) = \{n_0\}$ and $\boldsymbol{x}_f(a) = \emptyset$ for every $a \in A$.

A marking $\boldsymbol{x}$ *enables* an atom n if $n \in \boldsymbol{x}(a)$ for every $a \in P_n$, i.e., if every agent that parties in n is currently ready to engage in it. If $\boldsymbol{x}$ enables n, then n can take place and its parties agree on an outcome r; we say that (n, r) *occurs*. The occurrence of (n, r) produces a next marking $\boldsymbol{x}'$ given by $\boldsymbol{x}'(a) = \mathcal{X}(n, a, r)$ for every $a \in P_n$, and $\boldsymbol{x}'(a) = \boldsymbol{x}(a)$ for every $a \in A \setminus P_n$. We write $\boldsymbol{x} \xrightarrow{(n,r)} \boldsymbol{x}'$ to denote this, and call it a *small step*.

By this definition, $\boldsymbol{x}(a)$ is always either $\{n_0\}$ or equals $\mathcal{X}(n, a, r)$ for some atom n and outcome r. The marking $\boldsymbol{x}_f$ can only be reached by the occurrence of (n_f, r) (r being a possible outcome of n_f), and it does not enable any atom. Any other marking that does not enable any atom is considered a *deadlock*.

Reachable markings can be graphically represented by placing tokens (black dots) on the forking points of the hyperarcs (or in the middle of an arc). Thus, both the initial marking and the final marking are represented by no tokens, and all other reachable markings are represented by exactly one token per agent.

Figure 1 shows on the right the marking in which Father (`F`) is ready to engage in the atoms $n_{\texttt{FD}}$ and n_f, Daughter (`D`) is only ready to engage in $n_{\texttt{FD}}$, and Mother (`M`) is ready to engage in both $n_{\texttt{DM}}$ and n_f.

We write $\boldsymbol{x}_1 \xrightarrow{\sigma}$ to denote that there is a sequence

$$\boldsymbol{x}_1 \xrightarrow{(n_1,r_1)} \boldsymbol{x}_2 \xrightarrow{(n_2,r_2)} \cdots \xrightarrow{(n_{k-1},r_{k-1})} \boldsymbol{x}_k \xrightarrow{(n_k,r_k)} \boldsymbol{x}_{k+1} \cdots$$

of small steps such that $\sigma = (n_1, r_1) \ldots (n_k, r_k) \ldots$. If $\boldsymbol{x}_1 \xrightarrow{\sigma}$, then σ is an *occurrence sequence* from the marking $\boldsymbol{x}_1$, and $\boldsymbol{x}_1$ enables σ. If σ is finite, then we write $\boldsymbol{x}_1 \xrightarrow{\sigma} \boldsymbol{x}_{k+1}$ and say that $\boldsymbol{x}_{k+1}$ is *reachable* from $\boldsymbol{x}_1$. If $\boldsymbol{x}_1$ is the initial marking then we call σ *initial occurrence sequence.* If moreover $\boldsymbol{x}_{k+1}$ is the final marking, then σ is a *large step.*

3 Analysis Problems

Correct negotiations should be deadlock-free and, in principle, they should not have infinite occurrence sequences either. However, requiring the latter in our negotiation model is too strong, because infinite occurrence sequences may be excluded by fairness constraints. Following [1,2], we introduce a notion of partial correctness independent of termination:

Definition 4. *A negotiation is* sound *if* (a) *every atom is enabled at some reachable marking, and* (b) *every occurrence sequence from the initial marking is either a large step or can be extended to a large step.*

The negotiations of Figure 1 are sound. However, if we set $\mathcal{X}(n_0, \mathtt{M}, \mathtt{st}) = \{n_{\mathtt{DM}}\}$ instead of $\mathcal{X}(n_0, \mathtt{M}, \mathtt{st}) = \{n_{\mathtt{DM}}, n_f\}$, then the occurrence sequence $(n_0, \mathtt{st})(n_{\mathtt{FD}}, \mathtt{yes})$ leads to a deadlock.

The *final outcomes* of a negotiation are the outcomes of its final atom. Intuitively, two sound negotiations over the same agents are equivalent if they have the same final outcomes, and for each final outcome they transform the same initial states into the same final states.

Definition 5. *Given a negotiation $\mathcal{N} = (N, n_0, n_f, \mathcal{X})$, we attach to each outcome r of n_f a* summary transformer *$\langle \mathcal{N}, r \rangle$ as follows. Let E_r be the set of large steps of $\mathcal{N}$ that end with (n_f, r). We define $\langle \mathcal{N}, r \rangle = \bigcup_{\sigma \in E_r} \langle \sigma \rangle$, where for $\sigma = (n_1, r_1)\,(n_2, r_2) \ldots (n_k, r_k)$ we define $\langle \sigma \rangle = \langle n_1, r_1 \rangle\, \langle n_2, r_2 \rangle \cdots \langle n_k, r_k \rangle$ (remember that each $\langle n_i, r_i \rangle$ is a relation on Q_A; concatenation is the usual concatenation of relations).*

$\langle \mathcal{N}, r \rangle (q_0)$ is the set of possible final states of the agents after the negotiation concludes with outcome r, if their initial states are given by q_0.

Definition 6. *Two negotiations $\mathcal{N}_1$ and $\mathcal{N}_2$ over the same set of agents are* equivalent*, denoted by $\mathcal{N}_1 \equiv \mathcal{N}_2$, if they are either both unsound, or if they are both sound, have the same final outcomes, and $\langle \mathcal{N}_1, r \rangle = \langle \mathcal{N}_2, r \rangle$ for every final outcome r. If $\mathcal{N}_1 \equiv \mathcal{N}_2$ and $\mathcal{N}_2$ consists of a single atom, then $\mathcal{N}_2$ is a* summary *of $\mathcal{N}_1$.*

Notice that, according to this definition, all unsound negotiations are equivalent. This amounts to considering soundness essential for a negotiation: if it fails, we do not care about the rest.

3.1 Deciding soundness

The *reachability graph* of a negotiation $\mathcal{N}$ has all markings reachable from $\boldsymbol{x}_0$ as vertices, and an arc leading from $\boldsymbol{x}$ to $\boldsymbol{x}'$ whenever $\boldsymbol{x} \xrightarrow{(n,r)} \boldsymbol{x}'$.

The soundness problem consists of deciding if a given negotiation is sound. It can be solved by (1) computing the reachability graph of $\mathcal{N}$ and (2a) checking that every atom appears at some arc, and (2b) that, for every reachable marking $\boldsymbol{x}$, there is an occurrence sequence σ such that $\boldsymbol{x} \xrightarrow{\sigma} \boldsymbol{x}_f$.

Step (1) needs exponential time, and steps (2a) and (2b) are polynomial in the size of the reachability graph. So the algorithm is single exponential in the number of atoms. This cannot be easily avoided, because the problem is PSPACE-complete, and NP-complete for acyclic negotiations.

Recall that a language L is in the class DP if there exist languages L_1, L_2 in NP and co-NP, respectively, such that $L = L_1 \cap L_2$ [21].

Theorem 1. *The soundness problem is PSPACE-complete. For acyclic negotiations, the problem is co-NP-hard and in DP (and so at level Δ_2^P of the polynomial hierarchy).*

Proof. The soundness problem is in PSPACE.
Membership in PSPACE can be proved by observing that the soundness problem can be formulated in CTL, and then applying the PSPACE algorithm for CTL and 1-safe Petri nets of [13]. This algorithm only assumes that, given a marking, one can compute a successor marking in polynomial time, which is the case both for Petri nets and for negotiations. However, since we only need a very special case of the CTL algorithm, we provide a self-contained proof.

We show that both conditions for soundness can be checked in nondeterministic polynomial space. The result then follows from Savitch's theorem (NPSPACE=PSPACE).

The first condition is: every atom is enabled at some reachable marking. For this we consider each atom n in turn, and guess step by step an occurrence sequence ending with an occurrence of n. This only requires to store the marking reached by the sequence executed so far.

The second condition is: every occurrence sequence from the initial marking is either a large step or can be extended to a large step. This case is a bit more involved. Let S denote the problem of checking this second condition. We prove $S \in$ PSPACE.

(1) The following problem is in PSPACE: given some marking $\boldsymbol{x}$, check that no occurrence sequence starting at $\boldsymbol{x}$ ends with the final atom.
Let us call this problem NO-OCC. We have $\overline{\text{NO-OCC}} \in$ NPSPACE, because we can nondeterministically guess an occurrence sequence starting at $\boldsymbol{x}$ that ends with the final atom (we guess one step at a time). Since NPSPACE=PSPACE=co-PSPACE, we get NO-OCC $\in$ PSPACE.

(2) $\overline{S} \in$ NPSPACE.
$\overline{S}$ consists of checking the existence of a sequence σ, firable from the initial marking, that is neither a large step nor can be extended to it. For this we guess a sequence σ step by step that does not end with the final atom. Then we consider the marking $\boldsymbol{x}$ reached by the occurrence of σ. Clearly, we have $\sigma \in \overline{S}$ iff $\boldsymbol{x} \in$ NO-OCC. So it suffices to apply our deterministic polynomial-space algorithm for NO-OCC (see (1)).

(3) $S \in PSPACE$.
Follows from (2) and NPSPACE=PSPACE=co-PSPACE.

The soundness problem is PSPACE-hard.
For PSPACE-hardness, we reduce the problem of deciding if a deterministic linearly bounded automaton (DLBA) recognizes an input to the soundness problem. Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DLBA, and consider an input $w = a_1 \dots a_k \in \Sigma^*$. The construction is very similar to that of [13] for proving PSPACE -hardness of the reachability problem for 1-safe Petri nets, and so we do not provide all details. The negotiation $\mathcal{N}_A$ has a control agent C, a head agent H, and a cell agent T_i for every tape cell (i.e., $1 \leq i \leq k$). All agents have only one internal state, i.e., the internal states are irrelevant. The negotiation has an atom $n[q, h, a]$ (with

only one outcome) for every state q, every head position $1 \leq h \leq k$, and every $a \in \Sigma$, plus an initial atom n_0 and a final atom n_f. The parties of $n[q, h, a]$ are C, H, and T_h. The transition function $\mathcal{X}$ is defined so that $\mathcal{N}_A$ simulates A in the following sense: if A is currently in state q with the head at position h, and the contents of the tape are $b_1 \dots b_k$, then the current marking $\boldsymbol{x}$ of the negotiation satisfies the following properties:

– if $q \neq q_f$, then $\boldsymbol{x}(C)$ is the set of atoms $n[h', q', a]$ such that $q' = q$, and both h' and a are arbitrary; if $q = q_f$, then $\boldsymbol{x}(C) = \{n_f\}$;
– $\boldsymbol{x}(H)$ is the set of atoms $n[h', q', a]$ such that $h' = h$ and q', a are arbitrary, plus the final atom;
– $\boldsymbol{x}(T_i)$ is the set of atoms $n[h', q', a]$ such that $h' = i$, q' is arbitrary, and $a = b_i$, plus the final atom.

(Intuitively, agent C is only ready to engage in atoms for the state q; agent H is only ready to engage in atoms for the position h; and T_h is only ready to engage in atoms for the letter b_h.) These properties guarantee that the only atom enabled by $\boldsymbol{x}$ is $n[h, q, b_h]$ if $q \neq q_f$, or the atom n_f if $q = q_f$. So the negotiation $\mathcal{N}_A$ has only one initial occurrence sequence, which corresponds to the execution of A on w.

It remains to define $\mathcal{X}$ so that it satisfies these properties. For the initial atom we take (recall that the input of the DLBA A is the word $w = a_1 \dots a_k$):

$$\begin{aligned}
\mathcal{X}(n_0, C, \mathit{step}) &= \{n[h', q_0, a'] \mid 1 \leq h' \leq k, a' \in \Sigma\} \\
\mathcal{X}(n_0, H, \mathit{step}) &= \{n[1, q', a'] \mid q' \in Q, a' \in \Sigma\} \\
\mathcal{X}(n_0, T_i, \mathit{step}) &= \{n[i, q_0, a_i]\}
\end{aligned}$$

For the transition function of an atom $n[q, h, a]$ we must consider the three possible cases of the transition relation (head moves to the right, to the left, or stays put). We only deal with the case in which the machine moves to the right, the others being analogous. Assume $\delta(q, a) = (\hat{q}, \hat{a}, R)$. Then we take

$$\begin{aligned}
\mathcal{X}(n[h, q, a], C, \mathit{step}) &= \{n[h', \hat{q}, a'] \mid 1 \leq h' \leq k, a' \in \Sigma\} \\
\mathcal{X}(n[h, q, a], H, \mathit{step}) &= \{n[h+1, q', a'] \mid q' \in Q, a' \in \Sigma\} \\
\mathcal{X}(n[h, q, a], T_h, \mathit{step}) &= \{n[h, q', \hat{a}] \mid q' \in Q\}
\end{aligned}$$

Since A is deterministic, $\mathcal{N}_A$ has only one maximal occurrence sequence, which is a large step iff A accepts. So $\mathcal{N}_A$ is sound iff A accepts.

The soundness problem for acyclic negotiations is in DP.
We first observe that no occurrence of an acyclic negotiation contains an atom more than once (loosely speaking, once the tokens of the parties of the atom have "passed" beyond it, they cannot return). It follows that the length of an occurrence sequences is at most equal to the number of atoms. To check soundness we must check that (1) every atom can be enabled, and that (2) every occurrence sequence can be extended to a large step. Checking (1) can be done by guessing in polynomial time enabling sequences for all atoms, and so (1) is

in NP. Checking the negation of (2) can be done by guessing in polynomial time an occurrence sequence that cannot be extended to a large step, and so (2) is in coNP. So the conjunction of (1) and (2) is in DP.

The soundness problem for acyclic negotiations is co-NP-hard.
We reduce 3-CNF-SAT to non-hardness. Given a boolean formula ϕ with variables x_i, $1 \leq i \leq n$ and clauses c_j, $1 \leq j \leq m$, we construct a negotiation $\mathcal{N}_\phi$ with an agent X_i for each x_i, and an agent J (for judge). W.l.o.g. we assume that no clause of ϕ is a tautology. For each variable x_i, $\mathcal{N}_\phi$ has an atom $\mathit{Set\_x_i}$ with X_i as only party and outcomes `true` and `false`. For each clause c_j, the negotiation $\mathcal{N}_\phi$ has an atom False_j whose parties are the variables appearing in c_j and the judge J. The atom has only one outcome `false`.

After the initial atom, agent X_i engages in $\mathit{Set\_x_i}$ and sets x_i to a value $b \in \{\texttt{true}, \texttt{false}\}$ by choosing the appropriate outcome. After that, X_i is ready to engage in the atoms False_j satisfying the following condition: the clause c_j is *not* made true by setting x_i to b; moreover, it is also ready to engage in the final atom. As a consequence, False_j becomes enabled iff the assignment chosen by the X_i's makes c_j false. Finally, after the occurrence of a False_j, its parties are only ready to engage in the final atom.

After the initial atom, the judge J is ready to engage in all atoms False_j, and then, if any of them occurs, in the final atom.

We argue that $\mathcal{N}_\phi$ is sound iff ϕ is satisfiable. Notice first that, since by assumption no clause is a tautology, every False_j atom is enabled by some occurrence sequence. So all atoms but perhaps the final atom can be enabled by some sequence. So $\mathcal{N}_i$ is sound iff every occurrence sequence can be extended to a large step, and therefore it suffices to show that ϕ is satisfiable iff every occurrence sequence of $\mathcal{N}_\phi$ can be extended to a large step.

If ϕ is unsatisfiable then, whatever the assignment determined by the outcomes of the $\mathit{Set\_x_i}$'s, some clause is false, and so at least one of the False_j atoms is enabled. After some False_j occurs, the final atom becomes enabled, and so the computation can be extended to a large step.

If ϕ is satisfiable, then consider an initial occurrence sequence in which the atoms $\mathit{Set\_x_i}$ occur, and they choose the outcomes corresponding to a satisfying assignment. Then none of the False_j atoms become enabled. Moreover the final atom is not enabled either, because the judge J is not ready to engage in it. So the occurrence sequence cannot be extended to a large step. $\square$

3.2 A summarization algorithm

The *summarization problem* consists of computing a summary of a given negotiation, if it is sound. A straightforward solution follows these steps:

(1) Compute the reachability graph of $\mathcal{N}$. Interpret it as a weighted finite automaton over the alphabet of transformers $\langle n, r \rangle$, with $\boldsymbol{x}_0$ as initial state, and $\boldsymbol{x}_f$ as final state.

(2) Compute the sum over all paths σ leading from $\boldsymbol{x}_0$ to $\boldsymbol{x}_f$ of the transformers $\langle\sigma\rangle$. We recall a well-known algorithm for this based on state elimination (see e.g. [19]). The algorithm proceeds in phases consisting of the following three steps:

(2.1) Exhaustively replace steps $\boldsymbol{x} \xrightarrow{f_1} \boldsymbol{x}'$, $\boldsymbol{x} \xrightarrow{f_2} \boldsymbol{x}'$ by one step $\boldsymbol{x} \xrightarrow{f_1+f_2} \boldsymbol{x}'$.

(2.2) Pick a state $\boldsymbol{x}$ different from $\boldsymbol{x}_0$ and $\boldsymbol{x}_f$. If there is a self-loop $\boldsymbol{x} \xrightarrow{f} \boldsymbol{x}$, replace all steps $\boldsymbol{x} \xrightarrow{g} \boldsymbol{x}'$, where $\boldsymbol{x}' \neq \boldsymbol{x}$, by the step $\boldsymbol{x} \xrightarrow{g^*f} \boldsymbol{x}'$, and then remove the self-loop.

(2.3) For every two steps $\boldsymbol{x}_1 \xrightarrow{f_1} \boldsymbol{x}$ and $\boldsymbol{x} \xrightarrow{f_2} \boldsymbol{x}_2$, add a step $\boldsymbol{x} \xrightarrow{f_1 f_2} \boldsymbol{x}_2$ and remove state $\boldsymbol{x}$ together with its incident steps.

Clearly, step (1) takes exponential time in the number of atoms. Steps (2.1)-(2.3) can be seen as *reduction rules* that replace an automaton by a smaller one with the same sum over all paths. In the next section we provide similar rules, but at the syntactic level, i.e., rules which act directly on the negotiation diagram, and *not* on the reachability graph. This avoids the construction of the reachability graph. Two of the three rules are straightforward generalizations of (2.1) and (2.3) above, while the third allows us to remove certain useless arcs from a negotiation.

4 Reduction Rules

A *reduction rule*, or just a rule, is a binary relation on the set of negotiations. Given a rule R, we write $\mathcal{N}_1 \xrightarrow{R} \mathcal{N}_2$ for $(\mathcal{N}_1, \mathcal{N}_2) \in R$. A rule R is *correct* if it preserves equivalence, i.e., if $\mathcal{N}_1 \xrightarrow{R} \mathcal{N}_2$ implies $\mathcal{N}_1 \equiv \mathcal{N}_2$. Notice that, in particular, this implies that $\mathcal{N}_1$ is sound if and only if $\mathcal{N}_2$ is sound.

Given a set of rules $\mathcal{R} = \{R_1, \dots, R_k\}$, we denote by $\mathcal{R}^*$ the reflexive and transitive closure of $R_1 \cup \ldots \cup R_k$. We say that $\mathcal{R}$ is *complete with respect to a class of negotiations* if, for every negotiation $\mathcal{N}$ in the class, there is a negotiation $\mathcal{N}'$ consisting of a single atom such that $\mathcal{N} \xrightarrow{\mathcal{R}^*} \mathcal{N}'$.

We describe rules as pairs of a *guard* and an *action*; $\mathcal{N}_1 \xrightarrow{R} \mathcal{N}_2$ holds if $\mathcal{N}_1$ satisfies the guard and $\mathcal{N}_2$ is a possible result of applying the action to $\mathcal{N}_1$.

Merge rule. Intuitively, the *merge rule* merges two outcomes with identical next enabled atoms into one single outcome. It corresponds to the rule of step (2.1) in the previous section.

Definition 7. *Merge rule*
Guard: *N contains an atom n with two distinct outcomes $r_1, r_2 \in R_n$, such that $\mathcal{X}(n, a, r_1) = \mathcal{X}(n, a, r_2)$ for every $a \in A_n$.*
Action: *(1) $R_n \leftarrow (R_n \setminus \{r_1, r_2\}) \cup \{r_f\}$, where r_f is a fresh name.*
(2) For all $a \in P_n$: $\mathcal{X}(n, a, r_f) \leftarrow \mathcal{X}(n, a, r_1)$.
(3) $\delta(n, r_f) \leftarrow \delta(n, r_1) \cup \delta(n, r_2)$.

It is easy to see that the merge rule is correct for arbitrary negotiations.

Shortcut rule. The shortcut rule corresponds to the rule of step (2.3) in the previous section. We need a preliminary definition.

Definition 8. *Given atoms n, n', we say that (n, r)* unconditionally enables *n' if $P_n \supseteq P_{n'}$ and $\mathcal{X}(n, a, r) = \{n'\}$ for every $a \in P_{n'}$.*

Observe that if (n, r) unconditionally enables n' then, for *every* marking $\boldsymbol{x}$ that enables n, the marking $\boldsymbol{x}'$ given by $\boldsymbol{x} \xrightarrow{(n,r)} \boldsymbol{x}'$ enables n'. Moreover, n' can only be disabled by its own occurrence.

Loosely speaking, the shortcut rule merges the outcomes of two atoms that can occur one after the other into one single outcome with the same effect. Consider the negotiation fragment shown on the left of Figure 2. The guard of the rule will state that n must unconditionally enable n', which is the case. For every outcome of n', say r_1, the action of the rule adds a fresh outcome r_{1f} to n, and modifies the negotiation so that the occurrence of (n, r_{1f}) has the same effect as the occurrence of the sequence $(n, r)(n', r_1)$. In the figure, shortcutting the outcome (n, r) leaves n' without any input arc, and in this case the rule also removes n'. Otherwise we require that at least one input arc of a party $\tilde{a}$ of n' is an arc (i.e., not a proper hyperarc) from some atom $\tilde{n} \neq n$, annotated by $\tilde{r}$. This implies that after the occurrence of $(\tilde{n}, \tilde{r})$, n' is the only atom agent $\tilde{a}$ is ready to engage in.

Definition 9. *Shortcut rule*
Guard*: N contains an atom n with an outcome r, and an atom n', $n' \neq n$, such that (n, r) unconditionally enables n'. Moreover, if $n' \in \mathcal{X}(\tilde{n}, \tilde{a}, \tilde{r})$ for at least one $\tilde{n} \neq n$ with $\tilde{a} \in P_{\tilde{n}}$ and $\tilde{r} \in R_{\tilde{n}}$, then $\{n'\} = \mathcal{X}(\tilde{n}, \tilde{a}, \tilde{r})$ for some $\tilde{n} \neq n$, $\tilde{a} \in P_{\tilde{n}}$, $\tilde{r} \in R_{\tilde{n}}$.*
Action*: (1) $R_n \leftarrow (R_n \setminus \{r\}) \cup \{r'_f \mid r' \in R_{n'}\}$, where r'_f are fresh names.*
(2) For all $a \in P_{n'}$, $r' \in R_{n'}$: $\mathcal{X}(n, a, r'_f) \leftarrow \mathcal{X}(n', a, r')$.
For all $a \in P \setminus P_{n'}$, $r' \in R_{n'}$: $\mathcal{X}(n, a, r'_f) \leftarrow \mathcal{X}(n, a, r)$.
(3) For all $r' \in R_{n'}$: $\langle n, r'_f \rangle \leftarrow \langle n, r \rangle \langle n', r' \rangle$.
(4) If $\mathcal{X}^{-1}(n') = \emptyset$ after (1)-(3), then remove n' from N, where $\mathcal{X}^{-1}(n') = \{(\tilde{n}, \tilde{a}, \tilde{r}) \in T(N) \mid n' \in \mathcal{X}(\tilde{n}, \tilde{a}, \tilde{r})\}$.

Theorem 2. *The shortcut rule is correct.*

Proof. Let $\mathcal{N}_2$ be the result of applying the shortcut rule to $\mathcal{N}_1$. Assume atoms n and n' and $r \in R_n$ of n as in Definition 9. We say that an atom $\overline{n}$ occurs in an occurrence sequence σ if $(\overline{n}, \overline{r})$ occurs in σ for some $\overline{r} \in R_{\overline{n}}$. We call an occurrence sequence initial if it starts with the initial marking.
Claim 1. For each initial occurrence sequence σ of $\mathcal{N}_2$, replacing all occurrences of (n, r'_f) by $(n, r), (n', r')$ in σ yields an initial occurrence sequence of $\mathcal{N}_1$ that leads to the same marking as σ in $\mathcal{N}_2$.

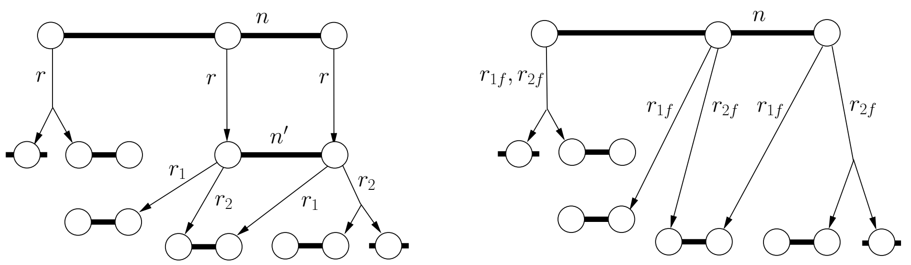

Fig. 2. An application of the shortcut rule

Claim 2. Each initial occurrence sequence σ of $\mathcal{N}_1$ can be uniquely divided into $\sigma_1(n,r)\sigma'_1(n',r_1)\sigma_2(n,r)\sigma'_2(n',r_2)\dots$ such that (n,r) does not occur in the sequences $\sigma_1,\sigma_2,\dots$ and n' does not occur in $\sigma'_1,\sigma'_2,\dots$.
Claim 3. If $\sigma = \sigma_1(n,r)\sigma'_1(n',r_1)\sigma_2(n,r)\sigma'_2(n',r_2)\dots(n',r_k)\sigma_{k+1}$ (notation as in Claim 2) then $\sigma_1(n,r_{f1})\sigma'_1\sigma_2(n,r_{f2})\sigma'_2\dots\sigma'_k\sigma_{k+1}$ is an initial occurrence sequence of $\mathcal{N}_2$ leading to the same marking as σ in $\mathcal{N}_1$. If there is an occurrence of n' in σ_i then a party $\tilde{a}$ of n' must have been enabled by a previous occurrence of some $(\tilde{n},\tilde{a},\tilde{r})$ with $n \neq \tilde{n}$. In this case n' still exists in $\mathcal{N}_2$.
Proof: For $i = 1,\dots k$ we have that σ'_i contains no atom with a party a of $P_{n'}$ because n' is the only atom with party a enabled after the occurrence of (n,r). So σ'_i and (n',r_i) can occur in arbitrary order. The sequence $(n,r)(n',r_i)$ can be replaced by the step (n,r_{fi}) of $\mathcal{N}_2$.
Claim 4. If $\sigma = \sigma_1(n,r)\sigma'_1(n',r_1)\sigma_2(n,r)\sigma'_2(n',r_2)\dots(n,r)\sigma'_k$ (as in Claim 2) then $\sigma_1(n,r_{f1})\sigma'_1\sigma_2(n,r_{f2})\sigma'_2\dots\sigma_k(n,r_{fk})\sigma'_k$ is an initial occurrence sequence of $\mathcal{N}_2$ leading to the same marking as $\sigma(n',r_k)$ in $\mathcal{N}_1$ for an arbitrary $r_k \in R_{n'}$.
Proof: Since (n,r) unconditionally enables n', n' is enabled after the last (n,r) in σ, and it remains enabled because σ'_k contains no occurrence of n'. So $\sigma(n',r_k)$ is also an initial occurrence sequence of $\mathcal{N}_1$. The claim follows using Claim 3 with σ_{k+1} being empty.
Claim 5. If $\mathcal{N}_2$ is sound then every atom $\overline{n} \in N_1$ appears in some initial occurrence sequence of $\mathcal{N}_1$.
Proof: If $\overline{n} \neq n'$ then this claim follows immediately from soundness of $\mathcal{N}_2$ and Claim 1. Otherwise, again by its soundness, $\mathcal{N}_2$ has an initial occurrence sequence with an occurrence of n. Clearly, the marking before the occurrence of n also enables some (n,r'_f), whence there is also a sequence including a step (n,r'_f). Claim 1 achieves the result.
Claim 6. If $\mathcal{N}_2$ is sound then every initial occurrence sequence σ of $\mathcal{N}_1$ can be extended to a large step.
Proof: By Claim 3 and Claim 4, the marking reached by σ or the marking reached by $\sigma(n,r_{fi})$ (for some $r_i \in R_{n_i}$) is also reachable in $\mathcal{N}_2$. Since $\mathcal{N}_2$ is sound, this sequence can be extended to a large step. By Claim 1, there is a corresponding occurrence sequence σ' of $\mathcal{N}_1$. So either σ' or $(n,r_{fi})\sigma'$ extends σ to a large step.
Claim 7. If $\mathcal{N}_1$ is sound then every atom $\overline{n} \in N_2$ appears in some initial occurrence sequence of $\mathcal{N}_2$.

Proof: If $\overline{n} \neq n'$ then this follows immediately from Claim 3 and Claim 4. So we only have to consider the case $\overline{n} = n'$. Then, in particular n' still exists in $\mathcal{N}_2$. So $\mathcal{X}^{-1}(n')$ contains some $(\tilde{n}, \tilde{a}, \tilde{r})$ where $(\tilde{n}, \tilde{r}) \neq (n, r)$. By the guard of the shortcut rule, we have $\{n'\} = \mathcal{X}(\tilde{n}, \tilde{a}, \tilde{r})$ for some $\tilde{n}$, $\tilde{a}$ and $\tilde{r}$, i.e., after the occurrence of $(\tilde{n}, \tilde{r})$ only the occurrence of n' can remove the token of $\tilde{a}$. Since $\mathcal{N}_1$ is sound, there is an initial occurrence sequence that enables $\tilde{n}$. This sequence can be extended to a large step, whence n' occurs after $\tilde{n}$. Call the entire sequence σ.

Taking the division of σ as in Claim 2, both the occurrence of $\tilde{n}$ and the subsequent occurrence of n' appear in some subsequence σ_i, because the agent $\tilde{a}$ is ready to engage only in n' during all σ'_i and can thus not participate in $\tilde{n}$. The transformation of Claim 3 (Claim 4, respectively) leads to an occurrence sequence of $\mathcal{N}_2$ that still includes all subsequences σ'_i and therefore also includes an occurrence of n'.

Claim 8. If $\mathcal{N}_1$ is sound then every initial occurrence sequence σ of $\mathcal{N}_2$ can be extended to a large step.

Proof: By Claim 1, the marking reached by σ is also reachable in $\mathcal{N}_1$. Moreover, the translated sequence has an occurrence of n' after its last occurrence of (n, r). By soundness of $\mathcal{N}_1$, σ can be extended by some σ' to a large step. Since this sequence ends with the empty marking, there is some occurrence of n' after the last occurrence of (n, r). So the entire sequence $\sigma\sigma'$ can be divided into

$$\sigma_1(n,r),(n',r_1)\sigma_2(n,r)(n',r_2)\ldots\sigma_l\sigma_{l+1}(n,r)\sigma'_{l+1}(n',r_{l+2})\ldots(n',r_k)\sigma_{k+1}$$

(notations as in Claim 2) where σ comprises everything up to σ_l. We apply Claim 3 and obtain the large step of $\mathcal{N}_2$:

$$\sigma_1(n,r_{f1})\sigma_2(n,r_{f2})\ldots\sigma_l\sigma_{l+1}(n,r_{fl+1})\sigma'_{l+1}\ldots\sigma_k(n,r_{fk})\sigma'_k\sigma_{k+1}.$$

Since the subsequence until σ_l reaches the same marking as σ, the result follows.

The proof of the theorem follows immediately from Claims 5 to 8.

□

Useless arc rule. Consider the negotiation on the left of Figure 3, in which all atoms have one outcome r. We have $\mathcal{X}(n_0, a, r) = \{n_1, n_f\}$, i.e., after the occurrence of (n_0, r) agent a is ready to engage in both n_1 and n_f. However, a always engages in n_1, because the only large step is $(n_0, r)(n_1, r)(n_2, r)(n_f, r)$. In other words, we can set $\mathcal{X}(n_0, a, n_f) = \{n_1\}$ without changing the behavior. Intuitively, we say that the arc (more precisely, the leg of the hyperarc) leading to the a-port of n_f is useless. The useless arc rule identifies and removes some useless arcs.

Definition 10. *Useless arc rule.*
Guard*: There are* $(n, a, r), (n, b, r) \in T(N)$ *and two distinct atoms* $n', n'' \in N$ *such that* $a, b \in P_{n'} \cap P_{n''}$, $n', n'' \in \mathcal{X}(n, a, r)$ *and* $\mathcal{X}(n, b, r) = \{n'\}$.
Action*:* $\mathcal{X}(n, a, r) \leftarrow \mathcal{X}(n, a, r) \setminus \{n''\}$.

The rule can be applied to the negotiation on the left of Figure 3 by instantiating $n := n_0$, $n' := n_1$, and $n'' := n_f$. It cannot be applied to the negotiation on the right. If we set $n := n_0$, $n' := n_1$, and $n'' := n_2$, then $a \notin P_{n_2}$.

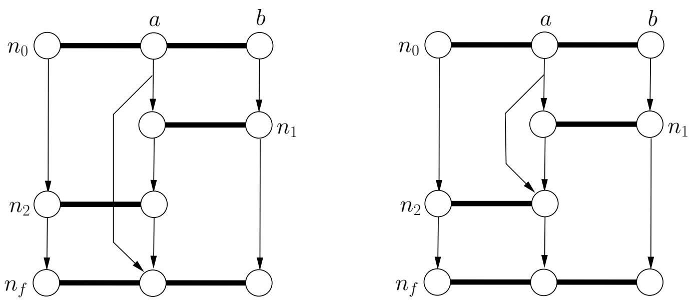

Fig. 3. The useless arc rule can only be applied to the left negotiation

Theorem 3. *The useless arc rule is correct.*

Proof. Let $\mathcal{N}_2$ be the result of applying the rule to $\mathcal{N}_1$. We adopt all notations from Definition 10 and claim that $\mathcal{N}_1$ and $\mathcal{N}_2$ have the same occurrence sequences. Let $\mathcal{X}_1, \mathcal{X}_2$ be the transition functions of $\mathcal{N}_1$ and $\mathcal{N}_2$, respectively. Since $\mathcal{X}_2(n, a, r) \subseteq \mathcal{X}_1(n, a, r)$ for every $(n, a, r) \in T(N)$, every occurrence sequence of $\mathcal{N}_2$ is also an occurrence sequence of $\mathcal{N}_1$. It remains to prove that every initial occurrence sequence σ of $\mathcal{N}_1$ is also an initial occurrence sequence of $\mathcal{N}_2$.

Assume the contrary, i.e., $\sigma = \sigma_1\sigma_2$ such that σ_1 is an initial occurrence sequence of both $\mathcal{N}_1$ and $\mathcal{N}_2$ whereas the first step of σ_2 is only possible in $\mathcal{N}_1$. Since both negotiations only differ w.r.t. $\mathcal{X}(n, a, r)$, this step must be an occurrence of agent n'' occurring after an occurrence of (n, r) such that no other atom of $\mathcal{X}(n, a, r)$ occurred in between. This holds in particular for $n' \in \mathcal{X}(n, a, r)$. But, since $\mathcal{X}(n, b, r) = \{n'\}$ and b participates both in n' and in n'', n' must occur before n'' after (n, r) – a contradiction. This concludes the proof showing that $\mathcal{N}$ and $\mathcal{N}'$ have the same occurrence sequences.

A first immediate consequence is that an atom can occur in $\mathcal{N}_1$ iff it can occur in $\mathcal{N}_2$. It remains to show that every occurrence sequence of $\mathcal{N}_1$ can be extended to a large step iff the same holds for $\mathcal{N}_2$. To this end, since both negotiations have the same occurrence sequences, we only have to show that an occurrence sequence is a large step of $\mathcal{N}_1$ iff it is a large step of $\mathcal{N}_2$. Respective markings in $\mathcal{N}_1$ and $\mathcal{N}_2$ reached by the same occurrence sequence differ only with respect to agent a: after the occurrence of (n, r), $\boldsymbol{x}(a)$ contains n' and n'' in $\mathcal{N}_1$ and $\boldsymbol{x}(a)$ contains n' in $\mathcal{N}_2$. Large steps, however, lead to the marking satisfying $\boldsymbol{x}(a) = \emptyset$ and are hence identical for both negotiations.

$\square$

5 (Weakly) Deterministic Negotiations

We introduce weakly deterministic and deterministic negotiations.

Definition 11. *An agent $a \in A$ is* deterministic *if for every $(n, a, r) \in T(N)$ such that $n \neq n_f$ there exists one atom n' such that $\mathcal{X}(n, a, r) = \{n'\}$.*

The negotiation $\mathcal{N}$ *is* weakly deterministic *if for every* $(n, a, r) \in T(N)$ *there is a deterministic agent* b *that is a party of every atom in* $\mathcal{X}(n, a, r)$*, i.e.,* $b \in P_{n'}$ *for every* $n' \in \mathcal{X}(n, a, r)$*. It is* deterministic *if all its agents are deterministic.*

Graphically, an agent a is deterministic if no proper hyperarc leaves any port of a. Consider the negotiations of Figure 1. In the acyclic negotiation both Father and Daughter are deterministic, while Mother is not. In the ping-pong negotiation only Daughter is deterministic. Both negotiations are weakly deterministic, because Daughter participates in all atoms, and so can be always chosen as the party b required by the definition. Observe that the notion of deterministic agent does not refer to the behavior of atoms, which is intrinsically nondeterministic with respect to its possible outcomes and even to its state transformations. Rather, it refers to the *composition* of negotiations: For each atom n, the next atom of a deterministic agent is completely determined by the outcome of n.

Weakly deterministic negotiations have a natural semantic justification. Consider a negotiation with two agents a, b and three atoms $\{n_0, n_1, n_f\}$. All atoms have the same parties a, b and one outcome r, such that $\mathcal{X}(n_0, a, r) = \{n_1, n_f\} = \mathcal{X}(n_0, b, r)$ and $\mathcal{X}(n_1, a, r) = \{n_f\} = \mathcal{X}(n_1, b, r)$. After the occurrence of (n_0, r) the parties a and b are ready to engage in both n_1 and n_f, and so which of them occurs requires a "meta-negotiation" between a and b. This meta-negotiation, however, is not part of the model and, more importantly, it can be difficult to implement, since it requires to break a symmetry. In a weakly deterministic negotiation this situation cannot happen. If $\mathcal{X}(n, a, r)$ contains more than one atom, then some deterministic agent b is a part of all atoms in $\mathcal{X}(n, a, r)$. If some atom of $\mathcal{X}(n, a, r)$ becomes enabled, say n', then because agent b is ready to engage in it, and, since b is deterministic, b is not ready to engage in any other atom. So n' is the only enabled atom of $\mathcal{X}(n, a, r)$, and n' is the negotiation that a will engage in next. This is very easy to implement: b just sends a message to a telling her that she should commit to n'.

Notice that, for deterministic negotiations, the second part of the guard of the shortcut rule is always satisfied. Using the notation of the shortcut rule, this condition requires that the atom n' is the only atom in $\mathcal{X}(\tilde{n}, \tilde{a}, \tilde{r})$ for some $(\tilde{n}, \tilde{r})$, provided n' is not only in $\mathcal{X}(n, a, r)$ for some $a \in P_n$ (i.e., provided n' is not removed by the application of the rule). This clearly holds if $\tilde{n}$ is deterministic, as all agents are deterministic in deterministic negotiations.

In the next section we show that, on top of their semantic justification, weakly deterministic and deterministic negotiations are also interesting from an analysis point of view. We prove that the shortcut and useless arc rules are complete for acyclic, weakly deterministic negotiations, which of course implies that the same rules plus the merge are complete, too. A second result proves that a polynomial number of applications of the merge and shortcut rules suffices to summarize any sound deterministic acyclic negotiation (the useless arc rule is irrelevant for deterministic negotiations).

6 Completeness and complexity

We start with the completeness result for the weakly deterministic case.

Theorem 4. *The shortcut and useless arc rules are complete for acyclic, weakly deterministic negotiations.*

Proof. Let $\mathcal{N}$ be a sound, acyclic, and weakly deterministic negotiation.

The proof has two parts:

(1) If $\mathcal{N}$ has more than one atom, then the shortcut rule or the useless arc rule can be applied to it.

Since $\mathcal{N}$ is acyclic, its graph generates a partial order on atoms in the obvious way ($n < n'$ if there is a path from n to n'). Clearly n_0 is the unique minimal element. We choose an arbitrary linearisation of this partial order. Since $\mathcal{N}$ has more than one atom, this linearisation begins with n_0 and has some second element, say n_1. Since $\mathcal{N}$ is sound, some occurrence sequence begins with an occurrence of n_0 and a subsequent occurrence of n_1. So n_0 has an outcome r_0 such that $n_1 \in \mathcal{X}(n_0, a, r_0)$ for every party a of n_1.

Consider two cases:

- $\{n_1\} = \mathcal{X}(n_0, a, r_0)$ for every party a of n_1.
  Then (n_0, r_0) unconditionally enables n_1. Moreover, there are no $\tilde{n}$, $\tilde{a}$ and $\tilde{r}$ such that $\tilde{n} \neq n_0$ and $\mathcal{X}(\tilde{n}, \tilde{a}, \tilde{r}) = \{n_1\}$ because otherwise, according to the above defined partial order, $n_0 < \tilde{n} < n_1$ and so $\tilde{n}$ would be between n_0 and n_1 in every linearisation. So the shortcut rule can be applied.
- $\{n_1\} \neq \mathcal{X}(n_0, a, r_0)$ for some party a of n_1.
  Then $n_1, n_2 \in \mathcal{X}(n_0, a, r_0)$ for some atom $n_2 \neq n_1$. Since $\mathcal{N}$ is weakly deterministic, there is a deterministic agent b that is a party of every atom in $\mathcal{X}(n_0, a, r_0)$, in particular of both n_1 and n_2. Since b is deterministic and $n_1 \in \mathcal{X}(n_0, b, r)$, we have $\mathcal{X}(n_0, b, r_0) = \{n_1\}$, and so the useless arc rule is applicable.

(2) The shortcut and useless arc rules cannot be applied infinitely often.
Let $\mathcal{N}_2$ be the result of applying any of the two rules to a negotiation $\mathcal{N}_1$. For every large step σ' of $\mathcal{N}_2$, let $\phi'(\sigma')$ be defined similarly to Theorem 2: if the useless arc rule has been applied, then $\phi'(\sigma') = \sigma'$; if the shortcut rule has been applied to atoms n, n', then let $\phi'(\sigma')$ be the result of replacing every occurrence of (n, r'_f) by the sequence $(n, r)(n', r')$. We have $|\sigma'| \leq |\phi(\sigma')|$. Moreover, if the shortcut rule is applied, then $|\sigma'| < |\sigma|$ for at least one large step σ', indeed for all large steps containing (n', r'_f). Since the set of large steps of an acyclic negotiation is finite, and the length of every large step is not negative, no infinite sequence of applications of the rules can contain infinitely many applications of the shortcut rule. So any infinite sequence of applications must, from some point on, only apply the useless arc rule. But this is also not possible, since the rule reduces the number of arcs.

By (2) every maximal sequence of applications of the rule terminates. By (1) it terminates with a negotiation containing one single atom. $\square$

Next we prove that a *polynomial number* of applications of the merge and shortcut rules suffice to summarize any sound *deterministic* acyclic negotiation (SDAN). For this we have to follow a strategy in the application of the shortcut rule.

Definition 12. *The* deterministic shortcut rule*, or* d-shortcut *rule, is the result of adding to the guard of the shortcut rule a new condition: (3) n' has at most one outcome (the actions of the shortcut and d-shortcut rules coincide).*

We say that a SDAN is *irreducible* if neither the merge nor the d-shortcut rule can be applied to it. In the rest of the section we prove that irreducible SDANs are necessarily atomic, i.e., consist of a single atom. The proof, which is rather involved, proceeds in three steps. First, we prove a technical lemma showing that SDANs can be reduced so that all agents participate in every atom with more than one outcome. Then we use this result to prove that, loosely speaking, every SDAN can be reduced to a "replication" of a negotiation with only one agent: if $\mathcal{X}(n, a, r) = \{n'\}$ for some agent a, then $\mathcal{X}(n, a, r) = \{n'\}$ for *every* agent a. In the third step, we show that replications can be reduced to atomic negotiations. Finally, we analyze the number of rule applications needed in each of these three steps, and conclude.

Lemma 1. *Let $\mathcal{N}$ be an irreducible SDAN and let $n \neq n_f$ be an atom of $\mathcal{N}$ with more than one outcome. Then every agent participates in n.*

Proof. We proceed in two steps.

(a) The atom n has an outcome r such that: either (n, r) unconditionally enables n_f, or (n, r) unconditionally enables some atom with more than one outcome.

This is the core of the proof. We first claim: if some outcome (n, r) unconditionally enables some atom, then (a) holds. Indeed: if (n, r) unconditionally enables some atom n', then either $n' = n_f$ or n' has more than one outcome, because otherwise the d-shortcut rule can be applied to n and n', contradicting the irreducibility of $\mathcal{N}$. This proves the claim.

It remains to prove that some outcome (n, r) unconditionally enables some atom. For this, we assume the contrary, and prove that $\mathcal{N}$ contains a cycle, contradicting the hypothesis.

Since the merge rule is not applicable to $\mathcal{N}$, n has two outcomes r_1, r_2 such that $\mathcal{X}(n, a, r_1) \neq \mathcal{X}(n, a, r_2)$ for some agent a. We proceed in three steps.

(a1) For every reachable marking $\boldsymbol{x}$ that enables n there is a sequence σ such that $\boldsymbol{x} \xrightarrow{(n,r_1)\,\sigma} \boldsymbol{x}_1$ and $\boldsymbol{x} \xrightarrow{(n,r_2)\,\sigma} \boldsymbol{x}_2$ for some markings $\boldsymbol{x}_1, \boldsymbol{x}_2$, and the sets N_1 and N_2 of atoms enabled by $\boldsymbol{x}_1, \boldsymbol{x}_2$ are nonempty and disjoint.

Let σ be a longest occurrence sequence such that $\boldsymbol{x} \xrightarrow{(n,r_1)\,\sigma} \boldsymbol{x}_1$ and $\boldsymbol{x} \xrightarrow{(n,r_2)\,\sigma} \boldsymbol{x}_2$ for some markings $\boldsymbol{x}_1, \boldsymbol{x}_2$ (notice that σ exists, because all occurrence sequences of $\mathcal{N}$ are finite by acyclicity). We have $N_1 \cap N_2 = \emptyset$, because otherwise we can extend σ with the occurrence of any atom enabled by both markings. We prove that, furthermore, $N_1 \neq \emptyset \neq N_2$. Assume w.l.o.g. $N_1 = \emptyset$.

Then, since $\mathcal{N}$ is sound, we have $\boldsymbol{x}_1 = \boldsymbol{x}_f$, which means that the last step of σ is of the form (n_f, r_f). So $\boldsymbol{x}_2$ is also a marking obtained after the occurrence of (n_f, r_f). Since every agent participates in n_f and $\mathcal{X}(n_f, a, r_f) = \emptyset$ for every agent a and outcome r_f, we also have $\boldsymbol{x}_2 = \boldsymbol{x}_f$. So $\boldsymbol{x} \xrightarrow{(n,r_1)} \boldsymbol{x}'_1 \xrightarrow{\sigma} \boldsymbol{x}_f$ and $\boldsymbol{x} \xrightarrow{(n,r_1)} \boldsymbol{x}'_2 \xrightarrow{\sigma} \boldsymbol{x}_f$, which implies $\boldsymbol{x}'_1 = \boldsymbol{x}'_2$. Since $\mathcal{N}$ is deterministic, we then have $\mathcal{X}(n, a, r_1) = \mathcal{X}(n, a, r_2)$, contradicting the hypothesis.

(a2) For every $n_1 \in N_1$ there is a path leading from some $n_2 \in N_2$ to n_1, and for every $n_2 \in N_2$ there is a path leading from some $n_1 \in N_1$ to n_2.
By symmetry it suffices to prove the first part. Since N_1 and N_2 are disjoint, n_1 is enabled at $\boldsymbol{x}_1$ but not at $\boldsymbol{x}_2$. Moreover, since $\mathcal{N}$ is acyclic, every atom can occur at most once in an occurrence sequence, and so neither n_1 nor n_2 appear in σ. Since, furthermore, the sequences $(n, r_1)\,\sigma$ and $(n, r_2)\,\sigma$ only differ in their first element, there is an agent a such that $\mathcal{X}(n, a, r_1) = \{n_1\}$ and $\mathcal{X}(n, a, r_2) = \{n'_2\}$ for some $n'_2 \neq n_1$ (n'_2 is not necessarily the $n_2 \in N_2$ we are looking for). So we have $\boldsymbol{x}_1(a) = \{n_1\}$ and $\boldsymbol{x}_2(a) = \{n'_2\}$ (see Figure 4).

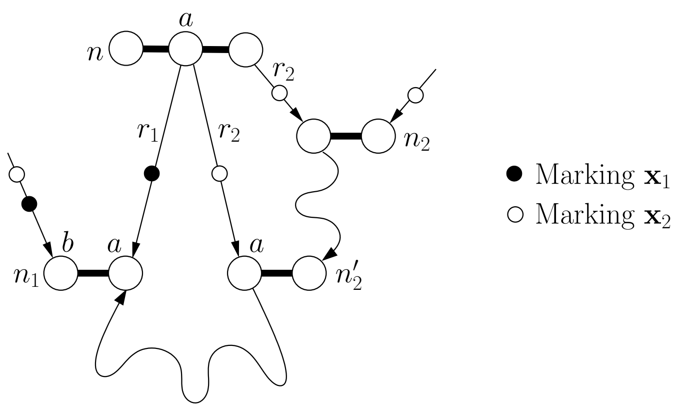

Fig. 4. Illustration of the proof of Lemma 1.

We first show that there is a path from n'_2 to n_1. By assumption, no outcome of n unconditionally enables any atom, and so (n, r_1) does not unconditionally enable n_1. So n_1 has a party $b \neq a$ such that either b is not a party of n or $\mathcal{X}(n, b, r_1) \neq n_1$. Since $\boldsymbol{x}_1$ enables n_1 we have $\boldsymbol{x}_1(b) = \{n_1\}$, and since b is not a party of n or $\mathcal{X}(n, b, r_1) \neq n_1$, we have $\boldsymbol{x}_2(b) = \{n_1\}$ as well. Since $\boldsymbol{x}_2(b) = n_1$, and $\mathcal{N}$ is a SDAN, there is an occurrence sequence τ such that $\boldsymbol{x}_2 \xrightarrow{\tau} \boldsymbol{x}'_2$ and $\boldsymbol{x}'_2$ enables n_1 (intuitively, the white token on the arc to the b-port can only leave the arc through the occurrence of n_1). Since $\boldsymbol{x}_2(a) = \{n'_2\} \neq \{n_1\}$, there is a path from n'_2 to n_1 (intuitively, the white token on the arc leading to the a-port of n'_2 has to travel to some arc leading to the a-port of n_1, and by determinism it can only do so through a path of a-ports that crosses n'_2).

We now prove that there is a path from some $n_2 \in N_2$ to n_2'. If n_2' is enabled at $\boldsymbol{x}_2$, then $n_2' \in N_2$ and we are done. If n_2' is not enabled at $\boldsymbol{x}_2$ (as in the figure) then, since $\boldsymbol{x}_2(a) = \{n_2'\}$ and $\mathcal{N}$ is a SDAN, there is a sequence τ such that $\boldsymbol{x}_2 \xrightarrow{\tau} \boldsymbol{x}_2'$ and $\boldsymbol{x}_2'$ enables n_2' (again, by soundness the white token on the arc to the a-port of n_2' can eventually leave the arc to move towards n_f, and by determinacy it can only leave the arc through the occurrence of n_2'). Since N_2 is the set of transitions enabled at $\boldsymbol{x}_2$, we have $\tau = (n_2, r)\,\tau'$ for some $n_2 \in N_2$. So some subword of τ is a path from some transition of N_2 to n_2'.

(a3) $\mathcal{N}$ contains a cycle.
Follows immediately from (a2) and the finiteness of N_1 and N_2.

(b) Every agent participates in n.
By repeated application of (a) we find a chain $(n_1, r_1) \ldots (n_k, r_k)$ such that $n_1 = n$, $n_k = n_f$, and (n_i, r_i) unconditionally enables n_{i+1} for every $1 \leq i \leq k-1$. By the definition of unconditionally enabled we have $P_1 \supseteq P_2 \supseteq \cdots \supseteq P_k = P_f$. Since $P_f = A$, we obtain $P_1 = A$.

Lemma 2. *Let $\mathcal{N}$ be an irreducible SDAN. Every agent participates in every atom, and for every atom $n \neq n_f$ and every outcome r there is an atom n' satisfying $\mathcal{X}(n, a, r) = \{n'\}$ for every agent a.*

Proof. We first show that every agent participates in every atom. By Lemma 1, it suffices to prove that every atom $n \neq n_f$ has more than one outcome. Assume the contrary, i.e., some atom different from n_f has only one outcome. Since, by soundness, every atom can occur, there is an occurrence sequence $(n_0, r_0)(n_1, r_1) \cdots (n_k, r_k)$ such that n_k has only one outcome and all of $n_0, \ldots, n_{k-1}$ have more than one outcome. By Lemma 1, all agents participate in all of n_0, n_1, n_{k-1}. It follows that (n_i, r_i) unconditionally enables (n_{i+1}, r_{i+1}) for every $0 \leq i \leq k-1$. In particular, (n_{k-1}, r_{k-1}) unconditionally enables (n_k, r_k). But then, since n_k only has one outcome, the d-shortcut rule can be applied to n_{k-1}, n, contradicting the hypothesis that $\mathcal{N}$ is irreducible.

For the second part, assume there is an atom $n \neq n_f$, an outcome r of n, and two distinct agents a_1, a_2 such that $\mathcal{X}(n, a_1, r) = \{n_1\} \neq \{n_2\} = \mathcal{X}(n, a_2, r)$. By the first part, every agent participates in n, n_1 and n_2. Since $\mathcal{N}$ is sound, some reachable marking $\boldsymbol{x}$ enables n. Moreover, since all agents participate in n, and $\mathcal{N}$ is deterministic, the marking $\boldsymbol{x}$ only enables n. Let $\boldsymbol{x}'$ be the marking given by $\boldsymbol{x} \xrightarrow{(n,r)} \boldsymbol{x}'$. Since a_1 participates in all atoms, no atom different from n_1 can be enabled at $\boldsymbol{x}'$. Symmetrically, no atom different from n_2 can be enabled at $\boldsymbol{x}'$. So $\boldsymbol{x}'$ does not enable any atom, contradicting that $\mathcal{N}$ is sound. $\square$

Theorem 5. *Let $\mathcal{N}$ be an irreducible SDAN. Then $\mathcal{N}$ contains only one atom.*

Proof. (Sketch) Assume $\mathcal{N}$ contains more than one atom. Fore every atom $n \neq n_f$, let $l(n)$ be the length of the longest path from n to n_f in the graph of $\mathcal{N}$. Let $n_{\min}$ be any atom such that $l(n_{\min})$ is minimal, and let r be an arbitrary outcome of $n_{\min}$. By Lemma 2 there is an atom n' such that $\mathcal{X}(n_{\min}, a, r) = \{n'\}$ for every agent a. If $n' \neq n_f$ then by acyclicity we have $l(n') < l(n_{\min})$, contradicting the minimality of $n_{\min}$. So we have $\mathcal{X}(n_{\min}, a, r) = \{n'\}$ for every outcome r of $n_{\min}$ and every agent a. If $n_{\min}$ has more than one outcome, then the merge rule is applicable. If $n_{\min}$ has one outcome, then, since it unconditionally enables n_f, the d-shortcut rule is applicable. In both cases we get a contradiction to irreducibility. □

Definition 13. *For every atom n and outcome r, let $shoc(n, r)$ be the length of a shortest maximal occurrence sequence containing (n, r) minus 1, and let $Shoc(\mathcal{N}) = \sum_{n \in N, r \in R} shoc(n, r)$. Finally, let $Out(\mathcal{N}) = \sum_{(P,R,\delta) \in N \setminus \{n_f\}} |R|$ be the total number of outcomes of $\mathcal{N}$, excluding those of the final atom.*

Notice that if $\mathcal{N}$ has K atoms then $shoc(n, r) \leq K - 1$ holds for every atom n and outcome r. Further, if $K = 1$ then $Shoc(\mathcal{N}) = 0 = Out(\mathcal{N})$.

Theorem 6. *Every SDAN $\mathcal{N}$ can be completely reduced by means of $Out(\mathcal{N})$ applications of the merge rule and $Shoc(\mathcal{N})$ applications of the d-shortcut rule.*

Proof. Let $\mathcal{N}$ and $\mathcal{N}'$ be negotiations such that $\mathcal{N}'$ is obtained from $\mathcal{N}$ by means of the merge or the d-shortcut rule. For the merge rule we have $Out(\mathcal{N}') < Out(\mathcal{N})$ and $Shoc(\mathcal{N}') \leq Out(\mathcal{N})$ because the rule reduces the number of outcomes by one. For the d-shortcut rule we have $Out(\mathcal{N}') = Out(\mathcal{N})$ because if it is applied to pairs n, n' such that n' has one single outcome, and $Shoc(\mathcal{N}') < Shoc(\mathcal{N})$, because $Shoc(n, r'_f) < Shoc(n, r)$. □

7 Conclusions

We have introduced negotiations, a formal model of concurrency with negotiation atoms as primitive. We have defined and studied two important analysis problems: soundness, which coincides with the soundness notion for workflow nets, and the new *summarization problem*. We have provided a complete set of rules for sound acyclic, weakly deterministic negotiations, and we have shown that the rules allow one to compute a summary of a sound deterministic negotiations in polynomial time.

Several open questions deserve further study. Our results show that summarization can be solved in polynomial time for deterministic, acyclic negotiations, and is co-NP-hard for arbitrary acyclic negotiations. The precise complexity of the weak deterministic case is still open. We are currently working on a generalization of Rule 2.2. of Section 3.2, such that we can completely reduce in polynomial time *arbitrary* deterministic negotiations, even if they contain cycles.

Related work. Previous work on Petri net analysis by means of reductions has already been discussed in the Introduction.

A number of papers have modelled specific distributed negotiation protocols with the help of Petri nets or process calculi (see [22,20,3]). However, these papers do not address the issue of negotiation as concurrency primitive.

The feature of summarizing parts of a negotiation to single negotiation atoms has several analogies in Petri net theory, among these the concept of zero-safe

Petri nets. By abstracting from reachable markings which mark distinguished "zero-places", transactions can be modelled by zero-safe Petri nets [8]. Reference [7] extends this concept to reconfigurable, dynamic high-level Petri nets.

A related line of research studies global types and session types to model multi-party sessions [18]. See [9] for an overview that also covers choreography-based approaches for web services. This research emphasises communication aspects in the formal setting of mobile processes. Thus, the aim differs from our aim. However, it might be worth trying to combine the two approaches.

Finally, the graphical representation of negotiations was partly inspired by the BIP component framework [4,6], where a set of sequential components (i.e., the agents) interact by synchronizing on certain actions (i.e., the atoms).

Acknowledgement. We thank the reviewers, in particular for their hints to related papers. We also thank Stephan Barth, Eike Best, and Jan Kretinsky for very helpful discussions.

References

1. W. M. P. van der Aalst. The application of Petri nets to workflow management. *J. Circuits, Syst. and Comput.*, 08(01):21–66, 1998.
2. W. M. P. van der Aalst, K. M. van Hee, A. H. M. ter Hofstede, N. Sidorova, H. M. W. Verbeek, M. Voorhoeve, and M. T. Wynn. Soundness of workflow nets: classification, decidability, and analysis. *Formal Asp. Comput.*, 23(3):333–363, 2011.
3. E. Bacarin, E. R. M. Madeira, C. B. Medeiros, and W. M. P. van der Aalst. *SpiCa*'s multi-party negotiation protocol: Implementation using YAWL. *Int. J. Cooperative Inf. Syst.*, 20(3):221–259, 2011.
4. A. Basu, S. Bensalem, M. Bozga, J. Combaz, M. Jaber, T.-H. Nguyen, and J. Sifakis. Rigorous component-based system design using the BIP framework. *IEEE Software*, 28(3):41–48, 2011.
5. G. Berthelot. Transformations and decompositions of nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Advances in Petri Nets*, volume 254 of *LNCS*, pages 359–376. Springer, 1986.
6. S. Bliudze and J. Sifakis. The algebra of connectors - structuring interaction in BIP. *IEEE Trans. Computers*, 57(10):1315–1330, 2008.
7. R. Bruni, H. C. Melgratti, and U. Montanari. Extending the zero-safe approach to coloured, reconfigurable and dynamic nets. In J. Desel, W. Reisig, and G. Rozenberg, editors, *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 291–327. Springer, 2003.
8. R. Bruni and U. Montanari. Executing transactions in zero-safe nets. In M. Nielsen and D. Simpson, editors, *Application and Theory of Petri Nets*, pages 83–102, 2000.
9. G. Castagna, M. Dezani-Ciancaglini, and L. Padovani. On global types and multi-party session. *Logical Methods in Computer Science*, 8(1), 2012.
10. J. Desel. Reduction and design of well-behaved concurrent systems. In J. C. M. Baeten and J. W. Klop, editors, *CONCUR*, volume 458 of *Lecture Notes in Computer Science*, pages 166–181. Springer, 1990.
11. J. Desel and J. Esparza. *Free choice Petri nets.* Cambridge University Press, New York, NY, USA, 1995.

12. B. F. van Dongen, W. M. P. van der Aalst, and H. M. W. Verbeek. Verification of EPCs: Using reduction rules and Petri nets. In O. Pastor and J. F. e Cunha, editors, *CAiSE*, volume 3520 of *LNCS*, pages 372–386. Springer, 2005.
13. J. Esparza. Decidability and complexity of Petri net problems - an introduction. In *Petri Nets*, volume 1491 of *LNCS*, pages 374–428. Springer, 1996.
14. J. Esparza and J. Desel. On negotiation as concurency primitive, 2013. To appear in the Proceedings of CONCUR 2013.
15. H. J. Genrich and P. S. Thiagarajan. A theory of bipolar synchronization schemes. *Theor. Comput. Sci.*, 30:241–318, 1984.
16. S. Haddad. A reduction theory for coloured nets. In G. Rozenberg, editor, *Advances in Petri Nets*, volume 424 of *LNCS*, pages 209–235. Springer, 1988.
17. S. Haddad and J.-F. Pradat-Peyre. New efficient Petri nets reductions for parallel programs verification. *Parallel Processing Letters*, 16(1):101–116, 2006.
18. K. Honda, N. Yoshida, and M. Carbone. Multiparty asynchronous session types. In G. C. Necula and P. Wadler, editors, *POPL*, pages 273–284. ACM, 2008.
19. J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.
20. S. Ji, Q. Tian, and Y. Liang. A Petri-net-based modeling framework for automated negotiation protocols in electronic commerce. In D. Lukose and Z. Shi, editors, *PRIMA*, volume 4078 of *LNCS*, pages 324–336. Springer, 2005.
21. C. H. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). In H. R. Lewis, B. B. Simons, W. A. Burkhard, and L. H. Landweber, editors, *STOC*, pages 255–260. ACM, 1982. ISBN 0-89791-070-2.
22. G. Salaün, A. Ferrara, and A. Chirichiello. Negotiation among web services using LOTOS/CADP. In L.-J. Zhang, editor, *ECOWS*, volume 3250 of *LNCS*, pages 198–212. Springer, 2004.
23. H. M. W. Verbeek, M. T. Wynn, W. M. P. van der Aalst, and A. H. M. ter Hofstede. Reduction rules for reset/inhibitor nets. *J. Comput. Syst. Sci.*, 76(2): 125–143, 2010.