# Dialogues for Negotiation: Agent Varieties and Dialogue Sequences


Fariba Sadri* Francesca Toni* Paolo Torroni†

* Department of Computing, Imperial College,
180 Queens Gate, SW7 London, UK
{fs,ft}@doc.ic.ac.uk

† DEIS, Università di Bologna
V.le Risorgimento 2, 40136 Bologna, Italy
ptorroni@deis.unibo.it



**Abstract.** This work presents a formal, logic-based approach to one-to-one agent negotiation, in the context of goal achievement in systems of agents with limited resource availability. The proposed solution is based on agent dialogues as a way, e.g., to request resources and propose resource exchanges. It relies upon agents agreeing solely upon a language for negotiation, while possibly adopting different negotiation policies, each corresponding to an agent variety. Agent dialogues can be connected within sequences, all aimed at achieving an individual agent's goal. Sets of sequences aim at allowing all agents in the system to achieve their goals, and can be used, e.g., to solve the resource reallocation problem. We model dialogues via logic-based dialogue constraints, and propose an execution model to be adopted for the automatic generation of dialogues. We also propose an agent dialogue cycle for the automatic generation of sequences of dialogues. Finally, we identify some desirable properties of dialogues, dialogue sequences and sets of such sequences, and prove/disprove these properties for concrete agent varieties.


## 1 Introduction

Negotiation is one of the main research areas in multi-agent systems. In many cases agents need to negotiate because they operate in environments with limited resource availability. The area of negotiation is broad and applies to different scenarios [5]. For example, *one-to-many* negotiation is used for auctions, where auctioneer and bidders reach an agreement on the cost of the items on sale. Most research in the area of one-to-many negotiation adopts a game-theoretic approach, with interesting results but under the often unrealistic assumptions that the computational resources of agents are unlimited and that their knowledge of the world is global and complete. Another example of negotiation is *one-to-one* negotiation, used, for instance, for task reallocation [15] and for resource reallocation [11], where the limited resources may be time, the computational resources of agents, or physical resources needed to carry out some tasks. Many approaches in the area of one-to-one negotiation are heuristic-based and, in spite of their experimentally proven effectiveness, they do not easily lend themselves to expressing theoretically provable properties. Other approaches present a good descriptive model, but fail to provide an execution model that can help to forecast the behaviour of any corresponding implemented system.

In this paper we tackle one-to-one negotiation between agents that share the same language for negotiation but may adopt different negotiation policies. We approach the problem in a logic-based manner that allows for proving properties such as correctness and completeness. We also propose an execution model to be used in order to implement the system. Although we concentrate on the specific problem of resource reallocation, our approach is more general, as far as the representation of policies and the reasoning that generates dialogues and their sequences, and could be extended to be applicable to other negotiation domains.

The framework we propose is based on agent dialogues, as a way to request resources, propose resource exchanges, suggest alternative resources, etc. We assume that the agents are equipped with a planner that, given a goal, is able to produce a sequence of actions to achieve it and an associated set of missing resources that are needed to carry out the plan. A single dialogue is used to obtain a resource. We describe an agent dialogue cycle that can be used to trigger a new dialogue after one is terminated, in order to allow the agent to collect all the resources needed to achieve its goal. In this way, the agents of a system produce sequences of dialogues, and in general sets of sequences of dialogues (a sequence for each agent that needs to obtain any missing resources).

We study some properties of dialogues, sequences of dialogues, and sets of sequences of dialogues in an agent system. We define the properties of termination, correctness and completeness, and show whether they hold for some concrete agent varieties we propose. Finally, we study how a heterogeneous system, i.e., a system composed by a variety of agents, can be used to implement resource reallocation policies. Note that we do not make any concrete assumption on the internal structure of agents, except for requiring that they hold beliefs, goals, intentions and, possibly, resources.

## 2 Preliminaries

Capital and lower-case letters stand for variables and ground terms, respectively.

---

**Definition 1** (*Performative or dialogue move*)
A *performative* or *dialogue move* is an instance of a schema $tell(X, Y, \mathbf{Subject}, T)$, where $X$ is the *utterer* and $Y$ is the *receiver* of the performative, and $T$ is the *time* when the performative is uttered. $\mathbf{Subject}$ is the *content* of the performative, expressed in some given *content language*.

---

A concrete example of a performative is $tell(a, b, \mathbf{request}(\mathbf{give}(nail)), 1)$, where $\mathbf{Subject}$ is $\mathbf{request}(\mathbf{give}(nail))$. Intuitively, this performative represents $a$'s request to $b$ for a nail, at time 1. We will often refer to a performative simply as $p(T)$, if we want to emphasise the time of the performative, or $p$.

---

**Definition 2** (*Language for negotiation*)
A *language for negotiation* $\mathcal{L}$ is a (possibly infinite) set of (possibly non ground) performatives. For a given $\mathcal{L}$, we define two (possibly infinite) subsets of performatives,

$\mathcal{I}(\mathcal{L})$, $\mathcal{F}(\mathcal{L}) \subseteq \mathcal{L}$, called respectively *initial moves* and *final moves*. Each final move is either *successful* or *unsuccessful*.

---

The following is a possible concrete language for negotiation:

$\mathcal{L}_1$ = { (1) $tell(X, Y, \mathbf{request}(\mathbf{give}(Resource)), T)$
(2) $tell(X, Y, \mathbf{accept}(Move), T)$
(3) $tell(X, Y, \mathbf{refuse}(Move), T)$
(4) $tell(X, Y, \mathbf{challenge}(Move), T)$
(5) $tell(X, Y, \mathbf{justify}(Move, Support), T)$
(6) $tell(X, Y, \mathbf{promise}(\mathbf{give}(Resource), \mathbf{give}(Resource')), T)$ }

where (4) is used by $X$ to ask $Y$ a reason (justification) for a past $Move$, (5) is used by $X$ to justify to $Y$ a past $Move$ by means of a $Support$, (6) is used by $X$ to propose a deal: $X$ will give $Resource'$ to $Y$ if $Y$ will give $Resource$ to $X$ (see Sadri et al. [14] for a more detailed description). The initial and final moves are:

$\mathcal{I}(\mathcal{L}_1) = \{tell(X, Y, \mathbf{request}(\mathbf{give}(Resource)), T)\}$
$\mathcal{F}(\mathcal{L}_1)$ = { [*successful moves:*]
$tell(X, Y, \mathbf{accept}(\mathbf{request}(\mathbf{give}(Resource))), T)$,
$tell(X, Y, \mathbf{accept}(\mathbf{promise}(\mathbf{give}(Resource), \mathbf{give}(Resource'))), T)$,
[*unsuccessful moves:*]
$tell(X, Y, \mathbf{refuse}(\mathbf{request}(\mathbf{give}(Resource))), T)$,
$tell(X, Y, \mathbf{refuse}(\mathbf{promise}(\mathbf{give}(Resource), \mathbf{give}(Resource'))), T)$ }.

As in this paper we are interested in negotiation for the exchange of resources, we will assume that there always exists a **request** move in the (initial moves of any) language for negotiation.

---

**Definition 3** (*Agent system*)
An *agent system* is a finite set $A$, where each $x \in A$ is a ground term, representing the name of an agent, equipped with a *knowledge base* $\mathcal{K}(x)$.

---

We will assume that, in an agent system, the agents share a common language for negotiation as well as a common content language.

In this paper, we presume a logic language for the knowledge base of agents, but make only three assumptions on the syntax of this language. In particular, we assume that it contains notions of *literal*, *complement* of sentences, *conjunction* of sentences, *true* and *false*, and that such language is equipped with a notion of *entailment*, such that, for every ground literal in the language, either the literal or its complement is entailed from the set of sentences that represent the knowledge of agents, and such that no literal and its complement are entailed at the same time. We also make concrete assumptions on the syntax of a component of the knowledge base, consisting of "dialogue constraints" (described in Section 3).

As far as the content of the knowledge base, we will assume that it describes, in addition to the dialogue constraints,

– domain-specific as well as domain-independent beliefs that the agent can use to generate *plans* for its *goal*, such as the knowledge describing preconditions and effects of actions. We will not make any assumptions on how such plans are gen-

erated; in particular, they could be obtained from a planner, either within the agent or external to it, or be extracted from a given library of plans, which is part of the knowledge of the agent;
– information about the resources available to the agent;
– information about the dialogues in which the agent has taken part;
– information on the selected *intention*, consisting of the given *goal*, a *plan* for the given goal, as well as the set of resources already *available* to the agent and the set of *missing* resources, both required for the plan to be executable.

Note that we assume that each agent is equipped with a *single (possibly conjunctive) goal*. For the sake of simplicity, we will also assume that this goal does not change during the dialogues. Such assumption could be relaxed, and our framework extended, for example by letting the agents' modify their goals after they have been achieved, or proven impossible to achieve, in accordance with the Rao and Georgeff's BDI theory [12] that requires goals to be believed feasible.

The purpose of negotiation is for the agent to obtain the missing resources, while retaining the available ones that are necessary for the plan in its current intention.

We do not make any assumptions on how the plan within an intention is obtained. However, we assume that the intention is such that the plan it contains allows the goal to be achieved, with respect to the knowledge base of the agent. We also assume that the missing resources in the intention and those available to the agent, as given by the knowledge base, have no intersection, and that the available resources in the intention are contained in the resources available to the agent, as given by the knowledge base.

In the sequel, we will refer to the knowledge base, intention and goal of an agent $x$ as $\mathcal{K}(x)$, $\mathcal{I}(x)$ and $\mathcal{G}(x)$, respectively, or simply as $\mathcal{K}$, $\mathcal{I}$ and $\mathcal{G}$, respectively, if clear from the context.

As an example, for some agent $x \in A$, the knowledge base $\mathcal{K}(x)$ might contain beliefs such as (0 stands for the initial time)

$have(picture, 0)$, $have(hammer, 0)$, $have(screwdriver, 0)$;

the goal $\mathcal{G}(x)$ might be $hung(picture)$, and the intention $\mathcal{I}(x)$ might be

$\{\ goal(\{hung(picture)\}),$
$plan(\{obtain(nail), hit(nail), hang(picture)\}, 0),$
$available(\{hammer, picture\}, 0),$
$missing(\{nail\}, 0)\ \}.$

This example is inspired by Parsons et al. [11]. There, two agents need to hang some objects (a picture the one, a mirror the other). Each agent could either use a hammer and a nail or a screw and a screwdriver, but neither agent owns both resources, hence the need for negotiation. The agent's knowledge is time-stamped as it might change over time, e.g., by acquisition/loss of resources. Moreover, as we will see in the remainder of the paper, the intention might also change as a result of negotiation, when resources are exchanged. For instance, after two agents, negotiating with the language $\mathcal{L}_1$, agree on a promise, both agents' knowledge bases will be different, since they will have different resources, and possibly different plans. Finally, note that we assume that actions within plans are implicitly temporally ordered (left-to-right). For simplicity, we do not address the issue of the time of actions and goals in this paper. For a more concrete description of the knowledge base of agents see [14].

In this paper, we will assume that (any ground instance of) the literal $have(R, T)$ holds (namely it is entailed by $\mathcal{K}(x)$, for the given agent $x$) if the agent has the resource $R$ at the time $T$ of interest, and (ground instances of) the literal $need(R, T)$ holds if the agent's intention identifies $R$ as $available$.

In the sequel, given any intention $\mathcal{I}$,

- $goal(G) \in \mathcal{I}$ will stand for '$G$ is the goal in $\mathcal{I}$'
- $plan(P, T) \in \mathcal{I}$ will stand for '$P$ is the plan in $\mathcal{I}$, time-stamped with time $T$'
- $missing(Rs, T) \in \mathcal{I}$ will stand for '$Rs$ are the resources needed to make the plan in $\mathcal{I}$ executable but currently missing, time-stamped with time $T$'
- $available(Rs, T) \in \mathcal{I}$ will stand for '$Rs$ are the resources needed to make the plan in $\mathcal{I}$ executable and currently available, time-stamped with time $T$'.

We will omit the time $T$ if clear from the context.

## 3 Dialogues

For a given agent $x \in A$, where $A$ is equipped with $\mathcal{L}$, we define the sets

- $\mathcal{L}^{in}(x)$, of all performative schemata of which $x$ is the receiver, but not the utterer;
- $\mathcal{L}^{out}(x)$, of all performative schemata of which $x$ is the utterer, but not the receiver.

Note that we do not allow for agents to utter performatives to themselves. In the sequel, we will often omit $x$, if clear from the context, and simply write $\mathcal{L}^{in}$ and $\mathcal{L}^{out}$.

Negotiation policies can be specified by sets of dialogue constraints:

---

**Definition 4** (*Dialogue constraint*)
Given an agent system $A$, equipped with a language for negotiation $\mathcal{L}$, and an agent $x \in A$, a *dialogue constraint* for $x$ is a (possibly non-ground) if-then rule of the form: $p(T) \wedge C \Rightarrow \hat{p}(T+1)$, where

- $p(T) \in \mathcal{L}^{in}(x)$ and $\hat{p}(T+1) \in \mathcal{L}^{out}(x)$,
- the utterer of $p(T)$ is the receiver of $\hat{p}(T+1)$, and the receiver of $p(T)$ is the utterer of $\hat{p}(T+1)$,
- $C$ is a conjunction of literals in the language of the knowledge base of $x$. [1]

Any variables in a dialogue constraint are implicitly universally quantified from the outside. The performative $p(T)$ is referred to as the *trigger*, $\hat{p}(T+1)$ as the *next move* and $C$ as the *condition* of the dialogue constraint.

---

Intuitively, the dialogue constraints of an agent $x$ express policies between $x$ and any other agent. The intuitive meaning of a dialogue constraint $p(T) \wedge C \Rightarrow \hat{p}(T+1)$ of agent $x$ is as follows: if at a certain time $T$ in a dialogue some other agent $y$ utters a performative $p(T)$, then the corresponding instance of the dialogue constraint is triggered

[1] Note that $C$ in general might depend on several time points, possibly but not necessarily including $T$; therefore we will not indicate explicitly any time variable for it.

and, if the condition $C$ is entailed by the knowledge base of $x$, then $x$ will utter $\hat{p}(T+1)$, with $y$ as receiver, at the next time $T+1$. This behaviour of dialogue constraints can be achieved by employing an automatic abductive proof procedure such as Fung and Kowalski's IFF [4] within an *observe-think-act* agent cycle [8], as discussed in [14]. The procedure implements a concrete concept of entailment with respect to knowledge bases expressed in abductive logic programming terms [7]. The execution of the proof procedure within the agent cycle allows agents to produce dialogue moves immediately after a dialogue constraint is fired.

The use of an abductive proof procedure, equipped with an operational semantics and theoretical results, has several advantages, beside the availability of an execution model. In particular, the proofs of the theorems stated later in the paper are implicitly grounded on the property of correctness, which holds for the IFF proof-procedure.

An example of a dialogue constraint allowing an agent $x$ to accept a request is:
$tell(Y, x, request(give(R)), T) \wedge [have(R, T) \wedge not\ need(R, T)]$
$\Rightarrow tell(x, Y, accept(request(give(R))), T+1)$
where the trigger is $tell(Y, x, request(give(R)), T)$, the condition is within square brackets and the next move is $tell(x, Y, accept(request(give(R))), T+1)$.

This is a simple example, but in general $C$ might contain other dialogue performatives, uttered in the past by other agents, within the same dialogue or other dialogues. For instance, one could decide to accept a request only if another agent made a certain move in the past.

We will refer to the set of dialogue constraints associated with an agent $x \in A$ as $\mathcal{S}(x)$, and we will call it the *agent program* of $x$. We will often omit $x$ if it is clear from the context or unimportant. As already mentioned in Section 2, the agent program of an agent is part of its knowledge base $\mathcal{K}$.

Note that different agents might be equipped with different agent programs.

---

**Definition 5** (*Dialogue*)
A *dialogue* between two agents $x$ and $y$ is a set of ground performatives, $\{p_0, p_1, p_2, \ldots\}$, such that, for some given $t \geq 0$:

1. $\forall\ i \geq 0$, $p_i$ is uttered at time $t+i$;
2. $\forall\ i \geq 0$, if $p_i$ is uttered by agent $x$ (resp. $y$), then $p_{i+1}$ (if any) is uttered by agent $y$ (resp. $x$);
3. $\forall\ i > 0$, $p_i$ can be uttered by agent $\alpha \in \{x, y\}$ only if there exists a (grounded) dialogue constraint $p_{i-1} \wedge C \Rightarrow p_i \in \mathcal{S}(\alpha)$ such that $\mathcal{K}(\alpha) \wedge p_{i-1}$ entails $C$.

A dialogue $\{p_0, p_1, \ldots p_m\}$, $m \geq 0$, is *terminated* if $p_m$ is a ground final move, namely $p_m$ is a ground instance of a performative in $\mathcal{F}(\mathcal{L})$.

---

By condition 1, a dialogue is in fact a *sequence* of performatives. By condition 2, agents alternate utterances in a dialogue. By condition 3, dialogues are generated by the dialogue constraints, together with the given knowledge base to determine whether the condition of triggered dialogue constraints is entailed.

Intuitively, a dialogue should begin with an initial move, according to the given language for negotiation. The kind of dialogue that is relevant to our purposes is that started with a request for a resource $R$. Such a dialogue will be initiated by an agent $x$ whose intention $\mathcal{I}$ contains $R$ in its set of *missing* resources.

---

**Definition 6** (*Request dialogue*)
A *request dialogue* with respect to a resource $R$ and an intention $\mathcal{I}$ of agent $x$ is a dialogue $\{p_0, p_1, p_2, \ldots\}$ between $x$ and some agent $y \in A$ such that

- $p_0 = tell(x, y, request(give(R)), t)$,
- $missing(Rs, t) \in \mathcal{I}$ and
- $R \in Rs$,

for some $t \geq 0$.

---

In general, after a terminated request dialogue $d$ with respect to a resource $R$ and an intention $\mathcal{I}$ of some agent $x$, the agent might have been successful or not in obtaining the resource $R$. If it has been successful, then its intention changes so that $R$ is not missing anymore, and the agent will engage in new dialogues to obtain the remaining missing resources, if any, in the intention. If $d$ has been unsuccessful, then the agent's intention does not change, and the agent might decide to engage in a dialogue to obtain $R$ from a different agent. In both cases, the plan in the intention does not change. We also allow a third possibility, of a successful termination with a change of intention possibly (and typically) involving a change of plan. This could happen, for example, if during the dialogue the other agent persuaded the requesting agent to modify its plan, with the promise of a different resource deal. In the sequel, we will assume that *a terminated request dialogue, for a given resource $R$ and intention $\mathcal{I}$, returns an intention $\mathcal{I}'$.*

---

**Definition 7** (*Types of terminated request dialogues*)
Let $\mathcal{I}$ be the intention of some agent $a$, and $R$ be a missing resource in $\mathcal{I}$. Let $d$ be a terminated resource dialogue with respect to $R$ and $\mathcal{I}$, and $\mathcal{I}'$ be the intention resulting from $d$. Then, if $missing(Rs), plan(P) \in \mathcal{I}$ and $missing(Rs'), plan(P') \in \mathcal{I}'$:

$i)$ $d$ is *successful* if $P' = P$, $Rs' = Rs \setminus \{R\}$;
$ii)$ $d$ is *conditionally* or *c-successful* if $Rs' \neq Rs$ and $Rs' \neq Rs \setminus \{R\}$;
$iii)$ $d$ is *unsuccessful* if $\mathcal{I}' = \mathcal{I}$.

Note that, in the case of c-successful dialogues, typically, but not always, the agent's plan will change ($P' \neq P$).

---

In [14], we impose that terminated request dialogues return intentions whose missing resources are no more than those in the original intention, in order to ensure the termination of the negotiation process of an agent, achieved by dialogue sequences. Here, we generalise this property to that of 'convergence':

---

**Definition 8** (*Convergence*)
An agent $x \in A$ is *convergent* iff, for every terminated request dialogue of $x$, with

respect to some resource $R$ and some intention $\mathcal{I}$, the *cost* of the returned intention $\mathcal{I}'$ is not higher than the *cost* of $\mathcal{I}$.

The *cost* of an intention can be defined as the number of missing resources in the intention.

## 4 Properties of Agent Programs

In this section we consider an individual agent $x \in A$, equipped with a language for negotiation $\mathcal{L}$ and an agent program $\mathcal{S}$ that, for the purposes of this section, we assume to be grounded.

An agent program should ideally guarantee that the agent's dialogues terminate, and that the agent is able to produce one and only one move at a time, in finite time, in response to any non-final move of other agents. In this section, we show how some such desirable properties can be ensured. We discuss the 'termination' property in an earlier paper [14] and in a companion paper [18].

Given a *grounded* agent program $\mathcal{S}$ and a *ground* performative $p$, we define the set $\mathcal{S}(p)$ of *conditions* associated with $p$:

$\mathcal{S}(p) \stackrel{def}{=} \{C \mid \text{ there exists } (p \wedge C \Rightarrow \hat{p}) \in \mathcal{S}\}$.

**Definition 9** (*Determinism*)
An agent $x \in A$ is *deterministic* iff, for each performative $p(t)$ which is a ground instance of a schema in $\mathcal{L}^{in}$, there exists at most one performative $\hat{p}(t+1)$ which is a ground instance of a schema in $\mathcal{L}^{out}$ such that $p(t) \wedge C \Rightarrow \hat{p}(t+1) \in \mathcal{S}$ and $\mathcal{K} \wedge p(t)$ entails $C$.

Namely, an agent is deterministic if for every given performative, at most one of the dialogue constraints triggered by it actually fires, in that it produces a next move.

Determinism can be guaranteed for "non-overlapping" agent programs:

**Definition 10** (*Non-overlapping agent program*)
$\mathcal{S}$ is *non-overlapping* iff for each performative $p$ which is a ground instance of a schema in $\mathcal{L}^{in}$, for each $C, C' \in \mathcal{S}(p)$ such that $C \neq C'$, then $C \wedge C' \equiv false$.

Namely, an agent program is non-overlapping if, for any given incoming dialogue move, there do not exist any two triggered dialogue constraints firing at the same time.

**Theorem 1** If the (grounded) agent program of $x$ is non-overlapping, then $x$ is deterministic.

***Proof*** (Sketch): for each incoming performative there exists at most one triggered dialogue constraint whose conditions are true, thus $\mathcal{K}$ only entails at most the condition of one constraint. ■

In order for a dialogue not to end before a final move is reached, it is important that an agent be "exhaustive":

**Definition 11** (*Exhaustiveness*)
An agent $x \in A$ is *exhaustive* iff, for each performative $p(t)$ which is a ground instance of a schema in $\mathcal{L}^{in} \setminus \mathcal{F}(\mathcal{L})$, there exists at least one performative $\hat{p}(t+1)$ which is a ground instance of a schema in $\mathcal{L}^{out}$ such that $p(t) \wedge C \Rightarrow \hat{p}(t+1) \in \mathcal{S}$ and $\mathcal{K} \wedge p(t)$ entails $C$.

Namely, an agent is exhaustive if, for every given performative, at least one of the dialogue constraints triggered by it actually fires, in that it produces a next move.

Exhaustiveness can be guaranteed for "covering" agent programs:

**Definition 12** (*Covering agent program*)
Let $\mathcal{L}(\mathcal{S})$ be the set of all (not necessarily ground) performatives $p(T)$ that are triggers in dialogue constraints:
$\mathcal{L}(\mathcal{S}) \stackrel{def}{=} \{p(T) \mid \text{ there exists } p(T) \wedge C \Rightarrow \hat{p}(T+1) \in \mathcal{S}\}$. (Obviously, $\mathcal{L}(\mathcal{S}) \subseteq \mathcal{L}^{in}$)
Then, $\mathcal{S} \neq \emptyset$ is *covering* iff for every performative $p$ which is a ground instance of a schema in $\mathcal{L}^{in}$, $\bigvee_{C \in \mathcal{S}(p)} C \equiv true$ and $\mathcal{L}(\mathcal{S}) = \mathcal{L}^{in} \setminus \mathcal{F}(\mathcal{L})$.

Namely, an agent program is covering if, for any given incoming dialogue move that is not a final move, there exists at least one triggered dialogue constraint which actually fires.

**Theorem 2** If the (grounded) agent program of $x$ is covering, then $x$ is exhaustive.

***Proof*** (Sketch): for each incoming performative there exists at least one dialogue constraint which is triggered and whose condition is true, thus $\mathcal{K}$ entails the condition of at least one constraint. ■

If an agent is exhaustive and deterministic, then at each step of the dialogue the agent will produce *exactly one* reply to a (non final) move made by the other agent. If both agents involved in a dialogue are exhaustive and deterministic, then at each step exactly one agent is guaranteed to produce only one dialogue move, unless a final move is made. [2] The IFF proof procedure and the agent cycle have features that allow us to produce dialogues as defined earlier, given agent programs that are exhaustive and deterministic.

## 5 Agent Varieties: Concrete Examples of Agent Programs

In this section we show some concrete agent programs and we discuss which properties hold for them and which ones do not. All the examples refer to a given agent $a$, equipped with a language for negotiation $\mathcal{L}_2 = \mathcal{I}(\mathcal{L}_2) \cup \mathcal{F}(\mathcal{L}_2)$, where:

- $\mathcal{I}(\mathcal{L}_2) = \{\ tell(X, Y, \mathbf{request}(\mathbf{give}(Resource)), T)\ \}$;
- $\mathcal{F}(\mathcal{L}_2) = \{\ tell(X, Y, \mathbf{accept}(Move), T), tell(X, Y, \mathbf{refuse}(Move), T)\ \}$.

[2] Note that we use *transaction*, rather than *real*, time, i.e., we assume that there is no time-point between two *consecutive* dialogue moves in a dialogue.

**Agent program** `P0`

$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ have(R, T)$
$\Rightarrow tell(a, X, \mathbf{accept}(\mathbf{request}(\mathbf{give}(R))), T+1)$
$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ need(R, T)$
$\Rightarrow tell(a, X, \mathbf{refuse}(\mathbf{request}(\mathbf{give}(R))), T+1)$

This program is neither deterministic nor exhaustive. In fact, if $a$ has and needs the resource that is requested at time $T$, the program allows to return two answers at time $T+1$, namely both $accept$ and $refuse$, while if $a$ does not have $R$, then the program will not be able to return any answer at time $T+1$.

**Agent program** `P1`

$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ have(R, T)$
$\Rightarrow tell(a, X, \mathbf{accept}(\mathbf{request}(\mathbf{give}(R))), T+1)$
$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ not\ have(R, T)$
$\Rightarrow tell(a, X, \mathbf{refuse}(\mathbf{request}(\mathbf{give}(R))), T+1)$

An agent programmed with `P1` simply gives away the requested resource if it has it, no matter whether it needs it or not. This program is exhaustive and deterministic.

**Agent program** `P2`

$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ have(R, T) \ \wedge \ not\ need(R, T)$
$\Rightarrow tell(a, X, \mathbf{accept}(\mathbf{request}(\mathbf{give}(R))), T+1)$
$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ not\ have(R, T)$
$\Rightarrow tell(a, X, \mathbf{refuse}(\mathbf{request}(\mathbf{give}(R))), T+1)$
$tell(X, a, \mathbf{request}(\mathbf{give}(R)), T) \ \wedge \ need(R, T)$
$\Rightarrow tell(a, X, \mathbf{refuse}(\mathbf{request}(\mathbf{give}(R))), T+1)$

An agent programmed with `P2` gives away the requested resource only if it has it and it does not need it. This program is exhaustive and deterministic.

## 6 Dialogue Sequences

In this section we model the negotiation process of an agent by means of "sequences of (request) dialogues" with respect to an intention, $I$, of a given agent $x$. The request dialogues aim at obtaining *all* the missing resources in $I$. We assume that all dialogues between $x$ and any other agents are *atomic*, i.e., it is not possible for $x$ to be engaged in two different dialogues at the same time.

In order to start a request dialogue with respect to an intention $I$, the set of missing resources of $I$ must be non-empty, i.e., $missing(Rs) \in I, Rs \neq \emptyset$.

Intuitively, given a goal $G$ such that $goal(G) \in I, missing(Rs) \in I$ and $plan(p) \in I$, a sequence of dialogues with respect to $I$ aims at decreasing the cost of $I$, either by obtaining one by one all the missing resources and making $p$ executable, or by finding an alternative plan (and thus an alternative intention) which can be made executable. Indeed, after each dialogue, the intention might change. We associate with a dialogue sequence $d_1, d_2, \ldots, d_n$ the corresponding sequence of intentions $I_1 = I, I_2, \ldots, I_{n+1}$. In general, after a request dialogue $d_i, i < n$, with respect to an intention $I_i$ and a resource $R$, is terminated, the agent will have an intention $I_{i+1}$. All such intentions have the same goal, but possibly different plans and missing/available resources.

**Definition 13** (*Sequence of dialogues*)
A *sequence of dialogues* $s(I)$ with respect to an intention $I$ of an agent $x$ with $goal(G) \in I$ is an ordered set $\{d_1, d_2, \ldots, d_n, \ldots\}$, associated with a sequence of intentions $I_1, I_2, \ldots, I_{n+1}, \ldots$ such that

- for all $1 \leq i$, $d_i$ is a request dialogue between $x$ and some other agent (not necessarily the same for all $i$), with respect to a resource $R_i$ such that $R_i \in Rs_i$, $missing(Rs_i) \in I_i$, $I_i$ is the intention of agent $x$, $goal(G) \in I_i$, and $I_{i+1}$ is the intention after dialogue $d_i$;
- $I_1 = I$;

We are interested in dialogue sequences that terminate:

**Definition 14** (*Termination of a dialogue sequence*)
A sequence of dialogues $\{d_1, d_2, \ldots, d_n\}$ with respect to an initial intention $I$ of an agent $x$ and associated with the sequence of intentions $I_1, I_2, \ldots, I_{n+1}$ is *terminated* iff there exists no possible request dialogue with respect to $I_{n+1}$ that $x$ can start.

Termination could occur for two reasons: either there are no missing resources in $I_{n+1}$ or $x$ could not obtain all the resources in $I_{n+1}$ but "decided" to give up the intention without forming a new one for the time being. Accordingly, we can assess the success of dialogue sequences:

**Definition 15** (*Success of a dialogue sequence*)
A terminated sequence of dialogues $\{d_1, d_2, \ldots, d_n\}$ with respect to an initial intention $I$ of an agent $x$ and associated with a sequence of intentions $I_1, I_2, \ldots, I_{n+1}$ is *successful* if $I_{n+1}$ has an empty set of missing resources; it is *unsuccessful* otherwise.

In the remainder of this section we will discuss how dialogue sequences can be generated by agents, by an "agent dialogue cycle" which repeatedly generates individual dialogues until a terminated sequence is produced. We will call $SD$ and $SI$, respectively, the sequence of dialogues and intentions, produced within the dialogue cycle:

**Definition 16** (*Agent dialogue cycle*)
Given an initial intention $I$ of agent $x$, containing a set of missing resources $Rs$, the *agent dialogue cycle* is the following:

1. set $SD = \emptyset$ and $SI$ to $\{I\}$;
2. until $Rs = \emptyset$ repeat
   (a) select a resource $R \in Rs$ and update $Rs$ to $Rs - \{R\}$;
   (b) until a successful or c-successful dialogue $d$ is found, between $x$ and some other agent, repeat

i. select an agent $y \in A - \{x\}$ such that there does not exist in $SD$ a request dialogue between $x$ and $y$ with respect to $R$;
ii. if such $y$ does not exist, then **failure** and restore $Rs$ to $Rs \cup \{R\}$;
iii. construct a request dialogue $d$ between $x$ and $y$ with respect to $R$ and $I$, returning an intention $I'$;

(c) add $d$ to $SD$ and $I'$ to $SI$;
(d) update $I$ to $I'$ and $Rs$ to $Rs'$ such that $missing(Rs') \in I'$;

3. **success**.

---

Note that the above agent dialogue cycle does not allow for the plan in an intention to change unless the change is achieved by an individual dialogue between $x$ and the chosen $y$.

Note also that the above agent dialogue cycle has the following properties:

- no agent is asked twice for the same resource within the dialogue sequence;
- if a resource is not obtained from one agent, then it is asked from some other agent, if any;
- if a resource is not obtained after asking all agents, then the agent dialogue cycle terminates with **failure**.

The idea behind the third property is that the agent will not carry on asking for the other resources, since, as at least one resource in the current intention cannot be obtained, the intention will not possibly be executable.

**Theorem 3** Given an agent $x \in A$, if $x$'s agent dialogue cycle returns **success** then there exists a successful dialogue sequence with respect to the initial intention $I$ of $x$.

***Proof*** (Sketch): The dialogue sequence $SD$, associated with the sequence of intentions $SI$, as generated by the agent dialogue cycle at step 2(c) during the last execution of the loop at step 2, is a successful dialogue sequence. ■

If the agent $x \in A$ is convergent, then all dialogues of a sequence with respect to an intention $I$ of $x$ will result in a non-increased cost of $x$'s intention. Thus, we can set an upper bound to the number of dialogues in a sequence initiated by $x$:

**Theorem 4** Given an agent $x$ with intention $I$, and a successful dialogue sequence $s(I)$ generated by $x$'s dialogue cycle, if $x$ is convergent, then the number of dialogues in $s(I)$ is bounded by $m \cdot |Rs|$, where $missing(Rs) \in I$ and $|A \setminus \{x\}| = m$, $A$ being the set of agents in the system.

***Proof*** (Sketch): $|Rs|$ can never increase, since $x$ is convergent. In the worst case, $x$ will have to request each resource $R \in Rs$ to all the other $m$ agents of the system, thus generating $m$ dialogues with respect to each resource. The cardinality of the set of missing resources decreases by only one unit after each successful dialogue. ■

## 7 Using Dialogue Sequences for Resource Reallocation

In this section, we study the use of multiple sequences of dialogues for the solution of the well-known and widely studied *resource reallocation problem* [2, 16]. The problem can be rephrased in our system as follows.

**Definition 17** (*Resource reallocation problem – r.r.p.*)
Given an agent system $A$, with each agent $x \in A$ equipped with a knowledge base $\mathcal{K}(x)$ and an intention $\mathcal{I}(x)$,

- the *r.r.p. for an agent* $x \in A$ is the problem of finding a knowledge base $\mathcal{K}'(x)$, and an intention $\mathcal{I}'(x)$ (for the same goal as $\mathcal{I}(x)$) such that $missing(\emptyset) \in \mathcal{I}'(x)$.
- the *r.r.p. for the agent system* $A$ is the problem of solving the r.r.p. for every agent in $A$.

A *r.r.p.* is *solved* if the required (sets of) knowledge base(s) and intention(s) is (are) found.

---

The new knowledge bases will differ from the old ones in the resources available to the agents and in the intentions. Typically, agents need to acquire new resources from other agents so that their intentions have no missing resources. In our framework, agents can obtain the new resources by means of dialogue sequences.

Note that, the knowledge and intention of an agent $x$ might change not only as a result of dialogues which have been initiated by $x$, but also as a result of dialogues initiated by other agents, but engaging $x$. Indeed, in such more 'passive' dialogues, $x$ might agree to give away a resource that it needs. This 'double change' of knowledge and intentions (of both agents involved in a dialogue) plays a role when agents need to engage in multiple concurrent dialogue sequences in order to achieve their goal. [3] In the remainder of this section, we will assume *generalised notions of (terminated) dialogue*, modifying the intentions of both agents engaged in it, and *dialogue sequence*, modifying the intentions of all agents involved in all dialogues in it. Then, the notion of *convergence* can be generalised as well, so that the cost of both agents' intentions returned by a dialogue is not increased, with respect to the cost of the agents' initial intentions. Note that convergent agents, in this generalised sense, correspond to *self-interested individual rational* agents as proposed in the literature (see Sandholm [15]).

We consider two properties of agents:

- *correctness*: if the dialogue cycle of some agent $x$ returns success, then the r.r.p of $x$ is solved, and, if the dialogue cycle of every agent in $A$ returns success, then the r.r.p. of the agent system is solved;
- *completeness*: if there exists a solution to the r.r.p of an agent system, then the dialogue cycle of every agent in $A$ returns success, producing such a solution.

The proof of correctness for a single agent is straightforward, by definition of successful dialogue sequence. In the case of an agent system, instead, correctness as given above can be proven only for convergent agents.
**Theorem 5** (*Correctness of the agent dialogue cycle with respect to the r.r.p.*) Let $A$ be the agent system, with the agent programs of all the agents in $A$ being convergent. If

[3] Note that this is not in contrast with the assumption that agents engage in at most one dialogue at a time. As we assume that dialogues are atomic, there will always be at most one dialogue taking place at any time.

all agent dialogue cycles of all agents in $A$ return **success** then the r.r.p. for the agent system is solved.
***Proof*** (Sketch): Since all agent programs are convergent, there will never occur a dialogue after which, and as a consequence of which, either of the two agents involved has more missing resources in its intention than it had before. Therefore, having $m$ successful dialogue sequences, one for each agent in $A$, is in the worst case equivalent to having $m$ *independent* dialogue sequences, after each of which the relevant agent will have solved its r.r.p., thus solving the r.r.p. of $A$. ■

In order to address completeness we need to deal with the issue of how intentions change, which is beyond the scope of this paper. But to illustrate the problem, consider the following example. Let $A = \{a, b, c\}$ be an agent system, with knowledge bases containing:
$a : \mathcal{R}(a) = \emptyset, \mathcal{I}(a) = \{ \texttt{plan}(\{\texttt{p(a)}\}), \texttt{missing}(\{\texttt{r}_2\}), \texttt{available}(\emptyset), \texttt{goal}(\{\texttt{g(a)}\})\}$
$b : \mathcal{R}(b) = \{r_2\}, \mathcal{I}(b) = \{ \texttt{plan}(\{\texttt{p(b)}\}), \texttt{missing}(\{\texttt{r}_3\}), \texttt{available}(\{\texttt{r}_2\}), \texttt{goal}(\{\texttt{g(b)}\})\}$
$c : \mathcal{R}(c) = \{r_3, r_4\}, \mathcal{I}(c) = \{ \texttt{plan}(\{\texttt{p(c)}\}), \texttt{missing}(\emptyset), \texttt{available}(\{\texttt{r}_3\}), \texttt{goal}(\{\texttt{g(c)}\})\}$
where $\mathcal{R}(x)$ stands for the resources available to agent $x$, as described within $x$'s knowledge base. Let us assume that there exists an alternative plan $\texttt{p}'\texttt{(b)}$ for $b$'s goal $\texttt{g(b)}$, and therefore an alternative intention, $\mathcal{I}'(b)$: $\mathcal{I}'(b) = \{ \texttt{plan}(\{\texttt{p}'\texttt{(b)}\}), \texttt{missing}(\{\texttt{r}_4\}), \texttt{available}(\emptyset), \texttt{goal}(\{\texttt{g(b)}\})\}$ It is easy to see that there exists a possible resource reallocation that makes all goals achievable ($r_2$ to $a$, $r_4$ to $b$ and $r_3$ to $c$), but in order to be found by negotiation, $b$ must be able to change its intention to $\mathcal{I}'(b)$.

We consider a weak notion of completeness, whereby agents have, as their own initial intentions, the intentions whose plan does not need to be changed by negotiation.

---

**Definition 18** (*Weak completeness*)
Let $A$ be an agent system consisting of $n$ agents. Let $\mathcal{R}(A)$ be the union of all resources held by all agents in $A$, and $\mathcal{R}(\mathcal{I}(A))$ be the union of all resources needed to make all agents' initial intentions $\mathcal{I}(A)$ executable.

$A$ is *weakly complete* if, given that $\mathcal{R}(\mathcal{I}(A)) \subseteq \mathcal{R}(A)$, then **there exist** $n$ successful dialogue sequences, one for each agent in $A$, such that the intentions $\mathcal{I}'(A)$ returned by the sequences have the same plans as $\mathcal{I}(A)$ and all have an empty set of missing resources.

---

**Theorem 6** (*weak completeness with respect to* `P1 + P2`) Let $A$ be an agent system consisting of $n$ agents. Then, if the agent program of every agent in $A$ is either `P1` or `P2`, then $A$ is weakly complete.
***Proof*** (Sketch): The sub-system composed by the agents programmed with `P2` is weakly complete, therefore the agents programmed with `P2` will obtain all missing resources, either from the agents programmed with `P1` or from the ones programmed with `P2`. The agents programmed with `P1` will obtain the missing resources either from the agents programmed with `P2` that have such resources and do not need them, or by the agents programmed with `P1`, which represent also a weakly complete sub-system. ■
Thus, we have shown that the r.r.p. can be solved within our approach, by diverse agents,

equipped with different agent programs, negotiating with one another. Note that such diversity can be used to implement a concept of priority between agents. Indeed, agents programmed with `P2` always retain all resources they need, whereas agents programmed with `P1` give them away, if asked. Thus, `P2`-agents have a higher priority than `P1`-agents, with respect to the allocation of resources.

## 8 Conclusions

This paper focused on one-to-one agent negotiation for achieving goals in a system with limited resources. The solution proposed is based on agent dialogues, as a way of requesting resources, proposing resource exchanges, and suggesting alternative resources. The negotiation framework presented is a general one for representing policies and generating dialogues. The resource reallocation problem is one possible application of this framework. This framework could be deployed to tackle other problems, e.g. to deal with dialogues starting with an offer or an advertisement rather than a request.

We assume that the agents are provided with a planner that, given a goal, is able to produce a sequence of actions to achieve it and an associated set of missing resources that are needed to carry out the actions. The framework proposed is independent of the planner, the specific application domain, the agent programs that regulate the dialogues, and even the language that agents use to communicate.

In the domain of resource reallocation, a single dialogue is used to obtain a resource. Our approach is logic based, and, in particular, the agents involved in a dialogue can use an abductive proof procedure to reason about their current beliefs and intentions, and produce a dialogue move. An agent dialogue cycle is used produce sequences of dialogues, and in general sets of sequences of dialogues in order for the agent to collect all the resources it needs to achieve its goal.

The logic-based approach allows us to define and prove some interesting properties of dialogues, sequences of dialogues, and sets of sequences of dialogues in an agent system. In particular, the adoption of an abductive proof procedure for which termination, correctness and completeness results hold, allows us to extend such general results to the specific case of a problem defined and addressed in the paper, the resource reallocation problem. We show various agent programs that can be used to tackle such a problem, and discuss to what extent the properties of termination, correctness and completeness, as defined in this paper, hold for them.

Introduction and surveys of automated negotiation can be found in [13] and [9, 6], respectively. An approach to negotiation via dialogue, that makes use of an argumentation system, is that of Sycara [17], and more recently Kraus et al. [10], where the authors adopt a modal logic approach and focus on the mental states of agents that move towards an agreement and encourage cooperation. The authors' formalisation of dialogues makes use of rules that, similarly to our dialogue constraints, are triggered when a performative is received, and may lead to another dialogue move under certain conditions, within an agent cycle. The execution is intertwined with the dialogue, allowing the agents to express threats, promises of rewards, etc. and to maintain a mental model of the other partners. Such an approach, focused on persuasion, leads to different results from ours, being somewhat more descriptive, and not aiming to determine specific properties of the negotiation policy.

Argumentation-based approaches to agent negotiation, that are in a way more general and less operational than ours, but are still very relevant, are those of Amgoud et al. [1] and Parsons et al. [11], from which we inherited the nail-and-hammer example.

Zlotkin and Rosenschein [19] adopt a game theory approach to the study of automated negotiation. However, their work – as most others that are game theory-based – considers agents that are not cooperative, and focuses on conflict resolution rather than collaboration. One of the interesting issues they address is the disclosure of private information. Our work is somewhat orthogonal, aiming at obtaining the resources the agents need, without taking into account the problems that could arise from disclosing private information. We agree that this is very important, especially in contexts such as e-commerce and virtual enterprises.

The use of negotiation to tackle the resource reallocation problem is not a new idea; among other contributions we cite that by Faratin [3], Sandholm [15], and Sathi and Fox [16]. Most of them study the problem adopting a game-theoretic approach and prove results that we consider orthogonal to ours.

This paper contributes to a number of issues. The introduction of a logic-based framework for negotiation allows us to prove properties that we can use to forecast the behaviour of a system of agents with no need for simulation and experimental results. The paper suggests an execution model that relies on the *observe-think-act* agent cycle of Kowalski and Sadri [8], and on the adoption of the IFF abductive proof procedure of Fung and Kowalski [4]. In this way, the implementation of the system, at least as far as a prototype, is straightforward, and does not require much more than writing high-level dialogue rules of the kind presented in the paper. This work extends our previous work [14] by dealing with dialogue sequences, discussing the application of the framework to a general resource reallocation problem, and proposing varieties of agents that can tackle such a problem.

There are a number of open issues. So far we assumed that agents do not change their goals. A natural extension would be to take into account the possibility of modifying goals as a consequence of an unsuccessful dialogue sequence, in accordance with the BDI theory that requires goals to be believed feasible.

Negotiation of sets of resources, rather that single items, is another interesting problem, as is the use of broadcast or multi-cast primitives as in electronic auctions.

It would also be interesting to study different cost functions for intentions, and to see what their impact is on the dialogues and on the dialogue properties. Currently, in moving from one intention to another, as a consequence of a deal (the acceptance of a $promise$), an agent can choose a plan that turns out not to be feasible, because of the lack of resources in the system. A more sophisticated cost function than the one proposed in the paper could result in the agent ensuring that all the resources for a new plan exist somewhere in the system, before agreeing to a deal.

## Acknowledgements

Thanks to Kostas Stathis and Paolo Mancarella for helpful discussions, and to the anonymous referees for useful comments.

## References

1. L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue and negotiation. In W. Horn, editor, *Proceedings 14th European Conference on Artificial Intelligence, Berlin, Germany*. IOS Press, August 2000.
2. M. Barbuceanu and W. Lo. A multi-attribute utility theoretic negotiation architecture for electronic commerce. *Proceedings of the Fourth International Conference on Autonomous Agents, Barcelona, Spain*, pages 239–247, 2000.
3. P. Faratin. *Automated Service Negotiation Between Autonomous Computational Agents*. PhD thesis, Department of Electronic Engineering, Queen Mary College, University of London, UK, 2000.
4. T. H. Fung and R. A. Kowalski. The IFF proof procedure for abductive logic programming. *Journal of Logic Programming*, 1997.
5. N. R. Jennings. Automated haggling: Building artificial negotiators (invited talk). In *AISB'01 Convention, York, UK*, March 2001.
6. N. R. Jennings, P. Faratin, A.R. Lomuscio, S. Parsons, C. Sierra, and M. J. Wooldridge. Automated Negotiation: Prospects, Methods and Challenges. *International Journal of Group Decision and Negotiation*, 10(2), 2001.
7. A. C. Kakas, R. A. Kowalski, and F. Toni. The role of abduction in logic programming. *Handbook of Logic in AI and Logic Programming*, 5:235–324, 1998.
8. R. A. Kowalski and F. Sadri. From logic programming to multi-agent systems. *Annals of Mathematics and AI*, 1999.
9. S. Kraus. *Strategic Negotiation in Multi-Agent Environments*. MIT Press, Cambridge, MA, 2000.
10. S. Kraus, K. Sycara, and A. Evenchik. Reaching agreements through argumentation; a logical model and implementation. *Artificial Intelligence*, 104:1–69, 1998.
11. S. Parsons, C. Sierra, and N. R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.
12. A. Rao and M. Georgeff. An abstract architecture for rational agents. In *Proceedings of the International Workshop on Knowledge Representation (KR'92)*, 1992.
13. J. S. Rosenschein and G. Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation Among Computers*. MIT Press, Cambridge, Massachusetts, 1994.
14. F. Sadri, F. Toni, and P. Torroni. Logic agents, dialogues and negotiation: an abductive approach. In *Proceedings AISB'01 Convention, York, UK*, March 2001.
15. T. Sandholm. *Negotiation among Self-Interested Computationally Limited Agents*. Computer science, University of Massachusetts at Amherst, September 1996.
16. A. Sathi and M. Fox. Constraint-directed negotiation of resource reallocation. In L. Gasser and M. Huhns, editors, *Distributed Artificial Intelligence*, volume II, pages 163–195. Morgan Kaufmann, San Mateo, CA, 1989.
17. K. P. Sycara. Argumentation: Planning other agents' plans. In *Proceedings 11th International Joint-Conference on Artificial Intelligence*, pages 517–523. Morgan Kaufman, 1989.
18. P. Torroni. A study on the termination of negotiation dialogues. Technical Report DEIS-LIA-01-008, University of Bologna (Italy), 2001. LIA Series no. 52.
19. G. Zlotkin and J. S. Rosenschein. Mechanisms for automated negotiation in state oriented domains. *Journal of Artificial Intelligence Research*, 5:163–238, 1996.