Fundamenta Informaticae XX (2016) 1–43
DOI 10.3233/FI-2015-0000
IOS Press

Now or Never: Negotiating Efficiently with Unknown or Untrusted Counterparts*

Toni Mancini C

Computer Science Department, Sapienza University of Rome, Italy

http://tmancini.di.uniroma1.it

Abstract. We define a new protocol rule, Now or Never (NoN), for bilateral negotiation processes which allows self-motivated competitive agents to *efficiently* carry out multi-variable negotiations with remote untrusted parties, where privacy is a major concern and agents know *nothing* about their opponent. By building on the geometric concepts of convexity and convex hull, NoN ensures a continuous progress of the negotiation, thus neutralising malicious or inefficient opponents. In particular, NoN allows an agent to derive in a finite number of steps, and *independently* of the behaviour of the opponent, that there is *no hope* to find an agreement. To be able to make such an inference, the interested agent may *rely on herself only*, still keeping the *highest freedom* in the choice of her strategy.

We also propose an actual NoN-compliant strategy for an automated agent and evaluate the computational feasibility of the overall approach on both random negotiation scenarios and case studies of practical size.

Keywords: Automated Negotiation, Rational Selfish Agents, Unmediated Agent Negotiation

*This is an extended and revised version of [1].
C Corresponding author

Contents

1. Introduction

Automated negotiation among rational agents is an important topic in Distributed Artificial Intelligence, being necessary in any domain where: (i) no single agent can achieve her own goals without interaction with the others (or she is expected to achieve higher utility with interaction), and (ii) constraints of various kinds (e.g., security or privacy) forbid the parties to communicate their desiderata to others (the opponent or a trusted authority), hence centralised approaches cannot be used.

In this paper we present a framework which allows two *self-motivated*, *competitive* agents to negotiate *efficiently* and find a mutually satisfactory agreement in a particularly *hostile* environment, where each party has *no information* on constraints, preferences, and *willingness to collaborate* of the opponent. This means that also the *bounds* of the domains of the negotiation variables are *not* common knowledge. Our framework deals with negotiations over multiple *constrained* variables over the type of *real numbers*, regarding integer or categorical variables as special cases.

The present setting is very different from what is often assumed in the literature (see also Section 8): the set of possible agreements is *infinite* and agents do not even know (or probabilistically estimate) possible opponent's types, variable domain bounds or most preferred values. It is not a split-the-pie game as, e.g., in [2] although with incomplete information, as in [3], and computing *equilibrium* or evaluating *Pareto-optimality* is not possible.

A major problem in our setting is that even *termination* of the negotiation process is not granted: it is in general *impossible* for the single agent to recognise whether the negotiation is making some progress, or if the opponent is just wasting time or arbitrarily delaying the negotiation outcome.

We solve this problem by proposing a new protocol rule, Now or Never (NoN) (Section 3), explicitly designed as to ensure a continuous progress of the negotiation. The rule (whose fulfilment can be assessed *independently* by each party using only the exchanged information) forbids the agents to reconsider already taken decisions, thus injecting a minimum, but sufficient amount of *efficiency* in the process. This leads to the *monotonic shrinking* of the set of possible agreements, which in turn allows each agent to derive in a finite number of steps, *independently* of the behaviour of the opponent, that there is *no hope* to find an agreement.

Furthermore, we discuss the notion of *non-obstructionist* agents, i.e., agents who genuinely aim at efficiently finding an agreement, even sacrificing their preferences (among the agreements they would accept). If both agents are non-obstructionist, the NoN rule guarantees that, whenever the termination condition arises, then *no agreement actually exists*. Hence, in presence of non-obstructionist agents, our approach is both *complete* and *terminates*.

We also propose (Section 4) a full NoN-compliant strategy for an agent which ensures termination independently of the behaviour of the opponent. The strategy, which takes into full account the presence of a utility function on the set of acceptable deals, is inspired to the well-known mechanism of Monotonic Concessions (MC) [4] and allows the agent to perform a sophisticated reasoning, based on the evidence collected so far on the behaviour of the opponent, to select the best deals to offer at each step and keep the process efficient.

In Section 5 we show how NoN and all the involved reasoning can be specialised to negotiations involving discrete and categorical variables.

Section 6 outlines our implementation of an automated negotiator employing the strategy of Section 4. Our system exploits a computational geometry library together with an all-SAT solver to perform the required reasoning.

In Section 7 we present experimental results on both random negotiation scenarios and real-world case studies, showing that enforcing the NoN rule in practical negotiation instances is computationally feasible. Section 8 discusses related work and Section 9 draws conclusions. Finally, Appendix A defines all the acronyms used in the paper and Appendix B gives proofs of all results.

2. Preliminaries and Negotiation Framework

In the following, we denote with $\mathbb{R}$, $\mathbb{R}^{\geq 0}$ and $\mathbb{N}^+$ the sets of all real numbers, non-negative real numbers and strictly positive integers, respectively.

Our framework deals with (possibly multi-deal) negotiations between *two* agents (agent 0 and agent 1) over *multiple constrained* variables. Agents do *not* have any information about constraints, goals, preferences, reasoning capabilities, willingness to collaborate, and strategy of the opponent. *The only knowledge common to both agents is the set of negotiation variables, the protocol rules and, after the process has started, the exchanged information*.

Definition 2.1 introduces the main concepts of our framework. Some of them are standard in the literature and are adapted to our framework to ease presentation of the following definitions and results.

Definition 2.1. (Negotiation Process)
A *negotiation process* is a tuple $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ where $\mathcal{V}$ is a finite set of negotiation *variables*, $s \in \{0, 1\}$ is the agent *starting* the negotiation, and $\mathcal{R}$ is the set of *protocol rules*.

The *negotiation space* is the set of assignments of real numbers to all variables in $\mathcal{V}$. To ease the presentation, without loss of generality we assume that set $\mathcal{V}$ is ordered. In this case, the negotiation space is the multi-dimensional real vector space $\mathbb{R}^{|\mathcal{V}|}$. Each point $D \in \mathbb{R}^{|\mathcal{V}|}$ is a *deal*. A *proposal* for π is a set of at most k deals or the distinguished element $\perp$. Value $k \in \mathbb{N}^+$ is the *maximum number of deals* that can be included in a single proposal.

Negotiation proceeds in *steps* (starting from step 1) with agents (starting from agent s) alternately exchanging proposals. The proposal exchanged at any step $t \geq 1$ is sent by agent $\mathrm{ag}(t)$, defined as s if t is odd and $1 - s$ if t is even.

The *status* of negotiation process π at step $t \geq 1$ is the sequence $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_t$ of proposals exchanged up to step t.

At each step, the status of π must satisfy the set $\mathcal{R}$ of *procotol rules*, a set of boolean conditions on sequences of proposals.

A *strategy* for agent $A \in \{0, 1\}$ for π is a function σ_A that, for each step t such that $\mathrm{ag}(t) = A$ and each status $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_{t-1}$ of π at step $t - 1$, returns the proposal $\mathcal{P}_t$ to be sent by agent A at step t, given the sequence of proposals already exchanged (σ_A is constant for $t = 1$ and $A = s$).

Our *alternating offers* [5] based framework primarily focuses on *real variables*. In Section 5 we discuss how more specialised domains (e.g., integers, categories) can be handled as special cases, and which is the added value of primarily dealing with real variables. Also, as each proposal can contain up to k deals, our framework supports *multi-deal* negotiations when $k > 1$. Section 4 discusses the added value given by the possibility of exchanging multi-deal proposals.

We will use Example 2.2 as a running example throughout the paper.

Example 2.2. (Alice vs. Bob)
Alice wants to negotiate with her supervisor *Bob* to schedule a meeting. At the beginning, agents agree on the relevant variables $\mathcal{V}$: (i) the start day/time t; (ii) the meeting duration d.

Deals are assignments of values to variables, as, e.g., $D = \langle t = \text{“Mon 11 am”}, d = \text{“30 min”} \rangle$. Deals can be easily encoded as points in $\mathbb{R}^2$.

Protocol rules are important to prevent malicious or inefficient behaviour. Well-designed rules are crucial when the process involves self-motivated and/or unknown/untrusted opponents. For protocol rules to be effective, agents must be able at any time to verify them using the current negotiation status only.

Given that our framework may allow multi-deal proposals (when $k > 1$), we assume that the following protocol rule is always enforced (hence belongs to set $\mathcal{R}$): for all $t > 1$, $|\mathcal{P}_t \cap \mathcal{P}_{t-1}| \leq 1$. This rule avoids ambiguity in case of acceptance (see Definition 2.3).

Definition 2.3. (Negotiation Outcomes)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process whose status at step $T > 1$ is $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_T$. We say that π *terminates* at step T if and only if T is the smallest value such that one of the following two cases holds:

- *success*: $\mathcal{P}_T \cap \mathcal{P}_{T-1} = \{D\}$ ($\mathrm{ag}(T)$ accepts deal D proposed by $\mathrm{ag}(T-1)$ at step $T-1$)
- *opt-out*: $\mathcal{P}_T = \perp$ ($\mathrm{ag}(T)$ opts-out).

If no such a T exists, then π is *non-terminating* (*non-term*).

Success, *opt-out* and *non-term* are the possible negotiation *outcomes*.

A negotiation process can be infinite (case *non-term*) or terminate in a finite number of steps, either with an *agreement* found (case *success*, where point D is the *agreement*) or with a failure (case *opt-out*, where one of the agents proposes $\perp$, which aborts the process).

For a deal to be *acceptable* to an agent, some *constraints* must be satisfied. Such constraints, which are *private information* of the single agent, are formalised by Definition 2.4.

Definition 2.4. (Feasibility region)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process. The *feasibility region* of agent $A \in \{0, 1\}$, denoted by R_A, is the subset of the negotiation space $\mathbb{R}^{|\mathcal{V}|}$ of deals *acceptable* to A.

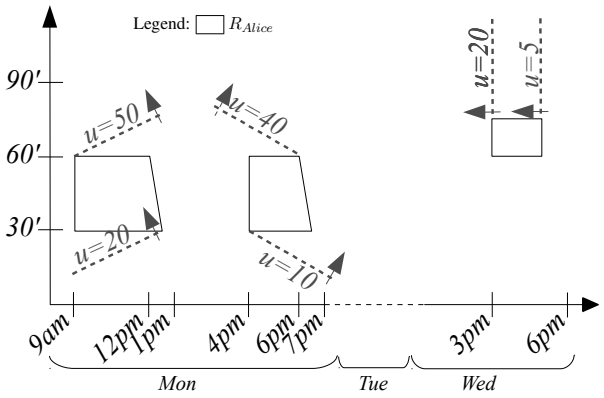
For agent A, any deal in R_A is better than failure. Conversely, any deal outside R_A would not be accepted by agent A. Thus, an *agreement* is any deal $D \in R_0 \cap R_1$.

Example 2.5. (Alice vs. Bob, continued)
Alice wants the meeting no later than Wednesday. Her (private) agenda up to Wednesday is shown in Figure 1a. Normally, Alice needs at least 30 minutes and does not want the meeting to last more than one hour; however, if she has to wait until Wednesday, she would have time during her Tuesday's trip to prepare new material to show; in this case she wants the meeting to last at least one hour, but no more than 75 minutes. Again, these constraints are private information. Conversely, Bob has his own, private, constraints.

Figure 1b shows Alice's feasibility region in the 2-dimensional real vector space $\mathbb{R}^2$, as the areas delimited by the three polygons. The region takes into account duties in her agenda (e.g., Alice is busy on Monday from 1pm to 4pm, see Figure 1a).

	Mon	Tue	Wed
Early morn		Trip to Paris	
09:00 AM			Project meeting
10:00 AM			
11:00 AM			
12:00 PM			
01:00 PM	Carol		
02:00 PM	Meeting w John		Juliet
03:00 PM			
04:00 PM			
05:00 PM			
06:00 PM			Laura's b'day party
07:00 PM	Gym		
08:00 PM			
Late even			

(a) Alice's agenda



(b) Alice's region and utility

Figure 1: Alice vs. Bob (Example 2.5)

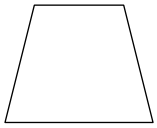
Agents may have *preferences* on the deals in their feasibility region. Such preferences are represented by a private *utility function*.

Definition 2.6. (Utility Function)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process. The *utility function* of agent $A \in \{0, 1\}$, denoted by u_A, is a function $R_A \to \mathbb{R}^{\geq 0}$ mapping each deal acceptable to A into a (without loss of generality non-negative) real number. For each pairs of deals $D_1, D_2 \in R_A$:

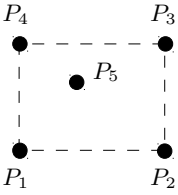
- if $u_A(D_1) > u_A(D_2)$ we say that agent A *prefers* an agreement on D_1 to an agreement on D_2;
- if $u_A(D_1) = u_A(D_2)$ we say that agent A considers D_1 and D_2 *equally* good.

Figure 1b shows that, e.g., the best agreement for Alice would be a 60-minute long meeting on Monday at 9am (having utility 50). Also, Alice considers, e.g., a 30-minute long meeting on Monday at 12.30pm and a meeting on Wednesday at 3pm (having whichever duration from 30 to 60 minutes) equally good (utility 20). The worst (but still acceptable) deal for Alice is a 60-minute long meeting on Wednesday at 5pm (ending up at 6pm just before Laura's birthday party). Such a deal has utility value 5. Again, utility information is kept private. We will handle the agent utility function in Section 4 when we present a full strategy for an agent.

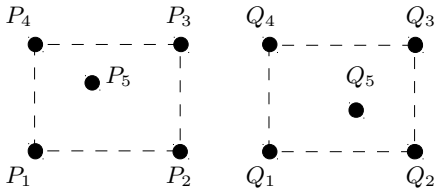
We assume (as typically done, see, e.g., [6, 7]) that agents offer only deals in their feasibility region (i.e., agents do not offer deals they are not willing to accept). This does not limit our approach, as suitable ex-post measures (e.g., penalties) can be set up to cope with the case where a deal offered by an agent (but *not* acceptable to her) is accepted by the other.



(a) A convex region



(b) $\mathrm{conv}(\{P_1, \ldots, P_5\})$ (region delimited by the dashed line)



(c) $\bigcirc\!\!\!\!\bigcup \boldsymbol{\mathcal{D}}$, with $\boldsymbol{\mathcal{D}} = \{\{P_1, P_2, P_3, P_4, P_5\}, \{Q_1, Q_2, Q_3, Q_4, Q_5\}\}$
(union of the two regions delimited by the dashed lines)

Figure 2: Examples of concepts of Definition 3.1 in $\mathbb{R}^2$.

3. Now or Never

We are interested in negotiations which are *guaranteed* to terminate in a finite number of steps (note that, being negotiation variables real-valued, the set of potential agreements is infinite), so we want to avoid case *non-term* of Definition 2.3. In this section we define a protocol rule, the Now or Never (NoN) rule, which is our key to drive a negotiation process towards termination, avoiding malicious or inefficient agents behaviour. The rule relies on the notions of Definition 3.1.

Definition 3.1. (Convex Region, Convex Hull, Operator $\bigcirc\!\!\!\!\bigcup$)
Let $\mathbb{R}^n$ be the n-dimensional real vector space (for any $n > 0$). Region $R \subseteq \mathbb{R}^n$ is *convex* if, for any two points D_1 and D_2 in R, the straight segment $\overline{D_1 D_2}$ is entirely in R.

Given a finite set of points $\mathcal{D} \subset \mathbb{R}^n$, the *convex hull* of $\mathcal{D}$, $\mathrm{conv}(\mathcal{D})$, is the smallest convex region of $\mathbb{R}^n$ containing $\mathcal{D}$.

Given a collection of finite sets of points $\boldsymbol{\mathcal{D}}$, $\bigcirc\!\!\!\!\bigcup \boldsymbol{\mathcal{D}}$ is the *union of the convex hulls* of all sets in $\boldsymbol{\mathcal{D}}$: $\bigcirc\!\!\!\!\bigcup \boldsymbol{\mathcal{D}} = \bigcup_{\mathcal{D} \in \boldsymbol{\mathcal{D}}} \{\mathrm{conv}(\mathcal{D})\}$.

Examples of the concepts above are shown in Figure 2 (in the 2-dimensional real vector space $\mathbb{R}^2$).

Convexity arises often in feasibility regions of agents involved in negotiations. An agent feasibility region is convex if, for any two acceptable deals D_1 and D_2, all *intermediate* deals (i.e., those lying on $\overline{D_1 D_2}$) are acceptable as well. In some cases [8, 7, 9] the feasibility region of an agent is *entirely* convex (consider, e.g., a negotiation instance over a single variable, the price of a good). In other cases this does not hold. However, a feasibility region may always be considered as the *union* of a number of convex sub-regions. Furthermore, in most real cases, this number is *finite* and *small*. Also, in most practical

situations, the closer two acceptable deals D_1 and D_2, the higher the likelihood that intermediate deals are acceptable as well.

Example 3.2. (Alice vs. Bob, continued)
Knowing that deals $\langle t = \textit{“Mon at 11am”}, d = \textit{“30 min”}\rangle$ and $\langle t = \textit{“Wed at 3pm”}, d = \textit{“1 hour”}\rangle$ are both acceptable to Bob would not be a strong support for Alice to assume that also $\langle t = \textit{“Tue at 1pm”}, d =$ “45 min”$\rangle$ would be acceptable to him. On the other hand, if $\langle t = \textit{“Mon at 9am”}, d = \textit{“40 min”}\rangle$ and $\langle t = \textit{“Mon at 9.30am”}, d = \textit{“20 min”}\rangle$ are both acceptable to Bob, it would not be surprising if also $\langle t = \textit{“Mon at 9.15am”}, d = \textit{“30 min”}\rangle$ is acceptable.

3.1. An Introductory Example

Before formalising the NoN rule (Definition 3.5), we introduce it using our example.

Example 3.3. (Alice vs. Bob, continued)
Steps below are shown in Figure 3.

Steps 1 and 2

Alice starts the negotiation by sending proposal $\mathcal{P}_1 = \{A_1^a, A_1^b\}$, thus hoping that Bob will accept one of them. Unfortunately, none of these deals are accepted by Bob, who sends back a proposal consisting of two counteroffers $\mathcal{P}_2 = \{B_2^a, B_2^b\}$ (see Figure 3a). As none of Bob's counteroffers, B_2^a and B_2^b, belong to $\mathrm{conv}(\{A_1^a, A_1^b\}) = \overline{A_1^a A_1^b}$, all deals in such a region are *removed* from further consideration (by exploiting the NoN rule). The rationale is as follows:

(a) Bob had *no evidence* that $\mathrm{conv}(\{A_1^a, A_1^b\})$ includes deals outside R_{Alice} (i.e., at the end of step 1 Bob had no evidence that this portion of R_{Alice} is not convex).

(b) Given that Bob has not proposed any such deal therein, then either $R_{Bob} \cap \mathrm{conv}(\{A_1^a, A_1^b\}) = \emptyset$ (in which case, Bob has no interest at all in proposing there), or Bob has chosen not to go for any such a deal *now* (as, e.g., he currently aims at higher utility).

(c) In the latter case, the NoN rule forbids Bob to reconsider that decision anymore (*never*): no deal in $\mathrm{conv}(\{A_1^a, A_1^b\})$ can be proposed or accepted by Bob in the sequel.

Step 3

Alice, having *no evidence* that $\mathrm{conv}(\{B_2^a, B_2^b\})$ includes deals outside R_{Bob}, proposes $\mathcal{P}_3$ containing deal $A_3^a \in \mathrm{conv}(\{B_2^a, B_2^b\}) \cap R_{Alice}$ (see Figure 3b): by proposing A_3^a she aims at closing the negotiation successfully *now*, believing that such a deal (intermediate to B_2^a and B_2^b) is likely to be acceptable also to Bob. Alice also includes in $\mathcal{P}_3$ deal A_3^b (according to her strategy, which is not of interest in this example).

Step 4

The situation at this point is shown in Figure 3b and the sequence of proposals exchanged so far is summarised in Figure 3c.

At step 4 it's Bob's turn again. By receiving $\mathcal{P}_3 = \{A_3^a, A_3^b\}$, Bob knows that such deals belong to R_{Alice}. Assume that Bob rejects $\mathcal{P}_3$ by sending a counteroffer. As there is no evidence that $\text{conv}(\{A_1^a, A_3^a, A_3^b\})$, $\text{conv}(\{A_1^b, A_3^b\})$, or $\text{conv}(\{A_1^b, A_3^a\})$ (the 3 light-grey areas in Figure 3b) include deals outside R_{Alice}, the NoN rule forces him to take a decision: either Bob includes, in his counteroffer $\mathcal{P}_4$, at least one deal (among the k deals he can include in his next proposal) belonging to at least one of such regions (showing Alice that he is potentially interested in reaching an agreement there), or he must forget those regions forever.

Note that the NoN rule does not apply to, e.g., $\text{conv}(\{A_1^b, A_3^a, A_3^b\})$, as this region contains B_2^b, which was part of a Bob's proposal already rejected by Alice. Hence, Alice already has evidence that some of the deals in $\text{conv}(\{A_1^b, A_3^a, A_3^b\})$ are acceptable to Bob and the NoN rule does not forbid agents to further explore that region.

3.2. Rationale and Impact of the NoN Protocol Rule

In this section we informally outline the rationale of the NoN rule and its impact on negotiation processes, before giving a full formalisation of the rule (Section 3.3) and of our main results (Section 3.4).

From Example 3.3 it can be seen that, at each step of the negotiation process, *if* the set of deals received so far *reveals* portions of the counterpart's region for which there is *no* evidence that they are *not* convex, then the NoN rule enforces the proposing agent to make a choice:

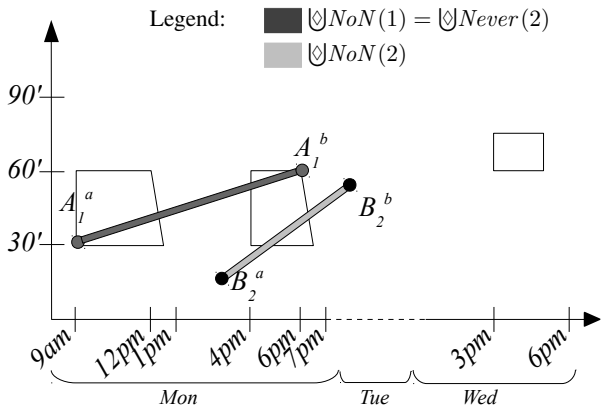
1. Either she includes, in her next proposal, *at least one* deal in *at least one* such portions (*now* decision); or
2. *No* deal in *any* such portions can be offered in the future (*never* decision).

In case 1 (*now decision*), the proposing agent shows the counterpart that she is *potentially interested* in reaching an agreement which is *intermediate* with respect to a subset of the deals received so far (one of the subsets of the deals received so far whose convex hull has not been already proven to contain points unacceptable to any party).

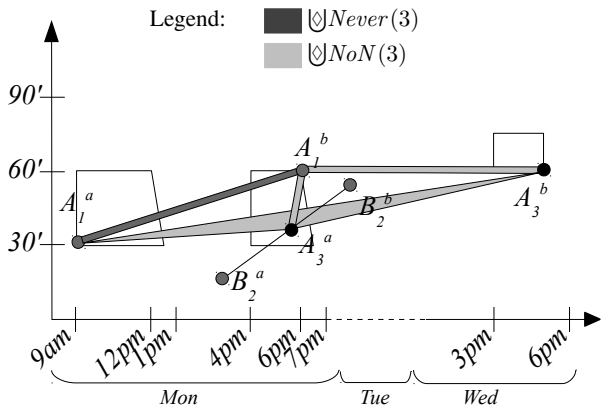
Note that, when taking a *now* decision, the proposing agent is free to choose *any* eligible subset of received deals where to propose next (and *none* of such subsets will be forbidden in the future). Furthermore, the agent is free to place her proposed deal *everywhere* within the convex hull of the chosen set of received deals. This makes the NoN rule *non-invasive*, as agents can exploit their (private) strategy to choose the deals to offer within the various regions.

Also, if *multi-deal* proposals are allowed, i.e., k (the maximum number of deals in a proposal) is greater than 1, then the proposing agent has *full freedom* in choosing the remaining $k - 1$ deals to be included in her next proposal. This makes it possible to adapt and exploit any agent strategy in a negotiation process which enforces the NoN rule.

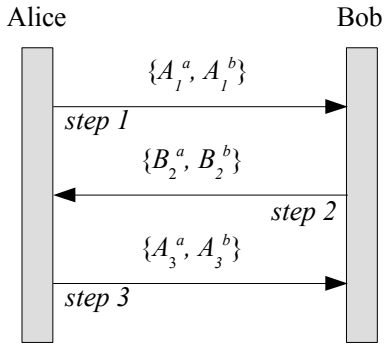
Case 2 (*never decision*) is used as a mechanism to inject a minimum amount of efficiency into the process. Although the proposing agent is obviously not forced to take a *now* decision whenever possible, the NoN rule forbids that this decision is reconsidered in the future, as this would lead to possible infinite negotiations.



(a) End of step 2



(b) End of step 3



(c) Summary of the exchanged proposals up to step 3

Figure 3: Alice vs. Bob (Example 3.3).

Proposition 3.6 shows that when a certain condition (which can be assessed independently by each agent using only information on the exchanged deals) is met, then there is no hope to find an agreement in the future, and the process can be safely terminated. Also, Section 4 shows that the NoN rule can be exploited by any agent to ensure that the termination condition arises in a *finite number of steps*. Importantly, termination is guaranteed *independently of the opponent behaviour*.

Summing up, the NoN rule can be thought as a (minimally invasive) *deterrent*, for each agent, to delay the negotiation by not exploiting promising portions of the space of possible agreements which, although containing deals acceptable to her, do not grant herself the utility she currently aims at. As the NoN rule forbids the agents to propose such deals in the future, any such "obstructionist" behaviour has a price in terms of opportunities that must be sacrificed forever.

Of course in our setting (where full privacy is guaranteed), agents have no means to derive that their counterpart is acting in an obstructionist way (i.e., that she does not take *now* decisions whenever possible). The NoN protocol rule does not forbid obstructionism (which might be crucial for successful negotiators), it does only enforce agents to keep a *coherent* behaviour during the course of the negotiation.

Non-obstructionist agents are formally defined in Definition 3.7. Non-obstructionist agents genuinely aim at finding an agreement *efficiently*, even sacrificing their preferences among deals they would accept. However, they are *not* necessarily collaborative, as they do not disclose to the opponent their constraints and preferences. Proposition 3.8 shows that, in case both agents are non-obstructionist, then the NoN rule guarantees *completeness*: when the termination condition arises, not only agents know that no mutually acceptable agreement can still be found; agents have also a *proof* that no mutually acceptable agreement actually *exists*.

3.3. Formalisation of the NoN Protocol Rule

Compliancy of an agent to the NoN rule can be assessed independently by the other agent, by computing sets $Never$ and NoN (Definition 3.4), using only public knowledge, i.e., the set of already exchanged deals.

Definition 3.4. (Sets $Never$ and NoN)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process and $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_T$ its status at step $T \geq 1$. For each agent $A \in \{0, 1\}$ and each step $1 \leq t \leq T$, let $\mathrm{deals}_A(t)$ be the set of all the deals in $\vec{\mathcal{P}}$ proposed by A up to step t (included). Sets $Never(t)$ and $NoN(t)$ are defined inductively for each $t \geq 1$ as follows:

$t = 1$**:** $Never(1) = \emptyset$, $NoN(1) = \{\mathcal{P}_1\}$

$t > 1$**:**

$$Never(t) = \begin{cases} Never(t-2) \cup NoN(t-1) & \text{if } \mathcal{P}_t \cap \Diamond\!\!\!\!\!\!\downarrow NoN(t-1) = \emptyset \\ Never(t-2) \cup \{\{D\} \mid D \in NoN(t-1)\} & \text{otherwise} \end{cases}$$

$$NoN(t) = \left\{\mathcal{D} \subseteq \mathrm{deals}_{\mathrm{ag}(t)}(t) \mid \mathrm{conv}(\mathcal{D}) \cap \Diamond\!\!\!\!\!\!\downarrow Never(t) = \emptyset\right\}$$

where $\mathcal{P}_t$ is the proposal sent by $\mathrm{ag}(t)$ at step t and $Never(0) = \emptyset$.

At each step t, $\Diamond\!\!\!\!\!\!\downarrow NoN(t)$ represents the region, defined by $\mathrm{ag}(t)$'s deals, for which the other agent $1 - \mathrm{ag}(t)$ needs, in the next step $(t + 1)$ to take a NoN decision: to include in her proposal at least one

deal therein (showing to $\mathrm{ag}(t)$ that she is potentially interested to that region) or to neglect that region forever. In case multi-deal proposals are allowed, i.e., $k > 1$, $\mathrm{ag}(t)$ can choose in full freedom up to $k-1$ other deals to be included in her next proposal.

Similarly, $\Diamond\!\!\!\!\!\!\bigcirc\, Never(t)$ represents the region, defined by $(1-\mathrm{ag}(t))$'s deals, for which $\mathrm{ag}(t)$ has taken a *never* decision. Deals therein cannot be offered any more.

We have that $\Diamond\!\!\!\!\!\!\bigcirc\, Never(t) \supseteq \Diamond\!\!\!\!\!\!\bigcirc\, Never(t-2)$ for all $t \geq 2$, i.e., sequences $Never(t)$ for odd and even values of t are monotonically non-decreasing. Figure 3 shows NoN and $Never$ regions at all steps of Example 3.3.

Definition 3.5 formalises our NoN protocol rule, which forbids agents to reconsider *never* decisions already taken.

Definition 3.5. (Now or Never Protocol Rule)
Status $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_T$ of negotiation process $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ satisfies the NoN protocol rule if, for all steps $2 \leq t \leq T$, $\mathcal{P}_t \cap \Diamond\!\!\!\!\!\!\bigcirc\, Never(t-2) = \emptyset$.

3.4. Main Results

Despite the simplicity of the NoN rule of Definition 3.5, Proposition 3.6 (proof delayed to Appendix B) shows that it is enough to allow agents to infer whether no further agreements are possible.

Proposition 3.6. (Termination Condition)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process where the NoN rule is enforced and let $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_T$ be the status of π at step $T \geq 2$.

If $R_{\mathrm{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\bigcirc\, Never(T-1) \cup \Diamond\!\!\!\!\!\!\bigcirc\, Never(T-2)$ and $\mathcal{P}_T$ is not a singleton $\{D\} \subseteq \mathcal{P}_{T-1}$, then:

(a) There exists no extension $\vec{\mathcal{P}}' = \mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{T-1}, \mathcal{P}_T, \ldots, \mathcal{P}_{T'}$ of $\vec{\mathcal{P}}$ to step $T' > T$ such that $\mathcal{P}_{T'} = \{D\} \subseteq \mathcal{P}_{T'-1}$

(b) For all $D \in R_0 \cap R_1$, there exists $1 < t_D < T$ such that $D \in \Diamond\!\!\!\!\!\!\bigcirc\, NoN(t_D - 1) \cap \Diamond\!\!\!\!\!\!\bigcirc\, Never(t_D)$.

A consequence of (a) is that, if at step $T \geq 2$, $R_{\mathrm{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\bigcirc\, Never(T-1) \cup \Diamond\!\!\!\!\!\!\bigcirc\, Never(T-2)$ and *agent* $\mathrm{ag}(T)$ *cannot or does not want to accept a deal offered in the last incoming proposal* $\mathcal{P}_{T-1}$*, she can safely opt-out* by proposing $\mathcal{P}_T = \perp$, as she has *no hope* to reach an agreement in the future. Also, from (b), for every mutually acceptable agreement D, there was a step $t_D < T$ in which agent $\mathrm{ag}(t_D)$ took a *never* decision on a NoN region containing D. This means that $\mathrm{ag}(t_D)$, although knowing that $D \in R_{\mathrm{ag}(t_D)}$ was likely to be acceptable also to the opponent (because she had no evidence, at that time, that the portions of the opponent region defined by deals in $NoN(t_D - 1)$ were not convex), *explicitly decided* not to take that chance and proposed elsewhere.

As anticipated in Section 3.2, NoN can be thought as a (minimally invasive) *deterrent*, for each agent, to delay the negotiation by ignoring plausible agreements which, although acceptable to her, do not grant herself the utility she currently aims at. As NoN forbids the agents to propose such deals in the future, any such "obstructionist" behaviour has a price in terms of opportunities that must be sacrificed forever.

Definition 3.7 defines *non-obstructionist agents*. As outlined in Section 3.2, non-obstructionist agents (although *not* being necessarily collaborative, as they do not disclose to the opponent their constraints and preferences) genuinely aim at finding an agreement *efficiently*, even sacrificing their preferences among deals they would accept.

Definition 3.7. (Non-obstructionist Agent)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process where the NoN rule is enforced. Agent $A \in \{0, 1\}$ is *non-obstructionist* if her strategy satisfies the following conditions for all $t \geq 2$ such that $\mathrm{ag}(t) = A$:

1. if $\mathcal{P}_{t-1} \cap R_A \neq \emptyset$, then $\mathcal{P}_t = \{D\} \subseteq \mathcal{P}_{t-1} \cap R_A$
2. else if $\Diamond\!\!\!\!\!\!\bigcirc NoN(t-1) \cap R_A \neq \emptyset$, then $\mathcal{P}_t \cap \Diamond\!\!\!\!\!\!\bigcirc NoN(t-1) \neq \emptyset$.

A non-obstructionist agent A accepts *any* acceptable deal $D \in R_A$ and takes a *now* decision at all steps t when $\Diamond\!\!\!\!\!\!\bigcirc NoN(t-1)$ intersects R_A.

Proposition 3.8 (proof delayed to Appendix B) shows that, in a negotiation process between two non-obstructionist agents, if one of the parties reaches the *termination condition* of Proposition 3.6, then no agreement actually exists (i.e., $R_0 \cap R_1 = \emptyset$).

Proposition 3.8. (Completeness)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process between two non-obstructionist agents where the NoN rule is enforced.

If π reaches, at step $T - 1 \geq 2$, status $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_{T-1}$ s.t. $R_{\mathrm{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\bigcirc Never(T-1) \cup \Diamond\!\!\!\!\!\!\bigcirc Never(T-2)$, then $R_0 \cap R_1 = \emptyset$.

4. A Terminating Strategy Based on Monotonic Concessions

Propositions 3.6 and 3.8 show that the Now or Never (NoN) rule allows each agent to detect when the negotiation process can be safely terminated, as no agreement can be found in the sequel. However, still the termination condition may not arise in a finite number of steps.

In this section we show that, with NoN, *termination can be enforced by any agent alone, without relying on the willingness to terminate of the counterpart*. To this end, from now on we focus on one agent only, which we call agent A (A can be either 0 or 1). To ease presentation, the other agent, agent $1 - A$, will be called agent B.

We make some assumptions on the feasibility region of agent A:

(a) R_A is *bounded* and defined as the union $P_1 \cup \cdots \cup P_q$ of a *finite* number q of convex sub-regions;

(b) Each convex sub-region P_i ($1 \leq i \leq q$) of R_A is defined by *linear constraints*, hence is a (bounded) *polyhedron* in $\mathbb{R}^{|\mathcal{V}|}$.

Any bounded feasibility region can be approximated arbitrarily well with a (sufficiently large) union of bounded polyhedra. However, in many practical cases, a small number of polyhedra suffices.

Deals in R_A may not be equally worth for agent A, who may have a (again, *private*) utility function u_A to *maximize*. We assume that u_A is *piecewise-linear* and defined (without loss of generality) by a linear function

$$u_A^i = c^i + \sum_{v \in \mathcal{V}} a_v^i \cdot v$$

(with a_v^i's and c^i being real coefficients) for each polyhedron P_i of R_A ($1 \leq i \leq q$). For this definition to be well founded, if a deal D belongs to two different polyhedra P_i and P_j of R_A, it must be $u_A^i(D) =$

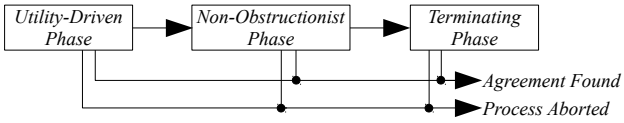


Figure 4: The three phases of our NoN-compliant agent strategy.

$u_A^j(D)$. Note that, again, any differentiable utility function can be approximated arbitrarily well with a piecewise-linear utility, provided R_A is decomposed in an enough number of polyhedra.

In this setting, we define a full strategy for agent A for negotiation processes $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ for which $k \geq 2$, i.e., in which exchanged proposals can contain multiple deals. Although our strategy is correct independently of the opponent region shape, it is designed for the common cases where agent A believes that the opponent feasibility region is the union of a small number of convex sub-regions (not necessarily polyhedra). Hence, a task of agent A while following the strategy is to *discover* non-convexities of the opponent region during negotiation and take them into account.

Our strategy is inspired by (but different from) the well-known mechanism of Monotonic Concessions (MC) [4]. It has three phases, *utility-driven*, *non-obstructionist*, and *terminating* phases, which are executed in the given order (see Figure 4).

Algorithm 1 shows high-level pseudo-code for the knowledge base and the main loop of our agent (agent A). As all procedures called by the main loop need to have access to most of the agent knowledge base and do modify components of the agent knowledge base, for simplicity of presentation we defined the agent knowledge base as a set of global variables. Pseudo-code of the procedures called by the main loop are reported in Algorithms 2, 3, 4 and 5.

4.1. Utility-Driven Phase

Agent A keeps and dynamically revises two utility thresholds, α and u, which are, respectively, the *responding* and the *proposing threshold*. At each step t such that $\mathrm{ag}(t) = A$, agent A uses:

(a) Threshold α to decide whether to take a *now* decision (if $t > 1$), by including, in the proposal $\mathcal{P}_t$ she will propose next, a deal in $\lozenge\!\!\!\!\bigcirc NoN(t-1)$ (possibly accepting one deal in $\mathcal{P}_{t-1}$) having at least utility α, and

(b) Threshold u to select the other deals to include in $\mathcal{P}_t$ ($t \geq 1$).

By generalising [2, 7], α is a function of the agent A utility of the best deal D_{next} that would be chosen in step (b). In particular, α is $u_A(D_{\text{next}}) - \mathit{span} \cdot \xi$, where *span* is the absolute difference of the extreme values of u_A in R_A and $0 \leq \xi \leq 1$ is a parameter (possibly varying during the negotiation) called *respond policy*. Hence, if $\xi = 1$, agent A accepts all acceptable deals and takes a *now* decision whenever possible, behaving in a non-obstructionist way (Definition 3.7). On the other extreme, if $\xi = 0$ the agent accepts only incoming deals $D \in R_A$ that are not worse than the best proposal D_{next} that would be chosen next in step (b), upon rejection of D. Pseudo-code for computing α is outlined as function *alpha()* in Algorithm 2. Note that, ultimately, α is a function of u.

// agent knowledge base
global variables
// parameters of agent A
R_A: feasibility region;
u_A: utility function;
ξ: respond policy;
span: abs. difference of extreme values of u_A in R_A;
// negotiation process status
t: current time step;
$\mathcal{P}$: sequence $\mathcal{P}_0, \ldots, \mathcal{P}_t$ of proposals exchanged so far;
u: proposing threshold;
current_phase: current phase of the negotiation process (i.e., utility-driven, non-obstructionist, terminating, done);

// main agent loop
procedure *run()*
$\mathcal{P} \leftarrow$ empty sequence of exchanged proposals;
$u \leftarrow$ maximum value of utility function u_A within R_A;
current_phase $\leftarrow$ utility-driven;
$t \leftarrow 1$;
$\mathcal{P}_0 \leftarrow \emptyset$;
while *current_phase* $\neq$ *done* **do**
  if $\mathrm{ag}(t) = B$ **then**
    // counterpart's turn
    recv $\mathcal{P}_t$;
    if $\mathcal{P}_t = \perp$ **then**
      current_phase $\leftarrow$ done ; *// counterpart opted out*
    else
      if $\mathcal{P}_t \cap \mathcal{P}_{t-1} = \{D\}$ **then**
        current_phase $\leftarrow$ done ; *// c'part accepted deal D in my last proposal*
  else
    // my turn: build proposal $\mathcal{P}_t$ to be sent
    respond(); *// possibly accept an incoming deal or take now decision*
    if *current_phase* $\neq$ *done* **then**
      // not accepted any of the last incoming deals
      propose<current_phase>();
    send $\mathcal{P}_t$;
  t++;

Algorithm 1: Our NoN-compliant MC-based strategy for Agent A.

Our strategy for this phase is decomposed into *responding*, *proposing* and *conceding* sub-strategies as in [10], after an *initialisation* phase (see Algorithm 1, line 11) where agent A sets u to the highest utility of deals in R_A (as in the spirit of MC).

4.1.1. Responding

At step $t \geq 2$, after that agent A has received proposal $\mathcal{P}_{t-1} \neq \perp$, proposal $\mathcal{P}_t$ is chosen as follows (see procedure *respond()* in Algorithm 2). Let $R_A^\alpha = \{D \in R_A \mid u_A(D) \geq \alpha\}$.

1. If $\mathcal{P}_{t-1}$ contains deals in $R_A^\alpha - \bigcup\!\!\!\!\!\diamond\, Never(t-2)$, then $\mathcal{P}_t = \{D\}$, where D is one such a deal giving agent A the highest utility (i.e., agent A accepts the best deal D among those acceptable in $\mathcal{P}_{t-1}$ granting herself at least utility α, see Algorithm 2, line 6).
   Otherwise:

2. $\mathcal{P}_t$ contains a deal in $(\bigcup\!\!\!\!\!\diamond\, NoN(t-1) \cap R_A^\alpha) - \bigcup\!\!\!\!\!\diamond\, Never(t-2)$ if and only if this region is not empty (*now* decision taken, see Algorithm 2, line 11).

Given that the closer deals in a set $\mathcal{D}$ defining $NoN(t-1)$ (see Definition 3.4) the more likely they belong to a single convex sub-region of R_B, as for case 2, agent A selects a deal with the highest utility in a set $\mathcal{D}$ having the minimum *diameter*.

Unless agent A has accepted an incoming deal (case 1), proposal $\mathcal{P}_t$ (see Algorithm 1, line 26) may contain *additional* deals (as to make $|\mathcal{P}_t| = k \geq 2$ if possible), chosen according the *proposing* sub-strategy.

4.1.2. Proposing

The *proposing* sub-strategy of the utility-driven phase is outlined as procedure *propose<utility-driven>()* in Algorithm 3.

Let $R_A^u = \{D \in R_A \mid u_A(D) \geq u\}$. Region R_A^u is again a union of bounded polyhedra, as u_A is piecewise-linear. Deals to be proposed in $\mathcal{P}_t$ are selected among *vertices* of R_A^u (some of them can be vertices of the overall region R_A) which do not belong to $\bigcup\!\!\!\!\!\diamond\, Never(t-2)$, as agent A needs to comply with the NoN rule. If $t > 1$, vertices of R_A^u to be proposed will be carefully selected by reasoning on the *evidence* provided by the past opponent behaviour. The reasoning is as follows.

Let $\hat{n}(t)$ be the *minimum* number of convex sub-regions that *must* compose $R_B - \bigcup\!\!\!\!\!\diamond\, Never(t-1)$, i.e., the opponent region minus the regions for which the opponent has taken a *never* decision (and in which, by the NoN rule, no agreements can be found in the sequel): $\hat{n}(t)$ is the minimum value such that there exists a $\hat{n}(t)$-partition $\{\mathcal{D}_1, \ldots, \mathcal{D}_{\hat{n}(t)}\}$ of $\text{deals}_B(t-1)$ (i.e., a mapping of each opponent deal to one sub-region) such that for all $1 \leq j \leq \hat{n}(t)$, $\text{conv}(\mathcal{D}_j) \cap \bigcup\!\!\!\!\!\diamond\, Never(t-1) = \emptyset$.

Agent A temporarily focuses on $\hat{n}(t)$, assuming that $R_B - \bigcup\!\!\!\!\!\diamond\, Never(t-1)$ is the union of *exactly* $\hat{n}(t)$ convex sub-regions. We call this assumption *Non-obstructionist Opponent Assumption (NOA)*. Under NOA, agent A tries to regard the past opponent behaviour as non-obstructionist, hence interprets the already taken *never* decisions as an admission that $R_B \cap \bigcup\!\!\!\!\!\diamond\, Never(t-1) = \emptyset$ (Proposition 3.8). Value $\hat{n}(t)$ is the minimum number of convex sub-regions that must compose R_B which is consistent with this (optimistic) hypothesis.

procedure *respond()*
  $\mathcal{P}_t \leftarrow \emptyset$;
  if $t = 1$ **then return**;
  $\alpha \leftarrow$ *alpha*();
  $R_A^\alpha \leftarrow \{D \in R_A \mid u_A(D) \geq \alpha\}$;
  if $\mathcal{P}_{t-1} \cap R_A^\alpha - \Diamond\!\!\!\!\!\!\Diamond Never(t-2) \neq \emptyset$ **then**
    // accept best acceptable deal in incoming proposal
    $D \leftarrow$ deal in $\mathcal{P}_{t-1} \cap R_A^\alpha - \Diamond\!\!\!\!\!\!\Diamond Never(t-2)$ with highest utility;
    $\mathcal{P}_t \leftarrow \{D\}$;
    current_phase $\leftarrow$ done;
  else
    if *current_phase* $\neq$ *terminating and* $(\Diamond\!\!\!\!\!\!\Diamond NoN(t-1) \cap R_A^\alpha) - \Diamond\!\!\!\!\!\!\Diamond Never(t-2) \neq \emptyset$ **then**
      // take now *decision*
      $D \leftarrow$ deal in $NoN(t-1) \cap R_A^\alpha - \Diamond\!\!\!\!\!\!\Diamond Never(t-2)$ with highest utility, belonging to a set $\mathcal{D} \in NoN(t-1)$ having minimum diameter;
      $\mathcal{P}_t \leftarrow \{D\}$;
function *alpha()*
  if *current_phase* $=$ *utility-driven* **then**
    // invoke propose() in order to discover best deal D_{next} *that would be chosen*
    $(u^{bkp}, \mathcal{P}_t^{bkp}) \leftarrow (u, \mathcal{P}_t)$; *// back up data possibly changed by propose()*
    $\mathcal{P}_t \leftarrow \emptyset$;
    propose<current_phase>(); *// invoke propose(), which populates* $\mathcal{P}_t$
    $D_{next} \leftarrow$ deal in $\mathcal{P}_t$ with highest utility;
    $(u, \mathcal{P}_t) \leftarrow (u^{\text{bkp}}, \mathcal{P}_t^{\text{bkp}})$; *// undo changes*
    return $u_A(D_{next}) - span \cdot \xi$;
  else return *minimum utility within* R_A ;

Algorithm 2: Procedure *respond()* and function *alpha()* (used in all phases).

```
1  procedure propose<utility-driven>()
2    current_phase ← utility-driven phase;
3    at_least_one_vertex_chosen ← false;
4    while |P_t| < k do
5      R_A^u ← {D ∈ R_A | u_A(D) ≥ u};
6      vertices ← vertices of R_A^u not in (⟡Never(t − 2)) ∪ Π(t) ∪ P_t;
7      if vertices = ∅ then
8        if at_least_one_vertex_chosen then
           // no more vertices, but at least one vertex has been chosen: P_t ready to be sent
9          break;
10       else
           // P_t still with no vertices: try to concede utility
11         conceded ← concede();
12         if not conceded then
             // cannot concede: move to non-obstr. phase
13           propose<non-obstructionist>();
14     else
15       D ← best vertex under NOA within vertices;
16       P_t ← P_t ∪ {D};
17       at_least_one_vertex_chosen ← true;
18 function concede()
19   if u ≤ min_{D∈R_A}(u_A(D)) then
       // u is already at its minimum
20     return false;
21   else
22     u ← u − Δu;
23     return true;
```

Algorithm 3: Utility-driven phase: procedure *propose<utility-driven>()* and function *concede()*.

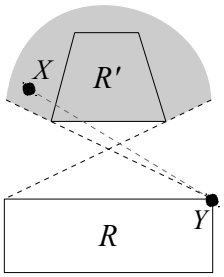


Figure 5: Projection of region R onto R' (unbounded grey area), notation $\mathrm{proj}(R, R')$.

Agent A computes the subsets $\mathcal{D}$ of the opponent deals that *might* belong to the same convex sub-region of $R_B - \Diamond\!\!\!\!-\, Never(t-1)$, provided that NOA is correct. We call these sets of deals *Possible Opponent Clusters (POCs)* (Definition 4.1).

Definition 4.1. (Possible Opponent Clusters)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process where the NoN rule is enforced. At any time step t such that $\mathrm{ag}(t) = A$, the set $\boldsymbol{\mathcal{K}}(t)$ of Possible Opponent Clusters (POCs) is defined as:

$$\boldsymbol{\mathcal{K}}(t) = \left\{ \mathcal{D} \subseteq \mathrm{deals}_B(t-1) \;\middle|\; \begin{array}{l} \exists\, \hat{n}(t)\text{-partition } \{\mathcal{D}_1, \ldots, \mathcal{D}_{\hat{n}(t)}\} \text{ of } \mathrm{deals}_B(t-1) \\ \text{s.t. } \forall j \in [1, \hat{n}(t)]\ \ \mathrm{conv}(\mathcal{D}_j) \cap \Diamond\!\!\!\!-\, Never(t-1) = \emptyset \end{array} \right\} \tag{1}$$

Definition 4.2 recalls the notion of *projection* from [8].

Definition 4.2. (Projection)
Let R and R' be two (bounded or unbounded) regions of $\mathbb{R}^n$ for some $n > 0$. The *projection of R onto R'*, notation $\mathrm{proj}(R, R')$, is the set of points X for which there exists $Y \in R$ such that $\overline{XY}$ intersects R'.

Region $\mathrm{proj}(R, R')$ is an unbounded polyhedron if both R and R' are polyhedra and $\mathrm{proj}(R, R' \cup R'') = \mathrm{proj}(R, R') \cup \mathrm{proj}(R, R'')$. Figure 5 shows an example of projection of R onto R' in $\mathbb{R}^2$, where $\mathrm{proj}(R, R')$ is the unbounded grey area.

Provided that NOA is correct, agent A can derive (Proposition 4.3, proof delayed to Appendix B) that region

$$\Pi(t) = \bigcap\nolimits_{\mathcal{D} \in \boldsymbol{\mathcal{K}}(t)} \mathrm{proj}(\mathrm{conv}(\mathcal{D}), \Diamond\!\!\!\!-\, Never(t-1)) \tag{2}$$

does not contain agreements that can be still reached.

Proposition 4.3. If, at step $t \geq 3$ such that $\mathrm{ag}(t) = A$, NOA is correct, then:

$$\Pi(t) \cap (R_B - \Diamond\!\!\!\!-\, Never(t-1)) = \emptyset.$$

Example 4.4. (Alice vs. Bob, continued)
Consider Figure 6. At step 4 Bob sent Alice proposal $\mathcal{P}_4 = \{B_4^a\}$. At step 5 (Alice's turn), $\hat{n}(5)$ is 3, as it is clear that B_2^a, B_2^b, and B_4^a belong to all-different convex sub-regions of $R_{Bob} - \Diamond\!\!\!\!-\, Never(4)$.

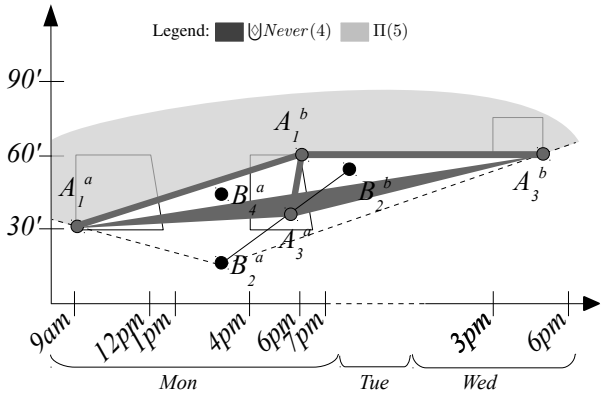


Figure 6: Alice vs. Bob (end of step 4).

POCs are $\mathcal{K}(5) = \{\{B_2^a\}, \{B_2^b\}, \{B_4^a\}\}$. Region $\Pi(5)$ is the area in light-grey: if NOA is correct ($R_{Bob} - \Diamond\!\!\!\!-Never(4)$ or, equivalently, R_{Bob} if Bob is non-obstructionist, consists of exactly 3 convex sub-regions), then no $X \in \Pi(5)$ can belong to $R_{Bob} - \Diamond\!\!\!\!-Never(4)$.

Besides always ignoring vertices in $\Diamond\!\!\!\!-Never(t-2)$ (as to comply with the NoN rule), as a result of Proposition 4.3 agent A (exploiting NOA) can *temporarily ignore* vertices of R_A^u in $\Pi(t)$ while choosing deals to propose at step t. By exploiting the fail-first principle, we define the following criterion (*best vertex under NOA*, used in Algorithm 3, line 15) to select the next vertices in $R_A^u - \Pi(t)$ (and not in $\Diamond\!\!\!\!-Never(t-2)$) to propose: those that, if rejected, would make the *highest* number of vertices be excluded in the next step, under NOA.

4.1.3. Conceding

When no more vertices in $R_A^u - \Pi(t)$ (and not in $\Diamond\!\!\!\!-Never(t-2)$) can be proposed, agent A reduces threshold u, if possible, by a given amount $\Delta u > 0$ (see function *concede()* in Algorithm 3, used in line 11 and defined from line 18). Value Δu is the agent *concession policy* and its value, possibly varying during time (see, e.g., [6]), depends on the application. Reducing u is in the spirit of MC (where the agent *increases* during time the *opponent* utility of the proposed deals). Differently from MC, here agent A *reduces own* utility of the deals she proposes (with the goal of approaching opponent's demand), as she has no information about opponent utility.

Given that k (the maximum number of deals in a proposal) is greater than 1, agent A proposes a new vertex in $R_A^u - \Pi(t)$ (and not in $\Diamond\!\!\!\!-Never(t-2)$) at each step t such that $\mathrm{ag}(t) = A$, independently of whether she has taken a *now* decision (point 2 of Section 4.1.1) or not. This ensures that agent A will *always* concede utility after a finite number of steps (where no more vertices in $R_A^u - \Pi(t)$ and not in $\Diamond\!\!\!\!-Never(t-2)$ are available), *independently of the behaviour of the counterpart* (on which *now* decisions taken by agent A ultimately depend). On the contrary, if $k = 1$, agent A could spend an arbitrarily high number of steps in taking *now* decisions (if the counterpart always proposes deals satisfying condition of point 2 of Section 4.1.1), and never concede utility.

Always conceding utility $\Delta u > 0$ (and greater than an arbitrarily small $\varepsilon > 0$) after a finite number

of steps guarantees that there will be a step $\hat{T}$ ($\mathrm{ag}(\hat{T}) = A$) in which agent A reduces u and R_A^u becomes equal to R_A (i.e., u cannot be further reduced). From step $\hat{T}$ onwards, the strategy of agent A moves to the *non-obstructionist* phase.

4.2. Non-Obstructionist Phase

Our strategy for this phase is decomposed into *responding* and *proposing* sub-strategies. As utility threshold u has already reached its minimum, in this phase there is no *conceding* sub-strategy.

4.2.1. Responding

The *responding* sub-strategy is identical to that of the *utility-driven* phase (see Section 4.1.1 and procedure *respond()* in Algorithm 2), but now α (as u) is set to the minimum utility within the agent feasibility region R_A. Hence, in the non-obstructionist phase agent A accepts any incoming acceptable deal and takes a *now* decision whenever possible. Thus, the agent is now certainly non-obstructionist, independently of the value of her respond policy ξ.

4.2.2. Proposing

As a result of acting in a non-obstructionist way, from step $\hat{T}$ onwards the following result (Proposition 4.5) holds. Proof is delayed to Appendix B.

Proposition 4.5. For each step $t \geq \hat{T}$ such that $\mathrm{ag}(t) = A$, $R_A \cap \lozenge Never(t-2) = R_A \cap \lozenge Never(\hat{T}-2)$.

Hence, for each step $t \geq \hat{T}$ such that $\mathrm{ag}(t) = A$, if agent A has not accepted an incoming deal by means of the *responding* sub-strategy, the region in which the additional deals to propose will be selected (as to make $|\mathcal{P}_t| = k \geq 2$ whenever possible), i.e., $R_A - \lozenge Never(t-2)$, is steadily equal to $R_A - \lozenge Never(\hat{T}-2)$.

In this phase, agent A aims at proposing *vertices* of $R_A - \lozenge Never(\hat{T}-2)$ with the goal of eventually covering this region with the *never* set of the opponent, as to reach the termination condition of Proposition 3.6. Pseudo-code for the *proposing* sub-strategy of the non-obstructionist phase is shown as procedure *propose<non-obstructionist>()* in Algorithm 4.

Unfortunately, as both R_A and $\lozenge Never(\hat{T}-2)$ are unions of polyhedra, their difference might *not* be represented as a union of polyhedra. Anyway, it can be always represented as a union of Not Necessarily Closed (NNC) polyhedra, i.e., polyhedra possibly defined by some *strict* inequalities, with some of their faces and vertices *not* belonging to them. In order to comply with the NoN rule, the agent must not propose vertices of $R_A - \lozenge Never(\hat{T}-2)$ not belonging to that region, as they would belong to $\lozenge Never(\hat{T}-2)$. The problem is solved (see Algorithm 4, line 5) by computing a suitable *under-approximation* $\lfloor R_A - \lozenge Never(\hat{T}-2) \rfloor \subseteq R_A - \lozenge Never(\hat{T}-2)$ which can be defined as a union of bounded (and closed) polyhedra. Note that such an under-approximation can be computed in order to make the error

$$R_A^{\mathrm{err}} = (R_A - \lozenge Never(\hat{T}-2)) - \lfloor R_A - \lozenge Never(\hat{T}-2) \rfloor$$

arbitrarily small. As a special case, if agent A was non-obstructionist from the beginning of the negotiation process, $R_A \cap \lozenge Never(\hat{T}-2) = \emptyset$ and $R_A^{\mathrm{err}} = \emptyset$.

```
1  procedure propose<not-obstructionist>()
2    current_phase ← non-obstructionist phase;
3    at_least_one_vertex_chosen ← false;
4    while |P_t| < k do
5      vertices ← vertices of ⌊R_A − ⦶Never(t − 2)⌋ not in Π(t) ∪ P_t;
6      if vertices = ∅ then
7        if at_least_one_vertex_chosen then
           // no more vertices, but at least one vertex has been chosen: P_t ready to be sent
8          break;
9        else
10         if Π(t) = ⦶Never(t − 1) then
             // NOA cannot be further relaxed: move to terminating phase
11           return propose<terminating>();
12         else relax NOA by artificially increasing n̂(t) by 1 (hence reducing Π(t)) ;
13     else
14       D ← best vertex under NOA within vertices;
15       P_t ← P_t ∪ {D};
16       at_least_one_vertex_chosen ← true;
```

Algorithm 4: Non-Obstructionist phase: procedure *propose<non-obstructionist>()*.

Agent A continues to use both NOA and $\Pi(t)$ as defined in the *utility-driven* phase. In particular, the agent proposes *vertices* of $\lfloor R_A - \text{⦶}Never(\hat{T}-2)\rfloor$ which are not in $\Pi(t)$. When no more such vertices can be proposed, NOA is gradually relaxed (i.e., $\hat{n}(t)$ is gradually increased) and the remaining vertices of $\lfloor R_A - \text{⦶}Never(\hat{T}-2)\rfloor$ are enabled (see Algorithm 4, line 12). By construction, $\hat{n}(t)$ cannot grow beyond the number of deals proposed by the opponent so far. If also in that case $\Pi(t)$ covers $\lfloor R_A - \text{⦶}Never(\hat{T}-2)\rfloor$, the agent sets $\Pi(t)$ to $\text{⦶}Never(t-1)$, hence assumes that R_B consists of at least one convex sub-region *not yet* disclosed by the opponent (i.e., not containing any of the past incoming deals).

As it happens in the *utility-driven* phase, given that *multi-deal* proposals are allowed ($k \geq 2$), all vertices will be proposed within a finite number of steps independently of the number of *now* decisions taken. When all vertices have been proposed and no agreement has been reached, agent A enters the *terminating phase* (see Algorithm 4, line 11).

4.3. Terminating Phase

In this phase, agent A continues by sending *empty proposals* until she receives and accepts an acceptable deal (see procedure *respond()* in Algorithm 2) or infers $R_A - R_A^{\text{err}} \subseteq \text{⦶}Never(t-1) \cup \text{⦶}Never(t-2)$ (see procedure *propose<terminating>()* in Algorithm 5). Proposition 4.6 (proof delayed to Appendix B) states that also this condition will arise in a finite number of steps.

Proposition 4.6. Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R}\rangle$ be a negotiation process ($k \geq 2$) where the NoN rule is enforced. If any agent $A \in \{0,1\}$ uses the strategy above, then, within a finite number of steps $T \geq \hat{T} \geq 2$ such that

procedure *propose<terminating>()*
  current_phase $\leftarrow$ terminating phase;
  if $R_A - R_A^{err} \subseteq \biguplus Never(t-1) \cup \biguplus Never(t-2)$ **then**
    // termination condition reached
    $\mathcal{P}_t \leftarrow \bot$;
    current_phase $\leftarrow$ done;

Algorithm 5: Terminating phase: procedure *propose<terminating>()*.

$\text{ag}(T) = A$, either an agreement is found or condition $R_A - R_A^{\text{err}} \subseteq \biguplus Never(T-1) \cup \biguplus Never(T-2)$ is satisfied.

Condition of Proposition 4.6 can be considered the *termination condition* of Proposition 3.6 in case agent A had admissible region $R_A - R_A^{\text{err}}$. Given that region R_A^{err} can be chosen as to be *arbitrarily small*, agent A can terminate the negotiation when this condition is reached. Any possible remaining acceptable deals would be in the (arbitrarily small) region R_A^{err}.

We stress again that, in case agent A is non-obstructionist from the beginning, for all $t \geq \hat{T}$ such that $\text{ag}(t) = A$, R_A^{err} can be made *empty*. Hence, as it happens for any acceptable deal in $R_A \cap \biguplus Never(t-2) = R_A \cap \biguplus Never(\hat{T}-2)$, any acceptable deal in R_A^{err} can be considered as an opportunity (with arbitrarily small Euclidean distance to $R_A \cap \biguplus Never(\hat{T}-2)$) that agent A had to sacrifice for having behaved in an obstructionist way (at most) up to step $\hat{T}-2$.

5. Handling Discrete and Categorical Variables

Discrete variables (e.g., integer and boolean) are very important to model requirements in many scenarios. In case all variables have a discrete and finite domain, the number of possible agreements is obviously finite. However, it would be practically *infeasible* to consider in turn all of them, given their huge number.

The NoN rule works also when (some of) the variables are discrete (e.g., integer or categorical), if we consider the union of the *integer hulls* [11] of the polyhedra in the NoN and $Never$ sets of Definition 3.4.

Integer Linear Programming results tell us that the integer hull of a polyhedron can still be represented with linear (plus integrality) constraints. Vertices of this new polyhedron have integer coordinates. Hence, the NoN rule as well as our NoN-compliant strategy of Section 4 and the underlying projection-based reasoning can be adapted to *prune* the space of the possible agreements: only the branches that deal with *now* decisions need to be refined (deals in $\biguplus NoN(t-1)$ proposed at step t need to have integer coordinates).

Also, $R_A - \biguplus Never(\hat{T}-2)$ can always be represented by an union of closed polyhedra, hence R_A^{err} can be always made empty.

Categorical variables can be tackled by fixing an ordering of their domain (common to both parties) and mapping them onto integers.

Example 5.1. (Alice vs. Bob, Categorical Negotiation Variables)
We consider a variation of the Alice vs. Bob example, where negotiation variables are t (start time) and

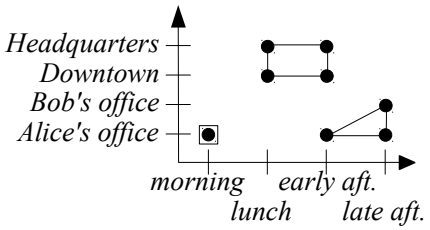


Figure 7: Alice vs. Bob with categorical variables: Alice's feasibility region (Example 5.1).

l (location). Both variables are *categorical*, with t having domain {morning, lunch, early aft., late aft.} and l having domain {Alice's office, Bob's office, Downtown, Headquarters}.

Figure 7 shows Alice's region. After having fixed a total ordering among the values of the domains of the two variables (common knowledge between agents), Alice's region can be regarded as the union of 3 bounded polyhedra, whose vertices have integer coordinates.

6. Implementation

Our framework has at its core well-studied tasks in computational geometry and (integer) linear programming [11]. However, to our knowledge, the exact complexity of the agent's reasoning is unknown, as these core tasks must be repeated on sets of exchanged deals. Still, existing libraries of computational geometry algorithms can manage the size of instances needed for practical scenarios.

We have implemented a Java system that uses the Parma Polyhedra Library (PPL) [12] to compute polyhedra, convex hulls, and projections, and an all-solutions SAT solver [13] to revise $\hat{n}(t)$ and Possible Opponent Clusters (POCs).

6.1. Computing Polyhedra, Convex Hulls and Projections

Our system uses the PPL [12] for computing and reasoning on polyhedra, convex hulls and projections.

PPL is a C++ library (with interfaces to other programming languages, among which Java) for the manipulation of numerical information that can be represented by points in some n-dimensional vector space. In particular, one of the key domains supported by PPL is that of convex polyhedra.

PPL is able to handle bounded and unbounded polyhedra, closed or not. Polyhedra are represented in two forms: the *implicit representation* by means of linear inequalities and the *explicit representation* by means of points, closure points (among which vertices), rays and lines. For each polyhedron of interest (e.g., convex hulls or projections) our system exploits the most efficient representation. Suitable operations among polyhedra (e.g., intersection) are executed by using the rich API provided by PPL.

6.2. Computing Possible Opponent Clusters

Our system computes and revises POCs by encoding the problem into a SAT instance, and uses Sat4j [13] to compute *all* solutions, as typically done in explicit model checking (see, e.g., [14, 15, 16, 17]).

Sat4j is a Java package for solving Boolean satisfaction and optimisation problems. It can solve SAT, All-SAT, Max-SAT, Pseudo-Boolean, and Minimally Unsatisfiable Subset problems. Being in Java, of course Sat4j cannot compete in speed with C/C++ solvers. However, as our All-SAT instances are quite small, using Sat4j resulted in a good compromise between performance and simplicity of interoperation with our Java implementation.

Our system implementing an agent (say agent A) computes, in each time point t such that $\mathrm{ag}(t) = A$, set $\boldsymbol{\mathcal{K}}(t)$ (see formula (1)) as follows. First, it incrementally computes and maintains the set of the $\subseteq$-minimal sets $\mathcal{D} \subseteq \mathrm{deals}_B(t-1)$ (where $B = 1 - A$) whose convex hull $\mathrm{conv}(\mathcal{D})$ intersects $\boxed{\Diamond} Never(t-1)$. Note that this is a dual representation of what requested by formula (1). Second, it reduces the problem of computing $\boldsymbol{\mathcal{K}}(t)$ into an instance of the Min Hyper-Graph Colouring Problem (Definition 6.1).

Definition 6.1. ((Min) Hyper-Graph Colouring Problem)
Let $H = (\mathcal{V}, \boldsymbol{\mathcal{E}})$ be a hyper-graph, where $\mathcal{V}$ is a finite set of nodes and $\boldsymbol{\mathcal{E}} \subseteq 2^{\mathcal{V}}$ is the set of hyper-edges (where each hyper-edge is a set of nodes). Let $n \in \mathbb{N}^+$ be a finite number of colours.

The *Hyper-Graph Colouring Problem* amounts to assign a colour (among the n available colours) to each node in $\mathcal{V}$ such that, for each hyper-edge $\mathcal{E} \in \boldsymbol{\mathcal{E}}$, not all nodes in $\mathcal{E}$ are assigned the same colour.

The *Min Hyper-Graph Colouring Problem* amounts to find a colouring to H satisfying the requirements above which uses the minimum number of distinct colours.

The reduction works as follows. The set of nodes of the hyper-graph is $\mathrm{deals}_B(t-1)$ and the hyper-edges are the sets $\mathcal{D}$ computed as described above. Intuitively, each such set defines a $\subseteq$-minimal set of incoming deals that *cannot* belong to the same POC (see formula (1)).

Note that *each* set of nodes (i.e., each set of opponent deals) coloured with the same colour in *each* solution of the Min Hyper-Graph Colouring Problem defines a POC. The whole set of POCs $\boldsymbol{\mathcal{K}}(t)$ is then the collection of the set of nodes coloured with the same colour in *all* solutions of the Min Hyper-Graph Colouring Problem.

At each time point t such that $\mathrm{ag}(t) = A$, our system (implementing agent A) encodes the Min Hyper-Graph Colouring Problem instance using the available agent knowledge into a SAT instance, assuming a fixed number of colours n, starting with $n = 1$. Then, it uses Sat4j to compute all solutions. If, at some time step t, the SAT instance for a given value of k is unsatisfiable, the system increases n by 1 and repeats the process.

The minimum value of colours which makes the SAT instance satisfiable is exactly the value $\hat{n}(t)$ used by Non-obstructionist Opponent Assumption (NOA). Note that, the minimum needed number of colours $\hat{n}(t)$ to colour the hyper-graph is a non-decreasing function of t. Hence, it is enough for the agent, in her next turn (at step $t+2$), to start the computation of $\boldsymbol{\mathcal{K}}(t+2)$ setting $n = \hat{n}(t)$.

For any given n (number of colours), the encoded SAT instance is defined on the following propositional letters:

$$\mathcal{L} = \{c_{v,i} \mid v \in \mathcal{V}, 1 \leq i \leq n\}\,.$$

The assignment $c_{v,i} = \text{true}$ in a model of the formula has the meaning that node v is coloured with colour i. The SAT instance is composed by the logical *and* of all clauses in all the following sets (along the lines of [18]):

At Least One Colour. Each node $v \in \mathcal{V}$ is assigned at least one colour. This yields the following

set of $|\mathcal{V}|$ clauses (one n-ary clause per node):

$$\{(c_{v,1} \vee \cdots \vee c_{v,n}) \mid v \in \mathcal{V}\}\,.$$

At Most One Colour. Each node $v \in \mathcal{V}$ is assigned at most one colour. This yields the following set of $|\mathcal{V}| \times \frac{n(n-1)}{2}$ clauses (a binary clause for each node and each pair of ordered distinct colours):

$$\{(\neg c_{v,i} \vee \neg c_{v,j}) \mid v \in \mathcal{V}, 1 \leq i < j \leq n\}\,.$$

Good Colouring. For each hyper-edge $\mathcal{E} \in \boldsymbol{\mathcal{E}}$, the nodes belonging to $\mathcal{E}$ are not all assigned to the same colour. This yields the following set of $n|\mathcal{E}|$ clauses (a $|\mathcal{E}|$-ary clause for each hyper-edge $\mathcal{E}$ and for each colour):

$$\{(\neg c_{v_1,i} \vee \cdots \vee \neg c_{v_e,i}) \mid \mathcal{E} = \{v_1, \ldots, v_e\} \in \boldsymbol{\mathcal{E}}, 1 \leq i \leq n\}\,.$$

Additional care is devoted in speeding-up the search of all solutions by Sat4j and in minimising RAM usage to store the set of all POCs. In particular, reformulation techniques on the SAT problem specification along the lines of [19, 20, 21, 22, 23, 24] have been employed to, e.g., perform symmetry breaking, and only the $\subseteq$-minimal POCs are kept in RAM, as the others are not needed to compute region $\Pi(t)$ (formula (2)), given that, for all regions R, R' and R'' such that $R'' \subseteq R$, it holds $\mathrm{proj}(R'', R') \subseteq \mathrm{proj}(R, R')$.

7. Experiments

In this section we present an empirical evaluation of the *computational feasibility* of the approach.

We evaluated our implementation on both random and structured negotiation scenarios using a *single* computer (a PC with a dual-core AMD Opteron 3GHz and 8GB RAM) for both agents. At each step, agents can exchange contracts of at most $k = 2$ deals.

Note that negotiations have been performed between two identical agents, although with different parameters. As a matter of facts, as our approach requires agents to comply with the NoN rule, it *cannot* be evaluated against other negotiators.

7.1. Random Negotiation Scenarios

In this section we describe our experimental results concerning random negotiation scenarios. In Section 7.1.1 we outline our random negotiation scenario generator. In Section 7.1.2 we describe the chosen experimental setting, in terms of the parameters we used to run our generator and of properties of the resulting random scenarios. Finally, in Section 7.1.3 we present our experimental results.

7.1.1. Generation of Random Negotiation Scenarios

In our framework a negotiation scenario consists of the following items: (a) The number of negotiation variables $n \in \mathbb{N}^+$; (b) Feasibility regions of the two agents; (c) Utility functions of the two agents; (d) Values Δu_0 and Δu_1 (positive reals), defining the concession policies of the two agents; (e) Values

ξ_0 and ξ_1 (reals between 0 and 1), defining the respond policies of the two agents. In turn, the feasibility region of each agent is the union of a finite number of bound polyhedra in $\mathbb{R}^n$ and its utility function is piece-wise linear.

Our random scenario generator takes as input the following parameters:

- A positive integer n, the number of negotiation variables;
- Two positive reals $\mathit{radius\_region}_0$ and $\mathit{radius\_region}_1$, defining the n-dimensional spheres in which the centres of all polyhedra of R_0 and R_1 will lie, respectively;
- Two positive reals $\mathit{radius\_poly}_0$ and $\mathit{radius\_poly}_1$, defining the maximum radius of each polyhedron of R_0 and R_1;
- Two bounded ranges $\mathit{n\_polys}_0$ and $\mathit{n\_polys}_1$ of $\mathbb{N}^+$, defining the minimum and maximum number of polyhedra defining R_0 and R_1, respectively;
- Two positive integers $\mathit{max\_vtx}_0$ and $\mathit{max\_vtx}_1$, defining the maximum number of vertices of each polyhedron in R_0 and R_1, respectively;
- The requested expected ratio of satisfiable scenarios *sat_ratio* (between 0 and 1), plus tolerance (a percentage value);
- Two bounded ranges conc_0 and conc_1 of $\mathbb{N}^+$, used to define the concession policy of agent 0 and 1, respectively.

Our generator does not take as input agents respond policies ξ_0 and ξ_1, as we aim at evaluating how the negotiation strategy of Section 4 behaves when changing values ξ_0 and/or ξ_1.

Our approach to random scenario generation consists of two main components: the feasibility region and the utility function (plus concession policy) generators.

7.1.1.1. Generation of Random Feasibility Regions In the literature, several models for random bounded polyhedra generation have been defined (see, e.g., [25] for a survey). We chose to adopt one of the simplest approaches, based on computing the convex hull of a given number of points randomly generated uniformly in a n-sphere of a given radius [26].

In details, for any agent $A \in \{0, 1\}$ our random polyhedra generator generates a number of polyhedra randomly chosen in interval $\mathit{n\_polys}_A$. Each such polyhedron P is computed by first randomly generating a point c_P in the n-sphere centred in the origin and having radius $\mathit{radius\_region}_A$. Polyhedron P is then computed by randomly generating $\mathit{max\_vtx}_A$ n-dimensional points in the n-sphere centred in c_P and having radius $\mathit{radius\_poly}_A$. Polyhedron P is defined as the convex hull of such random points. Each random point is generated by producing a random unit vector in $\mathbb{R}^n$ (to guess a direction) and a random length from 0 to $\mathit{radius\_poly}_A$. Of course, the generated polyhedron will have a number of vertices not greater than $\mathit{max\_vtx}_A$. It usually has less vertices, since some of them do not belong to the frontier of the computed convex hull. The generation of the feasibility region for each agent also avoids that two polyhedra overlap.

Finally, in order to control the ratio between satisfiable and unsatisfiable instances (i.e., with overlapping and non-overlapping regions), the feasibility region of one agent (agent 1, without loss of generality) is moved in order to be centred in a point having random coordinates ranging between 0 and *max_move*.

The choice of the value to assign to *max_move* is experimentally found by the generator itself during a preliminary *self-tuning* stage, given the requested ratio *sat_ratio* (plus tolerance) of satisfiable scenarios. In particular, during the self-tuning stage, the generator searches for a suitable value for *max_move* in a dichotomic fashion, each time generating a burst of scenarios, and measuring the ratio of satisfiable ones.

7.1.1.2. Generation of Random Utility Functions and Concession Policies For each agent $A \in \{0, 1\}$ and for each polyhedron P in R_A, we generate a random linear utility function u_A^P by computing a unit vector and the coefficients of the hyper-plane orthogonal to this. The utility function of agent A, u_A, is then the linear piece-wise function defined by u_A^P for each polyhedron P of R_A.

As for the concession policy Δu of each agent A, we first randomly choose the number of concessions $c_i \in \mathit{conc}_A$ to allow, and then compute Δu in such a way that the utility value could be decreased from its maximum exactly c_i times, i.e.,

$$\Delta u = \mathit{span}/c_i,$$

where *span* is the absolute difference of the extreme values of u_A in R_A.

7.1.2. Experimental Setting

We generated 100 random negotiation scenarios over 3 variables. Feasibility regions for both agents are unions of exactly 3 random polyhedra (i.e., $\mathit{polys}_0 = \mathit{polys}_1 = [3, 3]$), each with at most $\mathit{max\_vtx}_0 = \mathit{max\_vtx}_1 = 10$ vertices. Both regions lie within spheres having radius $\mathit{radius}_0 = \mathit{radius}_1 = 100$. As we aimed at generating both satisfiable and unsatisfiable scenarios, we set $\mathit{sat\_ratio} = 50\%$ (tolerance $\pm 10\%$). We set $\mathit{conc}_0 = \mathit{conc}_1 = [5, 5]$, as to force the concession policy of both agents to $\Delta u = \mathit{span}/5$.

In about 44% of the resulting instances we have $R_0 \cap R_1 \neq \emptyset$, i.e., an agreement does actually exist. The average volume of the intersection is 2.19% of the volume of each agent's region (standard deviation is 4.5%).

We set up a negotiation process for each random negotiation scenario described in Section 7.1.2 and for each combination of the agents respond policies ξ_0 and ξ_1, each one taking value in set $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$. In this way we were able to evaluate how the negotiation strategy of Section 4 behaves when changing values ξ_0 and/or ξ_1.

Overall, we defined and ran 3600 negotiation processes over the generated 100 random negotiation scenarios.

7.1.3. Experimental Results

All negotiation processes terminate in 20–30 steps. Even when agents were obstructionist (i.e., their respond policy ξ is lower than 1), agreements were actually found in $> 95\%$ of the scenarios for which $R_0 \cap R_1 \neq \emptyset$ (i.e., when agreements do actually exist).

Figure 8 shows average time, success rate (i.e., number of negotiations closed successfully divided by the number of negotiations such that $R_0 \cap R_1 \neq \emptyset$), and average *quality* of the agreement found for each agent as a function of the respond policies used (ξ_0 and ξ_1). The quality of an agreement D for agent $a \in \{0, 1\}$ is defined as:

$$\frac{u_a(D) - L_a}{H_a - L_a}$$

where H_a and L_a are, respectively, the highest and lowest values of agent a utility in $R_0 \cap R_1$. Hence, the quality of the agreement for agent a ranges from 0 to 1, and is 0 (respectively 1) if it yields her the lowest (respectively highest) utility among all existing agreements (i.e., deals in $R_0 \cap R_1$).

From Figure 8b it can be seen that *moderate* respond policies (intermediate values of ξ) lead to very high probabilities ($> 97\%$) of finding an agreement if one exists. Moreover, the quality of such agreements (Figure 8c) for the two agents is *similar* if their respond policies are *similar* (*fairness*). Conversely, if agents use very different values for ξ, the more conceding agent unsurprisingly gets lower utility with the agreement, but negotiations are more often aborted by the other, more demanding, agent. Finally, Figure 8a shows that negotiation time is always < 5 minutes.

7.2. Structured Negotiation Scenarios

We evaluated our system on multiple scenarios of three case-studies, as described in Section 7.2.1. Overall, we experimented with 6 structured negotiation processes.

7.2.1. Experimental Setting

Table 1 shows some relevant properties of these negotiation scenarios. Column "vars" gives the number of negotiation variables. Columns "polys" and "con" give, respectively, the number of polyhedra and the overall number of linear constraints defining each agent feasibility region, R_0 and R_1. The two last columns give the ratio of the volume of $R_0 \cap R_1$ (i.e., the volume of the space of the agreements) with respect to the volume of the feasibility region of each agent ("–" means that $R_0 \cap R_1$ is empty, hence no agreement is possible).

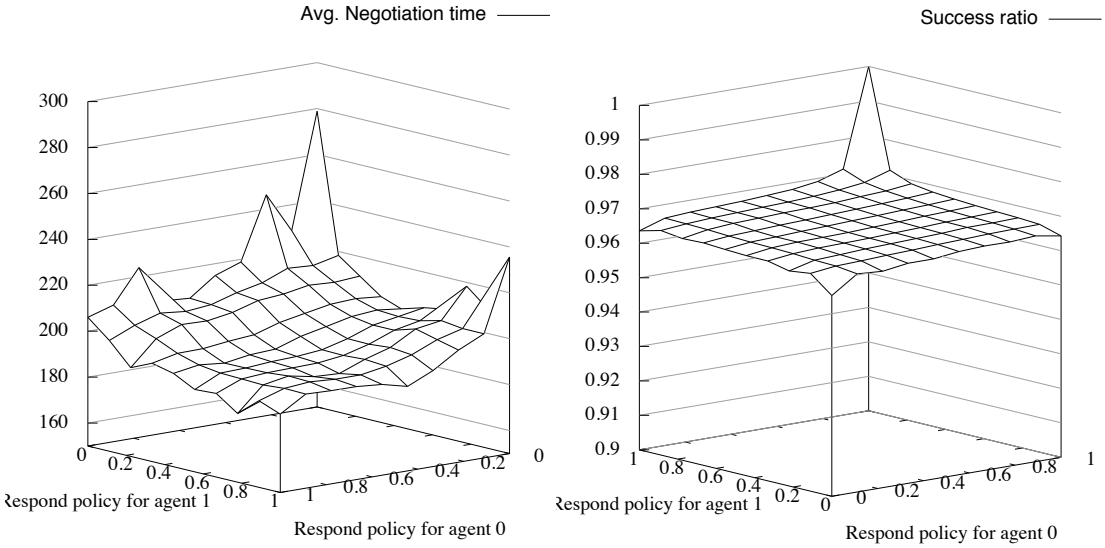
The following sections give more details on the structured negotiation problems and scenarios on which we ran our experiments. A full description of such scenarios is available in [27].

7.2.1.1. Alice vs. Bob This negotiation problem (over two negotiation variables: start day/time and duration) has already been introduced in Example 2.2 and used as a running example throughout the paper. We consider two negotiation scenarios (named AB1 and AB2), with different desiderata and requirements for the two agents.

From Table 1 we see that in scenario AB1 there is space for possible agreements (i.e., $R_0 \cap R_1 \neq \emptyset$) although the volume of this intersection is extremely small with respect to the volume of the feasibility regions of the two agents (i.e., $< 10^{-8}\%$). In scenario AB2 instead, there is no possibility to find an agreement, as $R_0 \cap R_1 = \emptyset$.

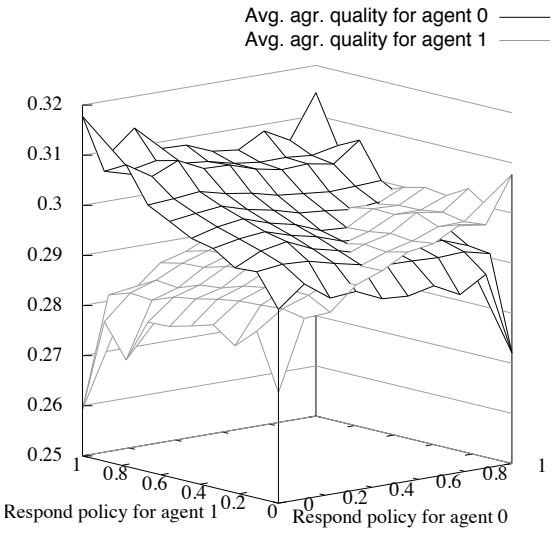
7.2.1.2. Summer House In this negotiation problem, agent Tenant is negotiating with agent Owner in order to rent a summer house. Negotiation processes are over 3 variables: the start day of the rental period, the rental duration (in days), the rental price per day.

We consider two scenarios of the Summer House problem (named SU1 and SU2), in which the Tenant has different preferences on the rental period (e.g., early June vs. late July), on the minimum and maximum rental duration, and different constraints about how much he/she is willing to accept to pay per day. The Owner has his/her own desiderata and constraints.



(a) Time (sec)

(b) Success rate



(c) Average agreement quality

Figure 8: Random negotiation scenarios: results.

From Table 1 we see that in scenario SU1 there is space for possible agreements (i.e., $R_0 \cap R_1 \neq \emptyset$) and the volume of this intersection is $1.5\% - 2.0\%$ with respect to the volume of the feasibility regions of the two agents. In scenario SU2 instead, there is no possibility to find an agreement, as $R_0 \cap R_1 = \emptyset$.

7.2.1.3. England vs. Zimbabwe In [28] a problem is described where England and Zimbabwe negotiate in order to reach an agreement evolving from the World Health Organisation Framework Convention on Tobacco Control.

There, the two parties want to find an agreement about four issues:

1. The total amount that England has to deposit into the Global Tobacco Fund to aid Zimbabwe to get rid of economic dependence on tobacco production;
2. The impact of the amount of the previous point to other aid programmes to Zimbabwe funded by England;
3. The possible changes to any trade barriers to incentivise or dis-incentivise import/export from/to the other country;
4. The possible creation of a forum to explore comparable arrangements for other long-term health issues (this could motivate Zimbabwe to follow the same approach to other global health agreements, turning out to be very costly to England).

We adapted this problem to our domain (real variables and no known bounds for their domains). Resulting negotiation processes are over 4 variables: the fund amount, other England-to-Zimbabwe aid reduction, increment of taxes on import by Zimbabwe government, increment of England import from Zimbabwe. We consider two negotiation scenarios (named EZ1 and EZ2), with different desiderata and requirements for the two agents.

From Table 1 we see that in both scenarios there are possible agreements (i.e., $R_0 \cap R_1 \neq \emptyset$), although the volume of this intersection is very small when compared to the volume of the feasibility regions of the two agents (namely: $0.04\% - 0.15\%$).

7.2.2. Experimental Results

Table 2 shows results of negotiation processes carried out on the above scenarios, under different values of the respond policies of each agent (ξ_0 and ξ_1). All processes have been run with k (the maximum number of deals in a proposal) equal to 2.

For each instance, column "agreement found" tells whether an agreement has been found (an agreement exists if and only if $R_0 \cap R_1 \neq \emptyset$, see Table 1), column "steps" gives the number of negotiation steps needed to conclude the negotiation process, column "overall time (sec)" gives the overall negotiation time in seconds, and column "polys" gives the overall number of polyhedra computed by Parma Polyhedra Library (PPL) during the process. For each negotiation instance, the number of All-SAT instances solved to compute POCs (see formula (1)) is equal to the number of negotiation steps.

Table 2 reports results for particularly meaningful values of ξ_0 and ξ_1, each one taking a value in set $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$. In particular, for scenarios in which an agreement exists, Table 2 reports a negotiation process where agents respond policies (ξ_0 and ξ_1) were set to the lowest values for which

Scenario	vars	R_0		R_1		$\frac{vol(R_0 \cap R_1)}{vol(R_0)}$	$\frac{vol(R_0 \cap R_1)}{vol(R_1)}$
		polys	con.	polys	con.		
AB1	2	3	12	3	12	$< 10^{-8}\%$	$< 10^{-8}\%$
AB2	2	3	12	5	20	–	–
SU1	3	3	12	4	16	1.5%	2.0%
SU2	3	3	12	4	16	–	–
EZ1	4	2	8	2	8	0.10%	0.05%
EZ2	4	2	8	2	8	0.15%	0.04%

Table 1: Properties of structured negotiation scenarios.

an agreement was actually found. In case either ξ_0 or ξ_1 is different from 0, Table 2 reports a second (unsuccessful) process for that scenario, run by further (minimally) decreasing one between ξ_0 and ξ_1 and by setting the other to 1 (non-ostructionist agent). For example, as for scenario AB1 (for which an agreement exists, see Table 1),Table 2 reports configuration $\xi_0 = 0$, $\xi_1 = 0.6$ for which an agreement was found. By further reducing ξ_1 to 0.4, no agreement was found even when maximally relaxing ξ_0 to 1 (i.e., making agent 1 non-obstructionist). Analogously, as for scenario SU1, Table 2 reports configurations $\xi_0 = 0.4$, $\xi_1 = 0$ (agreement found) and $\xi_0 = 0.2$, $\xi_1 = 1$ (agreement not found).

As for negotiation scenarios for which no agreement exists (namely: AB2 and SU2, see Table 1), Table 2 reports configuration $\xi_0 = 1$, $\xi_1 = 1$, where both agents are non-obstructionist (most favourable configuration).

7.3. Discussion

Our results on both random and structured negotiation scenarios show that enforcing NoN is computationally feasible. Negotiation processes with *hundreds* of interaction steps could be performed in minutes, even when NoN enforcement and agents reasoning require the computation of *millions* of polyhedra and the resolution of *hundreds* of All-SAT instances.

In particular, All-SAT instances are always trivial to encode and to solve, as overall the time needed to encode them, to run Sat4j and to loop across and to decode all returned models is always less than 15% of the overall time.

The most computationally intensive part of our negotiator is in using PPL (which takes, on average, more than 50% time for each negotiation step).

Although the number of sets in $\boldsymbol{\mathcal{K}}(t)$, NoN and $Never$ collections can grow exponentially, by keeping only their $\subseteq$-minimal (as for $\boldsymbol{\mathcal{K}}(t)$, see Section 6.1) and their $\subseteq$-maximal (as for NoN and $Never$) members (which is enough to enforce the NoN rule and to perform the needed reasoning), the overall memory requirements become, in the considered scenarios, compatible with the amount of RAM available on an ordinary PC.

Scenario	ξ_0	ξ_1	agreement found	steps	overall time (sec)	polys
AB1	0	0.6	Y	20	1.09	321
AB1	1	0.4	N	24	1.25	450
AB2	1	1	N	20	1.33	466
SU1	0.4	0	Y	347	15.22	5679
SU1	0.2	1	N	417	38.44	20 413
SU2	1	1	N	513	63.12	29 148
EZ1	0	0	Y	80	1237	3 115 508
EZ2	0	0	Y	92	2836	8 057 011

Table 2: Structured negotiation scenarios: results.

8. Related Work

Various protocols and algorithms for negotiation have been proposed in the literature, to handle the different needs of several application domains, as, e.g., resource allocation [29], scheduling [30], e-business [31], smart grids [32, 33, 34]. Available approaches to negotiation can be classified according to several factors, as the negotiation objects, the agents' decision making models, the degree of cooperation among the agents, the level of privacy about constraints and preferences of each agent, or the communication and computation costs (see, e.g., [4, 35, 36, 37] and citations thereof).

At the top level, existing approaches to automated negotiation can be classified depending on whether they make or do not make use of a central authority, called *mediator*, which is trusted by the negotiating parties. While mediator-assisted negotiation approaches (see, e.g., [38, 39]) can ensure a higher efficiency in the process, they are unsuitable in a setting as ours, where agents have strong privacy concerns and lack trust about their counterpart and any mediator.

The papers closest to ours are [8, 7, 9], where a computational geometry based approach to bilateral unmediated negotiation was investigated. However, in order to guarantee completeness and termination, such approaches make strong assumptions on the properties of the two agents. In particular, [8, 9] require that the feasibility region of both agents is a single convex bounded polyhedron and that agents are non-obstructionists. Convexity is partially relaxed in [7], where the concept of *safe lie* is defined: agents can lie by rejecting acceptable incoming proposals, but they must continue to *appear* (to their counterpart) as having a convex region for the approach to terminate. The Now or Never (NoN) protocol rule relaxes all such limitations and allows each single agent to employ a terminating strategy in a very hostile setting, where no hypothesis about the shape of the feasibility region and the willingness to collaborate and to be efficient can be made about the counterpart. Such terminating strategies for an agent (as the one presented in Section 4) do exist under very mild assumptions about her own feasibility region and utility, which are applicable to most practical circumstances.

The possibility of agents to *lie* has been explored also in [40], where negotiations involve agents ex-

changing *knowledge* beyond proposals. In that context, agents may deliberately send to their counterpart false knowledge (hence *dishonest proposals*), with the goal to deceive the reasoning of their opponent. The NoN protocol rule requires only a minimal negotiation framework (see Definition 2.1), where exchanged proposals contain *only* deals (and not additional information or knowledge). Although the NoN rule can be enforced in more sophisticated negotiation frameworks (see discussion in Sections 1 and 2), the only dishonest proposals which are strictly of interest for NoN are those including deals that the proposing agent is not really willing to accept. However, this is not a problem, as additional mechanisms (e.g., penalties) can be easily introduced in order to cope with situations where an agent accepts a deal proposed by the counterpart who then refuses to honour the agreement. On the other hand, the NoN protocol rule allows agents to lie when receiving a proposals, in that the receiving agent is not forced to accept an acceptable incoming deal or to take a *now* decision whenever possible. Hence, the NoN rule leaves the agents free to behave in an obstructionist way (see Definition 3.7), but at the same time injects a minimum amount of efficiency in the process in order to avoid infinite negotiations.

Game-theoretic approaches (see, e.g., [4, 41, 42, 43, 44, 45, 46]) typically assume that agents have complete or probabilistic information about their counterpart, and often focus on *split-the-pie* games, in order to ensure desirable properties of the agreements (maximum social welfare, envy-freeness, Pareto-optimality, etc.) For these reasons, although such methods are precious in several negotiation domains, they cannot be applied in a setting as ours, where the agents have *zero* knowledge and no trust about their counterpart.

Argumentation-based approaches emphasise the importance of exchanging information and explanations between negotiating agents during the process, in order to mutually influence their behaviours [47, 48]. Integrating argumentation theory in negotiation could of course supply additional information and help agents to convince each other by adequate arguments. However, such approach cannot be exploited in our context, where agents do not trust each other, hence would not believe in their arguments.

Heuristic approaches (see, e.g., [37, 2, 3, 49, 50]) tackle cases where agents have lack of information about their counterpart. However, they are intrinsically incomplete (i.e., they may fail to find an agreement if one exist) and are not guaranteed to terminate in a finite number of steps, without making assumptions on the two agents. On the other hand, in negotiation processes where the NoN rule is enforced, any single agent may exploit a terminating strategy, despite the properties and willingness to terminate of the counterpart. It also guarantees completeness (i.e., an agreement *will* be found if one exists) in case the negotiation process is carried our by non-obstructionist agents (Proposition 3.8). In case the agents are obstructionist, the approach is of course incomplete (as an obstructionist counterpart does *not* do her best to reach an agreement, e.g., she can reject acceptable offers, see Definition 3.7), but makes the responsibility of each agent about the negotiation failure explicit. In particular, Proposition 3.6 shows that, in case the negotiation process fails, for each existing mutually acceptable agreement D there was a negotiation step t in which the agent in charge of making a proposal during step t deliberately took a *never* decision on a NoN region containing D. Note that, even in presence of an obstructionist counterpart, the NoN rule makes the agent able to employ a terminating strategy.

9. Conclusions

In this paper we defined a new protocol rule, Now or Never (NoN), for bilateral unmediated negotiation processes which allows self-motivated competitive agents to *efficiently* carry out multi-variable negoti-

ations with remote untrusted parties, where privacy is a major concern and agents know *nothing* about their opponent. NoN has been explicitly designed as to ensure a continuous progress of the negotiation, thus neutralising malicious or inefficient opponents. The protocol rule (whose fulfilment can be assessed *independently* by each party using only *local* information) forces the agents to never reconsider already taken decisions, thus injecting a minimum, but sufficient amount of *efficiency* in the process. In particular, NoN allows each agent to independently infer, during the negotiation, that there is no hope to find an agreement (Proposition 3.6). When such a termination condition arises, the agent can safely opt-out.

The NoN protocol rule is non-invasive, in that it leaves maximum freedom to agents to exploit their (private) strategy. In particular, it does allow agents to behave in an obstructionist way (e.g., agents are free to reject acceptable deals if they are, e.g., currently aiming at higher utility). We also defined the notion of non-obstructionist agents. Although they are not necessarily collaborative, non-obstructionist agents genuinely aim at closing the negotiation efficiently, by also sacrificing their preferences among deals they are willing to accept. Proposition 3.8 shows that, in case both agents are non-obstructionist, then the NoN rule guarantees *completeness*: when the termination condition arises, not only agents know that no mutually acceptable agreement can still be found; agents have also a *proof* that no mutually acceptable agreement actually *exists*.

The enforcement of the NoN protocol rule in a negotiation process also allows each agent to exploit terminating strategies, i.e., strategies that are guaranteed to yield the termination condition after a finite number of steps. We have also presented one such NoN-compliant terminating strategy inspired to the well-known Monotonic Concessions (MC) approach. Our strategy, under mild assumptions on the agent feasibility region, allows the agent to derive, in a *finite* number of steps and *independently* of the behaviour of the opponent, that there is *no hope* to find an agreement. Our multi-phase NoN compliant agent strategy supports the presence of a private utility function. Also, the strategy includes a sophisticated reasoning activity of the agent (based on the evidence provided by the behaviour of her counterpart) in order to effectively prune the space of possible agreements.

We have presented a Java implementation of an agent employing our NoN-compliant strategy. Our implementation jointly exploits a computational geometry package (Parma Polyhedra Library (PPL) [12]) and an all-solutions SAT solver (Sat4j [13]) to implement the required reasoning.

We finally evaluated the computational feasibility of the overall approach on both random and structured instances of practical size. Experiments show that each agent can efficiently evaluate the compliancy of the opponent to the NoN protocol rule in real-world negotiation scenarios.

Acknowledgements

This research was founded by the EU 7th Framework Programme under grant agreements n. 317761 (SmartHG) and n. 600773 (PAEON). The author wishes to thank the anonymous reviewers for their useful comments and suggestions, which improved the quality of this paper.

References

[1] Mancini T. Now or Never: Negotiating Efficiently with Unknown Counterparts. In: Proceedings of 22nd RCRA International Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2015). vol. 1451. CEUR Workshop Proceedings; 2015. p. 47–61.

[2] Faratin P, Sierra C, Jennings NR. Using Similarity Criteria to Make Issue Trade-Offs in Automated Negotiations. Artificial Intelligence. 2002;142(2):205–237.

[3] Lin R, Kraus S, Wilkenfeld J, Barry J. Negotiating with Bounded Rational Agents in Environments with Incomplete Information using an Automated Agent. Artificial Intelligence. 2008;172(6–7):823–851.

[4] Rosenschein JS, Zlotkin G. Rules of Encounter: Designing Conventions for Automated Negotiations Among Computers. MIT Press; 1994.

[5] Rubinstein A. Perfect Equilibrium in a Bargaining Model. Econometrica. 1982;50(1):97–109. doi:10.2307/1912531.

[6] Faratin P, Sierra C, Jennings NR. Negotiation Decision Functions for Autonomous Agents. Robotics and Autonomous Systems. 1998;24(3–4):159–182.

[7] Mancini T. Negotiation Exploiting Reasoning by Projections. In: Proceedings of 7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009). Advances in Intelligent Systems and Computing. Springer; 2009. p. 329–338.

[8] Cadoli M. Proposal-based negotiation in convex regions. In: Proceedings of 7th International Workshop on Cooperative Information Agents (CIA 2003). vol. 2782 of Lecture Notes in Computer Science. Springer; 2003. p. 93–108.

[9] Costantini S, De Gasperis G, Provetti A, Tsintza P. A Heuristic Approach to Proposal-Based Negotiation: with Applications in Fashion Supply Chain Management. Mathematical Problems in Engineering. 2013;doi:10.1155/2013/896312.

[10] Lai G, Sycara K. A Generic Framework for Automated Multi-Attribute Negotiation. Group Decision and Negotiation. 2009;18(2):169–187. doi:10.1007/s10726-008-9119-9.

[11] Schrijver A. Theory of Linear and Integer Programming. John Wiley & Sons; 1998.

[12] Bagnara R, Hill PM, Zaffanella E. The Parma Polyhedra Library: Toward a Complete Set of Numerical Abstractions for the Analysis and Verification of Hardware and Software Systems. Science of Computer Programming. 2008;72(1–2):3–21.

[13] Le Berre D, Parrain A. The Sat4j Library, Release 2.2. Journal on Satisfiability, Boolean Modeling and Computation. 2010;7:59–64.

[14] Della Penna G, Intrigila B, Melatti I, Minichino M, Ciancamerla E, Parisse A, et al. Automatic Verification of a Turbogas Control System with the Murphi Verifier. In: Proceedings of 6th International Workshop on Hybrid Systems: Computation and Control (HSCC 2003). vol. 2623 of Lecture Notes in Computer Science. Springer; 2003. p. 141–155.

[15] Cesta A, Finzi A, Fratini S, Orlandini A, Tronci E. Validation and Verification Issues in a Timeline-Based Planning System. Knowledge Engineering Review. 2010;25(3):299–318. doi:10.1017/S0269888910000160.

[16] Mancini T, Mari F, Massini A, Melatti I, Tronci E. System Level Formal Verification via Distributed Multi-Core Hardware in the Loop Simulation. In: Proceedings of 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP 2014). IEEE; 2014. p. 734–742. doi:10.1109/PDP.2014.32.

[17] Mancini T, Mari F, Massini A, Melatti I, Tronci E. Anytime System Level Verification via Random Exhaustive Hardware In The Loop Simulation. In: Proceedings of 17th Euromicro Conference on Digital System Design (DSD 2014). IEEE; 2014. p. 236–245.

[18] Cadoli M, Mancini T, Patrizi F. SAT as an Effective Solving Technology for Constraint Problems. In: Proceedings of 16th International Symposium on Foundations of Intelligent Systems (ISMIS 2006). vol. 4203 of Lecture Notes in Computer Science. Springer; 2006. p. 540–549.

[19] Mancini T, Cadoli M. Detecting and Breaking Symmetries by Reasoning on Problem Specifications. In: Proceedings of 6th International Symposium on Abstraction, Reformulation and Approximation (SARA 2005). vol. 3607 of Lecture Notes in Computer Science. Springer; 2005. p. 165–181.

[20] Cadoli M, Mancini T. Automated Reformulation of Specifications by Safe Delay of Constraints. Artificial Intelligence. 2006;170(8–9):779–801.

[21] Cadoli M, Mancini T. Using a Theorem Prover for Reasoning on Constraint Problems. Applied Artificial Intelligence. 2007;21(4&5):383–404. doi:10.1080/08839510701252650.

[22] Mancini T, Cadoli M, Micaletto D, Patrizi F. Evaluating ASP and Commercial Solvers on the CSPLib. Constraints. 2008;13(4):407–436.

[23] Bordeaux L, Cadoli M, Mancini T. A Unifying Framework for Structural Properties of CSPs: Definitions, Complexity, Tractability. Journal of Artificial Intelligence Research. 2008;32:607–629. doi:10.1613/jair.2538.

[24] Mancini T, Cadoli M. Exploiting Functional Dependencies in Declarative Problem Specifications. Artificial Intelligence. 2007;171(16–17):985–1010.

[25] Molchanov IS, Stoyan D. Statistical Models of Random Polyhedra. CWI, Amsterdam, The Netherlands; 1995. BS-R9510.

[26] Stoyan D, Stoyan H. Fractals, Random Shapes and Point Fields. John Wiley & Sons; 1994.

[27] Mancini T. Now or Never: Negotiating Efficiently with Unknown or Untrusted Counterparts (Appendix); 2016. Available from: http://tmancini.di.uniroma1.it.

[28] Lin R, Kraus S, Baarslag T, Tykhonov D, Hindriks KV, Jonker CM. Genius: an Integrated Environment for Supporting the Design of Generic Automated Negotiators. Computational Intelligence. 2014;30(1):48–70. doi:10.1111/j.1467-8640.2012.00463.x.

[29] Conry SE, Kuwabara K, Lesser RA V R Meyer. Multistage Negotiation for Distributed Constraint Satisfaction. IEEE Transactions on Systems, Man and Cybernetics. 1991;21(6):462–477.

[30] Sycara KP, Roth S, Sadeh N, Fox M. Distributed Constrained Heuristic Search. IEEE Transactions on Systems, Man and Cybernetics. 1991;21(6):446–461.

[31] Jennings NR, Norman TJ, Faratin P, O'Brien P, Odgers B. Autonomous Agents for Business Process Management. Applied Artificial Intelligence. 2000;14(2):145–189.

[32] Wang Z, Wang L. Adaptive Negotiation Agent for Facilitating Bi-Directional Energy Trading Between Smart Building and Utility Grid. IEEE Transactions on Smart Grid. 2013;4(2):702–710. doi:10.1109/TSG.2013.2237794.

[33] Vrba P, Marik V, Siano P, Leitao P, Zhabelova G, Vyatkin V, et al. A Review of Agent and Service-Oriented Concepts Applied to Intelligent Energy Systems. IEEE Transactions on Industrial Informatics. 2014;10(3):1890–1903. doi:10.1109/TII.2014.2326411.

[34] Mancini T, Mari F, Melatti I, Salvo I, Tronci E, Gruber J, et al. Demand-Aware Price Policy Synthesis and Verification Services for Smart Grids. In: Proceedings of 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm 2014). IEEE; 2014. p. 794–799.

[35] Kraus S. Negotiation and Cooperation in Multi-Agent Environments. Artificial Intelligence. 1997;94(1–2):79–98.

[36] Yokoo M, Katsutoshi H. Algorithms for Distributed Constraint Satisfaction: A Review. Autonomous Agents and Multi-Agents Systems. 2000;3(2):185–207.

[37] Jennings NR, Faratin P, Lomuscio AR, Parsons S, Wooldridge M, Sierra C. Automated Negotiation, Prospects, Methods and Challenges. Group Decision and Negotiation. 2001;10:199–215.

[38] Hemaissia M, El Fallah-Seghrouchni A, Labreuche C, Mattioli J. A multilateral Multi-Issue Negotiation Protocol. In: Proceedings of 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007). IFAAMAS; 2007. p. 155. doi:10.1145/1329125.1329314.

[39] Ito T, Klein M, Hattori H. A Multi-Issue Negotiation Protocol among Agents with Nonlinear Utility Functions. Multiagent and Grid Systems. 2008;4(1):67–83. Available from: http://content.iospress.com/articles/multiagent-and-grid-systems/mgs00092.

[40] Son TC, Pontelli E, Nguyen NH, Sakama C. Formalizing Negotiations Using Logic Programming. ACM Transactions on Computational Logic. 2014;15(2):12:1–12:30. doi:10.1145/2526270.

[41] Zlotkin G, Rosenschein JS. Mechanism Design for Automated Negotiation, and its Application to Task Oriented Domains. Artificial Intelligence. 1996;86(2):195–244. doi:10.1016/0004-3702(95)00104-2.

[42] Zlotkin G, Rosenschein JS. Mechanisms for Automated Negotiation in State Oriented Domains. Journal of Artificial Intelligence Research. 1996;5:163–238.

[43] Fatima SS, Wooldridge M, Jennings NR. Multi-Issue Negotiation with Deadlines. Journal of Artificial Intelligence Research. 2006;27:381–417. doi:10.1613/jair.2056.

[44] Gottlob G, Greco G, Mancini T. Complexity of Pure Equilibria in Bayesian Games. In: Proceedings of 20th International Joint Conference on Artificial Intelligence (IJCAI 2007); 2007. p. 1294–1299.

[45] Chevaleyre Y, Endriss U, Estivie S, Maudet N. Reaching Envy-Free States in Distributed Negotiation Settings. In: Proceedings of 20th International Joint Conference on Artificial Intelligence (IJCAI 2007); 2007. p. 1239–1244. Available from: http://dli.iiit.ac.in/ijcai/IJCAI-2007/PDF/IJCAI07-200.pdf.

[46] Saha S, Sen S. An Efficient Protocol for Negotiation over Multiple Indivisible Resources. In: Proceedings of 20th International Joint Conference on Artificial Intelligence (IJCAI 2007); 2007. p. 1494–1499. Available from: http://dli.iiit.ac.in/ijcai/IJCAI-2007/PDF/IJCAI07-241.pdf.

[47] Kraus S, Sycara K, Evenchik A. Reaching Agreements through Argumentation: a Logical Model and Implementation. Artificial Intelligence. 1998;104:1–69.

[48] Amgoud L, Dimopoulos Y, Moraitis P. A Unified and General Framework for Argumentation-based Negotiation. In: Proceedings of 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007). IFAAMAS; 2007. p. 967–974.

[49] López-Carmona MA, Marsá-Maestre I, de la Hoz E, Velasco JR. A Region-Based Multi-Issue Negotiation Protocol for Nonmonotonic Utility Spaces. Computational Intelligence. 2011;27(2):166–217. doi:10.1111/j.1467-8640.2011.00377.x.

[50] Williams CR, Robu V, Gerding EH, Jennings NR. Negotiating Concurrently with Unknown Opponents in Complex, Real-Time Domains. In: Proceedings of 20th European Conference on Artificial Intelligence (ECAI 2012). vol. 242 of Frontiers in Artificial Intelligence and Applications. IOS Press; 2012. p. 834–839. doi:10.3233/978-1-61499-098-7-834.

A. List of Acronyms

MC Monotonic Concessions 36

NNC Not Necessarily Closed 22

NOA Non-obstructionist Opponent Assumption 41

NoN Now or Never 40

POC Possible Opponent Cluster 25

PPL Parma Polyhedra Library 36

B. Proof of Results

In this section we give proofs of our results.

Proposition 3.6. (Termination Condition)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process where the Now or Never (NoN) rule is enforced and let $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_T$ be the status of π at step $T \geq 2$.

If $R_{\text{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\Diamond Never(T-1) \cup \Diamond\!\!\!\!\!\!\Diamond Never(T-2)$ and $\mathcal{P}_T$ is not a singleton $\{D\} \subseteq \mathcal{P}_{T-1}$, then:

(a) There exists no extension $\vec{\mathcal{P}}' = \mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{T-1}, \mathcal{P}_T, \ldots, \mathcal{P}_{T'}$ of $\vec{\mathcal{P}}$ to step $T' > T$ such that $\mathcal{P}_{T'} = \{D\} \subseteq \mathcal{P}_{T'-1}$

(b) For all $D \in R_0 \cap R_1$, there exists $1 < t_D < T$ such that $D \in \Diamond\!\!\!\!\!\!\Diamond NoN(t_D - 1) \cap \Diamond\!\!\!\!\!\!\Diamond Never(t_D)$.

Proof:
Proof of (a) builds on the observation that, from Definition 3.4, $\Diamond\!\!\!\!\!\!\Diamond Never(t) \supseteq \Diamond\!\!\!\!\!\!\Diamond Never(t-2)$ for all $t \geq 2$. Assume, for the sake of contradiction, that there exists step $T' > T$ such that $\mathcal{P}_{T'} = \{D\} \subseteq \mathcal{P}_{T'-1}$ (i.e., agreement $D \in R_0 \cap R_1$ is accepted at step $T' > T$).

As $D \in R_0 \cap R_1 \subseteq R_{\text{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\Diamond Never(T-1) \cup \Diamond\!\!\!\!\!\!\Diamond Never(T-2)$, we have that:

1. If T' and T have the same parity (i.e., $T' = T + 2n$ for some integer $n \geq 1$):
   1.1. if $D \in \Diamond\!\!\!\!\!\!\Diamond Never(T-2) = \Diamond\!\!\!\!\!\!\Diamond Never(T'-2-2n) \subseteq \Diamond\!\!\!\!\!\!\Diamond Never(T'-2)$, then, by Definition 3.5, D cannot be part of $\mathcal{P}_{T'}$
   1.2. otherwise ($D \in \Diamond\!\!\!\!\!\!\Diamond Never(T-1) - \Diamond\!\!\!\!\!\!\Diamond Never(T-2) = \Diamond\!\!\!\!\!\!\Diamond Never(T'-1-2n) - \Diamond\!\!\!\!\!\!\Diamond Never(T-2) \subseteq \Diamond\!\!\!\!\!\!\Diamond Never(T'-3)$), by Definition 3.5, D cannot be part of $\mathcal{P}_{T'-1}$.

2. Otherwise (T' and T have different parity, i.e., $T' = T - 1 + 2n$ for some integer $n \geq 1$):

   2.1. if $D \in \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2) = \Diamond\!\!\!\!\!\!\Big\downarrow Never(T'-1-2n) \subseteq \Diamond\!\!\!\!\!\!\Big\downarrow Never(T'-3)$, then, by Definition 3.5, D cannot be part of $\mathcal{P}_{T'-1}$

   2.2. otherwise ($D \in \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-1) - \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2) = \Diamond\!\!\!\!\!\!\Big\downarrow Never(T'-2n) - \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2) \subseteq \Diamond\!\!\!\!\!\!\Big\downarrow Never(T'-2)$), by Definition 3.5, D cannot be part of $\mathcal{P}_{T'}$.

Hence, a contradiction arises and point (a) follows.

As for point (b), we know by hypothesis that, for all $D \in R_0 \cap R_1$, $D \in R_{\mathrm{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-1) \cup \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2)$. Let t_D be the smallest step such that $D \in \Diamond\!\!\!\!\!\!\Big\downarrow Never(t_D)$. From Definition 3.4 we have that $t_D > 1$. As $D \notin \Diamond\!\!\!\!\!\!\Big\downarrow Never(t_D - 2)$, Definition 3.4 also ensures that $D \in \Diamond\!\!\!\!\!\!\Big\downarrow NoN(t_D - 1)$.

□

Proposition 3.8. (Completeness)
Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process between two non-obstructionist agents where the NoN rule is enforced.

If π reaches, at step $T - 1 \geq 2$, status $\vec{\mathcal{P}} = \mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_{T-1}$ s.t. $R_{\mathrm{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-1) \cup \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2)$, then $R_0 \cap R_1 = \emptyset$.

Proof:
As agents $\mathrm{ag}(T-1)$ and $\mathrm{ag}(T) = \mathrm{ag}(T-2) = 1 - \mathrm{ag}(T-1)$ are non-obstructionist, we have:

(i) $R_{\mathrm{ag}(T-1)} \cap \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-1) = \emptyset$ and

(ii) $R_{\mathrm{ag}(T)} \cap \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2) = \emptyset$.

To see why, assume, for the sake of contradiction that, e.g., (ii) does not hold, i.e., there exists $D \in R_{\mathrm{ag}(T)} \cap \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2)$ (case (i) can be proved similarly). By Definition 3.4, there exists a step $1 < t_D \leq T - 2$ such that:

(a) $\mathrm{ag}(t_D) = \mathrm{ag}(T-2)$, and

(b) $D \in \Diamond\!\!\!\!\!\!\Big\downarrow NoN(t_D - 1) - \Diamond\!\!\!\!\!\!\Big\downarrow Never(t_D - 2)$ and $D \in \Diamond\!\!\!\!\!\!\Big\downarrow Never(t_D)$.

Agent $\mathrm{ag}(t_D)$, by not proposing in $\Diamond\!\!\!\!\!\!\Big\downarrow NoN(t_D - 1)$ containing $D \in R_{\mathrm{ag}(t_D)}$ at step t_D, did not satisfy requirement (2) of Definition 3.7.

Now, as for the main proof, assume, for the sake of contradiction, that exists a deal $D \in R_0 \cap R_1$. As $R_0 \cap R_1 = R_{\mathrm{ag}(T)} \cap R_{1-\mathrm{ag}(T)} \subseteq R_{\mathrm{ag}(T)} \subseteq \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-1) \cup \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2)$, we must have $D \in \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-1) \cup \Diamond\!\!\!\!\!\!\Big\downarrow Never(T-2)$. A contradiction arises by (i) and (ii). □

Proposition 4.3. If, at step $t \geq 3$ such that $\mathrm{ag}(t) = A$, Non-obstructionist Opponent Assumption (NOA) is correct, then:

$$\Pi(t) \cap (R_B - \Diamond\!\!\!\!\!\!\Big\downarrow Never(t-1)) = \emptyset.$$

Proof:
Assume that, at step $t \geq 3$, NOA is correct. As $\hat{n}(t)$ is the minimum number of convex sub-regions of $R_B - \biguplus Never(t-1)$ (where $B = 1 - A$), each such sub-region contains at least one deal already offered by the opponent (i.e., one of $\mathrm{deals}_B(t-1)$). Intuitively, this means that the opponent (agent $B = 1 - A$) has proposed at least one deal within each of the $\hat{n}(t)$ convex sub-regions of R_B.

As, by construction, members of $\mathcal{K}(t)$ are all and the only subsets of $\mathrm{deals}_B(t-1)$ that may belong to the same convex sub-region of $R_B - \biguplus Never(t-1)$ (as NOA is assumed to be correct), we have that for any $\mathcal{D} \subseteq \mathrm{deals}_B(t-1)$ such that $\mathrm{conv}(\mathcal{D}) \subseteq R_B - \biguplus Never(t-1)$ (i.e., for any set of the opponent deals that *really* belong to one such a convex sub-region), $\mathcal{D}$ is in $\mathcal{K}(t)$.

Assume now, for the sake of contradiction, that there exists point $X \in \Pi(t)$ such that $X \in R_B - \biguplus Never(t-1)$. We have that:

1. As $X \in \Pi(t)$, then we must have $X \in \mathrm{proj}(\mathrm{conv}(\mathcal{D}), \biguplus Never(t-1))$ for *each* $\mathcal{D} \in \mathcal{K}(t)$. In particular, this holds for those POCs $\mathcal{D}$ for which $\mathrm{conv}(\mathcal{D}) \subseteq R_B - \biguplus Never(t-1)$ (i.e., those whose points *really* belong to a single convex sub-region of $R_B - \biguplus Never(t-1)$).

2. This implies (see Definition 4.2) that for each $\mathcal{D} \in \mathcal{K}(t)$ such that $\mathrm{conv}(\mathcal{D}) \subseteq R_B - \biguplus Never(t-1)$ there exists a point $X' \in \mathrm{conv}(\mathcal{D})$ and $Y \in \biguplus Never(t-1)$ such that $Y \in \overline{XX'}$.

3. As $\mathrm{conv}(\mathcal{D} \cup \{X\})$, containing $Y \in \biguplus Never(t-1)$, does not entirely belong to $R_B - \biguplus Never(t-1)$, deal X cannot belong to the same convex sub-region of $R_B - \biguplus Never(t-1)$ as $\mathcal{D}$.

4. As this holds for all $\mathcal{D} \in \mathcal{K}(t)$ such that $\mathrm{conv}(\mathcal{D}) \subseteq R_B - \biguplus Never(t-1)$, a contradiction arises.

□

Proposition 4.5. For each step $t \geq \hat{T}$ such that $\mathrm{ag}(t) = A$, $R_A \cap \biguplus Never(t-2) = R_A \cap \biguplus Never(\hat{T}-2)$.

Proof:
By contradiction. Let $\hat{t} \geq \hat{T} + 2j$ ($j \in \mathbb{N}^+$) be the first step ($\mathrm{ag}(\hat{t}) = A$) such that $R_A \cap \biguplus Never(\hat{t}-2) \neq R_A \cap \biguplus Never(\hat{t}-4) = R_A \cap \biguplus Never(\hat{T}-2)$.

As $Never(\hat{t}-2) \supseteq Never(\hat{t}-4)$ (by Definition 3.4), this means that we have:

$$Never(\hat{t}-2) \supset Never(\hat{t}-4)$$

and

$$R_A \cap \biguplus Never(\hat{t}-2) \supset R_A \cap \biguplus Never(\hat{t}-4).$$

This implies that

$$\biguplus \left(Never(\hat{t}-2) - Never(\hat{t}-4)\right) \cap R_A \neq \emptyset.$$

From Definition 3.4, we have that one of the two cases below holds:

(a) $Never(\hat{t}-2) - Never(\hat{t}-4) \subseteq NoN(\hat{t}-3)$ (if agent A did not propose in step $\hat{t}-2$ any deal in $\biguplus NoN(\hat{t}-3)$)

(b) $Never(\hat{t}-2) - Never(\hat{t}-4) \subseteq \{\{D\} \mid D \in NoN(\hat{t}-3)\}$ (otherwise).

Case (a) would imply that $\Diamond\!\!\!\!-\!\!\!\!\Diamond NoN(\hat{t}-3) \cap R_A \neq \emptyset$ and would lead to contradiction, as agent A is certainly non-obstructionist from step $\hat{T}$ onwards, hence would have proposed in $\Diamond\!\!\!\!-\!\!\!\!\Diamond NoN(\hat{t}-3)$ at step $\hat{t}-2$.

Case (b) would imply that $\{\{D\} \mid D \in NoN(\hat{t}-3)\} \cap R_A \neq \emptyset$ and would lead again to contradiction, as agent A (non-obstructionist from step $\hat{T}$) would have accepted one of the acceptable deals $D \in NoN(\hat{t}-3)$ at step $\hat{t}-2$. □

Proposition 4.6. Let $\pi = \langle \mathcal{V}, s, k, \mathcal{R} \rangle$ be a negotiation process ($k \geq 2$) where the NoN rule is enforced. If any agent $A \in \{0,1\}$ uses the strategy above, then, within a finite number of steps $T \geq \hat{T} \geq 2$ such that $\mathrm{ag}(T) = A$, either an agreement is found or condition $R_A - R_A^{\mathrm{err}} \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-1) \cup \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-2)$ is satisfied.

Proof:
We first prove that, within a finite number of steps $T \geq \hat{T} \geq 2$, either an agreement is found or the following condition is satisfied:

$$\lfloor R_A - \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \rfloor \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(t-1) \cup \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(t-2). \tag{3}$$

Let $T_V \geq 1$ be the negotiation step ($\mathrm{ag}(T_V) = A$) in which agent A sent out proposal $\mathcal{P}_{T_V}$ containing the last vertex of $\lfloor R_A - \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \rfloor$. In case agent $B = 1 - A$ accepts one of the deals in $\mathcal{P}_{T_V}$, an agreement is found at step $T_V + 1 \geq 2$ and the thesis follows.

Otherwise, from step $T_V + 2$ onwards, agent A will send out only empty proposals, unless she decides to accept an acceptable deal (in which case, the thesis again follows).

For all steps $t \geq T_V + 2$, we have:

$$\lfloor R_A - \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \rfloor \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond NoN(t) \cup \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(t-1) \cup \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(t-2)$$

as no new point in $\lfloor R_A - \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \rfloor$ is disclosed.

At each step $t > T_V$ such that $\mathrm{ag}(t) = B$, three cases may arise:

1. Agent B proposes an acceptable deal and the negotiation terminates successfully. The thesis follows.

2. Agent B sends a proposal $\mathcal{P}_t$ with no deal in $\Diamond\!\!\!\!-\!\!\!\!\Diamond NoN(t-1)$ and no acceptable deals: $NoN(t-1)$ is added to $Never(t)$. At step $t+1$, agent A sends out an empty proposal again, and $NoN(t+1) = \emptyset$, as $\mathrm{deals}_A(t+1) = \mathrm{deals}_A(t)$. Condition (3) is now satisfied.

3. Agent B sends a proposal $\mathcal{P}_t$ with at least one deal in $\Diamond\!\!\!\!-\!\!\!\!\Diamond NoN(t-1)$ and no acceptable deals. As agent A replies with $\mathcal{P}_t = \emptyset$, $NoN(t)$ will shrink by at least one element, and condition (3) is one step closer to be satisfied.

Condition (3) can be equivalently rewritten as:

$$(R_A - R_A^{\mathrm{err}}) - \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-1) \cup \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-2).$$

The formula can be rewritten again into:

$$R_A - R_A^{\mathrm{err}} \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-1) \cup \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-2)$$

as, being $T \geq \hat{T}$, Definition 3.4 guarantees that either $\Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-2)$ (when $T = \hat{T} + 2j$, $j \geq 0$) or $\Diamond\!\!\!\!-\!\!\!\!\Diamond Never(\hat{T}-2) \subseteq \Diamond\!\!\!\!-\!\!\!\!\Diamond Never(T-1)$ (when $T = \hat{T} + 2j + 1$, $j \geq 0$). The thesis follows. □