IAM-ECS Technical Report

Formal Semantics of ABN Framework

Nishan C. Karunatillake, Nicholas R. Jennings, Iyad Rahwan, Peter McBurney

IAM Research Group,
School of Electronics and Computer Science,
University of Southampton,
Southampton, SO17 1BJ, UK.

E-mail: nnc@ecs.soton.ac.uk

Technical Report Number: ECSTR-IAMXX-YYY
ISBN: nnnnnnnnnnnnnn

Abstract

This technical document extend upon and give formal meaning to the syntactic definitions presented in [7] and forwards the formal semantics of our argumentation-based negotiation (ABN) dialogue game protocol, which allows agents to argue, negotiate and resolve conflicts in a multi-agent context.

Keywords: *Example Template*

1 The Formal Semantics

This technical document extend upon and give formal meaning to the syntactic definitions presented in [7] and forwards the formal semantics of our argumentation-based negotiation (ABN) dialogue game protocol, which allows agents to argue, negotiate and resolve conflicts in a multi-agent context.

Similar to the theory of program language semantics [6], there are a number of alternative approaches to developing semantics for agent communication languages and protocols, as first presented by van Eijk in [4]. The most common approach is the articulation of an *axiomatic semantics*, in which the pre-conditions and post-conditions of each legal utterance in the language are defined. Examples of such axiomatic semantics include the Semantic Language (SL) of the IEEE FIPA Agent Communication Language (FIPA-ACL) [5] which presents, semi-formally, the pre-conditions and post-conditions of each locution in terms of the mental states of the agents engaged in dialogue. In addition to defining these pre- and post-conditions, within the literature on agent dialogue games, axiomatic semantics also define the effects that each utterance has on the state of the dialogue [1, 2, 8]. In particular, the state of the dialogue may be assessed by the contents of any *commitment stores* as well as constraints on what utterances may validly precede or follow any given legal utterance. Accordingly, in line with this existing literature, we have defined an axiomatic semantics for the language and protocol given in [7] in terms of the effects of each legal utterance on the contents of commitment stores (CS) and information stores (IS) of agents and in terms of protocol constraints on the utterances which may precede or follow the given utterance. These semantics are presented in Section 2.

A second approach to defining a semantics for agent communications languages is an *operational semantics* [4, 9]. Under this approach, the participating agents and their shared dialogue are viewed conceptually as parts of a large virtual computer, whose overall state may be altered by the utterance of valid locutions or by internal decision processes of the participants; it is as if these locutions or decisions were commands in some computer programme language acting on the virtual machine. [1] The utterances and agent decision-mechanisms are thus seen as state transition operators, and the operational semantics defines these transitions precisely [4]. This approach to the semantics of agent communications languages makes explicit any link between the internal decision mechanisms of the participating agents and their public utterances to one another. The semantics therefore enables the relationships between the mental states of the participants and the public state of the dialogue to be seen explicitly, and shows how these relationships change as a result of utterances and internal agent decisions. Within the agent communications literature, operational semantics have now been defined for several agent communications languages and protocols, including information-passing interactions [3] and certain negotiation dialogues [8]. Accordingly, we have defined a formal operational semantics for the language, protocol, and the internal decision apparatus given in [7], and this semantics is presented in Section 3, along with an example of its application.

Inspecting the axiomatic semantics provided in section 2 and the operational semantics provided in Section 3, one will see that the two semantics are closely related. As with different types of semantics for programming languages, these different semantics may serve different purposes. By presenting the dialogue pre-conditions and post-conditions for each utterance, the axiomatic semantics defines the dialogue locutions and their rules of combination and ordering. Hence, this semantics identifies the circumstances under which particular utterances may legally be made (according to the protocol in question), and presents the effects of their being made on the dialogue as a whole. In contrast, the operational semantics links dialogue utterances with the decision-making mechanisms of the participating agents, by treating each utterance as a transition between states of an imaginary machine. Because this imaginary machine includes all the participating agents (something which, typically, no physical or conceptual system includes in an open multi-agent system), the oper-

[1] It is important to stress that this virtual machine may not exist in reality, or may be distributed across multiple physical machines or multiple conceptual systems.

ational semantics is able to articulate the linkages between public dialogue actions and the private decisions of the participants. From the perspective of a designer of a system as a whole, an axiomatic semantics is often the most useful, because the main system-level decision required is the run-time classification of proposed agent utterances as being legal or not under the protocol. From the perspective of the designer of an individual agent, however, the operational semantics is often the most useful, as agents need to know what decisions need to be made by themselves at what points in a dialogue; this information is provided by the operational semantics. In an open agent system, the designer of the overall system and the designers of individual agents may be different groups of people, working at different times and locations, so both models are provided.

2 Axiomatic Semantics

Here we present an axiomatic semantics for the multi-agent communications protocol whose syntax is given in Section 3.3. This semantics defines the dialogue pre-conditions and post-conditions of each legal locution of the language. These conditions are defined in terms of the effects of legal utterances on the Commitment Stores (CS) and the Information Stores (IS) of the participating agents and in terms of the legal utterances which may precede or follow each utterance. In the following paragraphs, we use the symbols "L1, L2" etc to refer to the eleven legal locutions permitted in the language.

1. **OPEN-DIALOGUE** (*Locutions L1 and L2*): This indicates the entry point of that agent to the dialogue. It would result in an entry in either agents' commitment stores corresponding to the dialogical commitment [10] of having made the move (i.e., commitment to the fact that the agent has uttered OPEN-DIALOGUE). An agent receiving an OPEN-DIALOGUE will retort back (if it hasn't already initiated it) by uttering the same. This would put both these agents in the opening stage and their negotiation over actions can commence. For simplicity, we assume that the first agent opening the dialogue is the one attempting to make its counterpart perform (or abstain from performing) an action. Thus, we denote that agent as a_p; the proponent of the dialogue and its counterpart as a_r the respondent. Using this notation, the following defines its axiomatisation giving the pre-conditions, valid responses, and their effects on the agents' commitment and the information stores respectively.

   - Usage:
     - L1 : OPEN-DIALOGUE(a_p, a_r) or
     - L2 : OPEN-DIALOGUE(a_r, a_p)
   - Meaning: By uttering the locution "OPEN-DIALOGUE(a_p, a_r)" the proposing agent a_p expresses its desire to initiate a negotiation dialogue. On the other hand, the locution "OPEN-DIALOGUE(a_r, a_p)" is used by the responding agent a_r to express its willingness to join that dialogue.

- Pre-conditions:
  - For OPEN-DIALOGUE(a_p, a_r): none
  - For OPEN-DIALOGUE(a_r, a_p): OPEN-DIALOGUE$(a_p, a_r) \in CS_{i-1}(r)$
- Valid Responses:
  - For OPEN-DIALOGUE(a_p, a_r): OPEN-DIALOGUE(a_r, a_p)
  - For OPEN-DIALOGUE(a_r, a_p): PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))$
- IS (information store) updates: none
- CS (commitment store) updates:
  - For OPEN-DIALOGUE(a_p, a_r):
    - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup$ OPEN-DIALOGUE(a_p, a_r)
    - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup$ OPEN-DIALOGUE(a_p, a_r)
  - For OPEN-DIALOGUE(a_r, a_p):
    - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup$ OPEN-DIALOGUE(a_r, a_p)
    - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup$ OPEN-DIALOGUE(a_r, a_p)

2. **PROPOSE** (*Locution L3*): Each proposal is composed of two basic elements; the *request* that the proponent wants the respondent to perform and the *reward* that the proponent is willing to perform in return. Therefore, in general, a proposal will have the form PROPOSE$(a_p, a_r,\ do(a_r, \theta_r), do(a_p, \theta_p))$. Here, the request from the proposing agent a_p to the respondent a_r is θ_r and the reward in return is θ_p. Here, both θ_p and θ_r could be a single atomic action (e.g., I will perform (or will not perform) a certain action in return or I will make a payment of a certain amount) or a composite action (e.g., I will perform action $(\theta_1$ *and* $\theta_2)$ or $(\theta_1$ *or* $\theta_2))$. Thus, this generic form of proposal allows the agents not only to make simple offers of payment over actions, but also to make simple or composite rewards and/or threats over actions. In this manner, this allows the agents to negotiate and also to use social influences as parameters within their negotiations to resolve conflicts (see [7]). Furthermore, both the elements request and reward can also be null. This allows the agents to express proposals that are mere requests without an explicit reward (such as demands, pleads, and orders) and solitary rewards (such as offers, gifts, and suggestions) that they deem to be viable during their negotiation. Once received, as an effect of the proposal, a_r will gain the information that a_p requires θ_r and that a_p has the ability to perform θ_p.

   - Usage:
     - L3 : PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))$
   - Meaning: By uttering the locution "PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))$" agent a_p forwards a proposal to agent a_r requesting a_r to perform θ_r and in return for a_p performing θ_p. Thus, the request of this proposal is $do(a_r, \theta_r)$ and the reward is $do(a_p, \theta_p)$. We assume that these rewards have a particular ordering based on the cost incurred by a_p when performing them. Thus, we denote the first reward as $do(a_p, \theta_{p_0})$, the next as $do(a_p, \theta_{p_1})$, the i^{th} as $do(a_p, \theta_{p_{(i-1)}})$, and the last as $do(a_p, \theta_{p_t})$.

- Pre-conditions:
  - For PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_{p0}))$: OPEN-DIALOGUE$(a_r, a_p)$
  - For PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_{p_i}))$:
    - REJECT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_{p(i-1)}))$ or
    - ASSERT(a_r, a_p, l)
- Valid Responses:
  - For PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_{p_i}))$:
    - ACCEPT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_{p_i}))$ or
    - REJECT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_{p_i}))$
- IS (information store) updates:
  - $IS_i(a_r) \leftarrow IS_{i-1}(a_r) \cup \mathit{need}(a_p, \theta_r) \cup \mathit{capable}(a_p, \theta_p)$
- CS (commitment store) updates:
  - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup$ PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))$
  - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup$ PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))$

3. **ACCEPT** (*Locution L4*): Upon receiving a proposal, the respondent agent a_r may choose to either accept or reject it. Now, in order to make this decision, it will need to evaluate the proposal (see Section 3.4.2 in [7] for a detailed discussion on this evaluation decision algorithm). During this evaluation, if the proposal satisfies the respondent acceptance conditions, it will retort back with an acceptance. Once accepted, both agents will incur commitments to perform their respective actions.

   - Usage:
     - L4 : ACCEPT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$
   - Meaning: By uttering the locution "ACCEPT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$", agent a_r indicates to agent a_p that a_r accepts the proposal made by a_p in a prior utterance of the locution "PROPOSE$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$". Thereby, a_r expresses its willing to perform the requested action θ_r in exchange for a_p performing the offered action θ_p.
   - Pre-conditions:
     - For ACCEPT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$
       - PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p)) \in CS_{i-1}(a_r)$
   - Valid Responses:
     - For ACCEPT$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$: CLOSE-DIALOGUE$(a_p, a_r)$
   - IS (information store) updates:
     - $IS_i(a_p) \leftarrow IS_{i-1}(a_p) \cup \mathit{capable}(a_r, \theta_r)$

- CS (commitment store) updates:
  - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup$ $[\textsc{Accept}(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))] \cup [do(a_p, \theta_p)] \cup [do(a_r, \theta_r)]$
  - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup$ $[\textsc{Accept}(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p))] \cup [do(a_p, \theta_p)] \cup [do(a_r, \theta_r)]$

4. **REJECT** (*Locution L5*): If the received proposal failed to satisfy the respondent acceptance conditions, it will retort back with a rejection. In effect both agents would record a dialogical commitment to the fact that the respondent rejected the proposal.

   - Usage:
     - L5 : $\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$
   - Meaning: By uttering the locution "$\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$", agent a_r indicates to agent a_p that a_r rejects the proposal made by a_p in a prior utterance of the locution "$\textsc{Propose}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$".
   - Pre-conditions:
     - For $\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$
       - $\textsc{Propose}(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_p)) \in CS_{i-1}(a_r)$
   - Valid Responses:
     - For $\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_{p_i}))$:
       - $\textsc{Propose}(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_{p_{(i+1)}}))$
       - $\textsc{Challenge}(\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_{p_i})))$
       - $\textsc{Close-Dialogue}(a_p, a_r)$
   - IS (information store) updates: none
   - CS (commitment store) updates:
     - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup \textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$
     - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup \textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$

5. **CHALLENGE** (*Locutions L6 and L7*): Upon rejection of a proposal by its counterpart (a_r), a_p may choose to either forward a modified proposal (i.e., if the reason is apparent such that there can be only one possibility) or challenge a_r's decision in order to identify the underlying reasons for rejection. Apart from this, an agent can also challenge a certain assertion by its counterpart if either that assertion or its contradiction is not within its knowledge. Using the notation $\Delta(a_i)$ to denote the agent a_i's knowledge-base we can axiomatise the CHALLENGE locution as follows.

   - Usage:
     - L6: $\textsc{Challenge}(a_p, a_r, \textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p)))$

- L7: $\textsc{Challenge}(a_{x_2}, a_{x_1}, \textsc{Assert}(a_{x_1}, a_{x_2}, l))$

- Meaning:
  - L6: By uttering the "$\textsc{Challenge}(a_p, a_r, \textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p)))$" locution, agent a_p challenges agent a_r's decision to reject a_p's prior proposal $\textsc{Propose}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$ and demands a_r's reason for rejecting it.
  - L7: By uttering the locution "$\textsc{Challenge}(a_{x_2}, a_{x_1}, \textsc{Assert}(a_{x_1}, a_{x_2}, l))$", agent a_{x_2} (either the proponent or the respondent) challenges the other agent a_{x_1} (in case of a_{x_2} being the proponent, a_{x_1} would be the respondent and vice-verse) demanding a_{x_1} to give its justification for uttering a prior assertion $\textsc{Assert}(a_{x_1}, a_{x_2}, l))$.

- Pre-conditions:
  - For $\textsc{Challenge}(a_p, a_r, \textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p)))$
    - $\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p)) \in CS_{i-1}(a_p)$ and
    - *reason*$(\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))) \notin \Delta_{i-1}(a_p)$
  - For $\textsc{Challenge}(a_{x_2}, a_{x_1}, \textsc{Assert}(a_{x_1}, a_{x_2}, l))$
    - $\textsc{Assert}(a_{x_1}, a_{x_2}, l) \in CS_{i-1}(a_{x_2})$ and
    - *reason*$(\textsc{Assert}(a_{x_1}, a_{x_2}, l)) \notin \Delta_{i-1}(a_{x_2})$

- Valid Responses:
  - For $\textsc{Challenge}(a_{x_2}, a_{x_1}, l)$
    - $\textsc{Assert}(a_{x_1}, a_{x_2}, H)$ where $H \vdash l$

- IS (information store) updates: none

- CS (commitment store) updates:
  - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup \textsc{Challenge}(l)$
  - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup \textsc{Challenge}(l)$

6. **ASSERT** (*Locutions L8 and L9*): An agent can assert a certain fact in two possible situations. First, if the agent is challenged for some justification on its decision it can assert that justification. Second, if its counterpart has made an assertion (l), but the agent has justification to believe its contradiction $(\neg l)$, then the agent can assert its negation to dispute its counterpart's assertion. This will allow agents to undercut and rebut each others' social reasoning, and, thereby, resolve conflicts (see [7]). Assert can result in one of five responses; namely (i) counterpart generating an alternative proposal (taking into account the reason given), (ii) re-forwarding the same proposal (if, while arguing, the agents realise that the original proposal was rejected due to incorrect reasons), (iii) challenging the justification for assertion, (iv) disputing the assertion by asserting its negation, or (v) closing the dialogue by agreeing to disagree.

   - Usage:
     - L8: $\textsc{Assert}(a_{x_1}, a_{x_2}, l)$

- L9: ASSERT$(a_{x_1}, a_{x_2}, \neg l)$

- Meaning:
  - L8: By uttering the locution "ASSERT(a_{x_1}, a_{x_2}, l)", agent a_{x_1} (either the proponent or the respondent) asserts a particular premise l to its counterpart agent a_{x_2}.
  - L9: By uttering the locution "ASSERT$(a_{x_1}, a_{x_2}, \neg l)$", agent a_{x_1} (either the proponent or the respondent) asserts the negation of a particular premise $(\neg l)$ to its counterpart agent a_{x_2}. Here, asserting the negation of a particular premise $(\neg l)$ would account to disputing the original premise l.

- Pre-conditions:
  - For ASSERT(a_{x_1}, a_{x_2}, l)
    - CHALLENGE$(a_{x_2}, a_{x_1}, l') \in CS_{i-1}(a_{x_1})$ where $l \vdash l'$
  - For ASSERT$(a_{x_1}, a_{x_2}, \neg l)$
    - ASSERT$(a_{x_2}, a_{x_1}, l) \in CS_{i-1}(a_{x_1})$

- Valid Responses:
  - For ASSERT(a_{x_1}, a_{x_2}, l)
    - PROPOSE$(a_p, a_r, do(a_r, \theta_r), do(a_p, \theta_{p+1}))$
    - PROPOSE$(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p))$
    - CHALLENGE$(a_{x_2}, a_{x_1},$ ASSERT$(a_{x_1}, a_{x_2}, l))$
    - ASSERT$(a_{x_2}, a_{x_1}, \neg l)$
    - CLOSE-DIALOGUE(a_p, a_r)

- IS (information store) updates: none

- CS (commitment store) updates:
  - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup$ ASSERT(l)
  - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup$ ASSERT(l)

7. **CLOSE-DIALOGUE** (*Locutions L10 and L11*): When either the counterpart has accepted a certain proposal or the proposing agent has no other feasible and worthwhile proposals to forward, an agent will utter CLOSE-DIALOGUE (echoed in return by its counterpart) to bring the dialogue to an end.

- Usage:
  - L10: CLOSE-DIALOGUE(a_p, a_r) or
  - L11: CLOSE-DIALOGUE(a_r, a_p)

- Meaning: By uttering the locution "CLOSE-DIALOGUE(a_p, a_r)" the proposing agent a_p expresses its desire to terminate an ongoing negotiation dialogue. On the other hand, the locution "CLOSE-DIALOGUE(a_r, a_p)" is used by the responding agent a_r to express its willingness to terminate that dialogue.

- Pre-conditions:

- For $\textsc{Close-Dialogue}(a_p, a_r)$
  - $\textsc{Reject}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p)) \in CS_{i-1}(a_p)$,
  - $\textsc{Accept}(a_r, a_p, do(a_r, \theta_r), do(a_p, \theta_p)) \in CS_{i-1}(a_p)$, or
  - $\textsc{Assert}(a_r, a_p, l) \in CS_{i-1}(a_p)$
- For $\textsc{Close-Dialogue}(a_r, a_p)$
  - $\textsc{Close-Dialogue}(a_p, a_r)$

- Valid Responses:
  - For $\textsc{Close-Dialogue}(a_p, a_r)$: $\textsc{Close-Dialogue}(a_r, a_p)$
  - For $\textsc{Close-Dialogue}(a_r, a_p)$: none

- IS (information store) updates: none

- CS (commitment store) updates:
  - For $\textsc{Close-Dialogue}(a_p, a_r)$:
    - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup \textsc{Close-Dialogue}(a_p, a_r)$
    - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup \textsc{Close-Dialogue}(a_p, a_r)$
  - For $\textsc{Close-Dialogue}(a_r, a_p)$:
    - $CS_i(a_p) \leftarrow CS_{i-1}(a_p) \cup \textsc{Close-Dialogue}(a_r, a_p)$
    - $CS_i(a_r) \leftarrow CS_{i-1}(a_r) \cup \textsc{Close-Dialogue}(a_r, a_p)$

3 Operational Semantics

Here we present an operational semantics for the multi-agent communications protocol whose syntax is given in [7]. As explained in [7], this semantics considers the effects of legal agent utterances as if they were program language commands acting on a virtual computer. In defining this semantics we bring together the protocol, which defines the rules of the interaction, with the internal decision-making mechanisms of the agents participating in the interaction. In the following paragraphs, we label the thirty-one transition rules of the operational semantics with the symbols “TR1”, “TR2”, etc.

We define our semantics using the *labelled terminal transition system* (LTTS) [9]. In more detail, the LTTS defines the operation of a system as a series of tuples $\langle \Gamma, A, \rightarrow, T \rangle$, where Γ represents a set of configurations, A a set of labels, $\rightarrow\ : \Gamma \times A \times \Gamma$ defines a transition relation, and T a set of terminal (or final) configurations; i.e., $\forall \gamma \in T$, $\nexists \gamma' \in \Gamma, \alpha \in A$ such that $(\gamma, \alpha, \gamma') \rightarrow$. Conventionally, $(\gamma_1, \alpha, \gamma_2') \rightarrow$ is sometimes written $\gamma_1 \xrightarrow{\alpha} \gamma_2$. This method of specifying operational semantics can be used at different levels of detail, and what counts as one transition for one purpose may be represented through many transitions when viewed in more detail [9].

In our specification, a configuration $\gamma \in \Gamma$ is itself a tuple $[a_i, P, o]$, where a_i is an agent, P is a decision mechanism being executed by agent a_i, and o is an output of the decision mechanism. Labels denote locutions (general message types) that cause the transition from one configuration to another (possibly in a different agent). Thus, the intuitive meaning of a transition statement $[a_i, P_1, o_1] \xrightarrow{L} [a_j, P_2, o_2]$ is that if we were in a configuration where agent a_i executes mechanism P_1 leading to output o_1, then after sending a message through locution L, the system moves to a configuration where agent a_j executes mechanism P_2 leading to output o_2. In certain instances, we also use the above notation to capture internal transitions where a certain internal decision mechanism leads to another state within an agent. Such transitions do not involve communications between different agents, but only changes in the internal state of a single agent. For this reason, these internal transitions are represented by arrows without labels. It also important to note that in our transition statements, we usually refer to output schema as opposed to specific output instances. Moreover, in certain instances we use the '.' notation to denote any type of output for a given mechanism. Finally, a special state T is used to denote the terminal state of the system. Given this, the following specifies the operational semantics of our ABN system and Figure 1 captures its operational flow.

TR1: If the agent does not require the services of another to accomplish a certain action θ, it will not require any argumentation, thus, will move to the terminal state T. To evaluate whether or not the agent requires the services of another, it would use its decision mechanism **P1 Recognise Need**:

$$[a_p, \text{P1}, \text{noNeedService}(\theta)] \rightarrow [a_p, \text{P1}, T]$$

TR2: If the agent recognises that it requires the services of another to accomplish a certain action, it will initiate a dialogue with that agent through the **L1: OPEN-DIALOGUE** locution. Similar to above, the agent uses the **P1: Recognise Need** decision mechanism to evaluate whether or not it requires the services of another. When its counterpart receives this locution it will initiate its decision mechanism **R1: Consider Participation**.

$$[a_p, \text{P1}, \text{needService}(\theta)] \xrightarrow{\text{L1}} [a_r, \text{R1}, .]$$

TR3: When an agent receives an invitation to enter into a dialogue via the **L1: OPEN-DIALOGUE** locution, it will indicate its readiness via its own **L2: OPEN-DIALOGUE** locution. Once the proponent receives this reply it will, in turn, initiate the decision mechanism **P2: Generate Proposals** attempting to formulate a viable and a feasible set of proposals.

$$[a_r, \text{R1}, \text{enterDialogue}(\theta)] \xrightarrow{\text{L2}} [a_p, \text{P2}, .]$$

TR4: Once an agent has generated a feasible and a viable set of proposals, it will initiate its own decision mechanism **P3: Rank Proposals** in order to obtain an ordered ranking on this set.

$$[a_p, \text{P2}, Q(\theta)] \rightarrow [a_p, \text{P3}, .]$$

TR5: Once the proposals are ranked, the agent will initiate its own **P4: Select Proposal** mechanism to a select a proposal to forward to its counterpart.

$$[a_p, \text{P3}, S(\theta)] \rightarrow [a_p, \text{P4}, .]$$

TR6: If there is no other proposal left to select (i.e., all possible proposals were forwarded and justifiably rejected) and the **P4: Select Proposal** mechanism returns null ($\emptyset$), then the agent will initiate its own **P11: Terminate Interaction** mechanism to end the dialogue.

$$[a_p, \text{P4}, \emptyset] \rightarrow [a_p, \text{P11}, .]$$

TR7: If the **P4: Select Proposal** decision mechanism returns a proposal (i.e., P4 will only return proposals that have not been previously forwarded and justifiably rejected within the encounter), then the agent will forward it to its counterpart via a **L3: PROPOSE** locution. Once received, the respondent will initiate the decision mechanism **R2: Evaluate Proposal** to consider whether to accept or reject this proposal.

$$[a_p, \text{P4}, S_i(\theta)] \xrightarrow{\text{L3}} [a_r, \text{R2}, .]$$

TR8: If the respondent decides to accept the current proposal within its **R2: Evaluate Proposal** mechanism, then it will indicate its decision via the **L4: ACCEPT** locution. Once a proposal is accepted, the proponent will initiate the decision mechanism **P11: Terminate Interaction** to bring the dialogue to an end.

$$[a_r, \text{R2}, \text{accept}(S_i(\theta))] \xrightarrow{\text{L4}} [a_p, \text{P11}, .]$$

TR9: If the respondent decides to reject the current proposal within its **R2: Evaluate Proposal** mechanism, then it will indicate its decision via the **L5: REJECT** locution. Once received, this REJECT will prompt the proponent to initiate the mechanism **P5: Find Justification, Continue Negotiation, or Terminate**, to decide its next course of action.

$$[a_r, \text{R2}, \text{reject}(S_i(\theta))] \xrightarrow{\text{L5}} [a_p, \text{P5}, .]$$

TR10: While considering its next course of action (via **P5**), if the proponent decides to terminate the dialogue, it will initiate its own decision mechanism **P11: Terminate Interaction** to bring the dialogue to an end.

$$[a_p, \text{P5}, \text{terminate}(S_i(\theta))] \rightarrow [a_p, \text{P11}, .]$$

TR11: If the proponent decides to continue negotiating with its counterpart (via **P5**), it will attempt to select and forward an alternative proposal to that agent. In order to select this alternative, the proponent will initiate its own decision mechanism **P4: Select Proposal**.

$$[a_p, \text{P5}, \text{continue}(S_i(\theta))] \rightarrow [a_p, \text{P4}, .]$$

TR12: The proponent may decide (via **P5**) to challenge its counterpart to establish the reason for rejecting its current proposal. In such cases, the proponent will construct an **L6: CHALLENGE** locution in order to challenge its counterpart for its justification to reject the proposal. Once a respondent receives such a challenge, it will, in turn, initiate its own **R3: Extract Justification** mechanism that will search within its knowledge-base (or formulate) the reason for the corresponding rejection.

$$[a_p, \text{P5}, \text{challengeReject}(S_i(\theta))] \xrightarrow{\text{L6}} [a_r, \text{R3}, .]$$

TR13: When the respondent extracts its justification for rejecting the proposal (using its decision mechanism **R3**), it will assert this via an **L8: ASSERT** locution to its counterpart. Once received, this will initiate the proponent's decision mechanism **P6: Evaluate Justifications**, which will attempt to compare its own justification with its counterpart's and analyse the cause of the conflict.

$$[a_r, \text{R3}, H_r] \xrightarrow{\text{L8}} [a_p, \text{P6}, .]$$

TR14: While evaluating justifications, if the agent still requires more information to evaluate the validity of one of its counterpart's premises $(l_r \in H_r)$, it will attempt to acquire this knowledge via challenging this assertion via the **L7: CHALLENGE** locution. This will, in turn, restart the opponent's **R3: Extract Justification** mechanism.

$$[a_p, \text{P6}, \text{needMoreJustification}(l_r)] \xrightarrow{\text{L7}} [a_r, \text{R3}, .]$$

TR15: While evaluating justifications, if the agent still requires more information to evaluate the validity of one of its own premises $(l_p \in H_p)$, it will restart its own **P7: Extract Justification** mechanism to establish the reasoning behind this premise.

$$[a_p, \text{P6}, \text{needMoreJustification}(l_p)] \rightarrow [a_p, \text{P7}, .]$$

TR16: While evaluating justifications, if the agent finds a premise within its own justification l_p to be invalid, then it will initiate its **P8: Update Knowledge** mechanism to update its own knowledge-base correcting the invalid premise.

$$[a_p, \text{P6}, \text{invalid}(l_p)] \rightarrow [a_p, \text{P8}, .]$$

TR17: While evaluating justifications, if the agent finds all premises within its counterpart's justification H_r to be valid, then it will initiate its **P8: Update Knowledge** mechanism to update its own knowledge by inserting this valid justification into its knowledge-base.

$$[a_p, \text{P6}, \text{valid}(H_r)] \rightarrow [a_p, \text{P8}, .]$$

TR18: While evaluating justifications, if the agent finds a premise within its counterpart's justification l_r to be invalid, then it will dispute this premise through an **L9: ASSERT** locution. Once received, the respondent will initiate its **R4: Consider Premise** mechanism to consider updating the invalid premise within its knowledge-base.

$$[a_p, \text{P6}, \text{invalid}(l_r)] \xrightarrow{\text{L9}} [a_r, \text{R4}, .]$$

TR19: While evaluating justifications, if the agent finds all premises within its own justification H_p to be valid, then it will assert its justification through an **L8: ASSERT** locution. Once received, the respondent will initiate its **R4: Consider Premise** mechanism to consider inserting this justification into its knowledge-base.

$$[a_p, \text{P6}, \text{valid}(H_p)] \xrightarrow{\text{L8}} [a_r, \text{R4}, .]$$

TR20: If the **P7: Extract Justification** decision mechanism is triggered to establish the reason behind a certain premise l_p, then it will extract this justification H'_p where $H'_p \vdash l_p$ from its knowledge and pass it back into its **P6: Evaluate Justifications** mechanism.

$$[a_p, \text{P7}, H'_p] \rightarrow [a_p, \text{P6}, .]$$

TR21: While considering a particular premise, if the respondent's **R4: Consider Premise** decision mechanism requires more justification to accept a particular premise, it will challenge the proponent for this further justification. Once received, this **L7: CHALLENGE** will trigger the proponent's **P7: Extract Justification** mechanism to extract further justifications.

$$[a_r, \text{R4}, \text{needMoreJustification}(l)] \xrightarrow{\text{L7}} [a_p, \text{P7}, .]$$

TR22: Once the proponent's **P7: Extract Justification** mechanism has extracted further justification in response to a particular challenge by the respondent, it will forward this justification H' via a **L8: ASSERT** locution. This will initiate the respondent's **R4: Consider Premise** mechanism to reconsider the relevant premise with this additional justification.

$$[a_p, \text{P7}, H'] \xrightarrow{\text{L8}} [a_r, \text{R4}, .]$$

TR23: While considering a particular premise l, if the respondent's **R4: Consider Premise** decision mechanism decides to accept that premise, it will incorporate (either update or insert) that into its knowledge-base. Once the knowledge its updated, it will, in turn, trigger the respondent's own **R5: Consider Counter Argument** mechanism to search for a possible counter argument within its updated knowledge-base.

$$[a_r, \text{R4}, \text{knowledgeUpdate}(l)] \rightarrow [a_r, \text{R5}, .]$$

TR24: Once the proponent updates its knowledge with a particular premise l via the **P8: Update Knowledge** mechanism, it will trigger the proponent's own **P9: Consider Counter Argument** mechanism to search for a possible counter argument within its updated knowledge-base.

$$[a_p, \text{P8}, \text{knowledgeUpdate}(l)] \rightarrow [a_p, \text{P9}, .]$$

TR25: Within the **P9: Consider Counter Argument** mechanism, if the proponent finds a valid counter argument it will restart its own **P6: Evaluate Justification** mechanism with this additional argument.

$$[a_p, \text{P9}, \text{hasCounterArg}(H_p)] \rightarrow [a_p, \text{P6}, .]$$

TR26: Within the **R5: Consider Counter Argument** mechanism, if the respondent finds a valid counter argument, it will forward this argument via a **L8: ASSERT** locution to the proponent. This will, restart the proponent's **P6: Evaluate Justification** mechanism with this additional argument.

$$[a_r, \text{R5}, \text{hasCounterArg}(H_r)] \overset{\text{L8}}{\rightarrow} [a_p, \text{P6}, .]$$

TR27: If the proponent, within its **P9: Consider Counter Argument** mechanism does not find a valid counter argument, it will initiate its own **P10: Terminate Challenge** mechanism to terminate this challenge.

$$[a_p, \text{P9}, \text{noCounterArg}()] \rightarrow [a_p, \text{P10}, .]$$

TR28: If the respondent, within its **R5: Consider Counter Argument** mechanism does not find a valid counter argument, it will indicate its agreement to the challenge to the proponent via a **L8: ASSERT** locution. Once, received, this will initiate the proponent's **P10: Terminate Challenge** mechanism.

$$[a_r, \text{R5}, \text{noCounterArg}()] \overset{\text{L8}}{\rightarrow} [a_p, \text{P10}, .]$$

TR29: Once initiated, the proponent's **P10: Terminate Challenge** mechanism will take steps to terminate the current challenge. Then it will initiate its own decision mechanism **P5: Find Justification, Continue Negotiation, or Terminate** thus, transferring control again back to the main negotiation strategy selection algorithm.

$$[a_p, \text{P10}, \text{evaluationComplete}()] \rightarrow [a_p, \text{P5}, .]$$

TR30: If the proponent decides to terminate the dialogue it will indicate this via a **L10: CLOSE-DIALOGUE** locution. Once the respondent receives this, it will, in turn, initiate its own **R6: Terminate Interaction** decision mechanism.

$$[a_p, \text{P11}, \text{exitDialogue}(\theta)] \overset{\text{L10}}{\rightarrow} [a_r, \text{R6}, .]$$

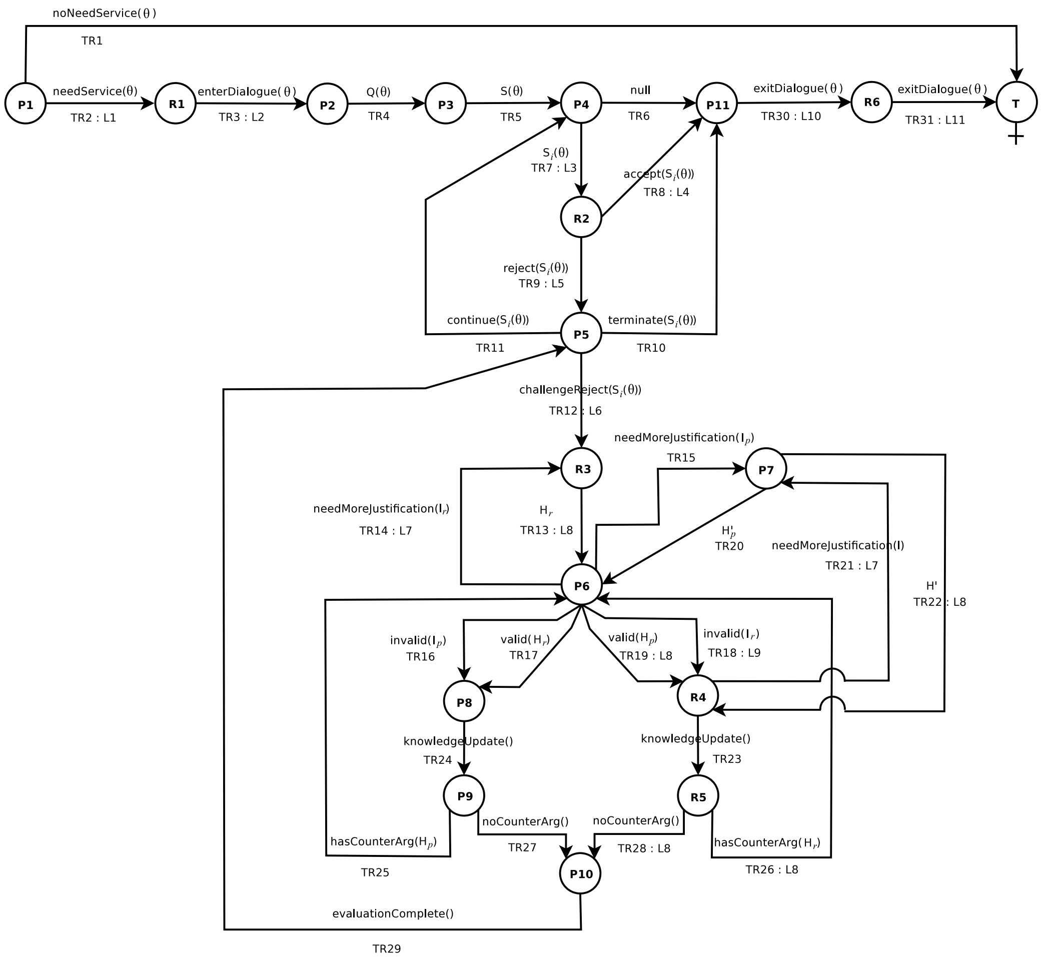

FIGURE 1: Operational flow.[2]

TR31: When the respondent's **R6: Terminate Interaction** is initiated, it will convey its willingness to close the dialogue via a **L11: CLOSE-DIALOGUE** locution. Thus, at this time both the proponent and the respondent will terminate their interaction. Once completed, the argumentation system would move to the terminal state T.

$$[a_r, \text{R6}, \text{exitDialogue}(\theta)] \xrightarrow{\text{L11}} [a_p, \text{P11}, T]$$

[2]Note that to simplify presentation, we used a single decision mechanism P7 to refer to the process of extracting justification used both (i) internally by the proponent agent via TR15 followed by TR20; and (ii) in response to a request for justification by another respondent agent via TR21 followed by TR22. The speech act transitions TR21 and TR22 are labelled with the relevant locutions (L7 and L8 respectively) to avoid any ambiguity.

References

[1] L. Amgoud, S. Parsons, and N. Maudet. Argument, dialogue and negotiation. In W. Horn, editor, *Proc. of the 14th European Conference on Artificial Intelligence (ECAI'00)*, pages 338–342, Berlin, 2000.

[2] K. Atkinson, T. Bench-Capon, and P. McBurney. A dialogue game protocol for multi-agent argument over proposals for action. *Journal of Autonomous Agents and Multi-Agent Systems*, 11(2):153–171, 2005.

[3] F. S. de Boer, R. Eijk, W. v. Hoek, and J.-J. C. Meyer. A fully abstract model for the exchange of information in multi-agent systems. *Theoretical Computer Science*, 290(3):1753–1773, 2003.

[4] R. Eijk. *Programming Languages for Agent Communications*. PhD thesis, Department of Computer Science, Utrecht University, Utrecht, The Netherlands, 2000.

[5] FIPA. Communicative Act Library Specification. Standard SC00037J, Foundation for Intelligent Physical Agents, 3 December 2002.

[6] C. A. Gunter. *Semantics of Programming Languages: Structures and Techniques*. MIT Press, Cambridge, MA, USA, 1992.

[7] N. C. Karunatillake, N. R. Jennings, I. Rahwan, and P. McBurney. Dialogue games that agents play within a society. *Artificial Intelligence*, submitted.

[8] P. McBurney, R. M. van Eijk, S. Parsons, and L. Amgoud. A dialogue-game protocol for agent purchase negotiations. *Journal of Autonomous Agents and Multi-Agent Systems*, 7(3):235–273, 2003.

[9] G. D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, University of Aarhus, 1981.

[10] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. SUNY Press, Albany NY, USA, 1995.