

Flexible Agent Protocols via Temporal and Resource-based Reasoning

Duc Q. Pham
RMIT University
GPO Box 2476V
Melbourne, 3001, Australia
qupham@cs.rmit.edu.au

James Harland
RMIT University
GPO Box 2476V
Melbourne, 3001, Australia
jah@cs.rmit.edu.au

ABSTRACT

In multi-agent systems, it has long been recognized that the traditional approaches to protocols which often used a fixed sequence of interactive actions as the basis are too brittle, and more flexible methods are needed. A crucial aspect of the development of such methods is the notion of commitments, which provides a mechanism for regulating interactions between agents. In this paper we investigate a rule-based approach to the specification of flexible protocols by means of a reasoning system which incorporates both temporal reasoning and resource-based reasoning and uses the notion of commitments.

Categories and Subject Descriptors

D.3.1 [Programming Languages]: Formal Definitions and Theory; I.2.5 [Artificial Intelligence]: Programming Languages and Software; I.2.4 [Artificial Intelligence]: Knowledge Representation Formalisms and Methods—*Representations (procedural and rule-based)*

General Terms

Languages, Theory

Keywords

Agent programming languages, Formal models of agency, Logics for agent systems

1. INTRODUCTION AND BACKGROUND

In traditional approaches to protocol specification, like those using Finite State Machines or Petri Nets in distributed computing or computer networks, protocols are often predetermined legal sequences of interactive behaviors. In frequently changing environments such as the Internet, such fixed sequences can quickly become outdated and are prone to failure. Hence, protocols in such environments need to

be flexible, and in particular, interaction protocols should ensure that agents have autonomy over their interactive behaviors. Also, agents should be allowed to adjust their interactive actions to take advantages of opportunities or handle exceptions that arise during interaction.

Our approach is to specify protocols in a declarative manner, i.e. in terms of what is to be achieved rather than how agents should interact. The concept of *commitments* is used to allow agents to perform means-end reasoning to determine an appropriate course of action. We do this reasoning in *temporal linear logic* [3], which combines temporal and resource-sensitive reasoning. It is well-known that temporal logic, in particular, is quite suitable for describing and reasoning about temporal constraints, and has been used in many agent systems. It is also known that linear logic is suitable for modeling resources [1]. Hence we used temporal linear logic to construct a commitment based interaction framework which allows both temporal and resource-related reasoning for interaction protocols.

A commitment refers to a strong promise to undertake some courses of action. Persistence in commitments gives agents a certain level of predictability about the actions of other agents, which is fundamental for issues of interdependencies, global constraints or resource sharing [4]. Commitments have been adapted to make protocols more flexible via commitment machines [6]. Conventions are also used to provide monitoring mechanisms for commitments.

In linear logic, formulas are constrained to be used exactly once and hence treated as resources. The connective $\otimes$ also allows a natural expression of proportion. If A encodes “having one dollar” then $A \otimes A \neq A$ and means having two dollars. Moreover, as remarked in [2], connectives $\&$ and $\oplus$ allow choices to be made clear between internal choices (one’s own), and external choices (others’ choice). For instance, to specify that the choice of places A or B for goods’ delivery is ours as the supplier, we use $A \& B$, or is the client’s, we use $A \oplus B$. Linear logic is a natural mechanism to provide the ability to match consumption and supply of resources among agents and hence can simplify the specification of resource allocation [2].

Temporal Linear Logic (TLL) [3] is the result of introducing temporal logic into linear logic and hence is resource-conscious as well as deals with time. The temporal operators used are $\bigcirc$ (next), $\Box$ (anytime), and $\Diamond$ (sometime) [3]. Formulas with no temporal operators can be considered as being available only at present. Adding $\bigcirc$ to a formula A , i.e. $\bigcirc A$, means that A can be used only at the next time

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS’06 May 8–12 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005 ...\$5.00.

and exactly once. Similarly, $\Box A$ means that A can be used exactly once and at any time, and the choice is internally decided, as appropriate to one's own capability. $\Diamond A$ means that A can be used once at some time and the choice of time is externally decided by others.

Section 2 introduces our framework based around an example of online sale interactions. Then we discuss the advantages and limitations of using our framework to model interaction protocols and achieve flexibility as well as conclusions.

2. OUR FRAMEWORK

2.1 Encoding In Temporal Linear Logic

We consider encoding of resources, capabilities and commitments in agent interaction.

A unit of consumable resources is modeled as a proposition. We also include information about the location and ownership into the encoding of resources. We denote $A@_a$ as that a resource A is located at ($@$) and owned (subscript $_a$) by agent a and is equivalent to $A \otimes \text{Locate}(A, a) \otimes \text{Own}(A, a)$. Numeric figures are further used to abbreviate a multiplicative conjunction of the same instances. For instance, $2 \text{ dollar} = \text{dollar} \otimes \text{dollar}$. Moreover, such $\bigcirc^3 A$ is a shorthand for $\bigcirc \bigcirc \bigcirc A$.

The capabilities of agents refer to producing, consuming and relocating resources. The general representation form is $\Gamma \multimap$, in which Γ describes the conditions before and describes the condition after an application of the capability.

We model non-conditional commitments by negative formulas. For example, a commitment to provide a cricket bat is represented as $\text{cricket.b}^\perp$. A non-conditional commitment is resolved when the committed agent brings about the respective resource or action as required by the commitment, which means there is a presence of a formula like cricket.b , which can remove the negative formula.

A conditional commitment to bring about some effects given certain conditions can be modeled by connecting the conditions to the effects via a linear implication. A general form is $\Gamma \multimap$ where Γ is the condition part and $\multimap$ is the effect part. If the condition Γ is derived, by consuming Γ , the linear implication will ensure that $\multimap$ is resulted. If the conditions can not be achieved, the linear implication can not be applied and hence the commitment is then released.

Capabilities and conditional commitments, despite their distinct meanings, can be viewed as rules of interaction. Protocols can then be specified by appropriate interaction rules.

2.2 Example Scenario

We consider agents whose states are determined by a set of resources available, interaction rules and capabilities together with commitments. The scenario is an on-line sale of cricket bats from a merchant (Mer) to a customer (Cus).

At agent Cus

Cus has an amount of 50 dollars available at any time, and can pay by credit card: $\Box \$50@C \otimes \Box \text{credit.paym}@C_C$. Cus has a commitment of obtaining 2 cricket bats at some time: $[\Diamond(2\text{cricket.b}@C_C)]^\perp$.

At agent Mer

Mer has available at any time 200 cricket bats for sale and 20 gifts: $200 \Box \text{cricket.b}@M_M \otimes 20 \Box \text{gift}@M_M$.

Rule 1: Mer commits offering two cricket bats and a gift

$(2\text{cricket.b}@M_M^\perp \otimes \text{gift}@M_M^\perp)$ to Cus ($2 \Box \text{cricket.b}@C_C \otimes \Box \text{gift}@C_C$) if Cus pays 20 dollars ($20\$@C_C$) either via Paypal ($\text{Paypal}@C_C$) or by credit card ($\text{credit.paym}@C_C$). The choice is at Cus. There are only 30 promotional offers like this.

$30 \Box [20\$@C_C \otimes (\text{Paypal}@C_C \oplus \text{credit.paym}@C_C)] \multimap 20 \Box \$@M_M \otimes 2 \Box \text{cricket.b}@C_C \otimes 2[\text{cricket.b}@M_M]^\perp \otimes \Box \text{gift}@C_C \otimes \text{gift}@M_M^\perp]$

Rule 2: Similarly, Mer will offer a cricket bat to Cus if Cus pays 10 dollars either via Paypal or by credit card. Access to rule 2 can be times, up to Mer's policy.

$\Box [10\$@C_C \otimes (\text{Paypal}@C_C \oplus \text{credit.paym}@C_C)] \multimap 10 \Box \$@M_M \otimes \Box \text{cricket.b}@C_C \otimes \text{cricket.b}@M_M^\perp]$

2.3 Interaction

To achieve goals, agents use TLL inference rules to construct proofs from state formulas and interaction rules. A simple model of negotiation is used. A rule can be proposed to another agent as a commitment it is willing to take. If the other agent agrees, both agents are involved in the commitment. Otherwise, the other agent can reject the proposal or counter-propose another rule.

The interaction begins by Cus trying to fulfill its pending commitment.

1. Cus asks Mer for two cricket bats to be delivered to Cus at some time.

C to M: **REQUEST** $[\Diamond(2\text{cricket.b}@C_C)]^\perp$

To meet the request, Mer looks for applicable rules. One application of rule 1 or two applications of rule 2 at present or any future time t can derive $\bigcirc^t 2 \Box \text{cricket.b}@C_C$ or $2 \bigcirc^t \Box \text{cricket.b}@C_C, t \geq 0$ and either of them can remove $[\Diamond(2\text{cricket.b}@C_C)]^\perp$.

2. Mer will propose each rule to Cus. n is any time up to Mer's choice. With similar analyzes, Cus determines that given their conditions can be satisfied, the proposals can help to derive its request.

We assume that Cus decides that it prefers having a gift and hence it will accept the proposal of rule 1 and refuse the proposal of rule 2.

M to C: **PROPOSE** $\bigcirc^n [20\$@C_C \otimes (\text{Paypal}@C_C \oplus \text{credit.paym}@C_C)] \multimap 20 \Box \$@M_M \otimes 2 \Box \text{cricket.b}@C_C \otimes 2[\text{cricket.b}@M_M]^\perp \otimes \Box \text{gift}@C_C \otimes \text{gift}@M_M^\perp]$

C to M: **ACCEPT**

M to C: **PROPOSE** $\bigcirc^n [10\$@C_C \otimes (\text{Paypal}@C_C \oplus \text{credit.paym}@C_C)] \multimap 10 \Box \$@M_M \otimes \Box \text{cricket.b}@C_C \otimes \text{cricket.b}@M_M^\perp]$

C to M: **REFUSE**

The conditions of the accepted proposal at the application time $n1$ is $\bigcirc^n (20\$@C_C \otimes (\text{Paypal}@C_C \oplus \text{credit.paym}@C_C))$. One way to satisfy the conditions is for Cus, at the next n time points, to derive 20 dollars ($\bigcirc^n 20\$@C_C$); and to pay via Paypal ($\bigcirc^n \text{Paypal}@C_C$) OR to pay by credit card ($\bigcirc^n \text{credit.paym}@C_C$).

3. $\bigcirc^n 20\$@C_C$: as Cus has 50 dollars, it can use 20 dollars: $20 \Box \$@C_C \vdash 20\$@C_C \vdash \bigcirc^n 20\$@C_C$.

There are two options for payment, the choice ($\oplus$) is at agent Cus.

4. We assume that Cus decides to use credit payment. We have $\Box \text{credit.paym}@C_C \vdash \bigcirc^n \text{credit.paym}@C_C$

5. (3) and (4) together allow the initial proposal to be applied to derive $\bigcirc^n \Box \text{gift}@C_C$ and $\bigcirc^n 2 \Box \text{cricket.b}@C_C$ for Cus and commitments of Mer of $\bigcirc^n \text{gift}@M_M^\perp$ and $\bigcirc^n 2[\text{cricket.b}@M_M]^\perp$, which are also resolved by the re-

sources $\Box gift@M_M$ and $2\Box cricket.b@M_M$ available at Mer ($\bigcirc^n gift@M_M^\perp \otimes \Box gift@M_M \vdash \perp$ and $\bigcirc^n 2[cricket.b@M_M]^\perp \otimes 2\Box cricket.b@M_M \vdash \perp$). Also, any value of n that satisfies $n \geq 0$ will allow Mer to fulfill Cus's initial request of $[\Diamond(2cricket.b@C_C)]^\perp$. The interaction succeeds as all commitments are resolved.

2.4 Flexibility

The flexibility desired has been achieved in the example. Mer can choose the time n to apply the sale proposal. In step 2, Cus can choose among the sale proposals offered by Mer to explore the opportunity in the promotional period. Also, it is Cus's decision to proceed with the preferred payment method - step 4. Moreover, attempts at deriving 20 dollars in step 3 or carrying out payment - step 4 - can be switched order, as can be the attempts of Mer at fulfilling the commitments of giving 2 cricket bats and giving the gift in step 5.

3. DISCUSSION AND FURTHER WORK

The basic execution mechanism can be defined by proof construction, i.e. building a proof of the goal formula from the current state and the interaction rules. This means that an appropriate interaction sequence for the agents can be extracted from this proof.

As we have seen, TLL allows a natural and expressive specification of sequences of agent interaction. Temporal operators ($\bigcirc$, $\Box$ and $\Diamond$) and their combinations help to allocate resources or actions in time and express the time order of events. Linear implication ($\multimap$) can express a removal or a consumption together with its consequences. Hence, "consumed resources $\multimap$ produced resources" models resource transformation; "a resource at one agent $\multimap$ that at another agent" models resource relocation; "a resource owned by one agent $\multimap$ one owned by another agent" models ownership change; and "one state $\multimap$ another" models state update. Linear implication also reinforces the commitments that effects achieved only when their conditions are fulfilled.

Furthermore, given that flexibility includes the ability to make a sensible choice, having the choices expressed explicitly ($\&$, $\oplus$, $\Diamond$ and $\Box$) in the specification of interaction protocols provides agents with an opportunity to reason about the right choices during interaction and hence explore the flexibility in them.

In addition, by having the order of rules' applications be determined dynamically by the proof construction rather than be predefined, agents can gain flexibility. Moreover, because changes in the environment can be regarded as removing or adding formulas onto the current state formulas, the proof construction by agents on the current state formulas may require a different set of interaction rules to be used for interaction. This alteration of interactive behaviors to deal with changes brings flexibility into agent interaction.

Moreover, as the proof construction selects a sequence of interaction rules for interaction, selecting from all the possible combinations of interaction rules as in our framework gives more chances and flexibility than selecting from a few fixed (predefined) sequences. It is also more likely for agents to handle exceptions or explore opportunities that arise.

However, as all the temporal operators in TLL refer to concrete time points, one drawback is that we can not express durations in time faithfully. One major disadvantage of simulating a duration of an event by spreading copies

of that event over adjacent time points continuously (like $\bigcirc A \otimes \bigcirc^2 A \otimes \dots \bigcirc^{10} A$) is that it requires the time range to be provided explicitly.

Another disadvantage of our approach is that the rules for interaction require some detailed knowledge of the formulas of temporal linear logic. Clearly it would be beneficial to have a visually-based tool similar to UML diagrams which would allow non-experts to specify the appropriate rules without having to learn the details of the formulas themselves.

Similar works in [5] explore the advantages of TLL by using partial deduction techniques to help agents figure out the missing capabilities or resources and based on that, to negotiate with other agents about cooperation strategies. Our approach differs in bringing the concept of commitment into the modeling of interaction, and providing a more natural and detailed map for specifying interaction, especially about choices, time and updating using the full propositional TLL.

Many other approaches like using Commitment Machines [6] to model protocols use the commitment concept to bring more meaning into agents' interactive actions and hence allow them to be more flexible. However, these approaches use logic systems that are limited in their expressiveness to model resources.

Our further work will include using TLL to verify various properties of interaction protocols such as liveness and safety. Also, we will investigate modeling commitments that can be breached together with the conventions that monitor agents' handling of commitments. This also includes managing conflicts among commitments.

Acknowledgments

We are very thankful to Binh Tran and Michael Winikoff for many stimulating and helpful discussions of this material.

4. REFERENCES

- [1] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
- [2] J. Harland and M. Winnikoff. Agent negotiation as proof search in linear logic (poster). In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002)*, Bologna, July 2002.
- [3] T. Hirai. *Temporal Linear Logic and Its Applications*. Doctoral Dissertation, Graduate School of Science and Technology, Kobe University, Japan, 2000.
- [4] N. R. Jennings. Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, 8(3):223–250, 1993.
- [5] P. Küngas. Temporal linear logic for symbolic agent negotiation. *Lecture Notes in Artificial Intelligence*, 3157:23–32, 2004.
- [6] P. Yolum and M. P. Singh. Commitment machines. In *Proceedings of the 8th International Workshop on Agent Theories, Architectures, and Languages (ATAL-01)*, pages 235–247. Springer-Verlag, 2002.