Using Formal Specification to Resolve Conflicts between Contracting Agents

Rose Gamble and Sandip Sen
Department of Mathematical & Computer Sciences
University of Tulsa
600 South College Avenue
Tulsa, OK 74104-3189
e-mail:gamble@tara.mcs.utulsa.edu, sandip@kolkata.mcs.utulsa.edu

Abstract

In this paper, we use formal specification techniques to provide precompiled and provably correct domain-specific knowledge to autonomous agents. This allows for verifying agent interactions in the environment. In particular, we show that negotiating agents can resolve mutually incompatible local preferences and settle for an acceptable compromise by reasoning with formally specified default and exception behaviors. We use Swarm, a formal specification language, to characterize the preference of negotiating agents that are using a variation of the contract-net protocol. The framework is illustrated on an example from the service interaction problem in telecommunications systems. The combination of formal specification and negotiating agents is a multi-paradigm approach that we believe will provide comprehensive solutions to a variety of multi-agent coordination problems.

1 Introduction

This work represents our initial efforts to combine formal specification and distributed artificial intelligence (DAI) to achieve flexible, yet timely and reliable, solutions to a class of problems that involve coordinating the efforts of two or more agents. Though formal specification methods and DAI have witnessed considerable research activity, the use of one of these methods alone does not produce a complete and timely solution for many practical problems. Whereas more flexibility is needed in formal methods, many DAI techniques do not provide the necessary guarantees of reliability needed to adopt them in critical applications. In this paper we propose one way to use formal specification to provide the necessary correctness guarantees for distributed negotiation. Additionally, the use of formal methods provides debugged, pre-compiled, and indexed domain-specific knowledge that can speed up negotiation by avoiding costly mistakes and eliminating online search for the most appropriate negotiation strategy.

Our multi-paradigm approach provides for the initial detection of inconsistencies and conflicts between agent preferences prior to implementing the distributed system. This detection is performed by examining the formal specification of the relationship between agents and the negotiation options that can be performed. Using the information about possible conflicts, we develop default and exception operations for agents to cover all foreseeable interactions. Once implemented, the system of agents can more reliably detect and resolve conflicting preferences or goals by using the formal specification.

We have used formal specifications to characterize negotiation between contracting agents [5] by developing predicates and their constraints to represent a contracting framework, and then defining safety and progress properties to instantiate appropriate domain-specific knowledge. The goal of this work is to formally characterize task requirements and agent capabilities and preferences, such that these specifications can be relaxed during on-line negotiation to arrive at task assignments that are acceptable to both parties involved in the negotiation process. A cost-based constraint relaxation and selection method is developed using default and exception operations for agents. We have shown that agent operations can be formally specified using the Swarm logic [2], [9] to aid negotiating agents to resolve conflicting requirements amicably and timely. In this paper, we concentrate on the practical application of this approach.

The target application problem is the service interaction problem in telecommunication systems [6]. This problem has been a subject of considerable research using formal specification [4], [14] and to a lesser extent, DAI [1], [7]. Our work bridges a gap between these research directions by providing a multi-paradigm solution to the service interaction problem that encompasses both *a priori* detection of service interactions and negotiation of solutions once the services are implemented.

2 Formal Specification for Negotiation

Negotiation provides a useful metaphor for the coordination of autonomous agents [12]. The most widely known negotiation protocol based on contracting is the Contract Net [11]. The basic premise of this form of negotiation is that, if an agent cannot solve an assigned problem using local resources/expertise, it will decompose the problem into subproblems and try to find other willing agents with the necessary resources/expertise to solve these subproblems. The problem of assigning the subproblems is solved by a contracting mechanism consisting of contract announcement by the manager agent, submission of bids by contracting agents in response to the announcement, and the evaluation of the submitted bids by the contractor, which leads to awarding a subproblem contract to the contractor(s) with the most appropriate bid(s).

We use a particular flavor of contract net as depicted in Figure 1, in that we assume the manager agent requests a service that can be provided by the contractor agent. In this variation and for the domain of problems on which we focus, a manager agent knows the contractor agent with which it should negotiate. The contracting process is used to identify which of the available subcontractors (requirement and bid agents) will be involved in the final contract. This variation of contract net is similar to the one used by Sen [10] to schedule meetings between attendees, in which the attendees (i.e., contractors) to a meeting are known, and the contract net is used to identify a mutually acceptable meeting time.

The managers and contractors belong to the class of *managing* agents; a particular managing agent can assume different roles in different negotiation instances. For a given instance of negotiation, the manager acquires a set of default requirements for the solution. Upon receiving a proposal from a manager, the contractor acquires a set of default bids for the services desired by the manager. For those default bids that conflict with the manager requirements, the contractor acquires a set of possible exceptions to the defaults that it sends to the manager for a decision. If a conflict still remains with the exception bids, the manager acquires exception requirements in a final attempt to conclude the negotiation positively.

Service agents are used by the manager and contractor to acquire the requirements and bids, respectively. A service agent has access to default and exception agents to provide the manager or contractor with sufficient information. A particular managing agent may be using any subset of the total set of available service agents. Since service agents are defined according to the problem domain, they are defined once and replicated among appropriate managing agents.

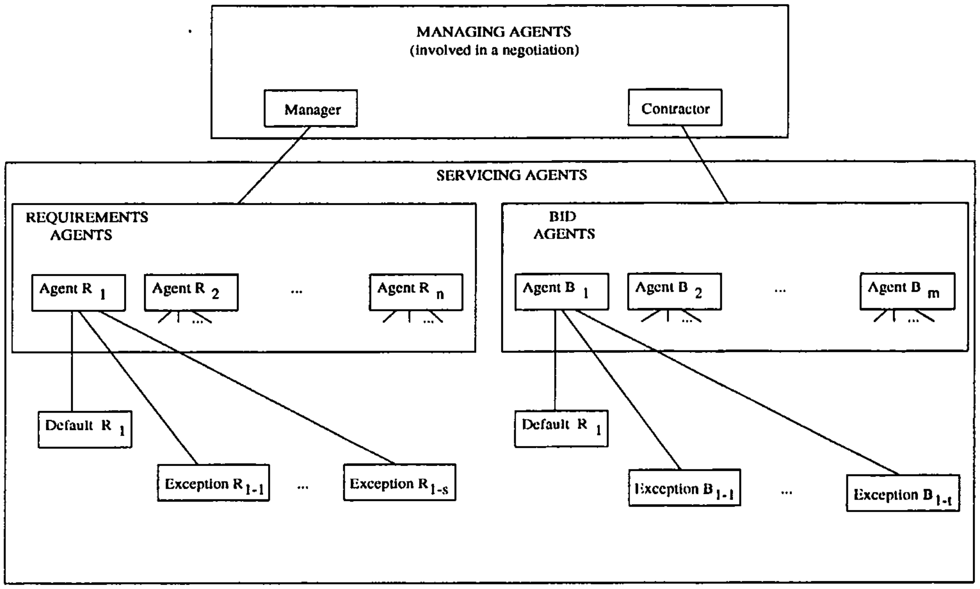

Figure 1: The Negotiation Framework

We use the negotiation framework by instantiating defined predicates within the problem domain. The actions of the negotiation are governed by safety and progress properties expressed over program states described by the predicates [5]. In the next section, we examine a specific problem domain in which we instantiate the defined predicates and illustrate the negotiation process.

3 The Service Interaction Problem

The *service interaction* problem in telecommunications systems falls into the class of problems in which the combined paradigms of formal specification and DAI can result in a comprehensive solution. A pressing problem in the telecommunications industry is how to define the services offered, such as caller ID and call waiting, to a consumer such that they do not conflict with each other and are amenable to the future addition of new services [6]. The conflict that arises between services is called a *service interaction.* In essence, a service interaction occurs when a service cannot be completed due to the presence of other active services.

In our negotiation framework, dialing numbers (DNs) are modeled as managing agents, and the services offered are modeled as servicing agents. The originating DN is the manager in a negotiation trying to connect to the destination DN, which is the contractor. The manager has a requirement agent for each caller service to which it subscribes, e.g., an unlisted number. The contractor has a bid agent for each callee service to which it subscribes, e.g., caller ID. Upon originating a call, the manager announces its default requirements that are needed to complete the call to the contractor. The contractor receives the requirements and requests its default bids from its bid agents. Each of the bid agents presents its needs (preconditions) for a connection to be established (postcondition). Default operations (both requirements and bids) have a cost of zero. The managers will perform the lowest cost option. Thus, performing the default operations is the most attractive solution to the problem, provided there are no conflicts. Once all default operations are obtained, the contractor determines which, if any, of the default bids conflict with the manager's requirements. If none conflict, the manager and the contractor agree to connect the call. If there is a conflict, the contractor requests agents to provide exceptions for those defaults found to be conflicting. These bids are passed directly to the manager for a decision. The manager will resolve any conflicts with the lowest cost exception operation if available. If a conflict still remains, the manager will request exception requirements for those requirements that are still in conflict. Again, the lowest cost exception operation resolving the conflict will be chosen. If no positive action described by the service agents can be taken due to remaining conflicts, the negotiation terminates by disconnecting the call. Each managing agent has knowledge of this action.

To instantiate the domain, we associate with each DN and service an *Agent* predicate, in addition to whether it is a managing agent or a servicing agent. Given an agent with a particular dialing number, DN_1, has an unlisted number, the following predicates are *true* in the environment:

$$\text{Agent}(DN_1),\ \text{Managing}(DN_1),\ \text{Agent(unlisted-number)},\ \text{Servicing(unlisted-number)}.$$

With an attempted call, the negotiation commences by instantiating the DNs to their respective types. For this example, we assume that the unlisted number service is only service to which DN_1 subscribes. The representation of the service resides with the requirement agents of DN_1 and its set of bid agents are empty as represented in the *Requirements* predicate below.

$$\text{Requirement}(DN_1,\{\text{unlisted}\}),\ \text{Bid}(DN_1,\{\})$$

The only restriction on connecting a call for the unlisted service is that the number should not be identified at the destination. We can instantiate a default operation to reflect this restriction. Each *Default* predicate has the service agent, a precondition of the operation, and a postcondition. The cost of a default is zero.

$$\underset{\textit{Agent}}{\text{Default(unlisted}},\ \underset{\textit{Pre}}{\neg\text{Identified}(DN_1)},\ \underset{\textit{Post}}{\text{connect})}$$

In the presence of a conflict with the unlisted number service, the caller may allow some name identification to be used as opposed to the number. Each exception operation has the service agent, a unique identifier, pre- and postconditions, and the cost of employing the exception, which can be customized by the subscriber.

$$\text{Exception}(\underset{\textit{Agent}}{\text{unlisted}},\ \underset{\textit{Num}}{1},\ \underset{\textit{Pre}}{\neg\text{Identified}(DN_1) \wedge \text{Name-Displayed(``Lee'')}},\ \underset{\textit{Post}}{\text{connect}},\ \underset{\textit{Cost}}{12})$$

We examine the following service interaction. A caller with an unlisted number attempt to connect a call to someone with caller ID, which requires disclosure of the calling number. This default operation leads to a conflict the between users' intentions when they subscribed to the features. One exception operation to resolve the conflict is to negotiate the delivery of alternative identifying information [13]. Additionally, since the number is protected by the unlisted service and the caller is not intentionally overriding the display of his or her number (i.e., using *67), another exception operation of the caller ID service could simply display "unlisted number" and allow the subscriber (or callee) to determine whether or not to answer the phone.

Assume that DN_1 calls DN_2 and DN_2 subscribes only to caller ID. We assert agent predicates for DN_2 and the caller ID service. In addition, we express that DN_2 subscribes to caller ID, making it part of DN_2's bid agents.

$$\text{Agent}(DN_2),\ \text{Managing}(DN_2),\ \text{Agent(caller-ID)},\ \text{Servicing(caller-ID)}$$
$$\text{Requirement}(DN_2,\{\}),\ \text{Bid}(DN_2,\{\text{caller-ID}\})$$

Given the three operations available for the caller ID service, DN_2 sets a cost on the negotiation strength of the bid agents. We assume that DN_2 wants the service to attempt to display some alternate identification before displaying the fact that the number is unlisted.

Default(caller-ID, (Identified(DN) $\vee$ Wanted(DN)),(connect))
only numbers that can be displayed or are wanted by subscriber are connected

Exception(caller-ID, 1, ($\neg$Identified(DN) $\wedge$ Name-Displayed(Name)), (connect), 2)
display alternate identification

Exception(caller-ID, 2, ($\neg$Identified(DN) $\wedge$ Unlisted(DN)), (display("unlisted")), 6)
displays the fact that the caller has an unlisted number

With this service information, the negotiation proceeds as follows. First the pre- and post-conditions of the requirements from DN_1 services would be given to DN_2. In our example, the requirements are *Default-Pre*(DN_1, *unlisted*) = $\neg$*Identified*(DN_1) and *Default-Post*(DN_1, *unlisted*) = *connect*. DN_2 then acquires default bids for all of its callee services, which are, *Default-Pre*(DN_2, *caller-ID*) = *Identified(DN)* $\vee$ *Wanted(DN)* and *Default-Post*(DN_2, *caller-ID*) = *connect*. A comparison by DN_2 shows the defaults to be in conflict. Thus, DN_2 acquires all exception operations for this conflicting service. Services not in conflict do not bid with exceptions. Thus, DN_2 returns the following.

Exception-Pre(DN_2, caller-ID, 1) = $\neg$Identified(DN) $\wedge$ Name-Displayed(Name)
Exception-Post(DN_2, caller-ID, 1) = connect
Exception-Pre(DN_2, caller-ID, 2) = $\neg$Identified(DN) $\wedge$ Unlisted(DN)
Exception-Post(DN_2, caller-ID, 2) = display("unlisted")

At this point, DN_1 examines the exceptions and finds that the conflict with its default remains. Therefore, DN_1 acquires its exception operations for all service requirements still in conflict.

Exception-Pre(DN_1, unlisted, 1) = $\neg$Identified(DN_1) $\wedge$ Name-Displayed("Lee")
Exception-Post(DN_1, unlisted, 1) = connect

Exception-Pre(DN_1, *unlisted, 1*) and *Exception-Pre*(DN_2, *caller-ID, 1*) can be matched successfully with *Name* = *"Lee"*. Thus, the connection between DN_1 and DN_2 is established.

In an attempt to resolve service interactions, many researchers have chosen a path of formal specification or DAI. An important distinction between our work and other DAI approaches to the service interaction problem is that local information is completely decoupled in that we do not rely on shared preference hierarchies [13], etc. Communication between agents involves the exchange of requirements and bid specifications. This information can change locally (e.g., when a DN subscribes to a new service or cancels an old one) without necessitating any changes in the knowledge of other agents. This allows the development of more robust and flexible systems.

4 Conclusion

In this paper, we have shown how distributed agents can resolve conflicts using formally specified requirements and capabilities. To develop a full-fledged multi-paradigm approach to negotiation, we need to develop mechanisms to use when the formal specification is incomplete. This development would require encoding knowledge to detect a situation where relevant pre-compiled negotiation knowledge is unavailable, and then formulating a negotiation strategy to deal with such circumstances. During such encounters, the system would preferably acquire new conflict resolution knowledge to fill in the gaps in the formal specification. We believe that we can use ideas from explanation-based generalization and learning [3], [8] to address some of these critical issues. We also intend to develop a CASE tool to aid specification input and refinement for services.

References

[1] E.J. Cameron and H. Velthuijsen. Feature interactions in telecommunications systems. *IEEE Communications*, 31(8):18–23, August 1993.

[2] H.C. Cunningham and G.-C. Roman. A UNITY-style programming logic for a shared dataspace language. *IEEE Transactions on Parallel and Distributed Systems*, 1(3):365–376, 1990.

[3] Thomas Ellman. Explanation-based learning: A survey of programs and perspectives. *ACM Computing Surveys*, 21(2):163–221, November 1989.

[4] A. Fekete. Formal models of communication services: a case study. *IEEE Computer*, 26(8):37–47, August 1993.

[5] R. Gamble and S. Sen. Using formal specification to aid negotiating agents. Technical Report UTULSA-MCS-94-2, University of Tulsa, Tulsa, OK, March 1994.

[6] N. Griffeth and Y.-J. Lin. Extending telecommunications systems: the feature-interaction problem. *IEEE Computer*, 26(8):14–18, August 1993.

[7] N.D. Griffeth and H. Velthuijsen. Win/win negotiation among autonomous agents. In *Proceedings of the Distributed Artificial Intelligence Workshop*, pages 187–202, 1993.

[8] T. Mitchell, R. Keller, and S. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1:47–80, 1986.

[9] G.-C. Roman and H.C. Cunningham. Mixed programming metaphors in a shared dataspace model of concurrency. *IEEE Transactions on Software Engineering*, 16(12):1361–1373, December 1990.

[10] Sandip Sen. *Predicting Tradeoffs in Contract-Based Distributed Scheduling.* PhD thesis, University of Michigan, October 1993.

[11] Reid G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, December 1980.

[12] Reid G. Smith and Randall Davis. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, 20:63–109, 1983.

[13] H. Velthuijsen. Distributed artificial intelligence for runtime feature-interaction resolution. *IEEE Computer*, 26(8):48–, August 1993.

[14] P. Zave. Feature interactions and formal specifications in telecommunications. *IEEE Computer*, 26(8):20–30, August 1993.