Dynamically Loaded Reasoning Models in Negotiating Agents

Garima Parakh, Marcin Paprzycki
Computer Science Department
Oklahoma State University
{garima, marcin}@cs.okstate.edu

Cristina Elena Nistor
Faculty of Computer Science
University of Iasi
nce@infoiasi.ro

Abstract

For the past decade the agent paradigm has been gaining popularity, particularly in applications of mobile agents in e-commerce. Here, support for negotiation is one of the more important research issues. Depending on the type of the transaction, different types of negotiations need to be utilized. Currently, very few frameworks are generic and flexible enough to handle multiple scenarios. This note proposes negotiating agents, which can change their reasoning model depending on circumstances. This goal is achieved through dynamic loading of reasoning models and their rule-based selection.

1. Introduction

The advent of Internet, particularly the rapid development of e-commerce, gave a boost to the research in agent technology. While there exist many definitions of agents [GATY], for the purpose of this note we will define them as follows. An agent is an encapsulated computer program that is situated in some environment, and that is capable of flexible, autonomous actions in that environment in order to meet its design objectives [WLD]. This typically implies that an agent is adaptable, intelligent, personalized, and possibly mobile. Some examples of agents are Personalized Travel Agent, Meeting Organizer, Workflow agent, Bargaining agent, Auction Agents etc. Agents are often considered in context of e-commerce applications. The number of e-commerce WWW-sites is currently estimated at more than 150000 worldwide (Gartner [GAR]) with revenue projections up to $1.5 trillion in 2004 (Andersen [AND] and Forrester [FOR]). Since e-commerce offers the opportunity to integrate and optimize the global production and distribution supply chain, automated trading using agents, should be able to considerably reduce the transaction costs. Even better results could be achieved if agents implement appropriate "intelligent" trading capabilities. This means that agents would be able to reason and negotiate, provide counter-offers and critiques regarding their proposals autonomously and dynamically during the negotiation process.

Most currently existing automated trading systems are not generic and focus mostly on a particular domain. For example the Kasbah Trading System [MAES] focuses on buying and selling but does not include participation in auctions. Some new systems for automated negotiation like SILKROAD [STB], FENAs [KOW1] and Inter-Market [KOW2] have been successful in providing the foundation of e-commerce systems, but they lack a concrete implementation. This paper proposes the possibility of agents, which can operate according to different business models including auctions, reverse auctions, trading, e-sales etc. This is made possible by incorporating a meta-level rule-based support for agents. In addition to this, each agent consists of three independent modules: protocol module, strategy module and communication module. These are designed in the form of plug-in components that can be remotely loaded on-demand. This is described in detail in the remaining parts of this note, which are organized as follows. First, in Section 2, we provide background information about negotiation in agents. In Section 3, we describe the proposed conceptual model. Section 4 contains the specification of the design of our model. Finally, in Section 5, we briefly discuss the proposed implementation.

2. Negotiations and the Internet

Negotiation is a method for coordination and conflict resolution. Conflict can be in the form of resolving goal disparities in planning, resolving constraints in resource allocation, and resolving task inconsistencies in determining organizational structure. As indicated above, in the context of the Internet it will be autonomous agents that will be involved in negotiations. Overall, research on agent-mediated negotiation can be divided into approaches based on game theory or artificial intelligence.

Game-theoretic approach is directed towards developing optimization algorithms (e.g. Rosenschein and Zlotkin [ZLT], AuctionBot [AUC]). This approach takes into account both cooperative and non-cooperative agents. In the case of cooperative agents, the problem space can be divided efficiently between all agents. In [TAMBE], agents form a team, each having its own local goals and a team goal. When a conflict arises, the team members can negotiate about the matter, and give evidence supporting their stance. However, the team members are assumed to have total knowledge about the system. In non-cooperative approach, theories like Nash equilibrium are applied to the bargaining problem to find the optimum solution for the agents. The drawback of game-theoretic approach is that it assumes unrealistic properties in the game. Agents are assumed to have the entire common knowledge and unbounded rationality. In addition, they are assumed to have unlimited computation power and indefinite negotiation time, making such approaches impossible to implement. Nevertheless, the extensive research carried out in this field helped develop other theories. An example of game-theoretic approach is the work on modeling and implementing techniques for agents participating in auctions e.g. Dutch auction, English auction, Vickery auction, etc.

Artificial Intelligence based approaches utilizes trading heuristics for different market mechanisms (e.g. Chavez et al. [MAES], Faratin et al. [JEN1]). AI techniques focus on the negotiation process rather than the outcome of the negotiation. Mostly learning approaches like decision trees, Q-learning and evolutionary algorithms have been used to improve bargaining strategies. The agents used are adaptable, realistic and sociable. AI theories are based on realistic assumptions of an imperfect world with bounded rationality and limited knowledge of the world. Since agents do not know a'priori what type of agents they are interacting with, this creates conflicts between agents. Simulations of agents learning different bargaining strategies, through genetic algorithms, have been carried out and their results have been quite promising.

When considering the practical aspects of designing multi-agent negotiations, the negotiation protocol, negotiation objects and the reasoning models [JEN2] need to be taken into account.

1. Negotiation protocol consists of a set of rules that govern the interaction among agents. Some examples of the rules are permissible types of participants: negotiators, third parties; negotiation states: accepting bids, negotiation closed; valid actions of the participant in particular states.

2. Negotiation objects are ranges of issues over which agreement must be reached. These depend on the environment and can be different for different environments.
3. Reasoning model is the apparatus that participants employ to act in line with the negotiation protocol in order to achieve their negotiation objectives. Reasoning models are the thinking machines behind the process of carrying out the negotiation. Reasoning model is a mechanism by which the next counter-offer is calculated during negotiation so that the price fits into the goal of buying some good within the specified range. Some of the strategies developed so far are argumentation, persuasion and heuristics-based. It can be safely assumed that the kind of reasoning model chosen depends on both the protocol and the negotiation object [JEN1]. Moreover, the complexity of former depends on latter.

Our architecture is based on the above fundamentals. Managing the three main areas: protocol, strategy and negotiation object is the essential issue concerning today's bargaining systems and we introduce one of the possible approaches to achieving this goal.

3. Conceptual model of the proposed system

It is often suggested that agent mobility is a source of an important advantage of agent systems over other approaches. Mobile agents can be used in disburdening the users with everyday tasks, while communicating mobile agent teams can handle simultaneously very large amounts of information. Users can be off-line whenever they want, while agents continue working for them. If transactions are monitored through a mobile device like a cell phone or a PDA, the cost of sending and receiving data is high. In such situations, mobile agents are ideal mechanism to carry out automated negotiation. Since mobility comes at a cost, mobile agents cannot be loaded with all the reasoning power available, as they have to be lightweight. We will therefore design our system around mobile agents.

The concept of dynamically loading agent's capabilities is powerful and is advantageous in the above-mentioned scenario. We merge this concept with that of a flexible negotiation system, which caters to needs of users interested in interacting in multiple domains. Most currently existing e-commerce systems use predefined non-adaptive negotiation strategies in generation of offers and counter offers. Since we believe that adding flexibility to the system can benefit both the buyer and the seller, in our system agents will be composed of plug-in modules that can be remotely loaded when the need arises. It should be stressed that not only the customer will benefit from such an approach, as it can be also applied on the seller site, making it more flexible and helping maximizing sales. However, due to the space restrictions we will concentrate our attention on the mobile agents representing buyers.

An example, somewhat similar to our proposal, is the Inter-market system [KOW2] and to Magnet [COL]. Inter-market comprises of mobile agents and intelligent decision-making agents offered as an add-on component to the commercial e-marketplace platform Inter-Shop. In Inter-market, there are two types of agents. Stationary agents run automated processes interacting with other agents or with users. These are provided by the Inter-Market system, their built-in functions cannot be extended or modified by any user of the Inter-Market system. Therefore, we can trust these agents. They are parameterized to perform trading tasks according to the users instructions. Mobile agents on the other hand, are used as means of communication when exchanging information between mobile devices and Inter-Market systems.

4. Design of the system

As mentioned above, an agent in our system is composed of plug-in components (the complete design of the agent is depicted in Figure 1):

1. **Communication module** – responsible for communication between the agents in a common, understandable way. Since FIPA, as a main agent standardization body, is supporting a number of communication technologies (e.g. the ACL communication language) [FIPA], we will omit this part as falling outside the scope of this note. This is especially so since this module is static one (while this could be an interesting research topic in its own rights – in this note we are not concerned with dynamically loading different communication models).

2. **Protocol module** – is to enable automated negotiation keeping the rules of the negotiation in mind. The first task an agent performs is matchmaking. A table as seen below can be used to find out the potential sellers of a commodity that we desire.

SELLER	PRODUCT	PROTOCOL	STRATEGY	SUCCESS RATE
Seller 1	Used Cars	Offer-Counter Offe	Tit-For-Tat	0
Seller 2	Used Cars	Offer-Counter Offe	Tit-For-Tat	60
Seller 3	Used Appliances	Argumentation	Persuade/Critique	90
Seller 4	Used Appliances	Auction	Heuristics	70
Seller 5	Travel package	Offer-Counter offe	Boulware + dependent	40
Seller 6	Travel package	Bidding		
Seller 7	Air tickets	Auction		

Table 1: Sample matchmaking table for negotiation initialization

This table lists sellers and protocol they use. The entries in this table could be a result of previous transaction with the seller or meta-negotiation with the seller in order to find out what protocol he/she is ready to conform with. The strategy column is dealt with in the strategy module. Once it is determined which protocol will be used, the agent dynamically loads the correct module from the user's local machine or any agent server (e.g. the nearest one). This way the agent is flexible enough to carry out any negotiation with any seller while being lightweight.

3. **Strategy module** – is designed to apply the proper reasoning module so that the negotiation ends in a success. The reasoning model contains policies, which are a set of goals, actions and action rules (triggers). In order to decide which reasoning model to use, the agent uses the mapping table. This records the earlier history of the transactions, which the agents made with the previous seller. It also lists what was the success rate for the transactions. If an entry is not found, then the agent resorts to a default strategy. To keep the agent lightweight it will carry only a part of the table, containing the "sites" most often negotiated with, while the remaining part of the

table will be kept on the user’s machine. In case the user’s machine is off-line the default strategy will be used, again (since this will involve only sites that are visited rarely, such an occurrence should not be detrimental to the overall behavior of our system). The strategy also depends on which protocol module has been chosen. In fact, the strategy is almost restricted by that decision. For example, a strategy used for argumentation cannot be used if the protocol is an auction protocol. It should be noted that it is possible define two levels of strategies: coarse and granular. The granular strategies are tit-for-tat, conceder etc, while the coarse strategies are heuristics, using knowledge base etc. Their usage will depend on the negotiation context and the details of the selection strategy will be further studied. We expect that a special (rule-based) strategy selection meta-module will be developed. This module will, again be split between the agent, that will carry the most important rules, and the user local machine that will contain the complete module. In this way the agent will be able to get involved in negotiations even if its contact with the user’s machine will be impossible to establish.

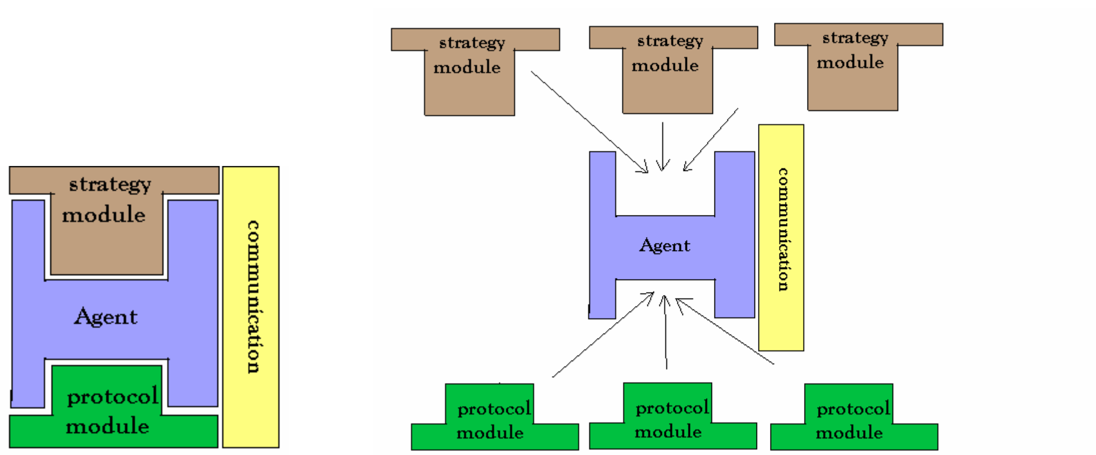

Figure 1. An Agent and its plug-in components

5. Proposed implementation

An implementation of the above-described scheme is proposed using Java's dynamically loaded classes. If a reasoning model is encapsulated in a Java class, then an agent can dynamically load its reasoning model locally or remotely by using reflection and Java's dynamic class loading. The modules can be viewed as intelligent agent implementations that act as plug in for different e-negotiation functions and tasks tailored to specific industries and market segments. Grifell et al. [GRF1, GRF2] have worked on designing plug-in modules and rule-based negotiation [GRF3]. We have taken ideas from such contemporary research and tried to build upon them. Componentware technology can be used, where new functionality can be added on the fly with need of reconfiguration but no recompilation. Each functional module is designed with well-defined interfaces.

6. Concluding remarks

This note presents a new approach to designing intelligent, flexible agents for e-commerce. We are particularly interested in e-commerce agents involved in negotiations and propose that they will be able to dynamically load their negotiation protocol module and their negotiation reasoning/strategy module. In the near future, we will attempt at implementing such a model in Jade agent environment.

References

[AND] Andersen Consulting. http://www.andersen.com

[AUC] AuctionBot. http://auction.eecs.umich.edu

[BEAM] Beam, Segev et al , On Negotiation and Deal Making in Electronic Markets

[BEN] Benameur, Chaib-draa and Kropf , Multi-item Auctions for Automatic Negotiation

[COL] J. Collins, M. Tsvetovat, B. Mobasher, and M. Gini, "MAGNET: A Multi-Agent Contracting System for Plan Execution", Workshop on Artificial Intelligence and Manufacturing: State of the Art and State of Practice, pp 63-68, AAAI Press, Albuquerque, NM, August 1998.

[FIPA] http://www.fipa.org

[FOR] Forrester. http://www.forrester.com

[GATY] V. Galant, J. Tyburcy (2001) Intelligent Software Agent (in Polish), in: A. Baborski (ed.), Knowledge Acquisition in Databases, Wrocław University of Economics

[GAR] Gartner. http://www.gartner.com

[GUT] Guttman, Maes (1998) , Agent-mediated Integrative Negotiation for Retail Electronic

[GRF1] M. T. Tu, F. Griffel, W. Lamersdorf , Integration of Intelligent and Mobile Agents for E-commerce

[GRF2] F. Griffel, W. Lamersdorf, M. Merz, A plug-in architecture for providing dynamic negotiation capabilities for mobile agents

[GRF3] F. Griffel, W. Lamersdorf, M. Merz, Interaction-Oriented Rule Management for mobile agent applications,

[KOW1] R. Kowalczyk ,On Fuzzy e-Negotiation Agents: Autonomous negotiation with incomplete and imprecise information

[KOW2] R. Kowalczyk, B. Franczyk, A. Speck, Inter-Market, towards intelligent mobile agent E-Market places

[KRS1] Sarit Kraus, Negotiation and Cooperation is Multiagent environment

[KRS2] Kraus, et al. (1998) ,Reaching agreements through argumentation: a logical model

[JEN1] Sierra, Faratin, Jennings, Service Oriented Negotiation Model between Autonomous Agents

[JEN2] Jennings Parsons (1998) , On Argumentation-Based Negotiation

[JEN3] Beer, d'Inverno, Luck, Jennings (1999) , Negotiation in Multi-Agent Systems

[LES] Lesser, Negotiation among computationally bounded self-interested agents

[MAES] Chavez and P. Maes, "Kasbah: An Agent Marketplace for Buying and Selling Goods "

[PAR] Parunak , Characterizing Multi-Agent Negotiation

[STB] Ströbel Michael "Design of Roles and Protocols for Electronic Negotiations"
Electronic Commerce Research Journal, Special Issue on Market Design, Vol.1

[TAMBE] Qiu, Tambe, Flexible Negotiation in Teamwork

[WLD] M. Wooldridge (1997) “Agent-based software engineering” IEEE Proc. on Software Engineering, 144 (1) 26-37.

[ZLT] Zlotkin, G.; Rosenschein, *J.S.* **,** Cooperation and conflict resolution via negotiation among autonomous agents in non-cooperative domains