Learning other agents' preferences in multiagent negotiation

H. H. Bui, D. Kieronska and S. Venkatesh
Department of Computer Science
Curtin University of Technology
Perth, WA 6001, Australia
buihh, dorota, svetha@cs.curtin.edu.au

Abstract

In multiagent systems, an agent does not usually have complete information about the preferences and decision making processes of other agents. This might prevent the agents from making coordinated choices, purely due to their ignorance of what others want. This paper describes the integration of a learning module into a communication-intensive negotiating agent architecture. The learning module gives the agents the ability to learn about other agents' preferences via past interactions. Over time, the agents can incrementally update their models of other agents' preferences and use them to make better coordinated decisions. Combining both communication and learning, as two complement knowledge acquisition methods, helps to reduce the amount of communication needed on average, and is justified in situations where communication is computationally costly or simply not desirable (e.g. to preserve the individual privacy).

Introduction

Multiagent systems are networks of loosely-coupled computational agents that can interact with one another in solving problems. In such systems, it is often not feasible for any agent to have complete and up-to-date knowledge about the state of the entire system. Rather, the agents must be able to work together, without prior knowledge about other agents' mental (internal) states.

Traditional work in distributed problem solving relies heavily on the communication between problem solving nodes in order to provide the kind of coordination necessary in a distributed system (Bond & Gasser 1988). Research in theories of agency based on the formulation of agents' mental states also uses communication as the only method for acquiring knowledge about other agents' mental states (Woodridge & Jennings 1995). Driven by the costs and problems associated with communication, recent work in multiagent learning has suggested learning as an alternative knowledge acquisition method. It has been shown that even without communication, agents can learn to coordinate their tasks in simple multiagent settings (Sen & Sekaran 1995),(Sen, Sekaran, & Hale 1994).

Our on-going research goal is to design a generic architecture for negotiating agents. In this domain, the importance of learning from previous experiences was documented in (Sycara 1989) where case-based reasoning techniques were used to reduce the communication overhead in the PERSUADER system. Since negotiation is a communication-intensive task, rather than using learning as a complete replacement for communication (Sen & Sekaran 1995), we view both communication and learning as two complementary knowledge acquisition techniques, each with its own strengths and weaknesses. Communication, typically, is more expensive (in terms of time and resource) than computation and can become a bottleneck of the negotiation process. However when one asks the right question and gets back the correct response, the information one gathers is certain. On the other hand, learning is performed locally by each individual agent and is thus less costly, however, the information acquired is mostly uncertain. The contrasting characteristics of the two knowledge acquisition methods make a hybrid approach an attractive alternative.

This paper describes how to integrate a learning component into a reactive agent architecture in which the agents negotiate by refining a joint intention gradually until a common consensus is reached (Bui, Venkatesh, & Kieronska 1995). Here, we assume that the agents are cooperative and sincere. We use a simple learning mechanism that allows an agent to make predictions about other agents' preferences by building statistical models of others' preference functions from its past interactions with them. The learning mechanism helps to reduce the amount of communication needed, and thus improves the overall efficiency of the negotiation process. The approach is illustrated with an example from the distributed meeting scheduling domain.

The paper is organised as follows: the following section introduces the negotiation context under which our agents interact; next, we describe the learning mechanism and how it is integrated into the agent's architecture; finally we show some initial experimental results in the distributed meeting scheduling domain and provide a preliminary evaluation of the approach.

The negotiation context

Definitions

We use the term *negotiation context* to refer to situations where a group of agents with different preferences are trying to achieve a common agreement. Due to the distributed nature of the problem, the agents, at best, possess only partial knowledge about other agents' preferences. Such problems turn out to be ubiquitous in Distributed Artificial Intelligence (Bond & Gasser 1988). Although a number of negotiation protocols (Smith 1980),(Conry, Meyer, & Lesser 1988) and agent architectures (Laasri *et al.* 1992) have been proposed, attempts to formalise and construct a generic agent architecture have proved to be quite complex (Woodridge & Jennings 1994).

In order to aid the clarity of further discussions, we define here a formal notion of a simple negotiation context $\mathcal{N}$ as follows:

- A group of agents $\mathcal{A}$ involved in the negotiation. Subsequently, we will use the capital letters $A, B, C \ldots$ to denote members of $\mathcal{A}$.
- A domain $\mathcal{D}$ represents the set of all possible agreements. Let $\delta \subset \mathcal{D}$ be a subset containing some agreements. We use the notion of intention $Int(A, \delta)$ to denote agent A "intends to look for the final agreement" within δ. Similarly, $JInt(\mathcal{A}, \delta)$ denotes the joint intention of all the agents in $\mathcal{A}$ to look for the final agreement within δ. If $JInt(\mathcal{A}, \delta)$ holds, δ is termed the current *agreement set* of the agents in $\mathcal{A}$.
- For each agent $A \in \mathcal{A}$, a function $f_A : \mathcal{D} \rightarrow \mathcal{R}$ (the set of real numbers) represents the *preferences* of agent A over the set of possible agreements $\mathcal{D}$. For $\delta \subset \mathcal{D}$, $\overline{f}_A(\delta)$ denotes the mean of $f_A(d), d \in \delta$. One can think of the preference $f_A(d)$ as the amount of money A would earn if d were accepted as the final agreement. The preferences are thus additive, and we define the preferences of the group $\mathcal{A}$ by the sum of its members' preferences $F_{\mathcal{A}}(d) = \sum_{A \in \mathcal{A}} f_A(d)$ [1].

The negotiation process

Throughout the negotiation process, the agents attempt to find a common agreement by refining their joint intentions incrementally. At the start of the negotiation process, $\delta = \mathcal{D}$. The incremental behaviour of the negotiation process is guided by an *agreement tree* defined as a tree structure whose nodes are agreement sets with the following properties: (1) the root node is $\mathcal{D}$, (2) all the leaf nodes are singleton sets, and (3) the set of all children of a node is a partition of that node.

At the k-th iteration of the negotiation process, each agent A would attempt to refine the current joint agreement set δ_k (at level k in the tree structure) to some new tentative agreement set $\delta^A_{k+1} \subset \delta_k$ (at level k+1). The choice of δ^A_{k+1} depends on A's perception of the expected utility of those possible agreements within the agreement set δ^A_{k+1}. The choice of refinement becomes the agent's individual intention and is broadcast to other agents in the group.

If all individual refinement choices agree, the group's refinement choice becomes the new joint agreement set of the agents. Otherwise, the differences in the individual refinement choices are resolved through further communication between the agents in three steps: (1) each agent collects other agents' preferences of its own refinement choice; (2) each agent calculates the group's preference for its refinement choice and uses this preference as a new ranking value for its own choice; and (3) the agents choose a winner among themselves on the basis of maximal ranking value (to assure a clear winner, a small random perturbation can be added to the ranking value in step 2). Subsequently, the winner's refinement choice is adopted by the whole group of agents.

At the end of the k-th iteration, all the agents in the group should form a new agreement set $\delta_{k+1} \subset \delta_k$ or decide that the agreement set δ_k is over-constrained and backtrack to δ_{k-1}. The iterative negotiation process ends when either an agreement set $\delta_s = \{d\}$ at the leaf level is reached, or $\delta_0 = \mathcal{D}$ is over-constrained itself. In the former case, a solution is found whereas in the latter case, the negotiation is regarded as failing.

Problems with incomplete knowledge

Crucial to the performance of the above negotiation protocol is the decision involved in choosing the refinement of an agreement set. Ideally, the agents should choose a refinement δ_{k+1} for δ_k so as to maximise the group's preferences $F_{\mathcal{A}}$.

Given that an agent A's preference value for a refinement choice δ is $\overline{f}_A(\delta)$, the sum of all group members' preferences for δ is $F_{\mathcal{A}}(\delta) = \sum_{A \in \mathcal{A}} \overline{f}_A(\delta)$. In the ideal case where every agent uses $F_{\mathcal{A}}(\delta)$ to select a refinement, all individual refinements and intentions will be the same, hence a new agreement set can be formed immediately without further complication.

[1] Since the product of positive numbers corresponds to a sum via a logarithmic transformation, the results of this paper still applies when the group's preferences are defined as the product of the (positive) individual's preferences.

To see why the ideal case might not happen in practice, let's rewrite $F_A(\delta)$ as:

$$F_A(\delta) = \overline{f}_A(\delta) + F_{other,A}(\delta)$$

where $F_{other,A}(\delta) = \sum_{B \neq A} \overline{f}_B(\delta)$

Unfortunately, the component $F_{other,A}(\delta)$ is usually not readily available to A since it requires knowledge about other agents' preference functions. In situations where other agents are eager to reveal their preference functions, A can directly ask other agents about their preferences (***ask-first selection method***). Such an approach requires additional communication and may not be feasible in circumstances where exposure of individual preference is not desirable.

When asking others is costly, the agents can choose the refinement by maximizing only their own preferences (***don't-ask selection method***). However, this approach usually leads to diverging and conflicting individual intentions and requires a lengthy conflict resolution stage.

We propose the use of learning as an alternative knowledge acquisition method to counter the problem of incomplete knowledge. If it is not desirable to acquire the knowledge from asking questions directly, why not learn to predict what the answers would be? Furthermore, in our negotiation context, making a false prediction will not result in a catastrophe (the worst situation is when extra exchange of messages is needed). With a mechanism to make reasonably good predictions about other agents' preferences, we are likely to improve the efficiency of the whole negotiation process.

Learning other agents' preferences

Learning data

A negotiating agent throughout its lifetime will participate in a potentially large number of different negotiation contexts. Although each negotiation context has a different set of participating members, closely affiliated agents are likely to engage in the same negotiation context more often. Furthermore, the domains of these negotiation contexts are usually subsets of a common domain. For example, in resource allocation, the set of resources to be allocated might be different from one negotiation to another, however, they are usually drawn out of one common set of resources frequently shared by the agents. In meeting scheduling, the time windows for the meetings to be scheduled are different, however, again, they are subsets of one common time line.

We denote this common domain of all negotiation contexts by $\mathcal{D}^*$. Formally, $\mathcal{D}^*$ is the union of the domains of all negotiation contexts: $\mathcal{D}^* = \cup_{\mathcal{N}} \mathcal{D}_{\mathcal{N}}$ where $\mathcal{D}_{\mathcal{N}}$ denotes the domain of the negotiation context $\mathcal{N}$.

An agent has the opportunities to acquire sample data about others' preference functions via the number of exchanges of preferences taking place in previous negotiation contexts. For example, from the agent A's viewpoint, the accumulated samples of f_B are the set of values $f_B(d)$ for some random d's drawn out of $\mathcal{D}^*$. This sample data in turn can help the agent in making predictions about others' preferences should they be in the same negotiation context in the future.

Learning mechanism

This subsection describes how an agent can use a simple learning mechanism to accumulate samples of other agents' preference functions and make statistically-based predictions of their future values.

To see how the mechanism works, consider a negotiation context with $\mathcal{A} = \{A, B, C\}$. Facing the problem of choosing a refinement for the current agreement set, agent A is trying to guess the values of agents B and C's preference functions f_B and f_C.

Like most learning methods, the first stage is feature selection. In this stage, the domain $\mathcal{D}^*$ is partitioned into a number of subsets $\{E_i\}$, where each subset corresponds to a region in the feature space. The values of $f_B(d)$ with d chosen randomly from E_i then define a random variable X_{B,E_i} on the sample space E_i. Given a point $d \in E_i$, the estimation of $f_B(d)$ is characterised by $P(f_B(d) = x | d \in E_i)$ which is the probability density function of X_{B,E_i}.

If we know that a refinement choice δ is a subset of E_i, we can proceed to approximate the function $F_{other,A}(\delta) = \overline{f}_B(\delta) + \overline{f}_C(\delta)$ by a random variable X_{E_i}, the sum of two random variables X_{B,E_i} and X_{C,E_i}, with the mean $\overline{X}_{E_i} = \overline{X}_{B,E_i} + \overline{X}_{C,E_i}$ and the standard deviation $\sigma^2(X_{E_i}) = \sigma^2(X_{B,E_i}) + \sigma^2(X_{C,E_i})$. For the purpose of predicting $F_{other,A}(\delta)$, agent A can use its expected value $\overline{X}_{E_i}$ with $\sigma(X_{E_i})$ as the prediction expected error [2].

To minimise the prediction expected error, the partition $\{E_i\}$ should be formed so that given an agent B, its preference values for the agreements in E_i is uniform (e.g. $\sigma(X_{E_i})$ is small). This partition is similar to the organizational structure for storing cases in the case-based reasoning approach (Sycara 1989). The formation of such a partition largely depends on the domain structure and knowledge. In the meeting scheduling domain, we choose to partition $\mathcal{D}^*$ (which is the time line) into periodic intervals such as all Monday mornings, Monday afternoons, Tuesday mornings, etc. Since the users tend to have appointments that

[2] A further conjecture based on the central limit theorem is that, when there are many agents participating in the negotiation context and when their preferences are statistically independent, the probability density function of X_{B_i} would be approximately gaussian.

happen on a regular basis, these periodic intervals can yield good predictive value.

If A has to choose among two refinement choices δ^1 and δ^2, the agent will use the following steps to decide which refinement choice to take:

- Identify the sample spaces that δ^i belongs to. Assume that $\delta^i \subset E_i$.

- From the samples of f_B and f_C accumulated from A's previous exchanges of preferences with B and C, calculate the average of $f_B(d)$ and $f_C(d)$ with $d \in E_i$. The results give the estimates of $\overline{X}_{B,E_i}$ and $\overline{X}_{C,E_i}$ respectively. If the distribution of X_{B,E_i} (or $X_{C,Ei}$) changes over time, a better estimation can be obtained if A only remembers and averages the most recent m values of $f_B(d)$, $(d \in E_i)$ for some positive integer m. We have:

$$F_A(\delta^i) = \overline{f}_A(\delta^i) + F_{other,A}$$
$$\approx \overline{f}_A(\delta^i) + \overline{X}_{B,E_i} + \overline{X}_{C,E_i} = F_A^{est}(\delta^i)$$

- Choose δ^i such that $F_A^{est}(\delta^i)$ is maximum.

Generally, for an arbitrary number of agents in the group, the function used by A in evaluating its refinement choices F_A^{est} is given by:

$$F_A^{est}(\delta) = \overline{f}_A(\delta) + \sum_{O \in \mathcal{A}-A} \overline{X}_{O,E_\delta}$$

where $E_\delta \supset \delta$.

From A's point of view, F_A^{est} is the expected value of the group's preference function F_A. The learning mechanism involves incrementally updating the function F_A^{est} when new data is available. To incorporate learning into the negotiation scheme, instead of using the usual function $\overline{f}_A(\delta)$ to evaluate A's refinement choices, the new function F_A^{est} is used. Since F_A^{est} includes the pattern of other agents' preferences, it can facilitate A and other learning agents in making better coordinated decisions.

Benefits of learning

Evaluation of the hybrid method requires the consideration of many factors, such as how often the agents need to conduct new negotiations and if there are any patterns to individual agent's preferences. In this section, we present preliminary results of applying the proposed hybrid method to solve the meeting scheduling problem.

The distributed meeting scheduling domain

We chose the distributed meeting scheduling domain as a testbed for the performance of the learning agents. In distributed meeting scheduling, the agents are the managers of the users' personal schedules. In a typical meeting scheduling situation, given an allowed time-window, a group of agents have to decide on a common slot within the given time-window as their agreed meeting time. Meanwhile, each member of the group has different preferences of what the desired slot should be and no agent has complete information about other agents' preferences. The problem is further complicated by the desired property to preserve the privacy of the personal schedules. This places an upper bound on the amount of information that can be exchanged among the group of agents.

A single meeting scheduling scenario involves a set of participating agents, a time window (W), the duration for the meeting being scheduled (l), and for each agent A a set of existing appointments ($App_A = \{app_i\}$) such that $\forall i \neq j, app_i \cap app_j = \emptyset$. For each appointment app we use $cost(app)$ to denote its cancellation cost. The continuous timeline is discretised and modelled by the set $Time = \{t_0 + i \mid i = 0, 1, \ldots\}$. Such a meeting scheduling scenario constitutes a negotiation context in which:

- The set of all agents $\mathcal{A}$ are the set of agents participating in the meeting scheduling.

- The set of all possible agreements $\mathcal{D}$ is derived from W and l as $\mathcal{D} = \{t \in Time \mid [t, t+l] \subseteq W\}$.

- For each agent A and a possible agreement $t \in \mathcal{D}$, the preference of A for t is

$$f_A(t) = \sum_{\substack{app \in App_A \\ app \cap [t, t+l] \neq \emptyset}} -cost(app) \qquad (1)$$

The domain of all negotiation contexts $\mathcal{D}^*$ becomes the timeline itself $\mathcal{D}^* = Time$. For the learning mechanism, we partition $Time$ into periodic intervals such as *morning*, *afternoon* (daily intervals) or *monday morning*, *monday afternoon* (weekly intervals). We choose this partition since the preferences of the agents also tend to have daily and weekly periods.

Preliminary analysis

Table 1 compares the expected performance of three refinement selection methods: *don't-ask* (choose the refinement to maximise own preference), *ask-first* (query other agents' preferences first before choosing a refinement), and *learning* (as described above). We assume

Refinement selection method	Function maximized	Prior-decision messages Querying others	Post-decision messages Resolving conflict
Ask-first	F_A	$log(L)n^2$	0
Don't-ask	f_A	0	$O(log(L)n^2)$
Learning	Expected value of F_A	0	$O(log(L)n^2)$

Table 1: Comparison between refinement selection methods

that the agents are using a binary tree as their agreement tree. The performance of each method is measured in terms of the total number of messages exchanged among the set of agents in one negotiation context. Here n denotes the number of participating agents and L is the number of possible agreements. The numbers show that the ask-first selection method always incurs a number of messages of $(log(L)n^2)$ order of magnitude, which is the expected performance of the *don't-ask* and *learning* selection method in the worst case. The trade-off exists, however, since the former method always guarantees to find the best optimal solution while the latter two do not.

Further, it is interesting to evaluate the relative performance of the agents using *don't-ask* selection and those augmented with a learning component. Since learning agents are more aware of other agents' preference functions, they can be better coordinated in selecting a refinement choice even without any prior-decision communication. Experiments with these two types of selection methods are presented in the next subsection.

Experiments

Our preliminary set of experiments involve two agents implemented in Dynaclips 3.1/Clips 6.0 running under SunOS operating system. The agents can run with or without the learning module. The aim of the experiment is to collect initial data confirming the benefits of the agents running with the learning module as opposed to those running without learning.

We model the timeline as a set of discrete points 30 minutes apart. Each day, and for a period of 20 days, the agents have to schedule a meeting with the duration of 2 units and within the time-window [8,16.30]. The possible agreement set $\mathcal{D}$ with its tree structure is shown in figure 1. The agents' preferences are periodic, with the period of 1 day. Noise can be added to the preference functions and this is interpreted as new non-periodic appointments, or the cancellation of existing periodic appointments.

The results of the experiment shown in figure 2 demonstrate that learning agents perform relatively superior when compared to the agents running without the learning module. The difference in performance, however, is reduced as the level of noise is increased.

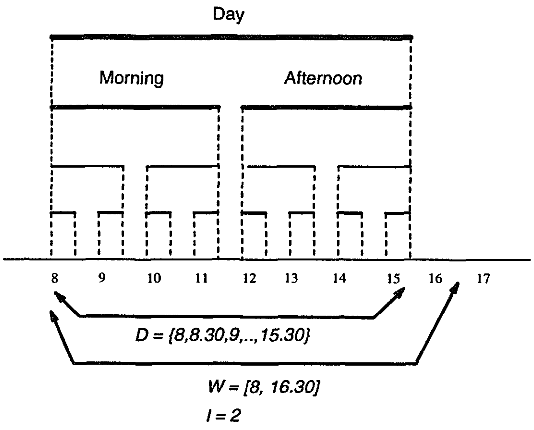

Figure 1: The domain $\mathcal{D}$ and its tree structure

This agrees with common sense as learning method would only show its benefits if the agents' preferences are periodic and can be learned. Also, the more often the non-learning agents are in conflict, the relatively better are the learning agents. This is because the learning mechanism works by learning from previous conflicts to prevent the same type of conflict from occurring in the future.

Discussion and Conclusions

In this paper, we have presented a method to incorporate a learning component into the negotiating agent architecture. This gives the agents the ability to learn about other agents' preferences from the interactions during their past negotiations. With the knowledge learned, the experienced agents are able to make better coordinated decisions in the future negotiations with their peers, thus improving the performance of the system over time. This illustrates that learning techniques can be used as an alternative knowledge acquisition to complement direct querying in negotiation. Although not designed to replace direct queries, the ability to complement direct queries with learning can be useful when communication costs are high, or when high level of inter-agent communication is not desirable (e.g. to preserve individual privacy).

Such a technique proves to be quite useful in the distributed meeting scheduling domain. In this do-

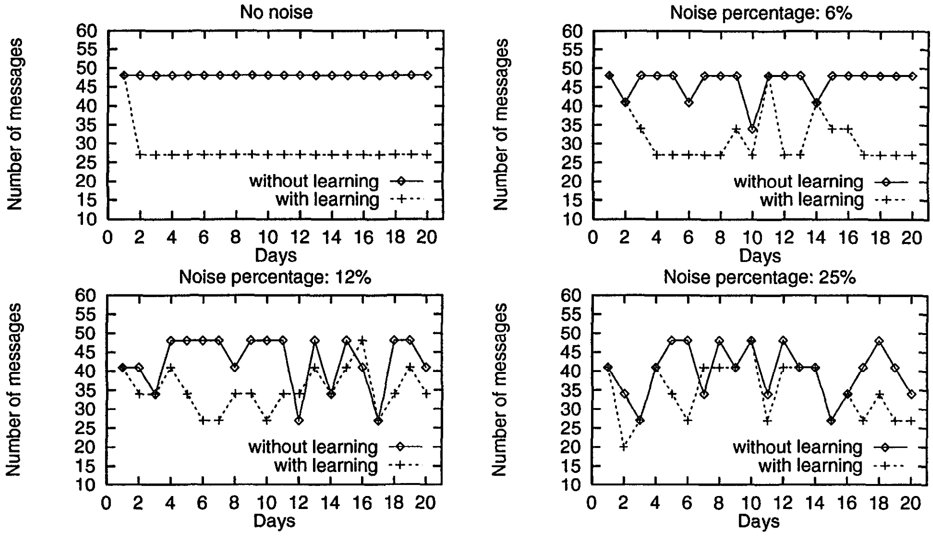

Figure 2: Relative performance of ordinary agents and learning agents

main, the agents' preferences tend to be repetitive with a fixed period; thus the learning mechanism can be simple and yet gives positive results. When there are a large number of agents involved, a saving in the amount of communication can save useful system resources required by the schedulers. Furthermore, it also preserves the privacy of the individual schedules.

The work presented here can be extended in a number of different directions. Firstly, based on the results of our initial experiments, we are planning to carry out more experiments to investigate the behaviour of the learning agents when there are a large number of agents and when the groups of agents are formed dynamically. Secondly, to evaluate the benefits of learning to its full extent, it is necessary to develop a common framework in which the accuracy of learned knowledge and the cost of communication can be examined together.

References

Bond, A. H., and Gasser, L., eds. 1988. *Distributed Artificial Intelligence*. Morgan Kaufman.

Bui, H. H.; Venkatesh, S.; and Kieronska, D. 1995. A multi-agent incremental negotiation scheme for meetings scheduling. *In proceedings ANZIIS-95, the Third Australian and New Zealand Conference on Intelligent Information Systems.*

Conry, S. E.; Meyer, R. A.; and Lesser, V. R. 1988. Multistage negotiation in distributed planning. In Bond and Gasser (1988), 367–384.

Laasri, B.; Laasri, H.; Lander, S.; and Lesser, V. 1992. A generic model for intelligent negotiating agents. *International Journal on Intelligent Cooperative Information Systems* 1(2):219–317.

Sen, S., and Sekaran, M. 1995. Multiagent coordination with learning classifier systems. In *Proceedings of the IJCAI-95 workshop on Adaptation and Learning in Multiagent Systems.*

Sen, S.; Sekaran, M.; and Hale, J. 1994. Learning to coordinate without sharing information. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-94)*, 426–431.

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

Sycara, K. 1989. Multiagent compromise via negotiation. In Gasser, L., and Huhns, M. N., eds., *Distributed Artificial Intelligence*, volume 2, 119–137. Morgan Kaufman.

Woodridge, M. J., and Jennings, N. R. 1994. Formalizing the cooperative problem solving process. In *Proceedings of the 13th International Workshop on Distributed Artificial Intelligence (IWDAI-94)*, 403–417.

Woodridge, M. J., and Jennings, N. R. 1995. Agent theories, architectures and languages: a survey. In Woodldridge, M. J., and Jennings, N. R., eds., *Intelligent Agents, ECAI-94 Workshop on Agent Theories, Architectures, and Languages*, number 890 in Lecture Notes in Artificial Intelligence, 1–39.