Augmenting Multi-Agent Environment Descriptions with a Normative Infrastructure*

Fabio Y. Okuyama[1], **Rafael H. Bordini**[2], **Antônio Carlos da Rocha Costa**[3]

[1]PPGC – Universidade Federal do Rio Grande do Sul
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

[2]Department of Computer Science, University of Durham
Durham DH1 3LE, U.K.

[3]Escola de Informática/PPGINF – UCPel
96.010-000 Pelotas, RS, Brazil.

okuyama@inf.ufrgs.br, R.Bordini@durham.ac.uk, rocha@atlas.ucpel.tche.br

Abstract. *This paper presents an approach for the development of environments for situated multi-agent systems. The approach includes features that allow designers to associate organisational structures to the environment. Spatially distributing the normative information over the environment is a natural way to simplify the definition of organisational structures and the development of large-scale multi-agent systems. By distributing the normative information in different spatial locations, we allow agents to directly access the relevant information needed in each environmental context.*

1. Introduction

Multi-agent systems (MAS) are usually composed of agents, an environment, organisational structures, and the interactions among these components. Organisational structures for multi-agent systems are usually defined independently of spatial structures. Therefore, when the MAS is situated in an environment, there is a conceptual gap between the environment and organisational structures.

We have been working on developing a platform for multi-agent based simulation called MAS-SOC presented in [Bordini et al. 2005a]. We have developed an environment description language presented in [Okuyama et al. 2004] and extended in [Okuyama et al. 2006] in order to associate to the environment an normative infrastructure that would link the spatial representation to the organisational structures.

In summary, the extensions support situated norms and leaves the necessary room for the inclusion of group structures that are spatially situated within a (simulated) physical environment. This is done using two means: first, a *normative infrastructure*, which is the structure that allows the distribution of normative information over a spatial environment; and second, a normative principle for *situated norms*, conceived as a special form of conditional rule, where an explicit condition on an agent's perception of a normative object appears: 'When playing the relevant role and being physically situated within the confines referred to by a situated norm $\mathcal{N}$ expressed in a normative object previously

*This work is partially supported by CNPq, CAPES and FAPERGS.

perceived, the agent is expected to reason about following norm $\mathcal{N}$; otherwise, it is exempted from reasoning about it'. We can therefore model things such as a sign saying that students are not allowed beyond the library desk, while staff members staff may go through.

2. The MAS-SOC Platform

One of the main goals of the MAS-SOC simulation platform (**M**ulti-**A**gent **S**imulations for the **SOC**ial Sciences) is to provide a framework for the creation of agent-based simulations which do not require too much experience in programming from users, yet allowing the use of state-of-the-art agent technologies. In particular, it should allow for the design and implementation of simulations with *cognitive* agents.

In our approach, an agent's individual reasoning is specified in the of AgentSpeak [Rao 1996] language. As interpreted by ***Jason*** [Bordini et al. 2005b] it offers extended features, among other things, for speech-act based agent communication, and there is ongoing work to allow for ontologies as part of an AgentSpeak agent's belief base. ***Jason*** is an Open Source platform based on Java (available at `http://jason.sf.net`).

The environments where agents are situated are specified in ELMS, a language we have designed for the description of multi-agent environments [Okuyama et al. 2004]. With the extensions to allow the definition of normative infrastructures, we intend to provide the means to link the environment to an organisation. It is important to note that the extensions are not an approach to describe organisations, it only links an organisation to the environment. The normative objects, normative places, and norm supervisors are meant to fill the gap between the environment and organisations, being a reflection of an organisation over the environment. For more details on MAS-SOC, refer to [Bordini et al. 2005a].

3. Modelling Multi-Agent Environments

According to [Wooldridge 1999], agents are computational systems situated in some environment, and are capable of autonomous action in this environment in order to meet their design objectives. Agents in a MAS interact with the environment where they are situated and interact with each other, typically through the shared environment. Therefore, the environment has an important role in a MAS, whether the environment is the Internet, the real world, or some simulated environment. As presented in [Okuyama et al. 2004], we developed a language to describe environments and the means to execute the simulated environments.

We understand as environment modelling, the modelling of external aspects that an agent needs as input to its reasoning and for deciding on its course of action. In a multi-agent scenario, how one agent perceives another is an important issue. Thus, modelling explicitly the agent's "body" should also be included in environment modelling. Further, it is necessary to model explicitly the physical actions and perception capabilities that the agents are allowed to perform in a given environment.

The language we designed for modelling environments is called ELMS (**E**nvironment Description **L**anguage for **M**ulti-Agent **S**imulation). Below we briefly review how a physical environment is described using this language.

3.1. Agents *Bodies* Modelling

In the environment description, the agents are defined through the definition of the *agent body*, agent sensorial capabilities, and agent effective capacities.

We call "agent body" the agent's characteristics that are perceptible to other agents. Using the ELMS language, classes of agent "bodies" are defined by a set of properties that characterise the agent and are perceptible to other agents. Such properties are represented as *string*, *integer*, *float*, and *boolean* values. Each *body* is associated with a set of actions that the agent can perform and of environment properties that the agent can perceive.

Each class of agent "body" have its sensorial capabilities (perception) defined. Each sensorial capability is used to specify which environment properties will be perceptible to each agent that has a "body" with such capacity. It determines the environmental properties that will be sent to the agent and the specific circumstances where they are possible (e.g., an agent may be able to see in a radius of 2 cells, but only if there is nothing obstructing its vision).

A perception type definition is formed by a name, an optional list of preconditions, and a list of properties that are perceptible. The listed properties can be any of those associated with the definitions of resources, agents, cells of the grid, or simulation control variables. If all the preconditions (e.g., whether the agent is located on a specific position of the grid) are all satisfied, then the values of those properties will be made available to the agent's reasoner as the result of its perception of the environment.

Finally, the agent effective capacities (actions) define the possible ways of changing environment that are made available to each class of agent "body". Each definition of an action determines the environmental changes that such action can make over the environment by an agent that has a "body" with such capacities. These changes are defined as assignments of values to the attributes of environments[1]. The production (instantiation) of previously defined resources (objects), and the consumption (deletion) of existing instances may also be part of an action description. Also, the conditions under which an action can be performed should be specified.

3.2. Environment Modelling

The environment is modelled by the definition of the objects and resources available in the environment, the reactions that may occur with such objects, and the spatial representation of the environment. Although objects and resources can be conceptually different, both are represented by the same structure in ELMS. Using the ELMS language, we can define the classes of objects and resources that are present in the environment. As mentioned before, agents interact with objects through the actions performed in the environment. Objects are defined by a set of properties that are relevant to the modelling and could be perceived by agents. A definition of a resource class includes the class name, a list of attributes, and a set of reactions.

Each instance of an object can "react", under specific circumstances, responding to actions performed by the agents in the environment. For each reaction, its name, a list of preconditions, and a sequence of commands are defined. In the same way as the

[1] Note that properties of agent bodies are also properties of the environment.

actions, the sequence of commands can be composed by assignment of values to properties, the creation of previously defined object instances, and the deletion of existing object instances.

The spatial representation of the environment can be done by a grid or a graph, according to the needs of the simulation or design preferences. For simulations where a precise positioning of agents and objects is relevant, a grid may be preferred. Otherwise, if the logical aspects of the spatial representation is more relevant over a precise positioning, a graph might be more suitable. Both spatial representation types are optional and not exclusive: if required by some particular simulation, both can be used at same time.

Using a grid, the space is divided into cells forming a grid that represents the spatial structure of the environment, both in 2 or 3 dimensions. As for resources, each cell can have reactions associated with them.

For the graph, the space is divided into nodes which are connected by links. As the cells of the grid, each node of the graph represents a spatial location, where reactions may occur in response to some action. The links may have weights or values associated into it, according to the needs of the project.

4. Spatially Distributed Normative Infrastructure

We often find ourselves in environments that have some objects aimed at informing agents about norms, give some advice, or warn about potential dangers. For example, a poster fixed on a wall in a library asking for "silence" is an object in the environment, but also informs about a norm that should be respected within that space. Another example are traffic signs, which give advice about directions or regulate priorities in crossings. The existence of such signs, which we call *normative objects*, implies the existence of a regulating code in such context, and such code is formed by what we call *situated norms*. We are interested in the use of such abstractions in the context of computational systems, particularly multi-agent systems.

In the examples above, the norms are only meant to be followed within certain boundaries of space or time, and lose their effect completely if those space and time restrictions are not met, which is the initial motivation for situated norms. Another important advantage of modelling some norms as situated norms is the fact that the spatial context where the norm is to be followed is immediately determined. Thus, the norm can be *pre-compiled* to its situated form, making it easier for the agents to operationalise the norm, and also facilitating the verification of norm compliance.

In this section, we present the extensions to ELMS that are meant to provide an infrastructure allowing the distribution of normative information within an environment. Such infrastructure is aimed at being a connection point between the environment and organisational structures, improving significantly the possibilities of our simulation platform. In particular, we present here a set of concepts for a spatially distributed infrastructure formed by *normative objects*, *normative places*, and *norm supervisors*. This infrastructure, in our view, facilitates the modelling and simulation of various real-world situation, which might be useful for social simulation as well as testing large-scale MAS that are to operate in real-world environments.

4.1. Normative Places

As described in previous sections, we have developed a language to describe environments for situated MAS. The description, based on the concepts of agent bodies, objects, and an optional grid, did not offer the means to define the notion of a "place", i.e., a set of cells where a set of connected activities are done or where groups or agent settings are related; we refer to them as *normative places*. They are effectively used to represent the physical spaces where organisations take place; that is, a normative place is the spatial scope of a particular organisation, and consequently the norms related to the activities of such organisation.

A *normative place* is defined simply by its name and the set of cells that constitute that place. For each normative place, there is a set of organisational roles available at such place. A normative place might have intersections with other normative places, or may even be contained by other normative places (e.g., a school may contain a library).

The roles available to each normative place are strictly relative to the function being performed in such place; such roles might have no direct relation to existing roles in other organisations. For example, in a normative place *street*, an agent with a car will be a *driver*, and another one without a car will be a *pedestrian*. In such case, from the point of view of the design of the environment and normative infrastructure, the agents are simply moving from one place to another, indenpendently of any other goal that an agent playing that role may have.

4.2. Normative Objects

To understand the notion of *normative object*, consider the posters one typically sees in public places such as libraries or bars saying "Please be quiet" or "No smoking in this area". Human societies often resort to this mechanism for decentralising the burden of regulating social behaviour; people then adopt such norms whenever they have visual access to such posters. This should be equally efficient for computational systems because it avoids the need to provide a complete, exhaustive representation of all social norms in a single public structure, known to all agents, as it is usually the case in approaches to agent organisations.

The normative information in a *normative object* is "read" by an agent through its sensing/perception abilities under specific individual conditions; that is, an agent can read a specific rule if it has the specific ability to perceive that type of object under its current circumstances. In the most typical case, the condition is simply being physically close to the object.

A definition of a normative object contains the norm itself and also meta-information. Normative objects can be defined before the simulation starts in a "norms definition file" or during the simulation, in both cases by the definition of the *type* of the information contained, which institution/group/agent issued the norm, the norm itself, the placement of the object, the condition for the object to be "read", and an *id* for edition or deletion. For the definition of the placement of the normative object, each normative object may be associated to a normative place of the environment.

It is worth noting that norm-abiding behaviour is not related just to the existence of a normative object at some place. Beyond the existence of such object, it is necessary

for the agent to perceive the normative object, and autonomous agents will also reason about whether to follow or not the norm stated by the normative object, after perceiving it.

4.3. Pre-compilation of Norms

Normative objects are not meant to be simply means to spread general norms. The norms informed through normative objects are supposed to be contextualised (as determined by the simulation designer), incorporating information which is specific to the normative place where it is relevant.

Since the spatial context of the norm is limited and determined by the normative places, a generic norm can then be "pre-compiled" with such information, so that it becomes less abstract. This process is meant to make it easier for agents to operationalise the norm. Other advantages of having less abstract norms are that the verification of norm compliance is facilitated and that they can reduce the misinterpretations that could happen with abstract, non-contextualised norms.

For example a norm that says "Be kind to the elderly", may be quite hard to operationalise and verify, in general. However, in a fixed spatial context such as a bus or train, with the norm contextualised as "Give up your seat for the elderly", or in a street crossing, with the norm contextualised as "Help elderly people to cross the street", the norm would be much more easily interpreted by the agents, and similarly much more easily verified by any norm compliance checking mechanism.

4.4. Norm Supervisors

Norm supervisor agents are a special class of agents that monitor the agents' compliance to norms within a MAS. The norm supervisor agents may be "system agents" specifically designed for such function, or autonomous agents (which are part of the system/simulation) with such functionality enabled.

Since the agents are free to reason about abiding or not to a norm stated in a normative object, besides having in general limited perception capabilities with respect to normative objects, there is also a need to monitor the behaviour of those agents. In order to be able to act as a norm supervisor, an agent may need extra information and perhaps extra abilities. For this reason, in the extended version of ELMS, it is possible to define an agent as a *norm supervisor*, which will enable it to receive relevant normative information as well as information about the actions being done by other agents at a given normative place.

Through the use of simple rules, a norm supervisor can check the compliance of agents to norms, and then according to the capabilities given to it, it may interrupt the course of action of another agent, issue penalties or appropriate "punishment", or simply report the breaching of the norm to some centralised agency or the user. It is important to note that norm supervisors are not meant only to try and stop rule breaking by other agents. The simulation designer may enable norm supervision in a agent just to help it achieving its goal, to use such information to monitor the simulation, or as an input to a reputation system, among other things.

For instance, according to [Conte and Castelfranchi 1995], an agent may be motivated to verify the compliance to norms by other agents in order to reassure itself

that the costs of norm adherence is being paid by the other agents too. A norm abiding agent will want that all the others addressees of the norms follow it too, otherwise the norm adhering behaviour may become some sort of competitive disadvantage. In [Conte and Castelfranchi 1995], the authors refer to agents with such behaviour as *norm defenders*.

5. Modelling Environments using ELMS

As the MAS-SOC platform does not enforce a particular agent-oriented software engineering methodology, designers can use the one they prefer. It is possible to model a MAS that will have an ELMS environment using any approach: starting from the system organisation (top-down), or starting from the agents and their interactions (bottom-up).

In both approaches, the modelling of the organisational structures and the agents' reasoning need fine tuning to achieve the desired results. To have a stable point on which to base the fine tuning of the agents' reasoning or the organisational model, we suggest the use of an explicitly defined environment description written in the ELMS language and the notions presented in the Section 3. The environment is an important part of a multi-agent system, and although it can be very dynamic, in regards to design it is usually the most "stable" part of the system.

We suggest that the MAS design should start with the environment definition, followed by the definition of the normative places. The environment modelling should proceed as follows:

1. Based on the requirements of the simulation, a choice between graph- or grid-based spatial representation can be made (see section 3.1).
2. Definition of which kinds of action each class of agents is able to perform in the environment. Actions typically produce effects on objects of the environment or other agents.
3. Based on the changes that the agents' effective capabilities are able to make in the environment, and the objectives of the simulation, the size and granularity of the grid or the nodes of the graph can be determined. For example, how many cells or nodes an agent can move within one action or simulation cycle, and in how many simulation cycles the agent would be able to traverse the simulated space.
4. Based on the granularity and size of the spatial environment, the sensorial capabilities of the agents can be modelled, determining for example in which range an agent can detect other agents and objects.
5. Based on an agent's sensorial capabilities and on its typical activities, it should be possible to define which attributes of that agent is important to declare as accessible to other agents. For example, if agents identify each other's role by the colour of their uniform, the "agent body" should have an attribute that represent the colour of the agent's uniform.
6. The types of objects or resources present in the environment should also be modelled based on which attributes will be perceptible by the agents and which actions can affect them.
7. Finally, instances of the agent and object classes should be placed in the environment, determining its initial state.

The definition of the environment should be followed by the definition of the normative infrastructure, as follows:

1. As the resource and agent instances are placed in the environment, the activities for each spatial place can be defined.
2. By grouping the neighbouring cells or nodes where similar activities are done, the normative places and its extensions can be defined.
3. Based on the activities done in a normative place, agent roles can be defined (see section 4.1).
4. By instantiating normative places into sets of cells or nodes, the *normative places* are created.
5. Then, based on the set of activities that can be performed in each type of normative place, the norms that are relevant to that type of place can be defined.
6. Finally, the types of *normative objects* can be defined and instantiated in the normative places, defining the locations where situated norms can be perceived.

Using the environment as the basis (i.e., using the information contained in the environment model), the agents' reasoning capabilities can then be defined so as to help agents achieve their goals as well as those of the groups to which they belong. Also, the detailed definitions of possible organisational structures can then be fine tuned in order to allow the overall system to achieve its (social) goals. In MAS-SOC, we use AgentSpeak to define the practical reasoning for each agent; in particular, we use the extended version of AgentSpeak as interpreted by ***Jason***; for details, see [Bordini et al. 2005b].

6. Related Work

The notion of artifacts [Viroli et al. 2005] and coordination artifacts [Omicini et al. 2004] resembles, in some aspects, our notion of *normative objects*. As defined in [Omicini et al. 2004], coordination artifacts are abstractions meant to improve the automation of coordination activities, being the building blocks to create effective shared collaborative working environments. They are defined as runtime abstractions that encapsulate and provide a coordination service to the agents. Artifacts [Viroli et al. 2005] were presented as a generalisation of coordination artifacts. Artifacts are an abstraction to represent tools, services, objects, and entities in a multi-agent environment.

As building blocks for environment modelling, artifacts encapsulate the features of the environment as services to be used by the agents. The main objective of a coordination artifacts is to be used as an abstraction of an environmental coordination service provided to the agents. However, coordination artifacts express normative rules only implicitly, through their practical effects on the actions of the agents, and so their normative impact does not require any normative reasoning on the part of the agents. In our work, rather than having a general notion of objects that by their (physical) properties facilitate coordination, *normative objects* are used specifically to store *symbolic* information that can be interpreted by agents, so that they can become aware of norms that should be followed within a well-defined location.

Our choice in regards to normative objects has the advantage of keeping open the possibility of agent autonomy, as suggested in [Castelfranchi et al. 1999]. Agents are, in principle, able to decide whether to follow the norms or not when trying to pursue their goals. This is something that is not possible if an agent's action can only happen if it is in accordance to implicit norms enforced by coordination mechanisms.

Another important difference is that normative objects are spatially distributed over a physical environment, with a spatial scope where they apply, and closely tied to the

part of the organisation that is physically located in that space. While the objective of the notion of coordination artifacts is to remove the burden of coordination from the agents, our work tries to simplify the way designers can guide the behaviour of each individual agent as they move around an environment where organisations are spatially located; this allows agents to adapt the way they behave in different social contexts.

In [Ferber et al. 2004], the authors present the AGRE model, an extension to the previous AGR model. Those latest extensions allow the definition of structures that represents the physical space. The approach defines organisational structures (i.e., groups) and the physical structures (i.e., areas) as "specialisations" of a generic space. We find, however, that in their approach, the social structures are not contextualised in the space as they are in our work, leaving the social and physical structures rather unrelated.

7. Conclusions

We believe that an explicit environment description is an important part of a MAS because it is a stable point from where the agent reasoning and the organisational structures can be fine-tuned so as to facilitate the development of agents and organisations that can achieve their goals. This paper presents the extentions made to our environment description language, associating the environment with a normative infrastructure and organisational aspects of the multi-agent system. It is important to note that our work is not an approach for modelling the organisational dimension of a MAS. With the definition of *normative places*, where group structures would be situated, we intend to fill a conceptual gap between the usual ways in which organisations and physical environments are modelled. We believe that the notion of a *spatially distributed normative infrastructure* that we have introduced in our work is a suitable approach to connecting definitions of organisations and definitions of environments. Additionally, distributing the organisational/normative information can facilitate the modelling of large organisations.

By distributing the normative information in the environment, it is possible to partition the environment in a functional way, thus helping the structured definition of large simulations, norms being associated only with the places where they are meant to be followed. It is also more efficient (by taking advantage of the natural distribution of certain environments) to have norms spread in an environment than having them in a centralised repository made available to the whole society, as it is usually the case. Another advantage of having the information distributed is the possibility of the *pre-compilation* of the norms with the spatial context information, which we refer to as *situated norms*. This makes it much easier to make norms operational, both for following norms and checking norm compliance.

As future work, we plan to make possible such association of our environment descriptions with existing approaches to modelling agent organisations, such as $\mathcal{M}$OISE$^+$ [Hübner et al. 2002], OperA/OMNI [Vázquez-Salceda et al. 2005], and approaches based on electronic institutions [Esteva et al. 2004]. Also, there are interesting consequences of distributing the normative information that may be subject of future work. Norms being spread over the environment may result in different "local" reputations within a single environment, leading to the notion of *locality of reputation*. It is interesting to note that, being conditioned on the possibility of checking the existence of a normative object, the normative reasoning required from agents that deal with normative

objects is necessarily of a non-monotonic nature; the experience of programming such reasoning in AgentSpeak is also something we also plan to do in the future.

References

Bordini, R. H., Costa, A. C. d. R., Hübner, J. F., Moreira, l. F., Okuyama, F. Y., and Vieira, R. (2005a). MAS-SOC: a social simulation platform based on agent-oriented programming. *Journal of Artificial Societies and Social Simulation*, 8(3).

Bordini, R. H., Hübner, J. F., and Vieira, R. (2005b). ***Jason*** and the Golden Fleece of agent-oriented programming. In R. H. Bordini *et al.*, editors, *Multi-Agent Programming: Languages, Platforms and Applications*, Springer-Verlag.

Castelfranchi, C., Dignum, F., Jonker, C. M., and Treur, J. (1999). Deliberative normative agents: Principles and architecture. In *6th International Workshop on Intelligent Agents VI, Agent Theories, Architectures, and Languages (ATAL)*, Lecture Notes In Computer Science, Vol. 1757, pages 364–378, London. Springer-Verlag.

Conte, R. and Castelfranchi, C. (1995). *Cognitive and Social Action*. UCL Press, London.

Esteva, M., Rosell, B., Rodríguez-Aguilar, J. A., and Arcos, J. L. (2004). Ameli: An agent-based middleware for electronic institutions. In *AAMAS*, pages 236–243. IEEE Computer Society.

Ferber, J., Michel, F., and Báez-Barranco, J.-A. (2004). AGRE: Integrating environments with organizations. In Weyns, D. *et al.*, editors, *Environments for Multi-Agent Systems, First International Workshop, E4MAS 2004*, New York, NY, USA, July, 2004, volume 3374 of *LCNS Series*, pages 48–56. Springer.

Hübner, J. F., ao Sichman, J. S., and Boissier, O. (2002). $\mathcal{M}$OISE$^+$: Towards a structural, functional, and deontic model for MAS organization. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'2002), Bologna, Italy*.

Okuyama, F. Y., Bordini, R. H., and da Rocha Costa, A. C. (2006). Spatially Distributed Normative Objects. In G. Boella *et al.*, editors, *Proc. of the Intl. Workshop on Coordination, Organization, Institutions and Norms in Agent Systems (COIN)*, held with ECAI, Riva del Garda, Italy, 2006.

Okuyama, F. Y., Bordini, R. H., and da Rocha Costa, A. C. (2004). ELMS: An environment description language for multi-agent simulation. In *Weyns, D., Parunak, H. V. D., and Michel, F., editors, Environments for Multi-Agent Systems, First International Workshop, E4MAS 2004, New York, NY, USA, July, 2004, volume 3374 of* LCNS Series*, pages 91–108*. Springer.

Omicini, A., Ricci, A., Viroli, M., Castelfranchi, C., and Tummolini, L. (2004). Coordination artifacts: Environment-based coordination for intelligent agents. In *AAMAS'04*.

Rao, A. S. (1996). AgentSpeak(L): BDI agents speak out in a logical computable language. In Van de Velde, W. and Perram, J., editors, *Proceedings of the Seventh Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96), 22–25 January, Eindhoven, The Netherlands*, number 1038 in LNAI Series, pages 42–55, London. Springer-Verlag.

Vázquez-Salceda, J., Dignum, V., and Dignum, F. (2005). Organizing multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 11(3):307–360.

Viroli, M., Omicini, A., and Ricci, A. (2005). Engineering MAS environment with artifacts. In Weyns, D. *et al.*, editors, *2nd Intl. Workshop "Environments for Multi-Agent Systems" (E4MAS 2005)*, pages 62–77, AAMAS 2005, Utrecht, The Netherlands.

Wooldridge, M. (1999). Intelligent agents. In Weiß, G., editor, *Multiagent Systems—A Modern Approach to Distributed Artificial Intelligence*, pages 27–77. MIT Press, Cambridge, MA.