Intelligent Agents and Multi Agent Systems

A Tutorial presented at IEEE CEC 99. © Vasant Honavar

Washington, D.C., July 1999

Vasant Honavar
Artificial Intelligence Research Laboratory
Department of Computer Science
Iowa State University

honavar@cs.iastate.edu
http://www.cs.iastate.edu/~honavar/
http://www.cs.iastate.edu/~honavar/aigroup.html
http://www.grad-college.iastate.edu/bioinformatics
http://www.elsevier.com/locate/cogsys

Tutorial Overview

- Intelligent Agents: Agent definitions, agents versus programs, agent properties, agent taxonomies, brief discussion of agent designs, applications
- Mobile Agents: Motivation, definition, advantages and disadvantages, mobile agent platforms
- Multi-Agent Systems: distributed problemsolving, coordination and control, interagent negotiation
- Sample applications
- Open research areas

What is an Agent?

- An agent is any entity that can be viewed as *perceiving* its *environment* through *sensors* and *acting* upon its environment through *effectors* Russel and Norvig, 1995]
- Agent is a *persistent software* entity *dedicated* to a *specific purpose* [Smith, Cypher and Spohrer]
- Intelligent agents *continuously* perform three functions: *perception* of *dynamic conditions* in the *environment*; *reasoning* to *interpret* perceptions, *solve problems*, *draw inferences*, and *determine actions* [Hayes-Roth, 1995]
- *Intelligent* agents are *software* entities that *carry out* some set of operations *on behalf* of a user or another program with some degree of *independence* or *autonomy,* and in so doing, employ some *knowledge* or *representation* of the user's goals or desires [IBM White paper, 1995]
- *Autonomous* agents are *computational* systems that inhabit some *complex dynamic environment*, *sense* and *act autonomously* in this environment, and by doing so, realize a set of *goals* or tasks for which they are designed [Maes, 1995]

What is an Agent?

- Software agents are programs that *engage in dialogs* and *negotiate* and *coordinate* transfer of information [Coen, 1995]
- Autonomous agents are systems capable of *autonomous, purposeful* action in the *real world* [Brustolini, 1991]
- A hardware and/or software-based computer system that is: *autonomous, socially adept, reactive,* and *pro-active* [Wooldridge and Jennings, 1995]
- An autonomous agent is a system *situated within an environment* and *acts on it, over time*, *in pursut of its own agenda and so as to affect what it senses in the future* [Franklin and Graesser, 1996]
- The term agent is often used in database, operating systems, and networking literature to mean a proxy for a computation or a site - a transaction, a process, a network router, that could be used to interact with the underlying entity. Thus, an agent is a formal interface to an arbitrary system. For example, Simple Network management protocol (SNMP) defines certain kinds of agents upon which a network management application can be designed.

I’ll know an agent when I see one!

Precisely defining an agent, like precisely defining life, is difficult. Perhaps it is easier to recognize an agent when we see one. Or is it really?

Which of the following is an agent?

- A spell checker
- A payroll program
- An electronic mail program
- A thermostat
- A Unix daemon
- A knowledge-based expert system controlling a nuclear power plant
- An electronic stock trader
- An autonomous land vehicle
- A personal information assistant

Properties of agents

- reactive versus deliberative
- purposeful (pro-active, goal-driven)
- transient versus persistent
- autistic versus communicative, social
- rigid verus adaptive, flexible
- preprogrammed versus instructible versus learning
- collaborative versus competitive versus antagonistic
- rational (utility maximizing)
- autonomous versus controlled
- stationary versus mobile
- believable
- ethical
- self-replicating
- evolvable

Properties of Environments

- Observable: what aspects of the environment are accessible to the agent through its sensors?
- Controllable: to what extent can the agent control its environment, i.e., take actions that lead to desired environmental states?
- Predictable: Deterministic?, stochastic?, chaotic?
- Episodic versus nonepisodic
- Static versus dynamic
- Discrete versus continuous
- Markovian versus non Markovian
- Purposeful: Are there other agents in the environment?
- Open versus closed

Agents and their environments are inextricably coupled! Simple agents can often display apparently complex behaviors when they interact with sufficiently rich environments**!**

A personal information assistant's Environment

Property	**User**	**System**
Observable	Yes	Yes
Predictable	Partially	Partially
Controllable	No	Partially
Episodic	Yes	No
Static	No	No
Markovian	No	Maybe
Purposefuk	Yes	Maybe
Discrete	Maybe	Typically

Agent Taxonomies

Task based classification

- interface agents
- personal information assistants
- data mining
- mail filtering
- travel planning
- meeting scheduling
- information brokering etc.

Classification based on mode of behavior

- Reactive, Deliberative, Goal-based, Utility-based (rarional), creative
- Cooperative, competitive, antagonistic
- Mobile versus stationary
- Controlled versus autonomous
- Autistic versus communicative
- Deterministic versus nondeterministic
- Preprogrammed versus adaptive versus learning
- Self-replicating, evolving

Agent Taxonomies

Classification based on Computational capabilities

- Boolean functions
- Finite state automata
- Stack automata
- Bounded linear automata
- Turing machines
- Super Turing machines

Classification based on AI paradigm

- procedural versus declarative
- symbolic versus connectionist versus hybrid
- deterministic -- logical (monotonic, nonmonotonic) versus probabilistic
- explicit versus implicit representation
- inference - logical, probabilistic, analogical, inductive, abductive

Models of Agency

- **Rational agency: logical** (consistency of beliefs, suitability of actions given beliefs and intentions). Example: Knowledge-based inference systems, Truth or reason maintenance systems
- **Rational agency: economic**;Agent holds preferences over world states and selects actions that result in maximizing its preferences. The decisions may be made with complete knowledge, or partial uncertainty. Rational agents might negotiate deals among themselves.
- **Social agency**: Cooperation, competition, coordination; Agents might make social commitments with other agents and work to achieve common objectives
- **Interactive agency**: Agents might interact with eachother through intended or unintended interactions; Intended interactions involve communication among agents, e.g., by means of a shared language (syntax, semantics, pragmatics).
- **Adaptive Agency**: Agents learn by interacting with their environment (which might include other agents)
- **Evolving Agency**: While individual agents may or may not learn or adapt, self-replicating agent populations adapt to their environments through evolution

A closer look at Rational Agents

A Rational Agent selects actions that maximize *measurable success.*
Rationality is *not* omniscience.
Rationality depends on:

- The performance measure or the agent's utility function
- The agent's perceptual history (what the agent has seen)
- What the agent *knows* about its environment (the agent's knowledge)
- The actions that the agent *can* perform including those that gather useful information

An *ideal rational agent* must, for each percept sequence, execute an action that is expected to maximise its performance given its percept sequence and its knowledge.

In practice, we often have to settle for bounded rationality

Historical context

Artificial Intelligence started with the whole agent problem but quickly realized that it was too difficult. 1960-1990 witnessed a great deal of progress in various sub-areas - knowledge representation and inference, machine learning, vision, planning, robotics, etc. Time to attack the whole agent problem!

Advances in computer science - multi-tasking, communicating processes, distributed computing, modern interpreted languages (e.g., Java), real-time systems, commmunication networks, networked environments, have made it possible, at least in principle, the design, implementation, and deployment of agent-based systems.

Applications involving distributed databases, smart user interfaces, world-wide web, mobile computing, distributed design and manufacturing, information gathering, decision support (using heterogeneous distributed data and knowledge sources), open systems, collaborative computing, etc. that exploit the advances in AI and computer systems argue for intelligent agents and multi-agent systems

A brief look at Agent Designs

Agent design has been the focus of study in artificial intelligence over the past four decades.

Sample agent designs [Russell and Norvig, 1995]

- Table driven agents
- Reflex agents
- Problem solving agents
- Deliberative Agents
- Decision-theoretic or Rational agents
- Knolwedge based agents
- Planning agents
- Reactive agents
- Proactive agents
- Adaptive agents
- Learning agents

Approaches to agent design:

- direct programming
- machine learning
- evolutionary techniques

Agent Languages

- Agent Implementation Languages (e.g., Java, Lisp)
- Agent Communication and Coordination Languages (e.g., KQML)
- Languages for modeling and describing the laws or behaviors of the environment (including that of other agents)
- Languages for representing and reasoning with and about knowledge (including facts about the world, intentions and beliefs of other agents, etc.)
- Agent specification languages

Sample Application: Personal Information Assistants

Traditional Approach: Direct Manipulation:

- Requires the user to tediously initiate and execute each action even in cases where sequences of actions are better automated e.g., locating, retrieving, and extracting relevant information from distributed data collections. What is practical and possible for tens of items becomes unwieldy and impractical for thousands.
- Does not offer support for initiating actions on behalf of the user in response to situations that might arise
- Does not facilitate easy composition of basic actions and objects into complex action structures (e.g., action sequences)
- Software is typically written for generic tasks rather than to solve specific problems for specific users.
- No improvement in behavior despite repeated use in similar tasks.
- User has to grapple with the complexity and heterogeneity of distributed, heterogeneous information sources and computing devices

Sample Application: Personal Information Assistants

Agent based approach: Indirect Manipulation

- Agents can automate routine tasks of locating, retrieving, and processing information from heterogeneous distributed information sources.
- Agents can hide the heterogeneity and complexity of the underlying information sources
- Agents are told what to do, not how to do it. They can compose complex action sequences as needed to accomplish a specific task
- Adaptive agents allow customization of generic software (e.g., news readers, mail programs, web browsers) to the needs and interests of specific users
- Agents can learn by obersving users and interacting with them, and thereby improve their behavior
- Agents can potentially work around unforeseen problems and exploit unforeseen opportunities as they go about doing their tasks

Sample Application: Personal Information Assistants

Approaches to designing personal information assistants E.g., user-programmed email filtering agents

- User-programmed approach
  - User has too recognize the opportunity to employ an agent
  - User has to take the initiative to create an agent
  - User has to program the agent with adequate domain-specific knowledge
  - User has to maintain the agent's knowledge base over time
- Model or Knowledge based approach e.g., interface agents designed to help novices to use Unix
  - Agent is endowed with extensive domain-specific knowledge as well as knowledge about a specific user or groups of users
  - Agent uses its knowledge at run time to recognize opportunities to contribute to a user's
  - Impractical except in cases where large groups of users can be modeled sufficiently accurately by knowledge engineers
  - Users tend to distrust autonomous agents whose operation they don't understand

Sample Application: Personal Information Assistants

Approaches to designing personal information assistants

- Machine Learning Approach
  - Agents are programmed with general knowledge of the domain
  - Agents learn a model of the user by observing the user, by taking advice from the user or other agents, by receiving positive and negative feedback from the user, by using examples provided by the user, etc.
  - Presents a testbed for automated software synthesis in relatively narrow task domains

Examples:

- Meeting scheduling agents (CMU, MIT)
- Recommender agents for documents, music, films, news, entertainment (MIT, Stanford, ISU)
- Bioinformatics Information Agents (ISU)
- Distributed Network Monitoring Agents (ISU)

Experience: Even relatively simple representations (e.g., bag of words representation of text) and learning algorithms often work surprisingly well.

Application: Distributed Information Infrastructure

Information rich or open environments

- Consist of a large number of data and knowledge sources and services e.g., databases, simulations and virtual environments, sensor arrays, etc.
- Have components that are heterogeneous in terms of computing platforms, software environments, structure and content of information sources
- Have components that are autonomously owned and operated and can be added and removed from the system with little central control or coordination
- Call for approaches to information retrieval that do not require the user to know where the information being sought is stored, how it can be accessed, or when and in what form it will be available.
- Need flexible information retrieval, extraction, assimilation, organization, and data driven knowledge discovery tools
- Call for multiagent systems wherein autonomous and/or semiautonomous agents can negotiate and collaborate
- Need support for carrying out computation where the resoruces are available, and not where results are needed

Current and Potential Applications

- Proactive, Reactive, and Context Sensitive Information retrieval
- Decision support using distributed, heterogeneous data and knowledge sources (e.g., in healthcare, defense, collaborative scientific discovery, etc.)
- Distributed design and manufacturing in virtual enterprises
- Electronic Commerce
- Adaptive self-managing complex dynamic systems (e.g., large communication networks, power systems, transportation systems)
- Adaptive user interfaces
- Mobile and ubiquitous computing
- Mail and message handling
- Collaborative work environments
- System monitoring, intrusion detection, and countermeasures
- Knowledge Discovery from heterogeneous distributed data and knowledge source (e.g., genome databases, protein databanks, laboratories)

Communicating Applications

Realizing the potential of intelligent agents calls for communicating applications which unlike standalone applications that run on today's personal computers, effectively use public networks to locate, use, and interact with heterogeneous, geographically distributed, autonomous data and knowledge sources and agents (including humans).

A case for public networks as computing platforms

- Applications need to be able to physically distribute themselves: they have to run not only on computers dedicated to individual users but also on remote servers that the users share.
- While users can install necessary software on their own computers, they cannnot do that on autonomously owned and operated servers on public networks to install and support software of their choice
- The success of personal computers is largely due to the fact that they provide hardware and software platforms on which third party software developers can build stand-alone applications.
- In order to realize the potential public networks, they have to be transformed into platforms that can support the development and deployment of communicating applications by third party developers

Mobile Agent Infrastructure

A mobile agent infrastructure offers an attractive approach to transforming public networks into computing platforms

A mobile agent

- consists of program code, persistent internal state, and other attributes (e.g., travel plan, movement history, access privilege information, etc.)
- moves in a network from host to host to accomplish its tasks
- is based on the remote programming paradigm as opposed to the more traditional remote procedure calling paradigm of distributed computing
- supports ongoing interaction without ongoing communication
- facilitates communicating applications involving distributed data and knowledge sources and services
- can reduce communication traffic over low bandwidth, high latency, high cost access networks typically employed by mobile computers
- enables small lightweight mobile computers to interact with heavyweight applications without prior detailed knowledge of the remote server's capabilities

Mobile Agent Infrastructure

A mobile agent

- can combine knowledge and data from the client and server and perform inference on the server i.e., where the data and computing resources are located
- support robust computing over unreliable public networks
- can provide better support for mobile computers that are only intermittent connection to a network and hence intermittent access to a server; have low bandwidth connection even when connected; and have limited storage and proceesing capacity
- can facilitate real-time interaction with a server (e.g., when the server is connected to a special purpose machine tool)
- enables users to create customized communicating applications by creating and deploying agents that take up residency at remote servers

While the individual advantages of mobile agents can be matched by other alternatives, there is no obvious alternative that matches all of the advantages of mobile agents taken together [Harrison et al., 1995]

Mobile Agent Platform

Mobile agent platforms provide

- host-independent execution environment for mobile agent programs
- standard communication languages using which agents and servers can engage in dialogs
- support for creation, deployment, transportation, registration, management, of mobile agents and other book-keeping functions
- support for employing public security services to enable secure and authenticated access to server resources and secure auditing and error recovery mechanisms
- provide a pervasive, open, flexible framework for the development and personalization of complex distributed information systems using agent-based communicating applications such as information gathering, electronic commerce targeted information dissemination, knowledge discovery from heterogeneous distributed data and knowledge sources

Mobile agent Facility (MAF) is an attempt to standardize certain aspects of mobile agent platforms to facilitate interoperability among mobile agent platforms

Sample Mobile Agent Platforms

The Voyager™ Mobile Agent Platform from Objectspace, Inc.

- Written entirely in Java™
- provides for host-independent execution
- provides for creation of remote objects, and sending of messages to objects
- provides for remote enabling of Java classes without modifying the source
- an agent is simply a special kind of object that can move independently, and can execute on different hosts as it moves
- provides services for mobile agents including directory services, messaging, scalable group communication
- has been used for developing mobile agent applications by multiple groups including:
  - Personal information assistants for selective document retrieval from distributed document collections (ISU AI Lab)
  - Data mining and knowledge discovery from distributed databases (ISU AI Lab)
  - Distributed knowledge networks for bioinformatics (ISU AI Lab)
  - Distributed manufacturing (Objectspace, Inc.)

Sample Mobile Agent Platforms

JavaMob: A MAF-Compliant Mobile Agent Platform from ISU AI Lab

- MAF Compliant to facilitate interoperabiliry with other MAF compliant mobile agent platforms
- Written entirely in Java to support host independent execution
- Supports creation, deployment, management, and authentication of agents
- Supports directory services to help agents locate services and other agents
- Supports multiple ways of organizing agents into collectives for efficient communication, access control, and modular organization
- Uses CORBA (Common Object Request Broker Architecture) for managing distributed objects
- Has been used to develop several prototype mobile agent applications in the ISU AI Lab including
    - Personal information assistants for selective document retrieval from distributed document collections
    - Data mining and knowledge discovery from distributed data and knowledge sources

- Data warehouses for information fusion from multiple sources

Multiagent systems

Potential of Multiagent systems:

- Open information systems will contain multiple autonomous agents or agents acting on behalf of autonomous users or entities
- Solution of complex problems require the services of multiple agents with diverse capabilities and needs e.g., the mediator-based approach to information systems
- Multi-agent systems can support distributed collaborative problemsolving by agent collections that dynamically organize themselves
- Multi-agent systems support a modular, extensible approach to design of complex information systems

Challenge: How can multiagent systems generate useful behaviors?

- Inter-agent communication (of knowledge, intentions, beliefs)
- Inter-agent collaboration (e.g., through negotiation among self-interested rational agents)
- Coordination and control in multi-agent systems

Interacting Agents

- An interaction occurs when two or more agents are brought into a dynamic relationship through a set of reciprocal actions
- Interactions develop as a result of a series of actions whose consequences influence the future behavior of agents.
- Interactions may be direct or indirect, intended, or unintended

Interaction assumes:

- Agents that are capable of acting and/or communicating
- Situations that promote interaction (e.g.,need to share resources, need for collaboration or competition)
- Dynamic elements allowing for local and temorary relationships among agents

Modes of interaction

- Independence (no interaction)
- Simple collaboration (compatible goals, sufficient resources, insufficient skills) e.g., through knowledge sharing
- Obstruction (incompatible goals, insufficient resources, sufficient skills) e.g., when an agent gets in another's way in accomplishing its tasks – calls for special techniques for coordination among agents
- Coordinated collaboration (compatible goals, insufficient resources, insufficient skills) e.g., in distributed manufacturing, network control, multi-robot systems
- Pure individual competition (incompatible goals, sufficient resources, sufficient skills) where there is no conflict about resources and the best competitior wins
- Pure collective competition (incompatible goals, sufficient resources, insufficient skills) wherein agents have to form teams to compete against others
- Individual conflict over resources (incompatible goals, insufficient resources, sufficient skills)
- Collective conflict over resources (incompatible goals, insufficient resources, insufficient skills)

Interagent Cooperation

- grouping
- communication
- specialization
- sharing tasks and resources
- coordination of actions
- conflict resolution by arbitration and negotiation

Inter-agent Communication

Communication among homogeneous agents in narrow, precisely defined domains (e.g., distributed routing in communication networks) is relatively straightforward to handle using suitably defined protocols with precisely specified syntax and semantics

Communication among heterogeneous agents in open information systems or collaborative problemsolving environments is much more challenging.

Effective communication requires:

- Shared knowledge of syntax
- Shared understanding of semantics and pragmatics
- Some means of exchanging sentences (or even signs or symbols) to communicate

Inter-agent Communication: The KSE approach

Mutual understanding of what is being communicated involves:

- translation from one representation language to another
- sharing of semantic (and often pragmatic) content of representation

Translation is not enough because each agent (or each knowledge base) holds implicit assumptions about what is being represented and what it means. Therefore these implicit assumptions must be shared either explicitly (using language) or implicitly through shared experience.

Sample prototype applications: ARPA Rome Lab Planning Initiative, CoBase, integration of general software systems

Inter-agent Communication: The KSE approach

- Knowledge Interchange Format (KIF) as a means of representing and encoding knowledge. KIF is essentially a prefix version of first order logic with extentions to handle nonmonotonic reasoning and definitions
- Ontologies for various domains that describe concepts and their relationships that are implicit in the use of databases knowledge bases (e.g., the fact that the “salary” field in a relational database stands for total wages and not taxable salary for instance). Mismatches between ontologies used by different agents may have to be resolved during communication.
- Knowledge Query Manipulation Language (KQML) for messages that reflect an agent’s attitude about the content that is being carried (e.g., in KIF or some other representation language). KQML is based on the theory of speech acts and supports a collection of performatives such as ask-if, tell, achieve, reply, etc.

```
(ask-one :sender Joe :content (price IBM ?price) :receiver stock-broker
    :reply-with ibm-stock :language LPROLOG :ontology NYSE-ticks)
```

Inter-agent Negotiation

Problem: How can self-interested agents share resources, exchange tasks, etc. in mutually beneficial ways in dynamic envioronments?

Example: Trucking companies exchanging tasks so as to minimize the cost of fufilling all delivery orders

Negotiation Protocol

- defines the rules that agents have to abide by in negotiating with other agents
- constrains the choice of negotiation strategy used by self-interested rational agents (e.g., if the protocol has certain properties, it is in the best interest of the agent to be not deceptive in its interactions with other agents)
- influences the overall behavior of multi-agent systems (e.g., whether or not negotiation process terminates with an optimal task allocattion in the distributed delivery problem)

The choice of negotiation protocol is influenced by the nature of the task domain: task oriented, state oriented and worth oriented (Rosenschein and Zlotkin, 1995)

Coordination using Contract nets

Contract net protocol for distributed problem solving

- Managers announce tasks, evaluate bids, make contracts, monitor results
- Contractors make bids, commit to contracts, execute contracts
- The same agent can be a manager as well as a contractor in different negotiations
- Contracts involve commitments to complete the agreed upon tasks among other agents
- Award decisions typically based on marginal utility calculations
- Well characterized in terms of necessary and sufficient contract types for negotiation (e.g., Sandholm, 1998)

Sample applications: distributed delivery, information agents, distributed design and manufacturing, electric power systems, electronic commerce.

Multi-agent organizations

Multi-agent organizations are defined by an assembly of classes of agents characterized by their assigned roles and a set of abstract relationships (e.g., acquaintance, subordination, conflict) among the roles

Possible approaches

- Horizontal modular
- Hierarchical
- Subsumption
- Democracies
- Economies
- Creative Anarchies
- Societies governed by conventions (e.g., negotiation protocols)
- Ant colonies (simple rules of interaction)
- Immune system
- Evolution

Some directions for further research

- Intelligent agent designs
- Tools for rapid design and prototyping of agents
- High level languages for specification and implementation of agents
- Interagent communication (a variety of approaches for specific classes of problem domains)
- Interagent negotiation
- Sophisticated mobile agent platforms
- Agents for information retrieval, information extraction, and knowledge discovery from heterogeneous, distributed data and knowledge sources
- Agents for collaborative problem solving
- Learning in multiagent systems
- Privacy, Security, and authentication
- Agent economics
- Performance studies of specific systems
- Evaluation of alternative approaches to design of agents, multi-agent sytstems, negotiation protocols, etc.
- Open information systems
- Personal information assistants

References

1. Bradshaw, J.M. (Ed). Software Agents. Cambridge, MA: MIT Press. 1997.
2. Brenner, W., Zarnekow, R., and Hartmut, W. Intelligent Software Agents. Berlin: Springer. 1998.
3. Ferber, J. Multi-Agent Systems. New York: Addison-Wesley. 1999.
4. Honavar, V. Intelligent Agents. In: Encyclopedia of Information Systems. Williams, J.G. and Sochats, K. New York: Marcel Dekker. In press.
5. Honavar, V., Miller, L., Wong, J. Distributed Knowledge Networks: Design, Implementation, and Applications. In: Proceedings of the IEEE Information Technology Conference. 1998
6. Huhns, M.N. and Singh, M.P. Readings in Agents. San Mateo, CA: Morgan Kaufmann 1998.
7. Jennings, N.R. and Wooldridge, M.J. Agent Technology Foundations, Applications, and Markets. Berlin: Springer. 1997.
8. Maes, P. Agents That Reduce Work and Information Overload. Software Agents. Cambridge, MA: MIT Press. 1997.
9. Nwana, H. Software agents: an overview. The Knowledge Engineering Review. vol. 11. no. 3. 1996.
10. Russell, S. and Norvig, P. Artificial Intelligence: A Modern Approach. New York: Prentice-Hall 1995.
11. Sycara, K., Dekker, K., Pannu, A., Williamson, M., Zeng, D. Distributed Intelligent Agents. IEEE Expert, 1996.
12. Wiederhold, G. and Genesereth, M. The Conceptual Basis of Mediation Services. IEEE Expert, vol. 12, No. 5, 1997.
13. White, J.E. Mobile Agents. In: Bradshaw, J. (ed). Software Agents. Cambridge, MA: MIT Press. 1997

14.Wooldridge, M.J. and Jennings, N.R. Intelligent Agents: Theory and practice. The Knowledge Engineering Review. vol. 10. no. 2. 1995.
15.Zlotkin, G. and Rosenschein, J.S. Rules of Encounter. Cambridge, MA: MIT Press.