Auton Agent Multi-Agent Syst
DOI 10.1007/s10458-006-0008-9

Learning to communicate in a decentralized environment

Claudia V. Goldman · Martin Allen · Shlomo Zilberstein

Published online: ■

Abstract Learning to communicate is an emerging challenge in AI research. It is known that agents interacting in decentralized, stochastic environments can benefit from exchanging information. Multi-agent planning generally assumes that agents share a common means of communication; however, in building robust distributed systems it is important to address potential miscoordination resulting from misinterpretation of messages exchanged. This paper lays foundations for studying this problem, examining its properties analytically and empirically in a decision-theoretic context. We establish a formal framework for the problem, and identify a collection of necessary and sufficient properties for decision problems that allow agents to employ probabilistic updating schemes in order to learn how to interpret what others are communicating. Solving the problem optimally is often intractable, but our approach enables agents using different languages to converge upon coordination over time. Our experimental work establishes how these methods perform when applied to problems of varying complexity.

Keywords Multi-agent systems · Learning · Communication · Probabilistic methods

C. V. Goldman
Caesarea Rothschild Institute,
University of Haifa, Mount Carmel, Haifa 31905, Israel
e-mail:clag@cri.haifa.ac.il

M. Allen (✉) · S. Zilberstein
Department of Computer Science,
University of Massachusetts Amherst, Amherst, MA 01003, USA
e-mail:mwallen@cs.umass.edu

S. Zilberstein
e-mail:shlomo@cs.umass.edu

1. Introduction

Cooperative decentralized planning is the problem of computing a set of local behaviors for a group of agents that act in the same environment while maximizing a global objective. Each local policy maps the information known or believed by the agent to actions (i.e., domain actions and possibly communication acts as well). It has been shown [4] that in the worst case, solving such multi-agent problems optimally is significantly more complex than is the case for single-agent sequential decision problems. One of the main sources of this difficulty is the fact that each individual decision-maker lacks global information when they compute their local behaviors. Allowing agents to share information may reduce uncertainty about this global information, for instance by reducing the number of possible belief-states that each agent needs to consider. In extreme cases, when communication is free and mutually understood, decentralized planning becomes equivalent to single-agent planning (e.g., see the MMDP model [8]). However, in practice, communication has some cost, be it the actual bandwidth used by a transmission, or some other function that quantifies the resources required for the information exchange. We have previously shown that computing policies involving costly communication can be as hard as computing optimal solutions without communication [22, 23]. Consequently, although communication can indeed be helpful in simplifying coordination at execution time, computing when to communicate and what to communicate may still be a complex process.

Beyond overcoming the computational problem of decentralized planning under uncertainty, we are interested in *robust* decentralized systems. Such robustness will often require agents to adapt their means of communicating in the face of new situations, or when miscoordination arises. Autonomous systems, developed separately, interact more and more often in contexts like distributed computing, information gathering over the Internet, and wide-spread networks of machines using distinct protocols. As a result, systems may be called on to deal with new situations and information, as autonomy increases and environments grow more complex. Agents that act cooperatively may not necessarily share the same language of communication, and may not simply be able to exchange a translation, mapping discrepancies in the languages, in an off-line manner. Such problems may occur, for instance, when the content of an agent's communication arises from observations available solely to that agent, in the midst of some shared task. Even agents with the same sensing apparatus may still lack the contextual information necessary to correctly interpret each other's messages. Alternatively, the ability to learn new meanings can guard against unintentional design-time errors. In many applications, the misinterpretation of messages can lead to miscoordination and an eventual decrease in performance. NASA's Climate Orbiter probe, for instance, crashed as a result of an unwitting use of different (metric and imperial) conventions of measure in calculations communicated between different design teams, causing the spaceship to follow an incorrect flight plan [29]. Such considerations also arise where users of a system may have different levels of competence—as in automated and interactive tutoring—or where it is practically infeasible to specify all necessary communication protocols at design time. The latter problem arises, for instance, in automated control and diagnosis. In such contexts, the range of ways a particular mechanism may go wrong cannot generally be known in advance, and encountered problems often require novel diagnoses and solutions [7]. Such problems are compounded when various mechanisms are combined as parts of a larger overall process, as is common in manufacturing plants.

The ability to communicate is thus a double-edged sword. While it has the potential to make multi-agent coordination problems much easier to solve, the possibility of miscommunication opens up difficulties of its own. In this paper, then, we focus on the problem of how agents may learn how to interpret the messages they exchange, while acting together in an uncertain and decentralized environment. We isolate this problem from the problem of computing optimal policies with costly communication and concentrate on: (1) algorithms that update the understanding of messages exchanged while improving the value of the joint policies of behavior; and (2) the properties of systems and environments that allow such learning to take place.

The standard of success by which we judge the particular process of learning to communicate is directly related to the system-wide measure of utility, rather than to the individual cost of using that language (as, e.g. [18]). Agents attempt to learn correlations between languages with pre-existing semantics, distinguishing the approach from such as [40], in which agents collectively learn new shared concepts for the purpose of learning to communicate per se and not in the framework of a planning problem. Furthermore, agents learn to communicate while attempting to maximize some global objective, which need not be the specific one of learning to communicate itself, as opposed to work in which agents are specifically rewarded for pre-determined "correct" responses to messages [42]. Finally, our work on cases where miscoordination arises is distinct from such research as that in verification systems [34], where the aim is to identify inconsistencies between a software specification and its execution code, which can then be fixed manually. Our purpose, on the other hand, is to explore methods by which agents automatically learn to correct a misinterpretation in addition to identifying it, in the framework of decentralized control.

We provide a general practical and formal framework for studying the problem of learning to communicate, and isolate sufficient and necessary conditions under which agents can maximize their joint expected utility by successfully solving that problem. We show this in particular for agents that communicate their observations and actions; in the process, we shed light on the difficulties involved with learning languages in general. One of the contributions of this work is in understanding how hard this problem can be. The basic framework is decision-theoretic; agents operate with probabilistic models of the meaning of languages of communication, and base decisions on the given probabilities. The problem of determining message meanings may then become unsolvable for languages in general—since it may be impossible to generate a meaningful probabilistic model of a sufficiently rich language—and can be quite complex even for relatively simple languages. Another contribution is an algorithm that, under the identified conditions, converges to some mutual understanding of a language which is not initially shared by the agents. We show how such convergence eventually leads to situations in which agents can maximize the value of their joint policy.

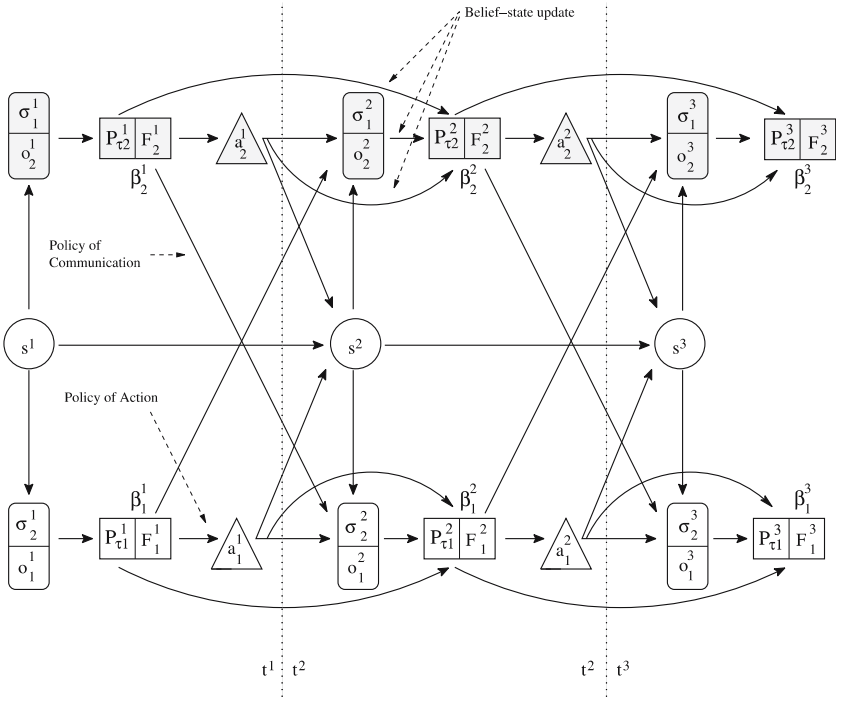
Sections 2 and 3 present the general language-learning process and show how the problem can be framed as a process of updating belief-states in the context of a decentralized Markov decision process. We explain this process further in Section 4. Important properties of decision problems and necessary and sufficient conditions for learning to communicate while acting are explained in Sections 5 and 6. These are followed by empirical evidence for the usefulness of the approach, for scenarios with increasing complexity, in Section 7. Finally, Sections 8 and 9 review other work on communication in multi-agent systems and give our conclusions. For brevity, full details of some proofs and other work have been omitted here; interested readers

are referred to the technical report containing a complete version of the contained research [20].

2. The general language-learning model

Figure 1 gives a graphical overview of the interrelations between the various components of the language-learning process as it occurs in the case of two inter-communicating agents. These elementary parts, which will be discussed in more detail later on, are as follows:

- *State*: The global state of the environment at any given time.
- *Observation:* How the global state actually appears to an individual agent.
- *Message*: The communications exchanged between agents.
- *Belief-state*: An agent's belief about the current state of affairs, consisting of:
  - *Translation*: An understanding of another agent's language, and of the most recent message.
  - *Belief-features*: All other beliefs about the current state of the environment.



s^i	State i	t^j	Time-step j
σ_i^j	Message j, sent by Agent i	o_i^j	Observation j of Agent i
a_i^j	Action j of Agent i	β_i^j	Belief-state j of Agent i
$P_{\tau i}^j$	Translation-distribution j of Agent i	F_i^j	Belief-features j of Agent i

Fig. 1 A sketch of the language-learning process

Such a process involves many elements that are familiar from the specification of problems involving agents acting under uncertainty. In particular, we will want to specify two important components:
State-action transitions: The environment transitions between global states s given the actions taken by each agent α at time t.
Observation functions: Each agent α makes some observation o at any global state s.
Furthermore, an agent in such a situation will possess, or learn a behavior which includes:
Policy of action: Given belief-states, agents take actions.
Policy of communication: Given belief-states, agents communicate something (maybe nothing).
Belief update: Given belief-states, actions, observations, and messages, agents generate new belief-states.

Language learning thus takes place within a general framework of action and communication. While the process of updating belief-states involving the interpretation of what other agents say has certain special features arising from the fact that language is involved, it is a special case of a general problem, to do with finding successful policies that are based on one's beliefs. The main distinguishing feature of this work is the notion of a *translation*. While the usual approach to many problems of action and communication assumes that the language of communication is shared, we make no such presumption. Instead, we include an agent's possibly imperfect understanding of another's language as part of the belief-state. We represent the degree to which agent α_i understands agent α_j by a correspondence between messages sent by α_j, and those that α_i might itself send. (Looking ahead a bit to the formal definitions found in Section 3, as far as α_i is concerned, the meaning of some received message forms a distribution over its own possible messages.)

Formally, we capture this model as a decentralized Markov decision process with direct communication, originally formulated in [23]. For simplicity of exposition we focus here on the case of two agents; the model can easily be extended to n agents.

Definition 1 (*Dec-MDP-Com*) A decentralized MDP with direct communication is given by the tuple $M = \langle S, A_1, A_2, P, R, \Sigma, C_\Sigma, \Omega_1, \Omega_2, O, T \rangle$ where:

- S is a finite set of world states with a distinguished initial state s^0.
- A_1 and A_2 are finite sets of control actions, with $a_i \in A_i$ an action performed by agent i.
- P is the transition probability function. $P(s'|s, a_1, a_2)$ is the probability of moving from state $s \in S$ to state $s' \in S$ when agents α_1 and α_2 perform actions a_1 and a_2, respectively.
- R is the global reward function. $R(s, a_1, \sigma_1, a_2, \sigma_2, s')$ represents the reward obtained by the system as a whole, when in state s agent α_1 executes action a_1 and sends message σ_1 and agent α_2 executes action a_2 and sends message σ_2, resulting in a transition to state s'.
- Σ is the alphabet of messages and $\sigma_i \in \Sigma$ is an atomic message sent by agent α_i. When this language is not mutually shared by all the agents, we use Σ_i to denote agent i's language of communication. (Definition 2 formalizes the idea of language precisely.)
- $C_\Sigma : \Sigma \to \Re$ gives the cost to transmit a message. A null message has zero cost to transmit.

- Ω_1 and Ω_2 are finite sets of observations.
- O is the observation function. $O(o_1, o_2|s, a_1, a_2, s')$ is the probability of observing o_1 and o_2 (respectively, by the two agents) when in state s agent 1 takes action a_1 and agent 2 takes action a_2, resulting in state s'. A Dec-MDP is jointly fully observable, i.e., there exists a mapping $J: \Omega_1 \times \Omega_2 \rightarrow S$ such that whenever $O(o_1, o_2|s, a_1, a_2, s')$ is non-zero then $J(o_1, o_2) = s'$.
- T is the time horizon of the problem (finite or infinite).

The proposed framework is based on several observations and assumptions. First, agents learn to communicate while acting to maximize some objective reward function. Second, agents share a context, in terms of their starting presumptions about the possible content of shared messages. While this may not be a single context, agents still consider only a relatively small set of plausible candidates. Third, agents base their understanding of one another upon probabilistic models of the relationship between language and the environment. Questions then arise about the problem of learning when such models are not accessible, but this is beyond the scope of this paper. Finally, we observe that it is not desirable to obscure message content by making it part of an extended state-space, to make the learning problem more general. Doing so will make the problem of learning to communicate intractable, given the complexity proofs and practical results for existing solution algorithms for Dec-POMDPs [4, 23]. Thus, we consider the use of communication separately from domain actions, allowing us to study techniques and features specific to language itself.

2.1. An applied example

The Dec-MDP-Com model can be applied to many real-world problems, so long as we can specify states, actions, and observations, all interacting in Markovian conditions. One such domain, which we have used in experimental work, involves two (2) agents, each controlling a set of pumps and flow-valves in a factory setting. At each time step, each agent separately observes fluid entering the system from one of two different inflow ducts, along with the pumps and valves under its own control. The agents seek to maximize flow out of the system through one of several outflow ducts, subject to the efficiency constraint that the number of ducts be minimized. Reward is directly proportional to outflow amount, minus the number of ducts used. Probabilistic effects arise because each of the pumps and valves is susceptible to variations in throughput, dependent upon whether the particular component was used to route flow in the prior time step.

Formally, we specify the problem as a Dec-MDP-Com $M = \langle S, A, P, R, \Sigma, C_\Sigma, \Omega, O, T \rangle$, where:

(a) S is the state set is described by flow through two inflow ducts, in_1 and in_2, and through a set of pumps and valves for each agent, $p_1^i, \ldots, p_n^i$ and $v_1^i, \ldots, v_m^i$. Initially, all such flows are set to zero (0).
(b) A is at each time-step each agent α_i chooses one action to control the pumps p_r^i (on, off, forward, back) or the valves v_s^i (open, shut).
(c) P is the transition function directs flow according to actions taken; however, pumps and valves fail to respond to commands probabilistically, based on whether or not they were used in the prior time-step.

(d) R is the total reward is given by (in/out)$-d$, where in is the total units of inflow, out is the total units of outflow, and d is the number of outflow ducts used.
(e) Σ is each agent α_i possesses messages corresponding to each of its possible actions, identifying labels for every pump or valve in the system, as well as the observed units of inflow through duct in_i.
(f) C_Σ is the cost of all messages is zero (0).
(g) Ω is each agent α_i can observe its own inflow duct in_i, along with all pumps p_r^i and valves v_s^i that it controls; further, agents observe the system-wide reward.
(h) O is the observation function takes any state of the system and returns each agent's own observations.
(i) T is the problem has an infinite time horizon.

While the state space of such a problem can be quite large, given the number of variables governing inflow and system settings, it is still efficiently solvable from a single-agent, centralized perspective. By taking the point of view of one agent observing all states globally, and acting in place of both agents simultaneously, it is solved offline using typical dynamic-programming methods. However, in situations where agents must share local observations and actions, a centralized perspective is not available, and agents need to communicate before optimal action is possible. The work here outlines conditions—involving both the problem domain and mechanism for communication—necessary to make this happen.

2.2. Solutions for Dec-MDP-Coms

In a Dec-MDP-Com with shared communication language Σ, the local behaviors of the agents are given by local policies of action and communication, $\delta_i^A : \Omega^* \times \Sigma^* \to A_i$ and $\delta_i^\Sigma : \Omega^* \times \Sigma^* \to \Sigma$, each based on sequences of observations and messages. In order to solve such decentralized processes optimally, we evaluate the possible local policies of behavior and find one optimizing the expected value of the joint policy. This expected value is computed as the expected reward obtained while following the policies, equal to the summed reward for each state-action transition, weighted by the probabilities of those transitions. We stress that while it is straightforward to define the value of such policies, actually calculating it to find the optimal solution is generally very difficult. In previous research, we have computed this value for the case when communication lead to joint full observability [23], but the general case is known to be extremely hard.

In the problems studied in this paper, on the other hand, agents do not share a common language of communication. Following Fig. 1, we notice that the local behaviors of the agents are mappings from belief-states to either domain actions or messages. That is, we wish to define the local policies of action and communication as functions from belief-state sequences, $\delta_i^A : \beta_i^* \to A_i$ and $\delta_i^\Sigma : \beta_i^* \to \Sigma_i$, respectively. These belief-states are composed of both translations of messages received and beliefs about the features of the states based on observations and message sequences. Section 4 discusses these policies and their value in full detail. Here, we simply note that our work is based on the idea that agents learn to communicate on-line while acting towards the maximization of some global objective. We assume that each agent already has been assigned some policy of behavior that includes a local policy of action and a local policy of communication. For every possible belief-state of its own, an agent will know how to act and what message from its own language to send. All they lack, then,

is a way to correctly interpret the messages received. Learning to act, to communicate and to interpret messages all at once on-line is beyond the scope of this paper. Here, we are interested in the process of updating these belief-states, resulting in updated interpretation of messages and consequently better coordination between agents.

3. Formal definitions

The elementary items comprising the theory of communication learning are *agents* taking *actions* and making *observations* in an environment, along with: (1) *languages* of communication between agents; (2) *translations* governing how messages in different languages are interpreted; (3) *beliefs* about the environment, governing which messages are sent and which actions are taken; and (4) *contexts* which govern how the interpretations of messages are generated and changed in response to feedback from the environment.

Definition 2 (*Language*) A language is a pair $\Sigma = (W, \int)$, where:

1. $W = \{\sigma_o, \sigma_1, \ldots, \sigma_n\}$ is a set of primitive words, the atomic units of which messages are comprised.
2. $\int$ is a syntax-function, defined as follows:
   - $W^n = \underbrace{W \times \cdots \times W}_{n}$ is the set of all n-word messages; $W^+ = \bigcup_{n\in\mathbb{N}} W^n$.
   - $\int : W^+ \to \{1, 0\}$.

A message $\sigma^+ \in W^+$ is legal for language Σ (written $\sigma^+ \in \Sigma$) iff $\Sigma = (W, \int)$ and $\int(\sigma^+) = 1$.

As given, the definition of a language does not specify any particular structure for a language; in general, the syntax-function can be arbitrary. For most purposes, of course, we would define such a syntax-function in some structured (perhaps compositional) way, so that the set of legal messages was similarly structured. Previous results [23] proved that agents controlling a Dec-MDP-Com benefit most by exchanging their own last observations when the cost of communication is constant over all messages. For more general cases, it may be worthwhile for the agent to send information about the actions it has performed. Since individual observations and actions are only known locally, they are natural candidates to be exchanged as messages between the agents. For example, agents acting in a 2D grid may be modeled as observing their own local coordinates. A language of communication can then be defined as the set of pairs of possible coordinates: the language $\Sigma_{xy} = (\mathbb{N}, \int)$ of coordinates in a 2D grid of size $(\mathrm{Max}X \times \mathrm{Max}Y)$ is given by the set of messages (x, y) where $x, y \in \mathbb{N}$ and $\int(x, y) = 1$ if and only if both $(0 \leq x \leq \mathrm{Max}X)$ and $(0 \leq y \leq \mathrm{Max}Y)$. Of course, other languages could have yet more complex syntaxes, as for example messages that include the actions taken as well. So far, there has been no study of Dec-MDPs featuring communication of messages that are different from observations. Our work here considers just such cases.

Note that we do not specify a semantics as part of the definition of a language. In fact, no detailed treatment of the meaning or truth-conditions of messages is given. Rather, the semantics of any language Σ arises implicitly from elementary practices of actual communication. Agents communicate particular messages, based simply on their own observations, actions and resulting beliefs, according to the agent's local policy of communication, which can be presumed to depend upon the global goals.

"Meaning," then, is simply a correlation between particular messages and the beliefs that cause their originator to send them; this is discussed in further detail below. Similarly, an agent establishes a *translation* between its own language and another by correlating the sets of messages in each (and so, indirectly, between messages in the other language and its own beliefs about the environment).

Definition 3 (*Translation*) Let Σ and Σ' be two sets of messages. A translation, τ, between Σ and Σ' is a probability function over message pairs, i.e., for any messages $\sigma, \sigma', \tau(\sigma, \sigma')$ is the probability that σ and σ' have the same meaning. $\tau^{+}_{\Sigma,\Sigma'}$ is the set of all translations between Σ and Σ'.

An agent translates between its own language and (some subset of) another by establishing a probabilistic correlation between the meanings of the messages in each.[1] Here we point out that "means the same thing" is given sense simply in terms of the agents' beliefs. For agent α_1 to say that message $\sigma_2 \in \Sigma_2$, sent by agent α_2, is likely to mean the same as message $\sigma_1 \in \Sigma_1$, is simply to say that it is likely that the situations in which α_2's beliefs cause it so send message σ_2 are the same as those in which α_1's beliefs would cause it to send σ_1.

For example, assume that the messages $\sigma_1 \in \Sigma_1$ and $\sigma_2 \in \Sigma_2$ name locations in a fully observable 2D grid, and that agent α_1 always sends out the name of its current location x, whenever it observes that it is in fact at x. Further, suppose that we have a completely specified translation τ between Σ_1 and Σ_2. Then, the probability $\tau(\sigma_1, \sigma_2)$ is simply the probability that the belief that causes α_2 to send σ_2 arises exactly when α_2 is in the same location that α_1 names using σ_1. Furthermore, in such a situation, any functional output $\tau(\sigma_1, \sigma_2) = 1$ will mean that α_1 is absolutely certain that σ_2 is to be translated as σ_1. This discussion can be made more precise by defining the notion of a belief-state.

Definition 4 (*Belief-state*) For an agent α in state-space S, a belief-state $\beta = (P_\tau, F)$, where:

1. P_τ is a probability distribution over translations.
2. F is a probability distribution over state-set S, with $F(s)$ the probability that the agent is in state s.

An agent's belief-state thus consists of a probability distribution over states—a familiar notion from the literature on partially observable Markov decision processes (POMDPs)—along with a probability distribution over the set of translations between the agent's own language and the language of another with which the agent communicates. In the case that there exist more than one such other agent, the belief-state will require multiple such translation-distributions; in this work, we will treat only of the two-agent case, and so only provide a single such distribution. Furthermore, since each translation τ^* is in itself a probability distribution over message pairs,

[1] In the most general case, the uncertainty of interpreting a message correctly can be captured by some general feasibility measure. We focus on probabilities as an example of such measure. Other measures are beyond the scope of this paper.

$(\sigma, \sigma') \in (\Sigma \times \Sigma')$, we can write $P_\tau(\sigma, \sigma')$ for the overall probability that some particular pair are equivalent, defined as:[2]

$$P_\tau(\sigma, \sigma') = \sum_{\tau^* \in \tau^+_{\Sigma,\Sigma'}} P_\tau(\tau^*)\tau^*(\sigma, \sigma'). \quad (1)$$

Thus, each agent may possess any number of distinct translations between its own language and some other, each assigned its own probability. The process of learning to communicate is thus the process of adjusting these sets of translations, replacing them with others that differ in the probabilities they assign to individual message pairs, or with respect to their own individual probabilities. In particular, we will assume that this process is governed by the observed outcomes of actions taken by the translator. That is, an agent α learns to communicate in some language Σ' as a result of taking actions based (at least partially) on messages received in that language, and then adjusting its current set of possible translations of Σ'.

In general, as given in Definition 3, a translation between language Σ and some other language Σ' involves a complete function over Σ. That is, the translation assigns values to translations for *every possible* message $\sigma \in \Sigma$. In real situations, of course, agents will not generally consider literally every possible meaning of a message σ received in some unknown language, for if they did, the process of interpretation and translation would never get off the ground. Rather, agents tend to restrict their attention to certain specific subsets of plausible meaning for the utterances of others, where such restricted subsets are given by contextual considerations of such things as relevance and expected intent. Effectively, then, a context allows an agent to bound the possible interpretations of other agents, or to pre-assign translations to specific parts of the language, as for example in cases where two agents already share part, but not all, of their two languages. The models and the bounds involved in contexts of translation can also correspond to such things as assumptions about the syntax of messages in the target language, or about the grammatical role of particular parts of certain messages. Such models, and their restrictions, are what makes language learning possible. Without some available context for translation—some model of the relationships between messages, actions, and outcomes—an agent α will be simply unable even to begin interpreting the language of another. Our notion of context is thus in the same vein as the *local models semantics/MultiContext Systems* (LMS/MCS) model reviewed in [33], in which it is assumed that an agent has a local theory containing all the knowledge it needs to solve a problem. We assume that agents share the same context, although each agent may have a different vocabulary associated with it. This paper studies algorithms that map one vocabulary to the other so that agents can correctly interpret messages sent in the relevant context.

4. On belief-state updates

The problems studied in this paper involve agents that do not share one language of communication, act in part based upon how they translate one another's messages. In

[2] A translation $\tau^* \in \tau^+_{\Sigma,\Sigma'}$ is total over an agent's own language Σ, but may be partial over the other language, Σ', especially as all the particular contents of that second language may not yet be known. In general, if some incomplete translation τ^* is such that the particular pair (σ, σ') is not part of its domain, then we set $\tau^*(\sigma, \sigma') = 0$ for the purposes of Eq. 1.

our framework, agents translate between sets of messages by establishing a probabilistic correlation between them. In most cases agents need to consider multiple possible translations between messages; that is, they possess beliefs about which translation to utilize in any given situation, and update those beliefs over time based on their experience. We now consider this update process, both in general and under some special simplifying assumptions. First, we discuss the way that the general process of updating translations, based on sequences of past observations, messages, and actions, interacts with policies of action and their value. Next, we look at a special case, where translation updates are based solely upon information from immediately prior time-steps.

4.1. Belief updates in general

Local behaviors of agents are mappings from sequences of belief-states to either domain actions or messages, $\delta_i^A : \beta_i^* \rightarrow A_i$ and $\delta_i^\Sigma : \beta_i^* \rightarrow \Sigma_i$. As given by Definition 4, the belief-state $\beta_i = (P_{\tau i}, F_i)$ is composed of probability distributions over translations and system states, respectively. We are particularly interested in updates of the translation portion, $P_{\tau i}$, of the belief-state (the portion F_i is itself dependent upon the current translation–distribution, along with the observation and message sequences), since the current translation will have direct effect upon an agent's current actions. Letting $P_{\tau i}^+$ be the space of all possible distributions over translations between languages Σ_i and Σ_j, we can write the local policies of each agent α_i in a manner analogous to those involving shared languages [23].

Definition 5 (*Local Policy of Action with Different Languages*)

$$\delta_i^A : \Omega_i^* \times \Sigma_j^* \times P_{\tau i}^+ \rightarrow A_i.$$

Definition 6 (*Local Policy of Communication with Different Languages*)

$$\delta_i^\Sigma : \Omega_i^* \times \Sigma_j^* \times P_{\tau i}^+ \rightarrow \Sigma_i.$$

That is, the policies for each agent are functions (to either actions or messages) from observation and message sequences, along with particular translation–distributions.

We define transition probabilities over state sequences given such a policy, and its expected value, just as for Dec-MDP-Coms with shared languages [23]. This task is complicated by the need to factor in updates to the translation–distributions. In general, agents will update their beliefs about translations—the distribution $P_{\tau_i}^+$—based on sequences of observations, messages, and actions.

Definition 7 (*Translation-Update Function*) Translation updates are functions:

$$U_{\tau i} : \Omega_i^* \times \Sigma_j^* \times A_i^* \times P_{\tau i}^+ \rightarrow P_{\tau i}^+.$$

By convention, if any of the three input sequences is empty, then $U_{\tau i}$ returns some designated distribution, $\hat{P}_{\tau i} \in P_{\tau i}^+$, the default distribution; that is:

$$U_{\tau i}(\epsilon, \overline{\sigma_j}, \overline{a_i}, .) = U_{\tau i}(\overline{o_i}, \epsilon, \overline{a_i}, .) = U_{\tau i}(\overline{o_i}, \overline{\sigma_j}, \epsilon, .) = \hat{P}_{\tau i}.$$

Over time, then, these updates influence actions taken, and the outcomes of those actions in turn influence further updates. Relative to a given local action policy, which generates the action sequences, we define the update function based on observation or message sequences alone.

Definition 8 (*Translation Updates for an Action Policy* δ_i^A) For a given local policy of action δ_i^A, and update function $U_{\tau i}$, we define the translation-updates for δ_i^A, written $\overline{U_{\tau i}^{\delta_i^A}}$, recursively on observation and message sequences as follows:

$$\overline{U_{\tau i}^{\delta_i^A}}(\epsilon, \epsilon) = \overline{U_{\tau i}^{\delta_i^A}}(\overline{o_i}, \epsilon) = U_{\tau i}^{\delta_i^A}(\epsilon, \overline{\sigma_j}) = U_{\tau i}(\epsilon, \epsilon, \epsilon, .) = \hat{P}_{\tau i}, \tag{2}$$

$$\overline{U_{\tau i}^{\delta_i^A}}(\overline{o_i}o_i, \overline{\sigma_j}\sigma_j) \quad = U_{\tau i}(\overline{o_i}o_i, \overline{\sigma_j}\sigma_j, \overline{\delta_i^A}(\overline{o_i}, \overline{\sigma_j}), \overline{U_{\tau i}^{\delta_i^A}}(\overline{o_i}, \overline{\sigma_j})), \tag{3}$$

where, for $\overline{o_i} = \langle o_i^1 o_i^2 \ldots o_i^n \rangle$ and $\overline{\sigma_j} = \langle \sigma_j^1 \sigma_j^2 \ldots \sigma_j^n \rangle$, we have action sequence, $\overline{\delta_i^A}(\overline{o_i}, \overline{\sigma_j})$ as:

$$\langle \delta_i^A(o_i^1, \sigma_j^1, \overline{U_{\tau i}^{\delta_i^A}}(o_i^1, \sigma_j^1))\, \delta_i^A(o_i^1 o_i^2, \sigma_j^1 \sigma_j^2, \overline{U_{\tau i}^{\delta_i^A}}(o_i^1 o_i^2, \sigma_j^1 \sigma_j^2)) \ \ldots \ \delta_i^A(\overline{o_i}, \overline{\sigma_j}, \overline{U_{\tau i}^{\delta_i^A}}(\overline{o_i}, \overline{\sigma_j}))\rangle.$$

That is, for either an empty observation or message sequence, the update returns the default translation–distribution, $\hat{P}_{\tau i}$. Further, for full sequences of observations and messages, the update is based upon those sequences, along with the particular actions and updates generated immediately prior, based on their prefix sub-sequences. Note that the given action sequence presumes that $\overline{o_i}$ and $\overline{\sigma_j}$ are of the same length; this is merely for convenience, and the definition can be rewritten easily for sequences of unequal length, given the proviso on $U_{\tau i}$ concerning empty sequences. Note also that by convention the action sequence for empty observation or message sequences is also empty: $\overline{\delta^A}(\epsilon, \epsilon) = \epsilon$.

To identify the expected reward for a joint policy, we first define the probability of transitioning over a sequence of states. This is given by the one-step transition probability of reaching a destination state $'s$ from any origin state q (part of the Dec-MDP-Com model), multiplied by the cumulative transition probability for any possible sequence suffix leading to q, and weighted by the probability of sensing the given observation sequence, just as in the case of policies with a shared language [23]. The difference here is that we must take account for the ongoing processing of messages received, i.e. the update function, $\overline{U_{\tau i}^{\delta_i^A}}$. Thus, when we consider the actions taken under each individual policy, we take into account not only the most recent sequence of observations and actions, but also the latest state of the translation, given those sequences.

Definition 9 (*Transition Probability Over a Sequence of States with Translations*) The probability of transitioning from a state s to a state s' following the joint policy $\delta = (\delta_1, \delta_2)$ in the presence of translations while agent 1 sees observation sequence $\overline{o_1}o_1$ and receives sequences of messages $\overline{\sigma_2}$, and agent 2 sees $\overline{o_2}o_2$ and receives $\overline{\sigma_1}$ of the same length, written $\overline{P_\delta}(s'|s, \overline{o_1}o_1, \overline{\sigma_2}, \overline{o_2}o_2, \overline{\sigma_1})$, can be defined recursively:

1. $\overline{P_\delta}(s|s, \epsilon, \epsilon, \epsilon, \epsilon) = 1$.
2. $\overline{P_\delta}(s'|s, \overline{o_1}o_1, \overline{\sigma_2}\sigma_2, \overline{o_2}o_2, \overline{\sigma_1}\sigma_1) =$

$$\sum_{q \in S} \overline{P_\delta}(q|s, \overline{o_1}, \overline{\sigma_2}, \overline{o_2}, \overline{\sigma_1}) * P(s'|q, \delta_1^A(\overline{o_1}, \overline{\sigma_2}, \overline{U_{\tau 1}^{\delta_i^A}}(\overline{o_1}, \overline{\sigma_2})),$$
$$\delta_2^A(\overline{o_2}, \overline{\sigma_1}, \overline{U_{\tau 2}^{\delta_i^A}}(\overline{o_2}, \overline{\sigma_1}))) *$$
$$O(o_1, o_2|q, \delta_1^A(\overline{o_1}, \overline{\sigma_2}, \overline{U_{\tau 1}^{\delta_i^A}}(\overline{o_1}, \overline{\sigma_2})), \delta_2^A(\overline{o_2}, \overline{\sigma_1}, \overline{U_{\tau 2}^{\delta_i^A}}(\overline{o_2}, \overline{\sigma_1})), s')$$

such that $\delta_1^\Sigma(\overline{o_1}o_1, \overline{\sigma_2}, \overline{U_{\tau 1}^{\delta_i^A}}(\overline{o_1}o_1, \overline{\sigma_2})) = \sigma_1$ and $\delta_2^\Sigma(\overline{o_2}o_2, \overline{\sigma_1}, \overline{U_{\tau 2}^{\delta_i^A}}(\overline{o_2}o_2, \overline{\sigma_1})) = \sigma_2$.

The value of the initial state s^0 of the Dec-POMDP-Com after following a joint policy δ for T steps can be defined as the expected reward attained over all possible sequences of states visited by δ starting in s^0. This is also analogous to the shared-language case [23], once again taking into account the update function $\overline{U_{\tau_i}^{\delta A}}$ to reflect how the sequence of past messages and actions is currently viewed, given current translation.

Definition 10 (*Value of an Initial State Given a Policy with Translations*) The value $V_\delta^T(s^0)$ after following policy $\delta = (\delta_1, \delta_2)$ from state s^0 for T steps is given by:

$$
\begin{aligned}
V_\delta^T(s^0) = \sum_{(\overline{o_1}o_1, \overline{o_2}o_2)} \sum_{q \in S} \sum_{s' \in S} & \overline{P_\delta}(q | s^0, \overline{o_1}, \overline{\sigma_2}, \overline{o_2}, \overline{\sigma_1}) * \\
& P(s' | q, \delta_1^A(\overline{o_1}, \overline{\sigma_2}, \overline{U_{\tau 1}^{\delta A}}(\overline{o_1}, \overline{\sigma_2})), \delta_2^A(\overline{o_2}, \overline{\sigma_1}, \overline{U_{\tau 2}^{\delta A}}(\overline{o_2}, \overline{\sigma_1}))) * \\
& R(q, \delta_1^A(\overline{o_1}, \overline{\sigma_2}, \overline{U_{\tau 1}^{\delta A}}(\overline{o_1}, \overline{\sigma_2})), \delta_1^\Sigma(\overline{o_1}o_1, \overline{\sigma_2}, \overline{U_{\tau 1}^{\delta A}}(\overline{o_1}o_1, \overline{\sigma_2})), \\
& \quad \delta_2^A(\overline{o_2}, \overline{\sigma_1}, \overline{U_{\tau 2}^{\delta A}}(\overline{o_2}, \overline{\sigma_1})), \delta_2^\Sigma(\overline{o_2}o_2, \overline{\sigma_1}, \overline{U_{\tau 2}^{\delta A}}(\overline{o_2}o_2, \overline{\sigma_1})), s'),
\end{aligned}
$$

where the observation and the message sequences are of length at most $T-1$, and both sequences of observations are of the same length l. The sequences of messages are of length $l+1$ because they considered the last observation resulting from the control action prior to communicating.

The optimal policy for a Dec-MDP-Com with given translation-update functions, $U_{\tau i}$ for each agent α_i, is thus the joint policy which maximizes the expected value of the starting state. As before, we stress the difference in difficulty between stating the value-function, and actually calculating the value so that we can determine an optimal value-maximizing policy. Solving such a problem will be no easier in general than for problems with shared languages. In any case, our work does not involve generating such policies. Rather, we are interested in the translation-update functions that produce behavior in concert with *given* policies. Therefore, we look now at ways in which these update functions can be simplified.

4.2. Belief updates with limited information

In the processes we consider, belief-state updates need not be based upon possibly unbounded sequences of observations, messages, and actions. Figure 1 (in Section 2) has given a general overview of the relationships between various components of the language-learning process. As shown there, the belief-state of an agent at any time t, β^t, depends upon the following pieces of information:

1. The prior belief-state, β^{t-1}.
2. The prior action, a^{t-1}, taken on the basis of that belief-state.
3. The most recent observation of the environment, o^t, resulting from that action.
4. The most recent message received, σ^t.

It is to be noted that this is an ideal case. In particular, it assumes that updates rely only upon information and action taken from the current and immediately prior time-step. The process can become quite complicated, if not infeasible, when updates must rely upon information from further in the past. Furthermore, we cannot always guarantee that these sorts of updates are always possible, even given just the time-limited

information specified here; we shall also examine cases in which updating the probabilities assigned to certain translation entries are simply infeasible, due to defects in the structure of the language in question.

As given by Definition 4, a belief-state is a two-part structure, consisting of an agent's current best belief about the meaning of another language (the probability distribution over translations, P_τ), along with a belief concerning other features of the environment (the distribution over states, F). The process of updating the belief-state therefore, involves updating each of these components, either separately or together. As we model this process, it takes place sequentially, updating each part in turn. Let $\beta_i^t = (P_{\tau i}^t, F_i^t)$ be the belief-state of agent α_i, to be calculated at time t; ideally, we want the probabilities reflected in the two component distributions to be set correctly by way of a two-step update:

1. First, agent α_i updates its belief about language, setting the probability distribution over possible translations based upon its prior belief-state and action, its own current observation, and any message just received from agent α_j. This update should be such that for any pair of messages, it yields the probability that this pair has the same meaning (written $\sigma_i = \sigma_j$):

$$(\forall \sigma_i, \sigma_j)\ P_{\tau i}^t(\sigma_i, \sigma_j) = \sum_{\tau^* \in \tau^+_{\Sigma_i, \Sigma_j}} P_{\tau i}^t(\tau^*)\tau^*(\sigma_i, \sigma_j)$$
$$= \mathbf{P}(\sigma_i = \sigma_j \mid P_{\tau i}^{t-1}, F_i^{t-1}, a_i^{t-1}, \sigma_j^t, o_i^t).$$

2. Second, the agent needs to update its belief about the state of the environment, setting the probability distribution over states according to the prior such belief and action, the most recent message and observation, and the current (just-updated) translation distribution:

$$(\forall s)\ F_i^t(s) = \mathbf{P}(s^t = s \mid P_{\tau i}^t, F_i^{t-1}, a_i^{t-1}, \sigma_j^t, o_i^t).$$

In our model, learning to communicate is therefore the process of systematically updating belief-states with respect to translations. Agent α_i chooses an action, a_i, based upon its local observation, o_i, any messages received, and the current belief-state, β_i^t, about how to translate those messages. The choice of a_i, along with the actions chosen by other agents, leads to some state-transition, which in turn results in some new observation, o_i^{t+1}. This observation then leads to an update to a new belief-state, β_i^{t+1}, further affecting how later messages are translated, and thus influencing future actions.

The procedure governing the update from belief-state β_i^t to β_i^{t+1} comprises the agent's *language model*: a function from actions, messages, and observations, to distributions over translations and system states. (Definition 7 and what follows, above, gives a formal treatment of the language portion of these updates.) Such probabilistic models can range from quite simple to highly complex, depending upon the nature of the languages and problem environment. For instance, in a Dec-MDP-Com in which agents already share a common language—so that translation is not an issue—and can freely communicate with one another, all that is required is that agents update their state-distributions. This, however, is elementary: since a decentralized MDP is *jointly fully observable* (Definition 1, clause 8), agents can compute the actual global state directly, and probabilities are strictly unnecessary. Where agents begin without a commonly understood language, things are obviously much more complicated,

and the prescribed belief updates can be difficult to compute correctly. Indeed, it is not clear that it would be possible to generate a meaningful, let alone usefully tractable, probabilistic model of a full-blown natural language. Thus, we concentrate upon languages restricted to suit the needs of agents working in decentralized MDPs, communicating specifically about their own observations and actions, and trying to optimize performance in a set task. Even such basic languages provide interesting challenges and difficulties; we discuss this further in [20, §4.2.1–2], including examples of vicious circles in the update process, or where agents require unbounded memory.

5. The language-learning problem

As we have explained, the general problem of learning to communicate while acting in a decentralized environment can be captured as the process of updating belief-states about system-states and translations, together with policies that allow agents to act based upon these beliefs Such problems may arise in any number of contexts. One plausible use for such techniques arises in cases of automated systems coping with errors in their design or specification, as for instance the problem of disparate metric and imperial measures encountered in the Mars orbiter program [29]. Such systems, when communicating with one another, or with human operators, may need to learn to reinterpret instructions or state specifications, in light of new information. In our Dec-MDP-Com framework, agents would each possess a model of the overall problem domain, in terms of possible states of the system, along with a probabilistic model of action effects in terms of state transitions, and a model of expected reward for actions. When observations indicate discrepancies between the world and how received messages are understood, agents will then need to learn how to re-translate those messages. The Mars orbiter, for example, upon observing that following its given flight-path information was leading it too close to the surface, might have been saved had it the capacity to learn how to adjust its understanding of those instructions by translating them into another system of measure.

Lacking any particular restriction on the updates and action choices, this general process can be very complex and is not guaranteed to converge to either a correct interpretation of the messages exchanged, or an optimal policy of action. In this section, we define the problem of learning to communicate precisely, giving criteria for what it means to solve it. Then, we identify desirable properties of algorithms for doing such learning, and give properties of Dec-MDP-Coms that allow us to give some guarantees on the performance of a system of probabilistic belief updates. The rest of the paper will concentrate on a particular implementation of an algorithm that solves the learning problem in such problem instances.

5.1. Solutions and solution algorithms

"Learning to communicate" comprises a broad set of behaviors and capacities. We give the phrase specific meaning in terms of convergence to optimal action policies in a decentralized MDP. In this context, agents learn to communicate while acting towards a jointly optimal solution of the Dec-MDP-Com, and we evaluate how successful learning has been according to its usefulness in achieving such a value-maximizing policy. To make the problem studied here precise, we refer to the expected reward of the optimal joint policy given some translations (Section 4.1). Then, we can

tell that a system of agents has learned to communicate if there exists some point in time such that the best plan based on translations from that point on is in fact optimal. We are not studying the optimality of the learning process itself, i.e. how much time the learning process takes until such a point in time is found. Instead, we consider the performance of a decentralized system when miscoordination may arise as a result of discrepancies in the interpretation of messages exchanged.

We begin by defining what it is for a joint policy, in combination with agents' update functions, to converge. We again restrict the formal presentation to the two-agent case for convenience.

Definition 11 (*Convergence with Translation Updates*) Let M^1 be a multiple-language Dec-MDP-Com:

$$M^1 = \langle S, A_1, A_2, \Sigma_1, \Sigma_2, C_\Sigma, P, R, \Omega_1, \Omega_2, O, T \rangle$$

and let agents α_1 and α_2 have translation-update functions $U_{\tau 1}$ and $U_{\tau 2}$. Additionally, let the agents take part in the joint policy $\delta_{M^1} = [(\delta_1^{A1}, \delta_1^{\Sigma 1}), (\delta_2^{A2}, \delta_2^{\Sigma 2})]$, with individual local policies as follows:

$$\delta_i^{Ai} \colon \Omega_i^* \times \Sigma_j^* \times P_{\tau i}^+ \to A_i \qquad \delta_i^{\Sigma i} \colon \Omega_i^* \times \Sigma_j^* \times P_{\tau i}^+ \to \Sigma_i.$$

Further, let M^2 be a Dec-MDP-Com identical to M^1, except for a single shared language, $\Sigma^+ = (\Sigma_1 \cup \Sigma_2)$:

$$M^2 = \langle S, A_1, A_2, \Sigma^+, C_\Sigma, P, R, \Omega_1, \Omega_2, O, T \rangle$$

and for any state s at time t, let $\delta^*_{M^2}(s^t) = \arg\max_\delta V_\delta^{T-t}(s^t)$ be the optimal policy for M^2, starting from s at t. Then, the first joint policy δ_{M^1}, with update functions $U_{\tau 1}$ and $U_{\tau 2}$, *has converged* at some state s and time t if and only if:

$$V_{\delta_{M^1}}(s^t) = V_{\delta^*_{M^2}(s^t)}(s^t).$$

That is, a policy converges, along with the associated update functions, when the agents reach a point in time after which their actions, in accord with their translations at that point and at all points afterwards, return the maximum value that could be expected if the agents did in fact share all their linguistic resources. This definition highlights the dual nature of convergence in the language-learning context, in that it depends upon both the policies of action and communication, as well as the translation-update functions in use. Given this notion of convergence, it is straightforward to give an exact statement of the learning problem.

Definition 12 (*Learning to Communicate while Acting*) Let M be a Dec-MDP-Com with multiple languages: and let agents α_1 and α_2 have translation-update functions $U_{\tau 1}$ and $U_{\tau 2}$, and joint policy $\delta_M = [(\delta_1^{A1}, \delta_1^{\Sigma 1}), (\delta_2^{A2}, \delta_2^{\Sigma 2})]$. The agents have solved the problem of learning to communicate while acting at some state s at time t if and only if δ_M, with update functions $U_{\tau 1}$ and $U_{\tau 2}$, *has converged* at s^t.

That is, the agents have solved the problem of learning to communicate whenever they arrive at a policy of action that, in combination with their given translation-update functions, leads to a course of action equal in value to one that would maximize return if the agents in fact shared all the same language. The definition therefore says nothing about the character of the translations at work. In particular, we are not committed

to any view as to whether these translations are "correct" or not. Rather, agents are said to have solved the problem whenever their actions, *however they translate* one another, are optimal. This opens the door to a number of possibilities, including translations that are only partially complete, or ones that conspicuously mistranslate certain language items. So long as these kinds of omissions or mistranslations do not affect the optimality of the associated policy, we accept that the agents have indeed learned to communicate.

5.1.1. Optimality of converged policies

A consequence of this manner of defining a solution to a communication problem is that it takes into account the possibility that the joint policy, along with the update functions, may converge to a pattern of behavior that is not *absolutely* optimal. That is, the optimality of that policy is relative to the point of convergence, s^t, and we do not guarantee optimal results if the agents were to "start over," implementing the policy from the initial Dec-MDP-Com start-state, s^0. This is a result of an unavoidable fact, namely that the time it takes to learn something in the setting of a decision problem may have negative consequences on the expected value in that environment. In general, we cannot guarantee that the time taken for learning does not cut agents off from some potential expected value. For instance, in some problem domains, there are states of the environment which are simply unreachable given the optimal joint policy of action. In such cases, an absolutely optimal policy need not prescribe actions that actually maximize value for such states; since they are never reached, any action whatsoever may be assigned to them without affecting overall optimality. When agents are learning, however, and taking sub-optimal actions along the way, they may very well find themselves in such states of the environment, and so their final policies may end up very different.

There is one important exception. In an *ergodic*, infinite-horizon Dec-MDP-Com, every state reachable with non-zero probability by following an action policy recurs infinitely often and aperiodically under that same policy; the set of such states is called the *recurrent set* [30]. If the recurrent set for every policy comprises the entire state space S, then the process is *irreducible*. It is well known that the expected value of policies in such environments is indifferent to the exact starting state, allowing us to establish the following.

Claim 1 (Optimality in Irreducible Dec-MDP-Coms) *Let M^1 be an irreducible two-agent and two-language Dec-MDP-Com, with an infinite time horizon. Let δ_{M^1}, with update functions $U_{\tau 1}$ and $U_{\tau 2}$, have converged at some s^t in M^1. Then δ_{M^1} is absolutely optimal for the shared-language version of the problem, M^2; that is, for initial state s^0 of M^2 :*

$$\delta_{M^1} = \delta^*_{M^2} = \arg\max_{\delta_2} V_{\delta_2}(s^0).$$

Proof δ_{M^1}, along with $U_{\tau 1}$ and $U_{\tau 2}$, have converged at s^t, and so by definition,

$$V_{\delta_{M^1}}(s^t) = V_{\delta^*_{M^2}(s^t)}(s^t), \tag{4}$$

where $\delta^*_{M^2}(s^t)$ is the policy that is optimal in the shared-language problem M^2 from point s^t onwards. Since M^1 is irreducible and infinite-horizon, so is M^2, as the problems are identical apart from the languages involved. Thus, starting at s^t in M^2, and

following policy $\delta^*_{M^2}(s^t)$, we will visit original initial state s^0 infinitely often. Therefore, this policy is absolutely optimal:

$$\delta^*_{M^2}(s^t) = \arg\max_{\delta_2} V_{\delta_2}(s^0). \tag{5}$$

Equations 4 and 5 establish Claim 1. □

Our empirical work, described in Section 7, has not generally involved problems that are fully irreducible, nor even properly ergodic. On the other hand, to impose natural stopping conditions on our learning algorithms, we dealt with problems in which agents eventually encountered every state in the environment often enough to drive their translations to a point at which those translations were in fact *complete* (in the sense of Definition 15, below). In such cases, and because starting conditions for decision tasks repeated randomly (so that there are no "dead-end" policies), it is easy to show that the convergent policies of action with translation are in fact absolutely optimal. We do not prove this fact here, as it is only incidental to the goals of this paper. Rather, we note it only to point out that the choice was one of convenience: our proposed methods require neither the special properties just mentioned, nor full irreducibility, to succeed. For instance, we prove in Claim 6, Section 6, that a particular protocol for action and translation update converges to an optimal policy. While this generally requires an infinite time horizon to allow enough time for learning, irreducibility is not assumed; the convergence proven is thus that of Definition 11.

5.1.2. τ-Learning algorithms

The central feature of any attempt to solve the communication-learning problem is an algorithm for updating translations. Here, we call such methods τ-*learning* algorithms; an example is given in Section 6. In our work, we concentrate upon instances in which policies of action and communication are given ahead of time. That is, the central problem facing an agent is that of updating its translations. Once this has been accomplished, these pre-selected policies dictate what is to be done, based upon the current interpretation. (More details of the policy-selection process are given below.) Of course, different τ-learning algorithms may result in different belief updates, and can thus have differing effects upon the values of the associated policies. For example, τ-learning algorithms that generate correct translations of only a proper subset of messages can result in an overall system utility that is lower than would be expected if agents were able to translate all messages properly. In other instances, however, this may not be the case, as for instance when not all messages are actually essential to the successful performance of the system.

In addition to a τ-learning algorithm, agents need a rule for how to choose one meaning for a message received out of its possible translations. We give an example of such a rule in the protocol presented in Section 6, where agents act simply upon a meaning with maximum probability of being correct. Another reasonable assumption about the learning process is that observations sensed after actions were taken based upon interpreted messages are *informative*. While this information need not necessarily tell an agent the actual correct meaning of a message, it should provide some guidance about whether it has acted correctly or not. There may be many ways to act right or wrong, and this observation is not required to distinguish among these. We avoid cases in which observations provide no useful information at all. Clearly, such

environments make language learning impossible, but this is a defect in the problems, not in the methods for solving them. Similarly, we are not interested in cases where any action at all, taken upon any interpretation, results in the same expected reward—in such cases there is clearly no more benefit to be had from learning language than from ignoring it altogether. This section focuses on some formal properties that τ-learning algorithms may have. These properties help us understand the optimality guarantees of learning algorithms. Two important characteristics of learning methods are the *certainty* and the *completeness* of their outputs.

Definition 13 (*A τ-learning algorithm is certain with respect to* (σ_i, σ_j)) A τ-learning algorithm is certain with respect to (σ_i, σ_j) if the algorithm is guaranteed to converge to a translation τ such that $\tau(\sigma_i, \sigma_j) = 1$ or $\tau(\sigma_i, \sigma_j) = 0$.

Definition 14 (*A τ-learning algorithm is ϵ-certain with respect to* (σ_i, σ_j)) A τ-learning algorithm is ϵ-certain with respect to (σ_i, σ_j) if the algorithm is guaranteed to converge to a translation τ such that $\tau(\sigma_i, \sigma_j) \geq \epsilon$, or $\tau(\sigma_i, \sigma_j) \leq (1 - \epsilon)$.

The latter type of learning algorithm enables us to compare levels of coordination between multiagent systems that have learned to communicate at different ϵ levels. This measure will thus serve us as a quantifier to compare the performances of such systems. Furthermore, we may be able to switch between contexts when the translations cannot be ϵ-certain for some ϵ. We leave this for future work.

Definition 15 (*A τ-learning algorithm is complete*) A τ-learning algorithm is complete if it is certain with respect to all pairs of messages in Σ_i.

An agent employing a complete τ-learning algorithm to learn how to communicate in some language Σ will have converged upon a translation that is ultimately certain in all its assignments to message-pairs. In order to use such a translation to actually generate an optimal policy of action, it can also be necessary that agents are able to communicate about all relevant features of the environment.

Definition 16 (*Fully describable*) A Dec-MDP-Com is fully describable if each agent α_i possesses a language Σ_i sufficient to communicate both: (*a*) any observation made, and (*b*) any available action.

The application of a complete τ-learning algorithm in a fully describable environment allows agents to solve Dec-MDP-Coms to optimality.

Claim 2 *Solving the problem of learning to communicate for a fully describable Dec-MDP-Com*

$$\overline{M}_2 = \langle S, A_1, A_2, \Sigma_1, \Sigma_2, C_\Sigma, P, R, \Omega_1, \Omega_2, O, T \rangle$$

with a complete τ-learning algorithm is a sufficient condition for solving $\overline{M}_2$ optimally.

Proof Since the τ-learning algorithm is complete, it is certain with respect to all pairs of messages. There thus exists some time t at which the agents are guaranteed to know with certainty a mapping between messages in both languages Σ_1 and Σ_2. Furthermore, since the Dec-MDP-Com is fully describable, completeness means that the agents know to map any possible observation and action in A_i and Ω_i to any action and observation in A_j and Ω_j. Therefore, given enough time for the learning algorithm to

converge and without loss of generality, agent α_1 can exchange messages in Σ_2 instead of in its original language Σ_1. Hence, the expected value of the joint policy of action and communication that solves $\overline{M}_2$ will be equal to the expected value of the optimal solution to the corresponding problem with a mutually shared language. □

While this result establishes that we can in fact solve a Dec-MDP-Com optimally given a complete method, such a τ-learning algorithm is not a necessary condition for optimality. A τ-learning algorithm may not be complete—that is, it may not reach full certainty for every possible pair of messages—and nevertheless the agents' performance may be optimal. For example, this may occur when messages that do not get translated with certainty are not relevant for the optimal solution of the problem. Such uncertain translations cannot affect the value of the joint policy of action and therefore, agents may behave and communicate optimally even though their translations are not complete. For example, assume a set of two agents is assigned a task of revealing a hidden treasure. Agents need to move around a two-dimensional grid in order to find the treasure. Agents can exchange messages about locations, for example, instructing the other agent to move towards a certain place. Discrepancies in the language of communication means in this example that agents do not have the same system of coordinates and therefore need to correlate pairs of locations in both languages. There may be some set of locations that is never used in the solution of the problem, so the fact that one agent may not know how to interpret the names of those locations correctly will have no effect on the quality of the solution. Similarly, agents may achieve optimal performance using an incomplete algorithm that does not assign complete certainty to translations, but is correct in terms of *relative* certainty (so that the correct translation of any message is always the most probable). In such a case, if agents always choose the most likely interpretation of any message as its actual interpretation, then the lack of absolute certainty will have no effect upon the value of those actions.

5.2. Suitability of Dec-MDP-Coms

The ability of agents to learn to communicate depends not only upon the algorithms employed to update belief-states, but also upon features of the problem instance. In the previous section, we identified one such feature, full describability (Definition 16); here, we identify further properties of Dec-MDP-Coms, and discuss their usefulness and possible necessity in the learning process. The first of these properties has to do with the cost of communication in the system; while the general definition of a Dec-MDP-Com allows arbitrary costs for each message sent, we are particularly interested in cases where communication is cost-free.

Definition 17 (*Freely describable*) A Dec-MDP-Com is freely describable if and only if the cost of communicating any message σ is 0.

Free communication has the potential to considerably simplify a Dec-MDP-Com, since agents may communicate as much as they like without affecting overall system utility. Where the system is not freely describable, optimal control policies need to consider not only what to communicate, but when, according to a cost–benefit analysis of the potential value of the information shared as compared to the price of communicating it. Thus, as discussed in Section 5.2.1 below, it may not be possible to generate optimal solutions to such problem instances in an effective manner. Detailed study

of the communication-learning problem in a non-free decentralized environment is beyond the scope of this paper.

When combined with free communication, the ability to describe a Dec-MDP-Com in full can radically simplify the problem at hand. For example, we have shown [1] that when agents in fact share a common language, a Dec-MDP-Com that is both freely and fully describable is reducible to the much simpler case of a multi-agent MDP (MMDP). The MMDPs, in which each agent effectively observes the entire global system state, are known to allow for polynomial algorithms that generate optimal policies, and are thus tractable in ways that Dec-MDP-Coms are not [8]. As discussed below, such properties also affect the possibility of optimal solutions when agents must first learn to communicate before they can undertake their policies.

As we have pointed out, optimal action in a Dec-MDP-Com is importantly related to agents' ability to communicate effectively about their observations and actions. Similarly, the ability to update belief-states so that agents learn to communicate about such factors depends upon their capacity to discern more likely interpretations over time. We thus define what it is for a Dec-MDP-Com to be *suitable* for language learning, allowing agents in such a system to update their translation belief-states in a monotonic fashion, so that correct translations become ever more likely. Among other things, such monotonic updates allow agents to avoid the sorts of vicious-circle problems discussed in [20, §4.2.2]. To define the required property, we first give some additional notation:

Notation 1 Let M be an n-agent Dec-MDP-Com. In some state s, at time t, suppose agent α_j observes o_j and intends to take action a_j, communicating both facts to other agents by messages σ_j^o and σ_j^a. Any agent α_i then assigns a probability

$$P_i^\sigma(o_j \mid \sigma_j^o, \beta_i^t) \tag{6}$$

to the possibility that α_j observes o_j, given message σ_j^o and α_i's current belief-state β_i^t. Similarly,

$$P_i^\sigma(a_j \mid \sigma_j^a, \beta_i^t) \tag{7}$$

is the probability that α_j will take action a_j, given message σ_j^a and α_i's current belief-state. Let $\max_i^\sigma(o_j)^t$ and $\max_i^\sigma(a_j)^t$ maximize (6) and (7) (i.e., the observation and action α_i considers *most likely* for α_j).

Notation 2 Let M be an n-agent Dec-MDP-Com. In some state s, at time t, suppose each agent α_j observes o_j and takes action a_j, causing a transition to state s', with observations $\langle o'_1, \ldots, o'_n \rangle$ at time $t+1$. Then, for any agent α_i, let

$$P_i^o(o_j \mid o'_i)^{t+1} \tag{8}$$

be the probability that agent α_j previously observed o_j, given that α_i now observes o'_i. Similarly,

$$P_i^a(a_j \mid o'_i)^{t+1} \tag{9}$$

is the probability that α_j took action a_j given α_i's current observation.

Two observations are worth making here. First, the probabilities in each case differ in character: where the first is essentially *subjective*, the second is *objective*. Notation 1

identifies a probability that each agent assigns to an event, given that agent's update scheme and language model. Notation 2, on the other hand, identifies a probability found in the Dec-MDP-Com model itself. As we will show, in certain circumstances these probabilities can be made to match, so that agents do in fact employ update schemes that capture the actual dynamics of their environment. The second important point concerns the nature of the observation o'_i, as it appears in Notation 2. In much research into decentralized MDP models, system-wide reward is separated from an agent's observations, and as such is not generally available; while the goal may be to find policies that maximize this reward, agents do not act directly in response to these rewards. In other research, however, agents do in fact observe system-wide reward; work using MDP models in reinforcement learning, for instance, often uses this reward as the primary motive force driving the agents toward optimal action over time [37]. In some of the examples we consider, we choose the latter path, making reward directly available to the agents. Nothing vital depends upon this choice, however, and our examples can be replaced with ones in which reward is unobserved, and updates draw upon other information. Further, we sometimes distinguish between components of the observation directly related to language learning, and those related to the rest of the environment. Again, this is just for convenience. As given in the notation, we are concerned simply with the probabilities of certain actions and observations, given what is observed during a state transition; if only certain features of the observation are informative, this will not affect the relevant probabilities.

Definition 18 (*Suitability*) Let M be any fully describable Dec-MDP-Com in which agents do not share a common language. In any state s at time t, let each agent α_i observe o_i and take action a_i, communicating both to other agents using messages σ_i^o and σ_i^a. We say that M is suitable just in case for any agents α_i and α_j, if $o_j \neq \max_i^\sigma (o_j)^t$, then for any time $t' \geq t$ at which α_j observes o_j (the same observation as at time t),

$$P_i^o(o_j \mid o'_i)^{t'+1} > P_i^o(\max_i^\sigma(o_j)^t \mid o'_i)^{t'+1} \tag{10}$$

and similarly for $a_j \neq \max_i^\sigma (a_j)^t$,

$$P_i^a(a_j \mid o'_i)^{t'+1} > P_i^a(\max_i^\sigma(a_j)^t \mid o'_i)^{t'+1}. \tag{11}$$

That is, in a suitable Dec-MDP-Com, suppose agent α_j observes o_j and takes action a_j, communicating both to other agents using messages σ_j^o and σ_j^a. However, based upon its belief-state at that time, some other agent α_i incorrectly considers a different observation $\max_i^\sigma (o_j)^t \neq o_j$ most likely for α_j. In such a case, at any later state (including the next one), α_i's resulting observation o'_i "corrects" the situation. That is, at the next time-step, α_i will now consider it more likely that α_j observed o_j, rather than the incorrect observation. And, since this property holds at all future time-steps, any time that α_j happens to observe that same o_j again, α_i will still consider that correct observation more likely than the one previously thought most likely. (And similarly for the action a_j taken by α_j.)

This property is quite complicated to state, and may take a little time to digest. However, while suitability is somewhat difficult to formalize precisely, we do not consider it to be an overly strong or artificial condition. As we argue in the next section, lack of suitability may make it simply impossible to update the probabilities assigned to action- or observation-messages in any meaningful fashion, and in unsuitable environments a decision-theoretic agent may be incapable of achieving optimal

performance. Furthermore, suitability is not necessarily an uncommon property of Dec-MDP-Coms. Note that the property as given does not require that the actual identity of the correct prior observation and action of other agents be determined exactly by individual observed outcomes. Rather, it is only needed that the correct such observation or action be *to some degree more likely* than ones that might have seemed most likely before. For instance, domains in which agents have no idea what actions others are taking, but can at least positively eliminate some incorrectly chosen candidate—assigning it zero (0) probability given the immediate effects of the action taken—can be suitable with respect to those actions (given the proper conditions on communication): the evidence after any action is taken will eventually eliminate incorrect candidates, while increasing the probability of the correct action towards eventual certainty. Similarly, environments in which one agent observes some state variable a time step before another can be suitable with respect to observation, since the latter agent will eventually be given positive evidence allowing the determination of the correct observations. Lastly, we note that the pump-network example we described in Section 2.1 can in fact be implemented as a suitable Dec-MDP-Com; Section 7.2, where we describe experiments with implemented versions of such a problem explains this point further.

5.2.1. Importance of the Dec-MDP-Com properties

We would argue that the three properties outlined in the previous sections—free communication, full describability, and suitability—are not overly strong or ad hoc. In each case, the lack of the stated property has the potential to make probabilistic updates of translations, and optimal action based on those updates, either intractable or impossible. In the absence of such properties, then, agents will either require additional restrictions upon the system, or must employ some completely different form of learning algorithm in order to converge to optimal behavior. Later, we will show an implementable technique for such convergence in the presence of all three properties, establishing their sufficiency with respect to communication learning. For now, we concentrate upon their necessity.

Claim 3 (Necessity of free communication) *Learning to communicate and act optimally in a Dec-MDP-Com that is not freely describable is generally intractable.*

Claim 4 (Necessity of full describability) *Learning to communicate and act optimally in a Dec-MDP-Com that is not fully describable is generally intractable.*

Claim 5 (Necessity of suitability) *In the absence of suitability, agents will be generally unable to update translation-belief probabilities so as to allow them to act optimally.*

We do not prove these claims here; discussion and supporting examples are found in [20]. For now, we only note that properly interpreting Claim 5 takes care. We do not say that without suitability, it is generally impossible for agents to learn to act optimally in a decentralized MDP. Rather, we simply mean that if agents are basing their actions upon the probabilities assigned certain interpretations, in a straightforward utility-maximizing decision-theoretic manner, they will need suitability to be able to make well-directed updates to those probabilities. This does not, then, militate against the application of completely different techniques to such problems (for instance, some form of reinforcement learning). We would argue that such simple and

direct learning methods, which do not focus on specific and structured interpretation of messages, are of limited applicability to language learning in general; however, fully comparing the advantages and possibilities of various policy-generation and learning algorithms is beyond the scope of this paper.

6. Particular protocols and algorithms for language learning

Learning to communicate while acting requires not only an update rule (the τ-learning algorithm), but also a protocol for coordinating communication, action and interpretation. Here, we present such a protocol, allowing agents in suitable and freely describable Dec-MDP-Coms to converge to optimal behavior. Following that, we describe the use of a Bayesian update method for calculating belief updates in this context, and describe how we have implemented that method in our tests and experiments (see Section 7).

Definition 19 (*Elementary action protocol*) Let s be a state of Dec-MDP-Com M, at time t, where agent α_i observes o_i. Each α_i follows the *elementary action protocol*:

(1) α_i communicates o_i to the others, using message σ_i^o.
(2) α_i calculates the *most likely observation sequence*,

$$o^\star = \langle \overset{\sigma}{\max_i}(o_1)^t, \ldots, o_i, \ldots, \overset{\sigma}{\max_i}(o_n)^t \rangle$$

and *most likely state*, $s^\star = J(o^\star)$. (In accord with joint full observability, as in Definition 1.)
(3) Proceeding in turn, α_i chooses an action by:
   (a) Calculating the *most likely action sub-sequence*,

$$a^\star = \langle \overset{\sigma}{\max_i}(a_1)^t, \ldots, \overset{\sigma}{\max_i}(a_{i-1})^t \rangle.$$

   (b) Choosing action a_i that is part of some joint action maximizing value for $s^\star$ at time t:

$$a^+ = \langle a^\star, a_i, a_{i+1}, \ldots, a_n \rangle.$$

   (c) Communicating a_i to the others by message σ_i^a.
(4) α_i takes action a_i after all agents complete step (3)
(5) The state transition from s to s' caused by joint action $\langle a_1, \ldots, a_n \rangle$ generates new observation sequence $\langle o'_1, \ldots, o'_n \rangle$ and reward r' at time $t+1$. α_i then updates its belief-state so that for any messages σ_j^o and σ_j^a received on the prior time step, and any observation o_j and action a_j, both:

$$P_i^\sigma(o_j \mid \sigma_j^o, \beta_i^{t+1}) = P_i^o(o_j \mid o'_i)^{t+1}, \tag{12}$$

$$P_i^\sigma(a_j \mid \sigma_j^a, \beta_i^{t+1}) = P_i^a(a_j \mid o'_i)^{t+1}. \tag{13}$$

It is important to note three features of this protocol. First, agents choose actions based upon the *observations* of all others, but the *actions* of only those that precede them. The reader can confirm that this allows agents that already understand each other to coordinate optimally, avoiding the coordination problems Boutilier [8] sketches. Agents that are still learning the language act in the way they believe

most likely to be coordinated. Second, in the agent's new belief-state, the probability assigned in observation or action given the most recently received messages—in other words, the *meaning* of the messages—is identical to the probability that the other agent actually made that observation or took that action. It is assumed that the translation of all other messages from each other agent is adjusted only to account for normalization factors. Section 6.1 describes the use of a Bayesian Filtering algorithm to actually accomplish these sorts of updates in practice. Finally, in general multi-agent coordination, such a straightforward procedure is not necessarily optimal, nor even necessarily close to optimal. As Boutilier [8] points out, correct choice of action in the presence of unreliable or inaccurate communication must consider how each action may affect that communication, along with the other more immediate rewards to be had. Thus, in our case, it might sometimes be better for agents to choose their actions based not simply upon what they thought the most likely state might be, but also upon how certain expected outcomes would affect their translations for future instances of the problem, perhaps trading immediate reward for expected long-term information value.

Claim 2 showed that fully describable problems can be solved optimally when translation updates are performed with a complete algorithm. The next claim shows that for suitable and fully describable problems, the elementary action protocol suffices to obtain optimal behavior. Intuitively, this protocol causes the agents to choose their actions in the *right* direction (because following the protocol, agents choose actions based on the *most likely* ones), and since the problem is suitable this behavior leads to optimal results.

Claim 6 *Agents following the elementary action protocol in a suitable and freely describable Dec-MDP-Com with infinite time converge upon a joint policy that is optimal for the states encountered afterwards.*

Proof As agents act at some time-step t, they choose actions based always on the observations and actions of others that they consider most likely. That is,

1. Agent α_i translates the observation- and action-messages, σ_j^o and σ_j^a, for other agents α_j.
2. For each such message, α_i chooses the most likely translation, $\max_i^\sigma(o_j)^t$ or $\max_i^\sigma(a_j)^t$.
3. These interpretations are then used, along with α_i's own observation o_i, to generate the most likely observation sequence, $o^\star = \langle \max_i^\sigma(o_1)^t, \ldots, o_i, \ldots, \max_i^\sigma(o_n)^t \rangle$, and the most likely action sub-sequence, $a^\star = \langle \max_i^\sigma(a_1)^t, \ldots, \max_i^\sigma(a_{i-1})^t \rangle$.

Now suppose, without loss of generality, that one of these observation–translations $\max_i^\sigma(o_j)^t$ is incorrect; that is, the prior observation of agent α_j is such that $o_j \neq \max_i^\sigma(o_j)^t$. Following the joint action a^+, α_i will receive its own observation o_i' and system-wide reward r'. Then, since the problem is suitable, we have that

$$P_i^o(o_j \mid o_i')^{t+1} > P_i^o(\max_i^\sigma(o_j)^t \mid o_i')^{t+1}$$

(and similarly for all future time-steps). That is, the correct observation will be more likely, given the observation, than that one previously thought most likely. Furthermore, updates of messages proceed directly in accord with these probability assignments, since the protocol simply assigns

$$P_i^\sigma(o_j \mid \sigma_j^o, \beta_i^{t+1}) = P_i^o(o_j \mid o_i')^{t+1}$$

and therefore, we have that

$$P_i^{\sigma}(o_j \mid \sigma_j^o,\, \beta_i^{t+1}) > P_i^{\sigma}(\max_i^{\sigma}(o_j)^t \mid \sigma_j^o,\, \beta_i^{t+1}).$$

That is, at all future time-steps, α_i will assign the correct translation of message σ_j^o a higher probability than the prior, incorrect interpretation. Finally, once the correct translation of any message is actually the most likely translation, it will remain so for all future time steps. Thus, since the number of possible actions and observations for any agent in a Dec-MDP-Com is finite by definition, agents will, when given enough time, choose the correct entries, since these will eventually become most probable, by process of elimination. After this point, the most likely observation and action sequences will in fact be the actual observation and action sequences for other agents, and α_i will implement a policy that is optimal from then on, since it is now acting based upon the actual states and next actions of the problem. (Note that the problem is freely describable, and so utility is not affected by the constant communication required by the protocol.) □

Usually, work on cooperation in decentralized environments has followed one of these approaches:

1. Agents cannot communicate [21].
2. Agents can communicate and understand each other correctly [11, 14, 15, 25].
3. Agents can communicate and learn a language until it is completely understood [18, 36, 40].

Our work here takes a different point of view, since in some cases decentralized systems can cooperate optimally even though the agents may not be able to interpret all messages exchanged fully and with certainty. In fact, cooperation may be achieved even though there remains confusion and uncertainty about the meanings of some messages. It follows from Claim 6 that at some point in the learning process of a suitable Dec-MDP-Com involving the elementary action protocol, $\max_i^{\sigma}(o_j)^t$ will become equal to o_j. This may occur when $P_i^o(\max_i^{\sigma}(o_j)^t | o_i')^{t+1} = \epsilon < 1$. The corresponding interpretation of o_j will be picked by agent i so long as it is most likely, even though not completely certain. Therefore, there exist cases when agents can effectively understand each other even before their translations became complete. We can conclude:

Corollary 1 *Optimal cooperation can be viable with an ϵ-certain τ-learning algorithm* $(0 < \epsilon < 1)$.

Note that characteristics of the environment such as symmetries could also be exploited by the agents when interpreting messages in order to behave optimally. In such cases, even though the probability of interpreting a pair of messages is less than one, due to interactions between these interpretations and the actual effects of the agents' actions, their coordination can be optimal. This is left for future research.

6.1. Bayesian filters

In order to calculate the probability updates required, we adopt the method of Bayesian filtering [12], used for example in robotics to handle localization [38]. Typically, agents using this method possess a set of beliefs, in the form of a probability distribution over possible states of the environment; the filtering algorithm updates

this distribution over time, under two basic circumstances. (1) An agent updates its belief function based on new *observations*: such observations may determine the state exactly, or imply a probability distribution over states, depending for instance on factors like noise. (2) Agents make belief updates *predictively*: before taking any action, the agent updates the probability distribution for each state in which it may end up.

In decentralized multi-agent contexts without communication, however, these updates are not generally feasible, as Bayesian updating cannot be performed correctly. In Dec-MDPs, observation and state-transition functions involve multiple distinct agents. That is, the observation function gives the probability $O(o_1, \ldots, o_n \mid s, a_1, \ldots, a_n, s')$ that each agent α_i observes o_i when actions $a_1, \ldots, a_n$ cause the state transition from s to s'. Similarly, the transition function gives the probability $P(s' \mid s, a_1, \ldots, a_n)$ of moving from state s to state s', given joint action $a_1, \ldots, a_n$. Now, an individual agent α_i, wanting to update its beliefs about the state of the system, needs to calculate both of these probabilities based solely upon its own observations and actions: $O(o_i \mid s, a_i, s')$ and $P(s' \mid s, a_i)$. If others chose actions according to known probabilistic (or deterministic) strategies, these quantities could be derived using straightforward marginalization. In general, however, such a presumption is invalid, and α_i can assign no meaningful prior probabilities to other actions a_j, especially in learning contexts where other agents are also adjusting their behaviors. (For an approach in which agents can generate such priors by modelling other agents, see [17].) Therefore, unless special independence assumptions are made about the Dec-MDP in question, Bayesian filtering is not suitable for updating beliefs about system states (e.g., previous studies [3, 23] have looked at decentralized problems with independent transitions and observations). We note also that our use of this method relies in particular upon the ability to update beliefs based solely upon information from the current and immediately prior time-step; as discussed in [20, Section 4.2.1], this is an important if not universally valid assumption.

In the context of communication, however, Bayesian filtering does allow individual agents to update their belief function appropriately. If a Dec-MDP is freely and fully describable (Definitions 16, 17), then agents that already understand one another can simply share their observations at all times; joint full observability (explained in Definition 1) then makes the process of identifying global system states elementary, since each such state is determined by the collected observations of each agent. Furthermore, even where agents do not fully understand one another, Bayesian filtering provides a method of updating *translations* over time. Figure 2 gives the Bayes-Filter algorithm, as used in this work. The algorithm is presented for a pair of agents; for n agents, the basic structure of the technique is the same, although it is more complicated to present notationally and schematically. As shown in the pseudocode, the algorithm is used to update agent α_i's belief-state distribution over translations, $P_{\tau i}(\tau)$, given a received message σ_j, and data $\mathbf{d}$, consisting of one of α_i's own observations o_i or own actions a_i. In the first case, where the algorithm receives a local observation o_i, it updates the distribution based on the prior probability of that observation given each possible existing translation, normalizing as appropriate. In the second case, when an agent is about to take an action a_i, it updates the belief distribution by projecting the probabilities of possible next translations, based upon the existing ones, along with the message received and action to be taken. For details of how the basic algorithm can be implemented to deal with actual problem instances (see Section 6.3).

Fig. 2 Bayes-filter algorithm

Bayes-Filter$(P_{\tau i}(\tau),\, \sigma_j,\, \mathbf{d})\{$
$NormFactor = 0$
If $\mathbf{d}$ is an observation o_i then
  For all translations τ do
    $P'_{\tau i}(\tau) = P(o_i \mid \tau,\, \sigma_j) P_{\tau i}(\tau)$
    $NormFactor = NormFactor + P'_{\tau i}(\tau)$
  For all translations τ do
    $P'_{\tau i}(\tau) = P'_{\tau i}(\tau) NormFactor^{-1}$
Else if $\mathbf{d}$ is an action a_i then
  For all translations τ do
    $P'_{\tau i}(\tau) = \sum_{\tau'} P(\tau \mid \tau',\, \sigma_j,\, a_i) P_{\tau i}(\tau')$
Return $P'_{\tau i}(\tau)$ $\}$

6.2. Bayesian filtering and the elementary action protocol

To perform the calculations described in the update algorithm, agents require two sets of prior probabilities:

(1) A sensor model, giving $P(o_i \mid \tau,\, \sigma_j)$, the probability of observation o_i, given any translation τ and message σ_j. This probability is used in the first, observation-update step of the filter algorithm, when calculating the probability distribution over translations, based upon observations and messages received.
(2) An action model, giving $P(\tau \mid \tau',\, \sigma_j,\, a_i)$, the probability of translation τ, following action a_i, and given translation τ' and message σ_j. This probability is used in the second update step, when predicting the next translation distribution based upon what is known now, and what action is intended.

The possibility of giving such models for a general Dec-MDP-Com clearly depends upon the content of the messages σ_j that are sent between agents. In particular, agents will generally need to share information about their own observations and actions in order to perform the correct updates.

We can understand these requirements better by focussing upon the elementary action protocol (EAP), as given in Definition 19. We note that in the EAP, agents swap messages at each step about their current observations and intended actions. Furthermore, an agent's translation of these messages induces probability distributions over those observations and actions of others. Thus, when agents swap information about their own local observations in the first step of the EAP, they can update their belief-states about translations using the first part of the Bayesian filtering algorithm, based upon these observations, what is known about actions at the prior time-step, and the (marginalized) prior probabilities given by the Dec-MDP-Com model. Following this step, agent α_i can perform the second step of the EAP, calculating the most likely observation for α_j given the updated new translation, and the most likely global state of the environment based upon that information, in combination with its own local observation.

In the third step of the EAP, agents exchange action-messages in order, using their current translations to choose most likely interpretations of those messages, and choosing actions that are optimal relative to those interpretations. Again, the current translation belief-state induces a probability distribution over the actions of the other agent, and so α_i can perform the second part of the Bayesian filtering algorithm, making its predictive update given the chosen action. Following this update step, agents

act jointly, a state transition occurs, and the process repeats, with new observations shared, and new Bayesian updates are performed.

6.3. An example of the update process

One possible representation of a translation τ is a two-dimensional table where each row in agent τ_i's translation table corresponds to some atomic component of the agent's own language Σ_i. Columns correspond to atomic components identified in messages received in language Σ_j. Entry (σ_i, σ_j) gives the probability that σ_j has the same meaning as σ_i. The overall structure of the table corresponds to bounds on the presumed structure of messages received, based on the assumption that cooperating agents communicate things related to the immediate task. This radically constrains the range of possible interpretations, in order to make communication generally feasible; such constraints correspond to the use of such considerations as relevance and context in real-world communication, in order to make interpretation easier, or even possible.

Probabilities play two important roles. First, the filtering algorithm assigns probabilities to translation tables, taken as part of the agent's local state. Second, agents choose actions based upon the probable meaning of recent messages, calculated according to the overall probability that any translation is correct, and the probabilities contained in the corresponding table. Interpretation is interleaved with action, in a process designed to narrow down possible translations, while still acting on current ones, however, uncertain.

To cope with particular features of this task, the basic filtering algorithm is modified in two main ways. First, agents do not *explicitly* enumerate and update all possible beliefs: since belief-states themselves contain combinations of continuous probability distributions, there will be infinitely many available at any time. Instead, only those states necessary are generated and updated at each step; the procedure is straightforward, and is sound so long as states not generated are properly taken to occur with zero probability. Second, the set of possible belief-states changes over time: since agents do not generally know all elements of the translated language in advance, new components must be added to the translation-tables along the way.

As an example, consider a simple gridworld problem and a language of communication given by the agents' observations ($\Sigma = \Omega$). The environment is a 2×2 grid; to identify locations, agents use unambiguous proper names, meaning (1) each square in the grid has exactly one name, and (2) each name identifies exactly one square. Suppose agent α uses the integers $\{1, 2, 3, 4\}$ to name the four squares; the goal is to find a mapping between these names and those used by another agent that uses as names the letters $\{A, B, C, D\}$. Agents communicate grid locations using their own naming conventions; the other agent receives the message, attempts to interpret it, and proceeds to the most likely square. An observation follows, indicating whether or not the agent has successfully identified the correct location.

Figure 3 shows how α adds newly received messages into its translation table. We assume that α begins in the sole belief-state τ^0 (i.e., α assigns translation τ^0 unit probability). We see that α has previously received two messages, "A" and "B"; further, α has successfully translated the first of these, since $\tau^0(A, 2) = P(A \text{ means } 2) = 1$. Two more features of the table stand out. First, each column sums to 1; α knows the components of its own language, and knows that the distribution for any message must sum properly. Second, rows *do not* sum to 1; before α has seen all of the words

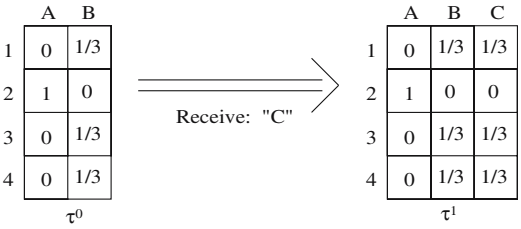


Fig. 3 A new message is added

in the other language, it cannot know how to fill out each row of its translation table. (Of course, in a simple example like this, α could reason out that there were only four possible messages, and fill out the table accordingly, but this is not generally possible.)

Suppose α now receives a new message, "C"; the belief-state is then updated from τ^0 to τ^1. A new column is added to the table: since α already knows that the name "2" corresponds to message "A", "C" receives a zero probability for this entry, and all remaining entries are uniformly distributed, reflecting the proper-name model of the language. Since "2" is translated as "A", it will not be translated as any other name, and so the 0 is inserted in the table; further, α has no reason to think "C" any more likely to name one remaining square in the grid than another, and so the remaining probability mass is distributed uniformly. Of course, for ambiguous languages, or ones in which locations can have more than one name, the first assumption would not hold. As well, in some cases an agent may have reason to favor one translation over another in advance, and probabilities need not be uniformly distributed.

Having expanded the table, α now chooses an action. Since the probability that "C" means "1", "3", or "4" is the same, it is indifferent which location it visits next. Of course, this need not be true; in general, the choice of an action is computed based on the most likely translation for the current message, weighted by the probability assigned to each possible translation table. Assume that α chooses "1" as the most likely translation: Fig. 4 shows the *action model* update of α's belief function (the "else" clause of the filtering algorithm in Fig. 2) before going to the square named by "1". We assume action outcomes are deterministic: after taking action $Go(1)$, α is sure to end up at the desired square. Furthermore, the observation sensed at the end of the action determines precisely whether or not α is correct in its translation (i.e., α perceives a particular observation if and only if it has correctly identified the chosen square). Thus, the update process replaces τ^1 with two new possible translations, τ^2 and τ^3.

Belief-state τ^2 reflects the outcome that the translation (C means 1) is *correct*. In this case, the entry $(1, C)$ in the table is set to unit probability, and all other entries in the row and column are set to 0. Again, this is a reflection of the unambiguous nature of the name language; more complex update scenarios are possible. Note also that the table in τ^2 also updates the probabilities contained in column "B"; since we normalize all columns, the probability mass for entry $(1, B)$ is distributed over the remaining possibilities. Belief-state τ^3, for its part, reflects the outcome that the translation (C means 1) is *incorrect*. Here, the only column affected is "C"; the entry $(1, C)$ is set to 0, and its probability mass distributed over the remaining entries in that column. Lastly, α assigns predictive probabilities to these two new belief-states before it takes action Go(1), based on what the outcome of that action can tell us about the chosen translation. In this case, $P(\tau^2|\tau^1, \text{Go}(1))$ is simply the original probability, 1/3, taken from table τ^1, that translation (C means 1) is correct. Similarly, the probability of the incorrect translation is $P(\tau^3|\tau^1, \text{Go}(1)) = (1-1/3) = 2/3$.

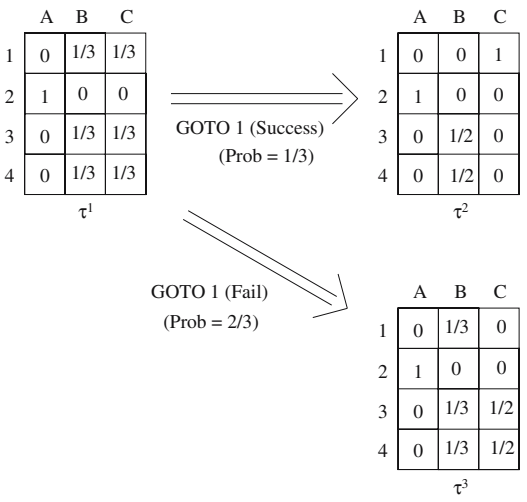


Fig. 4 The two translations believed possible after Go(1)

For this environment, the *sensor model* is also simple: α either senses appropriate observation for successfully translating "C" or not. We will use the notation OR when we refer to such an observation to distinguish from the usual observation of the environment. If α does receive *OR*, belief-state τ^2 is clearly correct and post-observation probability $P(\tau^2|\mathrm{OR})=1$, while $P(\tau^3|\mathrm{OR})=0$ and the latter belief-state can be discarded. If α does not receive OR, then the opposite holds. In either case, α is left with a single belief-state on which to base its next actions. If that message is one of "A", "B", or "C", then an action chosen based upon the existing probabilities, and the procedure shown in Fig. 4 is repeated. If it is some new message, "D", then that message is added to the table of translations as in Fig. 3, and the entire procedure repeats itself. Again, we stress that for more complex cases, the relevant updates can be more complicated. In particular, it need not be the case that α is left with but one translation at the end of the process of action and observation, since observed rewards may not determine single states, and actions may not have determinate outcomes. Still, the basic principles remain the same.

7. Experimental results

We present results from experiments on two sets of different scenarios of increasing complexity. The first scenario (first described in [19]) deals with some relatively simple examples of suitable problems, where agents do not need to communicate their actions, only their observations. In that domain, two agents work to meet at points in a gridworld environment, following a relatively simple procedure, with each acting in turn, according to the best estimate of the location of the other. Messages describing each agent's location are exchanged, and translations of those messages are updated after each step, depending upon whether or not the agents do in fact meet one another. Since agents are certain after checking some location whether or not the other agent was in fact there, the probability that the other observed that location is either 0 or 1, and the suitability of the Dec-MDP-Com follows immediately. Messages are chosen from the agents' observations, and were tested with different levels of structure. The

local policies of action were assigned based on a locally goal-oriented mechanism [24]. That is, each agent acted towards a local goal that was computed as a function of the agent's own local observation and the interpretation of the message received.

The second domain (first described in [1]), implements the pumping-station example described in Section 2.1. Agents control a network of pumps, attempting to maximize network flow while minimizing the number of outflow ducts in use. In this case, messages concern both observations and actions. Local policies of actions are given to the agents based on a prior computation of the centralized solution assuming a mutual language. Translations are again updated based on the Bayes filtering algorithm. In both cases, the process is Markovian and therefore belief updates are based only on the last translation and last belief-state.

We note one important fact about these examples, before considering the details. In both sets of experiments, agents update translations until such a time as they have completely translated one another's languages. However, in neither case is the agent directly rewarded for a correct or complete interpretation. That is to say, while learning language facilitated the optimal solution of the relevant tasks, it was not in and of itself the main motive force. In the grid-location task, for instance, our agents happened to visit all squares of the grid over time, and so deriving optimal reward involved learning the vocabulary covering the entire grid. Nothing would have been changed, however, if one or both of the agents simply did not visit certain parts of the grid. In such a case, agents would work only on translating messages they actually received; portions of the vocabulary that covered unvisited grid-regions would not have been translated effectively, but this would make no difference to overall reward earned. Furthermore, as we show in consideration of the pump domain, the average rate of accumulated reward is very nearly maximized long before the entire language of each agent is entirely translated, suggesting a role for approximate methods that forego complete translation where it no longer brings appreciably greater reward (see the discussion centered around Fig. 10). In each case, then, agents are driven directly by the rewards in their environment, not by some special class of rewards specifically tied to the form of their translations—so far as language learning allows them to accumulate more such reward, they update their translations, and otherwise, they do not.

7.1. Gathering example

Our initial experimental results demonstrate the basic feasibility of the given approach to the language-learning problem, and illustrate how performance is affected by various features, such as the structure of the observations or of the language itself. We ran a number of tests involving the basic cooperative task for two agents in a simple gridworld environment, averaging results over 100 runs with random starting locations. Work proceeds in turns; on even-numbered rounds:

(1) Agent 1 is the actor, randomly choosing a square and sending a message to agent 2 to that effect.[3]

[3] It would be interesting to consider cases in which agents choose messages non-randomly. Here, if the agents had already learned some part of the language, then they might choose locations corresponding to unlearned messages (to facilitate learning), or already-learned ones (to get more reward), instead of choosing uniformly.

(2) Agent 2 is the translator, choosing a square based on its current translation of agent 1's language.
(3) Agent 2 receives an observation, OR, depending upon its choice, and updates its beliefs accordingly.

On odd numbered-rounds, agents 1 and 2 switch roles; the task continues until one agent has successfully translated the other's language. Currently, this involves one agent assigning unit probability to a *complete* translation table, i.e., every row contains exactly one entry with probability 1 (and the rest 0). Obviously, this could be modified to involve other, perhaps more tolerant, criteria of success.

In step (2) of the process, the choice of a square is based on the current message and belief-state of the translator. This belief-state is a probability distribution over translation tables; for each such table τ_i, let P_{τ_i} be the probability that τ_i is the correct translation. Each table τ_i maps components of received messages to possible meanings; for any complete received message σ_j (in the actor's language) and meaning σ_k (in the translator's own language), let $\tau_i(\sigma_k, \sigma_j)$ be the probability computed from table τ_i that (σ_j means σ_k). The most likely meaning of the message σ_j is thus calculated as that σ_k satisfying: $\max_k \sum_i P_{\tau_i} \cdot \tau_i(\sigma_k, \sigma_j)$. The various possible tables τ_i are chosen based on the presumed structure of the two given languages; the probabilities P_{τ_i} are updated using the Bayes-Filter algorithm.

Our first study involves a simple language of *unambiguous names*, as discussed in the example considered in Section 6.3. Each agent assigns each square in the grid a *fixed*, *unambiguous* name, meaning that each square has but one name, and each name corresponds to but one square. Each agent attempts to learn the other's mapping from names to squares. Observations are simple: translators sense 0 as the value of OR for correctly identifying the square chosen, and otherwise the observation's value is 1; translations are updated in either case. Fig. 5 charts the average number of turns as translator an agent takes to arrive at a complete translation for this language. The ratio of grid size to number of turns is relatively stable: for grid size $G \times G$, the number of turns is roughly $G^2/4$. We can see that agents pursue a decision-theoretically optimal course of action here. Translators initially assign messages a uniformly distributed probability of naming squares for which the name is not already known; further, after visiting a square, the translator knows the exact (0/1) probability that the latest message names it, based on its observation. Once some message σ_i is known to correspond to the name of square x, the translator always visits x upon receiving σ_i. Conversely, the translator never visits x once it knows that current message σ_j cannot name it. Finally, for messages not yet translated with certainty, the most probable translation is chosen (breaking ties randomly). Since the translation-action strategy is optimal with respect to expected reward, it will not in general be possible to do better in terms of average number of attempts before achieving a complete translation. That the algorithm does not converge a little faster is explained by the fact that the observation information gained by the translating agent is not shared with the acting agent; the actor may thus randomly select a square multiple times, even after the translator already knows how to translate the name given that square.

This example shows the elementary feasibility of the filtering approach, but we are also interested in testing more complicated languages and observation structures. Within the same gridworld context, we also investigate messages in a language of (x, y)-coordinate pairs. Due to this structure, translation updates carry more information than when simple names are concerned. However, the number of possible translations

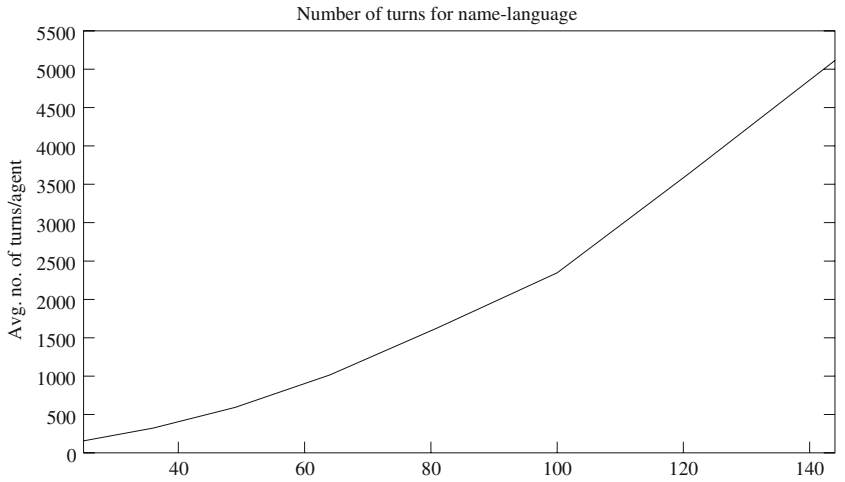


Fig. 5 Results for simple-name language

may increase drastically after each update. In the name-language, the observation after any action determined a single possible belief-state for the translator: either the translation of some name is known with certainty, or it is absolutely certain that one particular translation is incorrect. In a language of coordinate pairs, with a basic success/failure observation structure, things are not so simple. If selection of some square is successful, of course, the translation of message (σ_x, σ_y) is known with certainty, and any other possibilities for that pair are eliminated. If selection is unsuccessful, however, there are three options: either both translations of σ_x and σ_y are incorrect, or only one of them is. In the worst case, then, the essentially uninformative (0/1) observation structure can cause a large increase in the number of belief-states an agent has to entertain, affecting overall performance. Indeed, this increase may be so great that while translation uses a smaller number of *turns* to achieve certainty, overall *time* spent is much worse, as agents have to update many more beliefs in any given turn (see Table 1). In general the problem becomes potentially intractable for the combination of the coordinate-language and an essentially uninformative observation function.

To improve the situation, we consider other possible observation structures, which give the translator more information about what may have gone wrong. These are: OR **0/1/2**: agents observe a 0 if translation is correct, 1 if translation fails but one coordinate is translated correctly, and 2 if failure results because both coordinates are translated incorrectly; OR**0/1/2/3**: agents receive 0 if translation is correct, 1 if it fails but σ_x is translated correctly, 2 if it fails but σ_y is translated correctly, and 3 if failure results because both are translated incorrectly. The more informative observations greatly improve performance in terms of all problem dimensions: number of

Table 1 Two languages on a 25×25 grid

Language	Turns	Time (s)	Max. beliefs
Names	156	0.5	1
Coordinates	27	119.3	13,812

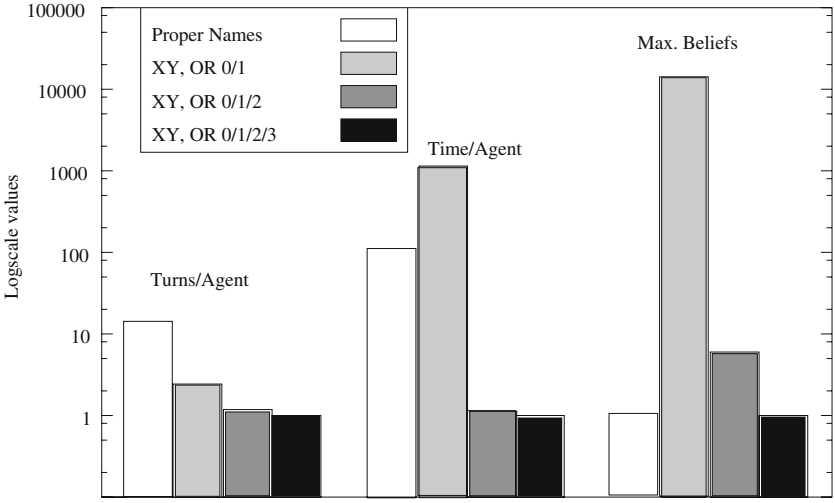


Fig. 6 Different languages on a 25×25 grid

turns before translation is successful, maximum number of beliefs that an agent must entertain at any time, and overall time of completion.

Figure 6 compares results for the name-language and for the different variations of the coordinate-language, in a 25×25 grid. Scale is logarithmic, with values normalized to the OR 0/1/2/3 case; results average over 100 random runs. In general, the extra information provided by the coordinate-language as opposed to the name-language significantly reduces the number of turns taken. Further, with respect to the coordinate-language, the more informative observation functions lead to significantly better results in terms of maximum number of belief-states ever updated in one pass of the filtering algorithm, and thus in terms of overall time taken. Such relations are not absolute, however. For instance, although the maximum number of beliefs is somewhat larger for the OR 0/1/2 case than for the name-language, the increase in information gained in the former case by using coordinates still decreases the number of turns enough to reduce overall time. In the single case shown, and in general, the performance of the two cases with more structured observations was essentially the same; while the 0/1/2/3 structure led to slightly improved performance, the differences were not significant (see Fig. 7). Finally, Fig. 8 shows the increase in the number of turns taken as the grid grows in size for both the name-language and for the coordinate-language with 0/1/2 observation structure. We did not test the name-language for very large grids, since the time taken quickly became unmanageable; in any case, the difference is evident for the limited range considered.

7.2. The network of pumps example

To explore the viability of our approach in more realistic settings, we implemented our language-learning protocol for a reasonably complex Dec-MDP-Com, based on the pumps domain discussed in Section 2.1. Each instance of the domain involves two (2) agents, each in control of a set of n pumps and m flow-valves in a factory setting, with parameters n and m varied to generate problem instances of different sizes. At each time step, each agent separately observes fluid entering the system from one of two

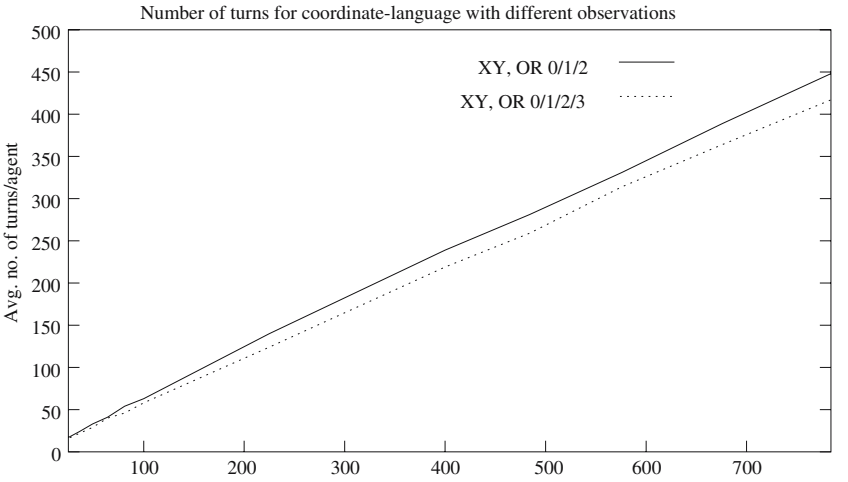


Fig. 7 Comparing observation functions

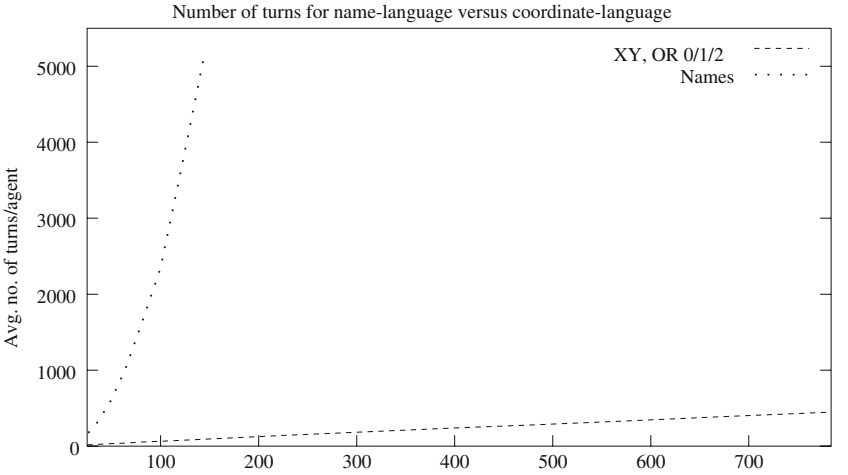


Fig. 8 Names versus coordinates

different inflow ducts, along with the pumps and valves under its own control. Agents seek to maximize flow out of the system through one of several outflow ducts while minimizing the number of ducts used. Probabilistic effects arise due to variations in throughput for ducts and valves, dependent upon whether the particular component was used to route flow in the prior time step. Any excess flow not routed through the system on a given time step is considered wasted, and is dropped from consideration. While agents in such a setting might be presumed to share an initial language for communication, there are cases where this does not hold, involving for instance diagnosis and trouble-shooting when they begin with imperfect or incomplete models of the environment [7], or where initial specifications of the domain were incorrect. In such cases, agents might begin with some common language, and only need to learn new meanings for particular expressions as they arose in practice. In order to establish the

general-purpose strength of our approach, however, our agents begin with completely different languages.

we note that this environment, while relatively complex, is in fact an example of a suitable Dec-MDP-Com. The problem is both freely describable, by the cost function C_Σ, and (for the purposes of solving the problem) fully describable, as given by the set of messages Σ. Furthermore, agents are aware of the overall structure of the pumping system, and can observe certain basic effects of each other's actions, by observing how many units of flow are routed through their own observable pumps and valves. These observations, combined with the observed total reward, allow them to reason backwards to what those actions may have been, as well as to the total number of units of flow entering the system through the other agent's inflow duct. While certain actions may fail to have the desired effect, given pump or valve failure, actions never affect the wrong pump or valve; furthermore, no pump or valve fails permanently. Thus, the observed effect of any action taken by the other agent will either completely confirm which action was taken, or give the agent no evidence to update its translation of the last message. Taken together, these conditions ensure that incorrect interpretations are eventually eliminated in favor of correct translations. While this requires agents to know the overall structure of the domain, this is simply the same assumption required for usual optimal offline methods of solving such problems, and so we consider it no real defect in our method.

Following the elementary action protocol, agents swap observation messages, choose and communicate actions based on their current beliefs, and then act, repeating the process to converge toward mutual understanding and optimal action. These experiments expand upon those of the previous section, both by including the language of actions as well as observations, and by extending the method to a appreciably more complicated domain. Using their model of the environment, agents update belief-states using the two-step Bayesian Filtering algorithm as before, first projecting possible belief-states before acting, then updating those beliefs given the results of their actions (see Section 6.1). Note that there is no update between the two sets of observation- and action-messages. Rather, the agents collect all their messages, then do the steps of the algorithm as before. That is, they first do the predictive update, generating possible next belief-states before taking the action; then they act, and do the retroactive update after the next observation comes in. Actions are chosen based on the translation as it exists following the observation on the prior action step.

Agents interact until each learns the language of the other with certainty—achieved when each agent α_i reaches a belief-state β_i with distribution $P_{\tau i}$, and for any message σ_j received from the other agent, there exists exactly one message σ_i such that $P_{\tau i}(\sigma_i, \sigma_j) = 1$. In this work, certainty provides a useful stopping condition, since the domain is one in which agents do in fact learn all of each other's messages in the course of optimizing action. We are now investigating cases in which complete certainty is not necessary, as when agents do not need to learn every part of another's language in order to achieve optimal performance, and convergence actually happens more quickly than where the entire set of messages is learned.

Our results show that the elementary protocol converges to optimal policies in each problem instance. Time of convergence depends upon the basic size of the problem, and thus the vocabulary of the agents necessary to describe all actions and observations, and also upon the frequency of certain rare states or actions. As conditions vary probabilistically, some states in the environment are encountered very infrequently, and agents do not learn related terms in the other's language. By design, we insured

that all states and actions are eventually encountered; current work also investigates cases where agents do not ever visit some parts of the state space, and so whole parts of the language are unnecessary to optimal action.

The most interesting and suggestive results concern the rates at which agents accumulate reward, relative to how much language they have learned. Figure 9 gives one example, for a problem featuring 100 vocabulary items for each agent. The graph shows the percentage of total accumulated reward, and total shared vocabulary, at each time step in the process of learning and acting. In a problem of this size, agents converge upon a complete understanding of one another, and are able to act entirely optimally from then on, in approximately 12,000 time steps, involving only a few minutes of computing time.

As can be seen, the language-learning process (top, dotted line) proceeds quite steadily. The rate of reward accumulation, on the other hand, grows with time. Initially, language learning outpaces reward gain given that knowledge, as agents still find many actions and observations of others hard to determine. After about 2,900 time steps, fully 25% of the language has been learned, but only just over 6% of the eventually accumulated reward. By the time 50% of the language has been learned ($\approx 6,200$ steps), things have improved somewhat, and some 27% of the reward has been earned. As time goes on, the rate of accumulation of reward actually increases to the point that it narrows the gap considerably, as agents now know much of what they need to communicate, and spend more time accumulating reward in already familiar circumstances, without learning anything new about the language of the other agent. Essentially the same curves, although differing in their time of convergence, are exhibited by problem instances of all sizes.

Figure 10 plots normalized average rate of reward accumulation over time. As can be seen, by far the largest proportion of the overall reward is accumulated early in the learning process. Such performance profiles reveal the possibility of approximate approaches, which may decide to stop the learning process early without sacrificing much in overall value. Such approximations are open topics for future research.

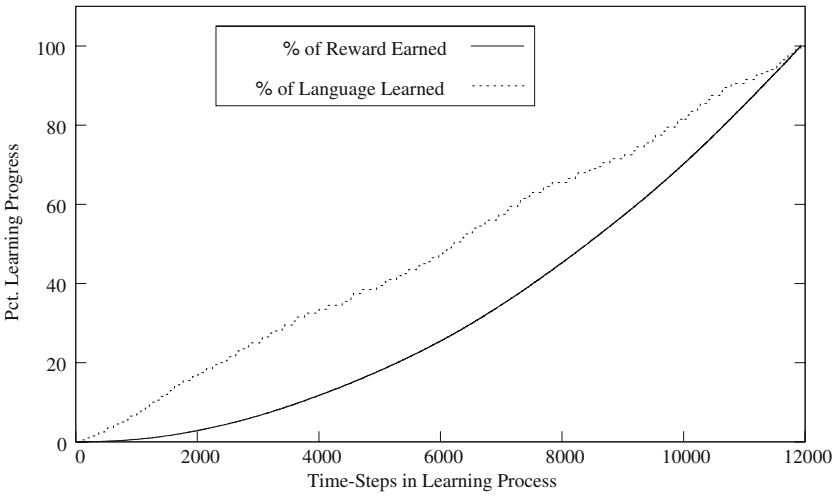


Fig. 9 Reward accumulated as language is learned

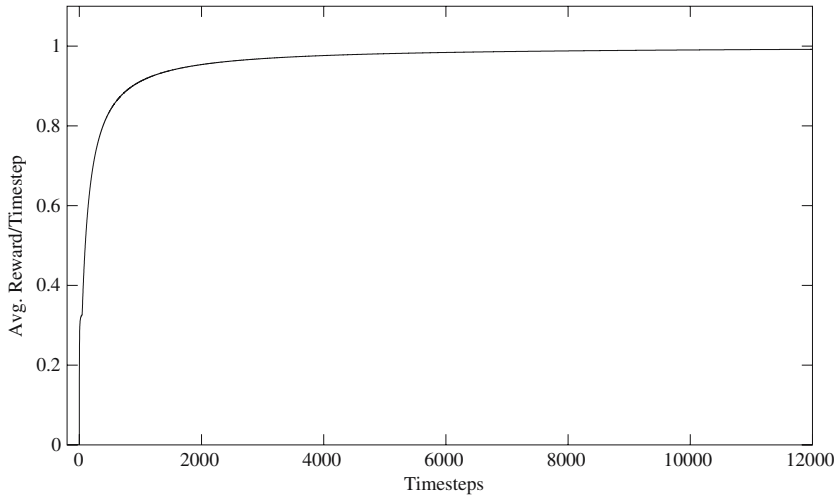


Fig. 10 Normalized average rate of accumulated reward

It is to be stressed that these results are first steps in the process of dealing with the problem of learning to communicate in decentralized settings. In particular, there are presently no ready candidates for comparison between different algorithms, since the communication-learning problem is somewhat new. Note also, that giving upper and lower baselines for these results is straightforward, but uninteresting. The lower bound on performance is given by agents who do not communicate at all. In such cases, since we are not working on solving the problem of learning how to act at the same time as learning how to translate, there is nothing more for the agents to learn. Given their model of initial starting conditions, they will simply choose actions maximizing expected value, and performance will be constant. (For instance, in the gridworld problems, agents will simply pick a square at random, guessing where the other agent is; for a grid of n squares in total, the expected average reward over time is then the constant function n^{-2}.) While it would be interesting to extend our work to cases in which agents still had learning to do even in the absence of learning about language, this would generally make the problem of translation updates prohibitively difficult; therefore, we leave that aside in the current work. Similar considerations apply to the case of upper bounds, given by agents who already share a language of communication. In such cases, as we have shown, the fully and freely describable nature of the problems makes generating optimal solutions straightforward, and agents will simply accumulate maximum expected average reward at all times. Again the plot of this reward will be a simple constant function. (For instance, the reward plotted for the pump-world case as shown in Fig. 10 would simply be the unit line, which the learning case already shown would converge towards.) Again, if we added learning how to act into our problems, this would be more interesting, but for now it is not.

8. Related work

We approach the problem of learning to communicate by integrating it into a general decentralized control process. Agents learn to interpret messages while acting

toward some global objective. We are interested in a learning process that will improve the agents' interpretations of each other's messages, leading consequently to optimal choices of action. This is a different approach from the reinforcement learning approach taken by Yanco [42] and Yanco and Stein [43] where a reinforcement signal is given explicitly for the interpretation of the messages. Moreover, her communication learning occurs in a stateless environment. We reinforce the agents only for their actions, which implicitly are chosen according to the agents' interpretation of the messages received. Our approach is thus more general and it can handle more complex language structures. We have also shown necessary and sufficient conditions under which learning to communicate can lead to optimal behavior. Yanco's work does not provide any finite convergence proof. Her tests, based on the interval-estimation algorithm [26] consider the frequency of the positive reinforcement obtained for messages and the actions taken as a consequence. Balch and Arkin [2] also study robotic communication. Their approach is inspired by biological models and refers to specific tasks such as foraging, consumption, and grazing. Agents are assumed to share a language, since their aim was not to study the language-learning problem.

Studies on the evolution of language in humans [5] and machines [28] have been recently pursued using computational models. Komarova and Niyogi [27] study how one language can be affected by other languages evolving in the same population. They also consider languages as probabilistic associations between form and meaning. Although agents are embedded in some world, the process of learning to communicate is not related directly to the agents' actions. In our work, on the contrary, this is fundamental, as agents learn to interpret each others' messages with respect to goals or some other coordinated behavior.

At the intersection between cognitive science and computational simulations, research has studied the evolution of lexicons and the problem of learning a language from positive examples [16, 39]. Here, we study how agents can learn to interpret each other's messages in order to improve global performance. Adaptive language games [36] and the anchoring problem ([9] and the cites therein) are also relevant areas of study. In particular, the latter work raises questions concerning formal models for the study of the problem. We present such a formal framework and a solution to the problem, formalizing the communication-learning problem as one of decentralized control. Our previous research has focused on the computation of optimal joint policies when a shared language of communication is assumed (in particular, *when* and *what* agents should communicate) [23]. Here, agents must learn such a language to optimize their joint behavior.

The distributed artificial intelligence community has studied how autonomous agents can coordinate their actions while acting in the same environment (e.g., see [13, 25, 35]). A known and fixed language of communication was assumed when communication was allowed. KQML [15] is an example of one standard designed to pre-set the possible communication between the agents. We believe that robust decentralized systems require that agents adapt their communication language when new situations arise or when miscoordination occurs possibly due to misunderstandings. Such miscoordination can either be revealed in practice, or in simulation, and serves as a signal for reinterpretation of messages received.

Other work has proposed that rational and self-interested agents can negotiate to evolve a shared communication language [18]. In such a context, conflicts arise because each agent prefers a distinct communication language, based on the cost of employing it and the local utility it may yield. We are interested instead in communication that

enables efficient coordination of agents toward a mutual goal. Communication serves to increase the overall utility of the system as a whole; the particular language learned will thus be directly related to this system-wide utility, rather than to the individual cost of using that language.

Our work relies on the idea that agents treat communicated messages as having some sort of *meaningful structure*. Initially, agents presume that others involved in a shared cooperative task are communicating information relevant to that task. This reduces the number of possible interpretations an agent has to consider, making the learning process more manageable. As well, treating messages as having meaningful structure speeds learning and allows for generalizations between various environments. Our experiments confirm the advantages and complications of this approach. Treating messages as having some semantic structure can allow agents to learn their meanings more quickly; at the same time, the specification of this structure and the learning updates related to it can become more difficult. The concentration on semantics distinguishes our approach from such as [40], in which a generalization of the perceptron algorithm was proposed to allow a multi-agent system to collectively learn a single shared concept, in a process that does not consider semantic structure explicitly.

Game theorists also consider the benefits of communication between players, although the focus is not usually on learning language. For example, Wärneryd [41], and Blume and Sobel [6] study how the receiver of a message may alter its actions in games where only one agent can send a single message at no cost.

Finally, philosophically, this work stems from thought originating with the American philosophers Quine, Davidson, and Putnam. Quine [32] argues that interpreting speakers of foreign languages is essentially the same as interpreting speakers of our own. On this view, we are always constructing "translation manuals" between one another's utterances, and we understand others by relating what they say and do to what we ourselves would say and do in the context of our shared environment. Further, translation is always under-determined by available evidence, and there are often multiple competing possible translations of another's language. Davidson [10] and Putnam [31] extend these ideas, considering how all understanding of others is a fundamentally indeterminate procedure of making and adjusting predictions about their behavior, and how changing context can alter the meaning of even apparently well-understood terms in a language. Together, these ideas suggest that designers of communicative agents must allow that even well-defined protocols and languages can lead to cases in which the interpretation of messages become ambiguous or error-laden, and must be adjusted in order to make coordination possible.

9. Conclusions and future work

The design of genuinely autonomous agents requires that those agents have the ability to learn how to interpret each other's messages, and consequently act adaptively. Multi-agent systems can be made more robust if they can autonomously overcome problems of miscoordination, arising when they encounter new situations or receive messages that are not completely understood. This paper presents and analyzes a formal framework in which agents learn to communicate while they are acting to maximize some global objective. Specifically, we have combined the problem of learning to communicate with that of maximizing overall system utility in a decentralized decision problem. We have explained how agents can, in general, improve their coordination

while improving their interpretation of messages exchanged in languages that they do not initially share. This model involves the notion of a translation between languages, and solves the problem of learning to communicate by adjusting these translations using probabilistic updating schemes. We lay the groundwork for further investigation by presenting and analyzing a formal decision-theoretic model of the problem, and by initiating empirical studies into algorithmic methods for solving it.

As shown, agents employing such update schemes can achieve value-maximizing policies so long as the decision problems they are working with have other important features. Prior work on decentralized MDP and communication has focused on messages that are composed simply of basic elements of the model, such as observations and actions. We have shown that even for such simple languages elements, the process of learning to interpret these messages in a desirable manner can become very difficult if our problem instances do not have the right structural properties. These features, such as free and full describability, along with our defined notion of suitability, are both necessary to, and sufficient for, the feasibility and success of the updates and learning methods we present, at least for the purposes of learning to communicate in languages composed of agents' observations and actions. We are interested in further research into more complex languages, allowing agents to convey other forms of information about the problems they are attempting to solve. While such richer languages may allow optimal policies to be enacted in a more efficient manner, they present additional difficulties for the learning process. We would like to investigate the possible benefits arising from more complex languages, and look at what new features of problem instances may be necessary if communication in such languages is to be learned.

Finally, our work so far has involved translation updates which converge to certainty, and lead to action policies that are optimal for all points encountered from the point of convergence onwards. This raises interesting questions. In particular, we are interested in the possibility of optimal behavior even in the absence of certainty about translation. Similarly, we are interested in the possibility of near-optimal behavior, and the trade-offs between optimality and certainty in the translation process. Our experimental results suggest that in many problems it will be possible to achieve very nearly optimal results without certainty about large portions of the language of communication. We are therefore interested in investigating approximate approaches to the problem of learning to communicate.

Acknowledgements This work was supported in part by the National Science Foundation under grants number IIS-0219606 and IIS-0535061 and by the Air Force Office of Scientific Research under grants number F49620-03-1-0090 and FA9550-05-1-0254.

References

1. Allen, M., Goldman, C. V., & Zilberstein, S. (2005). Learning to communicate in decentralized systems. In *Proceedings of the workshop on multiagent learning, twentieth national conference on artificial intelligence* (pp. 1–8). (AAAI Tech. Report WS-05-09). PA: Pittsburgh.
2. Balch, T., & Arkin, R. C. (1994). Communication in reactive multiagent robotic systems. *Autonomous Robots, 1*, 1—25.
3. Becker, R., Zilberstein, S., Lesser, V., & Goldman, C. V. (2004). Solving transition independent decentralized MDPs. *Journal of Artificial Intelligence Research, 22*, 423–455.
4. Bernstein, D. S., Givan, R., Immerman, N., & Zilberstein, S. (2002). The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research, 27*(4), 819—840.
5. Bhattacharjee, Y. (2004). From heofonum to heavens. *Science, 303*, 1326–1328.

6. Blume, A., & Sobel, J. (1995). Communication-proof equilibria in cheap-talk games. *Journal of Economic Theory, 65*, 359–382.
7. Bonarini, A., & Sassaroli, P. (1997). Opportunistic multimodel diagnosis with imperfect models. *Information Sciences, 103*(1–4), 161–185.
8. Boutilier, C. (1999). Sequential optimality and coordination in multiagent systems. In *Proceedings of the sixteenth international joint conference on artificial intelligence* (pp. 478–485). Sweden: Stockholm.
9. Coradeschi, S., & Saffiotti, A. (2003). An introduction to the anchoring problem. *Robotics and Autonomous Systems, 43*(2–3), 85–96.
10. Davidson, D. (1984). *Inquiries into Truth and Interpretation*. Oxford, England: Oxford University Press.
11. Decker, K. S., & Lesser, V. R. (1992). Generalizing the partial global planning algorithm. *International Journal of Intelligent Cooperative Information Systems, 1*(2), 319–346.
12. Doucet, A. (1998). On sequential simulation-based methods for bayesian filtering. *Technical Report CUED/F-INFENG/TR.310* Department of Engineering, University of Cambridge.
13. Durfee, E. H. (1988). *Coordination of Distributed Problem Solvers*. Boston, MA: Kluwer Academic.
14. Durfee, E. H., & Lesser, V. R. (1988). Using partial global plans to coordinate distributed problem solvers. In A. H. Bond & L. Gasser (Eds.), *Readings in Distributed Artificial Intelligence* (pp. 285–293). San Mateo, CA: Morgan Kaufmann Publishers, Inc.
15. Finin, T., Labrou, Y., & Mayfield, J. (1997). KQML as an agent communication language. In J. Bradshaw (Ed.), *Software Agents*, Cambridge, MA: MIT Press.
16. Firoiu, L., Oates, T., & Cohen, P. (1998). Learning regular languages from positive evidence. In *Proceedings of the twentieth annual meeting of the cognitive science society* (pp. 350–355). WI: Madison.
17. Gmytrasiewicz, P., & Doshi, P. (2005). A framework for sequential planning in multiagent settings. *Journal of AI Research, 24*, 1–31.
18. Gmytrasiewicz, P. J., Summers, M., & Gopal, D. (2002). Toward automated evolution of agent communication languages. In *Proceedings of the thirtyfifth hawaii international conference on system sciences*, Hawaii, p. 79.
19. Goldman, C. V., Allen, M., & Zilberstein, S. (2004). Decentralized language learning through acting. In *Proceedings of the third international joint conference on autonomous agents and multiagent systems* (pp. 1006–1013). New York, NY, 2004.
20. Goldman, C. V., Allen, M., & Zilberstein, S. (2006). Learning to communicate in a decentralized environment. *Technical Report UM-CS-2006-16, 2006*. Department of Computer Science, University of Massachusetts.
21. Goldman, C. V., & Rosenschein, J. S. (1994). Emergent coordination through the use of cooperative state-changing rules. In *Proceedings of the twelfth national conference on artificial intelligence* (pp. 408–413). WA: Seattle.
22. Goldman, C. V., & Zilberstein, S. (2003). Optimizing information exchange in cooperative multi-agent systems. In *Proceedings of the second international joint conference on autonomous agents and multi-agent systems* (pp. 137—144). Melbourne, Australia.
23. Goldman, C. V., & Zilberstein, S. (2004). Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research, 22*, 143–174.
24. Goldman, C. V., & Zilberstein, S. (2004). Goal-oriented Dec-MDPs with direct communication. *Technical Report UM-CS-2004-44*, Department of Computer Science, University of Massachusetts.
25. Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence, 86*(2), 269—357.
26. Kaelbling, L. P. (1993). *Learning in Embedded Systems*. Cambridge, MA: MIT Press.
27. Komarova, N., & Niyogi, P. (2004). Optimizing the mutual intelligibility of linguistic agents in a shared world. *Artificial Intelligence, 154*, 1–42.
28. MacLennan, B. (1990). Evolution of communication in a population of simple machines. *Knoxville, Technical Report CS90-99*, Department of Computer Science, University of Tennessee.
29. NASA (1999). Mars climate orbiter failure board report. *Available at*: ftp://ftp.hq.nasa.gov/pub/pao/reports/1999/MCO_report.pdf
30. Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY: Wiley.
31. Putnam, H. (1975). The meaning of 'meaning'. In *Mind, Language and Reality: Philosophical Papers* (pp. 215–271). Vol. 2 Cambridge, England: Cambridge University Press.
32. Quine, W. V. O. (1960). *Word and Object*. Cambridge, MA: MIT Press.
33. Serafini, L., & Bouquet, P. (2004). Comparing formal theories of context in AI. *Artificial Intelligence, 155*, 41–67.

34. Sharygina, N., & Peled, D. (2001). A combined testing and verification approach for software reliability. *Formal Methods for Increasing Software Productivity, LNCS, 2021*, 611–628.
35. Smith, R. G. (1988). The contract net protocol: High level communication and control in a distributed problem solver. In A. H. Bond & L. Gasser (Eds.), *Readings in Distributed Artificial Intelligence* (pp. 357–366). San Mateo, California: Morgan Kaufmann Publishers, Inc.
36. Steels, L., & Vogt, P. (1997). Grounding adaptive language games in robotic agents. In *Proceedings of the fourth european conference on artificial life* (pp. 474–482). UK: Brighton.
37. Sutton, R. S., & Barto, A. G. (2000). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.
38. Thrun, S., Fox, D., Burgard, W., & Dellaert, F. (2001). Robust Monte Carlo localization for mobile robots. *Artificial Intelligence, 128*, 99–141.
39. Vogt, P., & Coumans, H. (2003). Investigating social interaction strategies for bootstrapping lexicon development, *Journal of Artificial Societies and Social Simulation, 6*(1). http://jasss.soc.surrey.ac.uk/6/1/4.html
40. Wang, J., & Gasser, L. (2002). Mutual online concept learning for multiple agents. In *Proceedings of the first international joint conference on autonomous agents and multi-agent systems* (pp. 362–369). Bologna, Italy.
41. Wärneryd, K. (1993). Cheap talk, coordination, and evolutionary stability. *Games and Economic Behavior, 5*, 532—546.
42. Yanco, H. A. (1994). *Robot communication: Issues and implementation*. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.
43. Yanco, H. A., & Stein, L. A. (1993). An adaptive communication protocol for cooperating mobile robots. In J. A. Meyer, H. L. Roitblat & S. W. Wilson (Eds.), *From Animals to Animats: Proceedings of the second international conference on the simulation of adaptive behavior* (pp. 478–485). Cambridge, MA: MIT Press.