The Negotiation and Acquisition of Recursive Grammars as a Result of Competition Among Exemplars

John Batali
Department of Cognitive Science
University of California at San Diego
La Jolla, CA 92103-0515
batali@cogsci.ucsd.edu

June 3, 1999

1 Introduction

Of the known animal communication systems, human languages appear to be unique in their use of recursively characterizable structural relations among sequences of sounds or gestures and the meanings those sequences can be used to express.

The patterns of structural relations that recursion makes possible can serve a range of communicative functions, tremendously extending the expressive resources of the system. Certain structural relations may be used to express specific meanings, or to modify, or extend, or restrict the meanings conveyed by words and other simple constituents.

Despite the unbounded complexity it makes possible, a recursive communicative system can be learned relatively easily because the constituents of a complex construction may themselves be simpler instances of that same kind of construction, and the properties of complex constructions are often predictable from simpler counterparts.

The research described in this paper is an investigation of how recursive communication systems can come to be. In particular, the investigation explores the possibility that such a system could emerge among the members of a population as the result of a process I characterize as "negotiation," because each individual both contributes to, and conforms with, the system

as it develops. The members of the population are assumed to possess general cognitive capacities sufficient for communicative behavior, and for learning to modify their behavior based on observations of others. However they are given no external guidance about how their communication system is to work, and their internal cognitive mechanisms impose few specific constraints.

A specific model of the cognitive capacities required to participate in the negotiation of a communication system is presented in this paper. This model has been implemented as a computer program and used to simulate the negotiation process. The results of the simulations show that highly accurate recursive communication systems can indeed be developed through this process. The negotiated systems use structural properties of utterances to encode aspects of their meanings, often in ways that resemble syntactic processes in human languages.

Section 2 presents the assumptions underlying this research, and a summary of the computational model.

The formalism used to represent the meanings that agents in the computational model attempt to convey to each other is presented in Section 3. Section 4 describes how agents analyze relationships between signals and the meanings, and how they use such analyses as they communicate. The algorithms implementing the agents' learning abilities are described in Section 5.

Section 6 fills in some remaining details about the computational simulations, and describes the progress a typical simulation of the negotiation process. Analyses of several of the communication systems that emerge from the simulations are presented in Section 7.

2 Negotiated Communication Systems

I assume that the inventors of language were pretty smart. Their cognitive capacities almost certainly included the ability to categorize individuals and events, and to internally represent situations as involving one or more events and some configuration of participants. Individuals might have used such capacities to record past situations, to understand and respond to ongoing events, and to anticipate and plan for the future.

I also assume that the inventors of language were gregarious and mutually reliant, but not entirely naive. They watched each other closely. Sometimes their attention was based on concern, sometimes on curiosity, sometimes on fear. They learned enough from their observations to be able to to predict,

control, assist, thwart, and sometimes avoid, each other, with varying degrees of success.

Coordinated social activity required coordination of the representations that guided each individual's actions. A major impetus on the development of language was its use as an external and public medium for the expression of what had previously been private and internal. In particular, the ability to describe situations and events enables the members of a group to pool their perceptual abilities, to plan joint activities, and to entertain each other afterwards.

2.1 Systematic Communication

In the research described in this chapter, I treat communication as involving the transfer of information from one agent to another. One of the agents, who will be called the **sender**, begins an episode of communication with an internal representation M_1 of some meaning. The sender performs a publicly visible signaling behavior S in order to express M_1. The second agent, who will be called the **receiver**, observes the performance of S and interprets it as conveying a meaning the receiver represents as M_2. To the degree that the representations M_1 and M_2 are equivalent, the meaning was conveyed accurately, and the communicative episode is considered successful.

Communicative behavior is **systematic** if it occurs as the result of repeatable mechanisms in the agents and depends, at least partly, on meanings and signals having certain general properties, or instantiating certain types or categories. Therefore, if the sender were to express the same type of meaning in another situation, it might use the same signal.

If the communicative behavior of each member of a population is systematic, and if, as a result, the agents often communicate accurately with each other, I will speak of the population as possessing a **communication system**. A communication system may exhibit regularities involving meanings and the signals used by members of the population to convey them, and will often be characterized in terms of those regularities. However the system does not exist except as a manifestation of the individual behavior of the agents, and each agent might implement its communicative behavior differently.

A population's communication system will be called **conventional** if it is learned, and if, in principle, each member could have learned a different system. I assume that accurate communication is mutually beneficial for

both senders and receivers, at least often enough for the agents to attempt to convey meanings to each other, to interpret signals, and to learn how to improve their abilities at both tasks. In a population where this is true, a conventional communication system can result from a process of of **negotiation**, as the agents alternate between contributing to, and conforming with, the emerging system.

The negotiation of communication systems in which a set of simple signals is used to convey elements from a small set of meanings has been explored by researchers from a number of different fields. Lewis (1969), in philosophy, discusses the issue in the context of an investigation of conventional behavior. In economics, Spence (1973) Crawford and Sobel (1982), and Canning (1992) apply the mathematical theory of games. Hurford (1989), in linguistics, describes computational simulations comparing the effect that different learning proceures have on the communication systems that emerge from their use. In the field of robotics, Steels (1996) applies symbolic learning methods to explore the development and subsequent modification of communication systems. In cognitive science, Hutchins and Hazlehurst (1991) use neural-network learning algorithms in a simulation of the negotiation process; Oliphant (1997) explores various learning mechanisms, as well as the relations between communication and altruistic behavior.

2.2 Conventional Analyses of Structural Mappings

A conventional communication system could be a implemented as a simple table in which each meaning to be conveyed is associated with a unique signal. However if the set of meanings is large, agents won't get a chance to observe all of the entries, and wouldn't be able to remember them all anyway.

If the sets of meanings and signals are large, the agents probably don't use a distinct internal representation for each of them. Their representations are most likely constructed from a relatively small number of component types. The interpretation of an internal representation depends on the specific components use to construct it, and how they are configured. I will refer to this as the **structure** of the representation.

When an agent is given a meaning to express, it will, in general, use aspects of the structure of its representation of the meaning in its derivation of the structure of the signal used to express it. The receiver's interpretation of the signal will make use of the signal's structure to derive a representation of the meaning it conveys. The derivations performed by the sender and

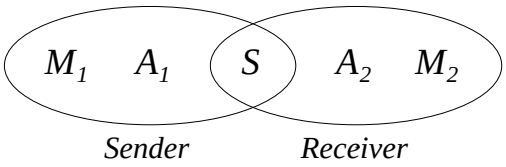


Figure 1: Communicating structured meanings. The sender derives the signal S it uses to express meaning M_1 according to analysis A_1 of the mapping from the structure of M_1 to S. The receiver derives its interpretation M_2 of the signal according to analysis A_2 of the mapping from the structure of S to M_2.

receiver therefore constitute implicit analyses of the relations between the structures of meanings and signals. The process is illustrated schematically in Figure 1.

The structural derivations performed by the sender and the receiver might have no relation to one another, except that they both involve the same signal, and, if the agents are are lucky, equivalent meanings.

For them to achieve better than chance accuracy, I assume that the members of a population obey a set of negotiated conventions regarding possible derivational relationships between the structures of signals and meanings.

Analyses of structural mappings between signals and meanings are used by learners as they attempt to discover the conventions the users of a communication system obey, and as they attempt to perform derivations of signals and meanings in accord with the conventions.

In general, a set of conventions might allow more than one analysis of the mapping between the structure of a meaning and a signal, and may allow more than one signal to be mapped to a given meaning, and vice versa. The set of conventions would therefore include procedures for determining which of a set of alternative analyses is preferred.

A communication system is negotiated by the members of a population as they learn from observations of each other's communicative attempts. The individuals analyze their observations as well as they can, and use their recorded analyses whenever possible, as follows:

> For a meaning M_1, a sender finds a signal S and an analysis A_1 of the derivation of S from M_1, such that according to the sender's observations, A_1 is licensed by the population's communication

system, and is preferred over alternatives with the same meaning. The signal S is performed by the sender to express M_1.

For a signal S, a receiver finds a meaning M_2, and an analysis A_2 of the derivation of M_2 from S, such that according to the receiver's observations, A_2 is licensed by the population's communication system, and is preferred over alternatives with the same string. The signal S is interpreted by the receiver as M_2.

The ability to create and manipulate mappings between the structures of representations may be an important aspect of cognition in domains other than communication. Gentner (1983) presents a theory of analogical reasoning that involves such mappings. A model of how structure mappings are computed is described by Falkenhainer, Forbus and Gentner (1989). An alternative model of analogy and perception, which also relies on the computation of structural mappings, is presented by Chalmers, French and Hofstadter (1992). An implementation of the model is described by Mitchell (1993).

2.3 Exemplars

Individuals must record analyses of their learning observations in a way that can be used to guide their subsequent attempts to express meanings and to interpret signals. This could take the form of a set of rules or principles that govern the allowed structural mappings between meanings and signals. These rules or principles would be induced from the set of observations the learner has analyzed, and would later be used to derive analyses of other signal/meaning pairs.

The alternative explored in this research is that learners simply store all of their analyzed observations as **exemplars**. No rules or principles are induced from them. Instead, exemplars are used directly to convey meanings and to interpret signals. An exemplar may be used intact, to express the meaning, or to interpret the signal, recorded in the exemplar. Exemplars may also be modified to construct new analyses of the mapping between a signal and a meaning.

The use of recorded exemplars based on individual observations, rather than rules or abstractions induced from them, is central to proposals made in a number of different cognitive domains. Medin and Schaffer (1978) and Barsalou (1989) have explored models of categorization based on recorded

exemplars. Aha, Kibler and Albert (1991) develop an "instance-based" theory of learning, and Hammond (1990) describes a "case-based" model of planning, in which stored solutions to previous problems are modified and reused. Bod (1998) develops a "data-oriented" model of parsing that uses a corpus of previously parsed examples to find the most likely structure of a sentence.

In general, exemplar sets are almost certain to contain inconsistent and redundant entries. Mechanisms must exist for choosing which of a set of alternative exemplars, or modified analyses based on them, will be used in a particular communicative episide. I assume that learning involves a process of **competition** among exemplars: Those that are repeatedly found to be consistent with learning observations become more likely to be used to create new signal/meaning mappings; Exemplars that are found to be inconsistent with learning observations, or with other exemplars, are used less often. This process is similar to that described by MacWhinney and Bates (1987) in a competition-based account of first language acquisition.

2.4 The Computational Model

These ideas are implemented in a computational model that is used to simulate the negotiation process and analyze its results.

Agents are implemented with a set of data structures and algorithms that correspond to the cognitive abilities required to participate in a negotiation. However the agents begin a negotiation with no shared communication system, and are given no guidance whatsoever as they develop one.

The agents begin the negotiation with the ability to manipulate internal representations of situations, called **formula sets**, which will serve as the meanings they convey to each other. The agents can also create and manipulate sequences of characters (called **strings**), which will be used as signals.

The mapping between a formula set and a string of characters is represented with a data structure called a **phrase**. A phrase may simply record that a given meaning was observed to be expressed with a given string, or it may involve an analysis of how the string is broken into constituents, and how their meanings are combined. Phrases are constructed and manipulated by the agents as they determine the strings they will use to express meanings, and as they interpret strings generated by others.

Phrases are also used by the agents as they learn to communicate. In

some situations, an agent is assumed to observe both the string sent by another agent, as well as the meaning that the other agent used that string to express. The learner creates one or more **exemplar** phrases to record this observation, and will use its set of exemplars to guide its subsequent communicative behavior.

Each phrase is assigned a numeric **cost**, whose value depends on how the phrase was constructed. Costs are used to implement the the choice of a preferred phrase among several that equally satisfy an agent's requirements. The primary algorithm that implements an agent's communicative behavior and its analysis of learning observations is a search through the agent's exemplars to find or construct the cheapest possible phrase with a specific meaning and/or string.

An agent can create a new phrase by combining and modifying some of its exemplar phrases. The cost of the resultant phrase equals the sum of the costs of the exemplars that were used, plus small additional costs for each modification step. If an agent has no exemplars, or none that satisfy its current search requirements, the agent can create entirely new phrases, or combine some of its exemplar phrases with newly created structure. The costs assigned to these new phrases are relatively high, so that the agents will tend to use exemplar phrases, or phrases that can be constructed out of them, if possible.

All exemplar phrases are given the same initial cost. An exemplar's cost may later be increased or decreased in subsequent learning observations, thereby changing the likelihood that the exemplar will be used by the agent. The cost of an exemplar is reduced if it are found to be consistent with exemplars used by other agents in the population, and an exemplar's cost is increased if it is found to be inconsistent with other exemplars the agent has acquired.

At the start of a negotiation, the agents have no exemplars to guide their communicative attempts, and so they just babble randomly and fail miserably to understand anything. As the agents acquire sets of exemplars, and as the costs of their exemplars are adjusted, their communicative accuracy steadily increases, and a shared system emerges.

	Formula Set	*Gloss*
a.	{(goose 1) (sang 1)}	A goose sang.
b.	{(insulted 1 2) (pig 2)}	(Something) insulted a pig.
c.	{(duck 1) (tickled 2 1) (cow 2)}	A cow tickled a duck.
d.	{(rat 1) (slapped 1 1)}	A rat slapped itself.
e.	{(goose 1) (sang 1) (noticed 1 2) (snake 2)}	A goose that sang noticed a snake.
f.	{(snake 1) (goose 2) (sang 2) (noticed 2 1) (bit 1 3) (moose 3) (danced 3)}	A snake that a goose that sang noticed bit a dancing moose.

Figure 2: Example formula sets.

3 Internal Representations

The agents begin a negotiation with the ability to create and manipulate internal representations of situations and their participants. I assume that such cognitive abilities are prior to, and independent of, language, at least for the simple kinds of meanings described here. Many species of mammals and birds exhibit behaviors that seem to require such representations, and a great deal of experimental and observational evidence suggests that apes and other primates use internal representations even more sophisticated than those described here. (Premack, 1983; Thompson, 1995; Pough, Heiser and McFarland, 1996.)

3.1 Formula Sets

In the computational model, internal representations are implemented as **formula sets**. Each **formula** is composed of a **predicate**, written as an English word, followed by one or two **variables**, which will be referred to as the formula's **arguments**, written as Arabic numerals. The predicate and arguments of a formula are written inside parentheses. Example formula sets are shown in Figure 2.

The variables used in a formula set are interpreted as designating participants some situation. A formula set's predicates are interpreted as designat-

ing properties of, and relations between, those participants. A given variable is interpreted as designating the same participant everywhere it occurs in a formula set.

In the manipulations of formula sets performed by the agents, the only distinction made among predicates has to do with the number of arguments they take. A predicate that takes a single argument will be referred to as a **property** whether it is the name of a kind of animal or an intransitive verb. A two argument predicate will be called a **relation**. The words used to represent predicates are entirely fanciful, and the modeled agents do no inference or other interpretation of formulae based on their predicates.

English sentences are used to gloss the meanings of the formula sets in Figure 2, and elsewhere in this chapter, to simplify the presentation and analysis of the model. However the formalism captures only those aspects of the meanings of sentences that involve characterizing configurations of actions, events, and participants. For most formula sets, a number of different sentences would serve equally well as glosses. For example, the meaning of formula set c in Figure 2 could be glossed as "A duck was tickled by a cow."

3.2 Renaming Arguments

Two formula sets that have exactly the same possible interpretations will be said to be **equivalent**. Two situation descriptions with the same set of formulae are equivalent no matter the order in which the formulate are written or stored. Two formula sets may also be equivalent if the variables in one of them can be uniquely renamed to yield the other. In the following examples, formula set 3.a can be used to describe a situation involving two participants, one of which, a cow, bit the other:

3.a {(cow 1) (bit 1 2)}
3.b {(cow 2) (bit 2 1)}
3.c {(cow 2) (bit 1 2)}
3.d {(cow 1) (bit 1 1)}

The same interpretation can be given to 3.b, and so the formula sets 3.a and 3.b are equivalent. Formula set 3.c also involves two participants in a biting incident, however the cow is the victim, not the perpetrator. In formula set 3.d, only one participant is involved. These two formula sets are therefore not equivalent to 3.a.

Formula Set	*Amap*	*Result*
{(rat 1) (sang 1)}	1 : 2	{(rat 2) (sang 2)}
{(cow 1) (bit 1 2)}	1 : 1 2 : 1	{(cow 1) (bit 1 1)}
{(fox 1) (chased 1 2) (duck 2)}	1 : 2 2 : 1	{(fox 2) (chased 2 1) (duck 1)}
{(chicken 1) (noticed 1 2)}	1 : 1 2 : 2	{(chicken 1) (noticed 1 2)}

Figure 3: Example argument maps. The argument maps in the center column are applied to the formula sets on the left to yield the formula sets on the right.

The operation of renaming the arguments in a formula set is formalized by introducing a set of operators called **argument maps**. An argument map is written as a two-column table of variables. When applied to a formula, each of the variables in the left-hand column of the table is replaced the variable in the right hand column of the same row. An argument map is **invertible** if no variable appears twice in its right-hand column, and is **non-invertible** otherwise. As just illustrated, the application of an invertible argument map to a formula set yields an equivalent formula set, while the application of a non-invertible argument map yields a formula set whose interpretation is more restricted. The set of argument maps also includes **identity argument maps** that that rename each variable to itself, and therefore yield formula sets that are identical to those they are applied to. Example argument maps are shown in Figure 3.

3.3 Combining Formula Sets

Two formula sets can be combined by taking their union. The result can be used to represent a more complex situation.

In the following examples, 3.e represents a situation in which a rat was present, and 3.f represents a situation in which something sang. The formula set obtained by combining them, 3.g represents a situation in which the rat

was the singer.

3.e {(rat 1)}
3.f {(sang 1)}
3.g {(rat 1) (sang 1)}

To represent a particular configuration of participants in a situation, the arguments of one or both of a pair of formula sets may be renamed with argument maps before their formulae are combined. For example, formula set 3.g could be combined with 3.h to yield 3.i, to represent that a singing rat was chasing something. If the variable in 3.g is renamed with the argument map $\boxed{1:2}$ before the formula sets are combined, the result, 3.j, can be used to represent a situation in which the sonorous rodent was the object of pursuit.

3.h {(chased 1 2)}
3.i {(rat 1) (sang 1) (chased 1 2)}
3.j {(rat 2) (sang 2) (chased 1 2)}

3.4 Formula Sets as Meanings

The agents in the computational simulations are given formula sets as meanings to convey. Formulae are constructed from a set of 22 property predicates and 10 relation predicates. The formula sets given to the agents are randomly selected from candidates that contain from two to seven formulae and involve at most three different variables. (So the last example in Figure 2 is as hard as they get.)

There are 2.3×10^{13} different (i.e., non-equivalent) formula sets that may be selected. This number represents the magnitude of the task the agents face in negotiating a communication system. They must somehow develop a way to express any of these possible meanings after having attempted to convey only a small subset.

Although there are a lot of them, the representational capacities of formula sets are extremely impoverished. They can't be used to represent propositional attitudes, counter-factual situations, or anything else that involves embedded meanings.

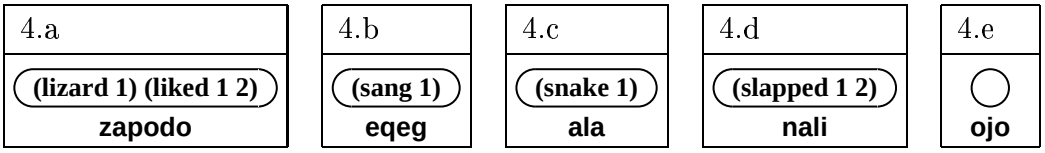


Figure 4: Tokens. A token contains a formula set, printed inside a balloon, and a string of characters, printed below the balloon.

The decision to use such a restricted formalism was made deliberately. Any formal system for representing embedded meanings would have to include operators for creating and manipulating embedded meanings, along with rules governing the use of those operators. The agents' internal representations would therefore have recursive structure. It is possible that the syntax of the formalism for representing embedded meanings might influence the ways the agents find to express such meanings in their communication systems. This possibility is consistent with theoretical proposals about the relationship between syntax and semantics (for example in the work of Montague, 1974), but can't occur in this model. Any recursive regularities in the systems the agents develop to express these non-recursive meanings will be entirely of their own invention.

4 Phrases

Analyses of mapping between formula sets and strings are represented with data-structures called **phrases** that the agents create and manipulate as they communicate, and as they record their observations of each other's communicative attempts.

4.1 Tokens

A simple phrase, which will be called a **token**, contains only a formula set and a string, with no further analysis of the structure of either. Some example tokens are shown in Figure 4.

A token's formula set may contain any number of elements, including zero. If its formula set contains a single element, the token will be referred to as a **singleton token**. Examples 4.b, 4.c, and 4.d in Figure 4 are singleton tokens. If the predicate in a singleton token's formula takes one argument,

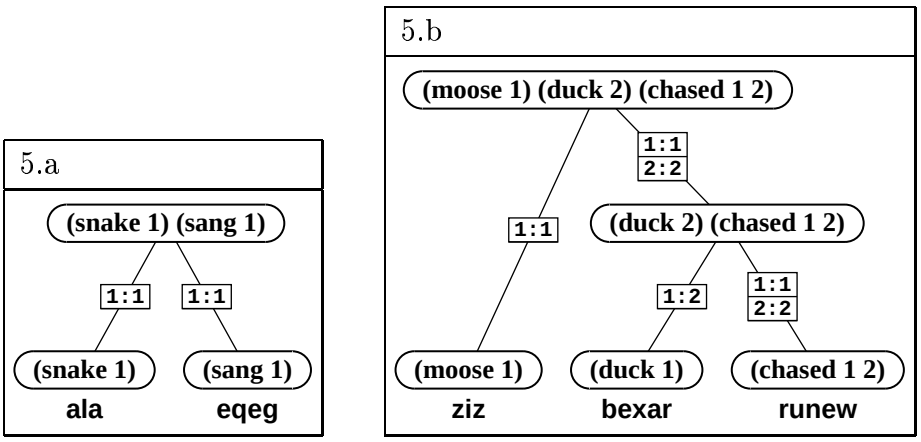


Figure 5: Complex Phrases. The left and right subphrases of a complex phrase, and the argument maps applied to each constituent's formulae, are drawn below the balloon containing the complex phrase's meaning.

the token will be called a **property token**; if the predicate in a singleton token's formula takes two arguments, the token will be called a **relation token**. Examples 4.b and 4.c and are property tokens, and 4.d is a relation token. Tokens whose formula sets contain no elements will be called **empty tokens**; an example is 4.e.

4.2 Complex Phrases

A **complex phrase** is used to record a structural analysis of how the mapping from a string and meaning depends on the meanings and strings of a pair of constituents. The mapping is represented as involving the combination of two other phrases, will subsequently be referred as its **left-subphrase** and **right-subphrase**. The complex phrase's string is obtained by concatenating the strings of its left and right subphrases. The meanings of the constituents are combined according to the technique described in Section 3.3: An argument map is applied to the formulae of each phrase's meaning, and the resulting formulae are combined to obtain the formula set of the complex phrase. The constituents of a complex phrase may be tokens, or they may be other complex phrases. Example complex phrases are shown in Figure 5.

I will use the term "phrase" to mean either tokens or complex phrases, unless the discussion is specific to one or the other. When discussing the

structure of a complex phrase, I will use the term **subphrase** to refer to a complex phrase's left or right subphrase, or one of their constituents.

4.3 Exemplars

Phrases are used to represent the structural mapping between a meaning and a string used to express it. For a shared communication system to emerge, the agents must come to agreement on how these mappings are constructed and analyzed. They do so by recording their observations other agents communicative attempts, and by using their recorded observations in their subsequent communicative behavior.

An **exemplar** is a phrase created by an agent to record an observation of another agent using a string to convey a meaning. An exemplar phrase may consist simply of a token with the observed string and meaning. Such an exemplar is nothing more than a literal record of the observation. Alternatively, the learner may construct a complex phrase with the given string a meaning. In this case the exemplar phrase records the learner's analysis of how the observed string and meaning might have been been obtained by combining simpler phrases.

Each exemplar has a numerical **cost**. New exemplars are all given a cost of 1.0, and exemplar costs are subsequently modified by the learning procedures described in Section 5.2. Exemplar costs are used to decide among alternatives as agents use their exemplars to construct phrases to communicate and to learn to communicate.

4.4 Combining and Modifying Exemplar Phrases

Given a set of exemplar phrases and their associated costs, it is possible to create new phrases from them. Each new phrase will be assigned a cost that depends on the costs of the exemplars used to construct it, and the specific modifications that were made to them.

Any pair of exemplar phrases may be combined into a new complex phrase by specifying the order in which the phrases are to be combined, and the pair of argument maps to be applied to the formula sets of each constituent, as described above. The cost of the new complex phrase equals the sum of the costs of the constituent phrases, plus the value of the "New Structure Cost" parameter, 1.5.

An exemplar phrase may be modified by replacing one of its subphrases with another phrase. The modified phrase's string is obtained by replacing the string of the subphrase that was removed with the string of the phrase that replaced it. The modified phrase's meaning is obtained by removing any formulae contributed by the replaced phrase from the meaning of the original phrase, and replacing them with the formulae of the new subphrase's meaning, after applying the argument maps above the modified phrase. The cost of the new phrase thus created equals the sum of the costs of the two phrases that were used, plus the value of the "Replacement Cost" parameter, 0.1. The process is illustrated in Figure 6.

A phrase can replace a subphrase of another phrase provided that the replacement phrase's formula set uses the same set of variables as the subphrase it replaces. I will refer to this restriction as the **Replacement Condition**. If it is satisfied, the argument maps that were applied to the variables in the deleted phrase's meaning can be applied to those in the phrase that replaces it, and the meaning of the modified phrase is the same as it would have if the phrase had been constructed from its token constituents.

4.5 Searching a Set of Exemplars

The algorithm that underlies all communicative behavior in the model involves a search over a set of exemplars for a phrase with a given meaning or string. The string or meaning thus given will be called the **target** of the search, and a phrase whose meaning or string matches the target of a search will be called a **solution** to the search. For some targets, the set of exemplars will contain one or more phrases that can be used intact as solutions. In other searches, new phrases must be created by combining and modifying exemplars. Whenever more than one solution to a search can be found or constructed, the cheapest alternative is selected.

When an agent acts as a sender in a communicative episode, it searches its set of exemplars for the cheapest phrase with the meaning it is given to express. The string from that phrase is passed to another agent, who searches its own set of exemplars for a phrase with that string. The formula set of the receiver's phrase represents the receiver's interpretation of the meaning of the string.

An agent also searches its set of exemplars to record a learning observation of a string being used to express a meaning. The agent locates the cheapest phrase with both the observed string and meaning, and uses its solution to

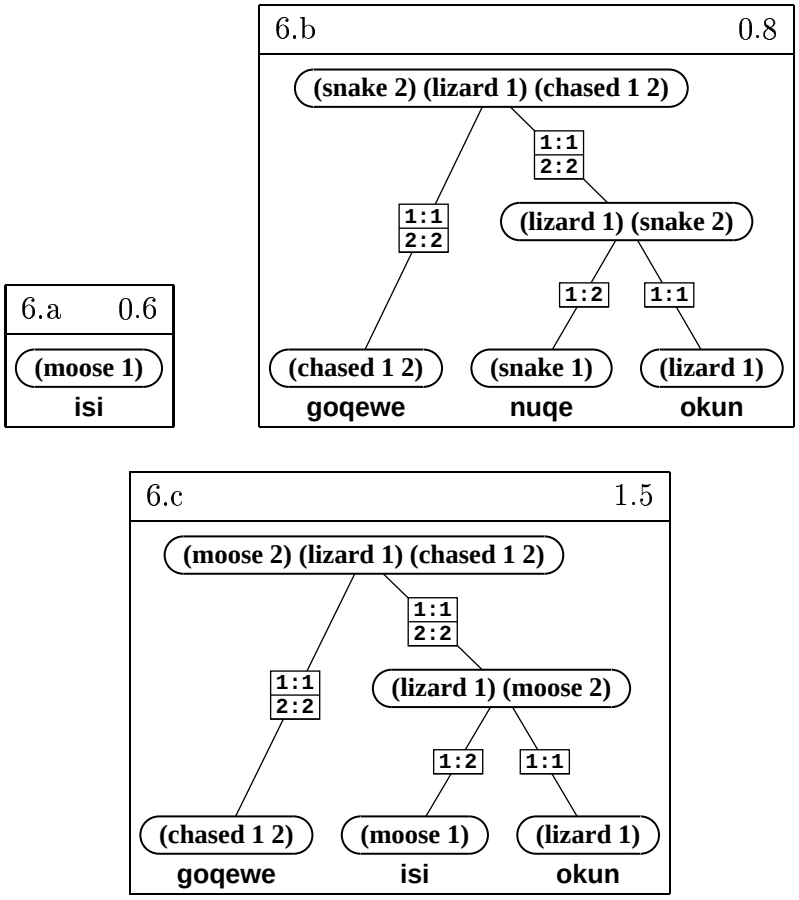


Figure 6: Modifying a phrase. Phrase 6.a replaces the token with the string '`nuqe`' in phrase 6.b, yielding 6.c. The cost of each phrase is shown at its upper right.

construct new exemplars, and to guide the adjustment of the costs of other exemplars, as described in Section 5.

An exemplar set may not contain phrases with some of the strings and/or meanings required to satisfy a given search target. (This will occur, for example, at the start of a negotiation, before the agents have acquired any exemplars at all.) In such cases, an entirely new token with any required string and meaning may be created. The cost of a new token equals the sum of the number of characters in its string and the number of formulae in its meaning. The new token is added to the exemplar set for the duration of the search in which it was created and is discarded afterward.

If an agent is given a meaning to express, and needs to create new tokens because it has none with one or more of the formulae in the meaning, it will generate a random string, consisting of alternating vowels and consonants, and create a new token with that string and the required formula or formulae.[1] If an agent is given a string to interpret, and has no token exemplars with that string, or one of its subsequences, the receiver will create a new token with an empty meaning. Learners also create new tokens if needed, and such tokens may be recorded as exemplars.

The values of the Replacement Cost and New Structure Cost parameters, and the costs assigned to new tokens, are such that agents will, in general, rely on their exemplars as they communicate and analyze their learning observations. If a search's requirements can be satisfied by using exemplar phrases, or a phrases that can be constructed by modifying exemplar phrase, such solutions will almost always be cheaper than phrases that require the creation of new tokens or complex phrases.

4.6 The Origins of Phrases

While it may be plausible that the ability to create and manipulate internal representations of situations and participants, described in Section 3, is prior to, and independent of, communication, the same cannot be said of the data structures and algorithms described in this section. The model of tokens and phrases is based on human language, and is intended to be as simple as possible, consistent with the following generalizations about human languages:

[1] The alternation of vowels and consonants in newly created random strings is not based on any deep theoretical motivation, but is done so that the agents' strings can be pronounced by the Festival Speech Synthesis system developed at the University of Edinburgh (Black and Taylor, 1997); URL: `www.cstr.ed.ac.uk/projects/festival`.

1. Utterances tend to be linear sequences of sounds or gestures.

   This generalization is implemented in the strings the agents use as signals.

2. The constituent structure of utterances tends to group adjacent elements.

   Implemented in the binary-branching structure of complex phrases.

3. The meanings of utterances often depend on the meanings of their constituents.

   Implemented in the operatation of combining the formula sets of the constituents of complex phrases.

The abilities to combine tokens into complex phrases, and to further combine complex phrases, could be seen an implementation of what Chomsky (1975) refers to as the "structure dependency" of the analyses performed by humans as they learn a language. The fact that all known human languages rely on structure dependent rules is used by Chomsky as evidence for the existence of innate computational mechanisms, useful only for language, and found only in humans.

An alternative, argued by O'Grady (1987) recognizes that the human ability to learn and use language might require biological support, but suggests that "the contributioon of the genetic endowment is restricted to the specification of concepts and learning strategies that are required independent of language" (p. xi).

As discussed in Section 2, the mechanisms of structural mappings and exemplar-based learning are similar to those proposed in the domains of perception, analogical reasoning, and planning, and so might turn out to be the result of cognitive mechanisms of general utility, consistent with O'Grady's suggestion.

Although the use of tokens and phrases with branching structure might not seem to be of general cognitive utility, these properties of language might occur as solutions to the problem of encoding complex meanings into linear sequences. In previous research (Batali, 1998) agents negotiating a system for communicating meanings that had a predicate/argument structure, developed signals in which the predicate tended to be expressed at the start of the signal, and the referent at the end.

In any case, although the agents begin a simulation with the ability to use embedded phrase structure, they are unable to communicate with each other unless and until they negotiate a shared system. Whether and how they do so is the main focus of this research.

5 Learning

The agents acquire their exemplars by recording observations of other agents expressing meanings. A learner finds the cheapest phrase with the observed string and meaning that can be created by combining or modifying phrases from its existing set of exemplars, creating new tokens and phrases if necessary.

Section 5.1 presents a sequence of observations made by a hypothetical agent at the beginning of a simulation, and the exemplars that are constructed to record those observations. The algorithms for subsequently modifying exemplar costs are described in Section 5.2.

5.1 Acquiring Exemplars

The agent begins the negotiation with no exemplars. If it is given a meaning to express before it has made any learning observations, the agent will create a new token containing a random string. The agent makes no record of the new token it just created, and so if were later given the same meaning to express, it would almost certainly create a token with a different random string. If the agent is given a string to interpret before it has any exemplars, it will create a new empty token, and simply fail to interpret the string as conveying any meaning at all.

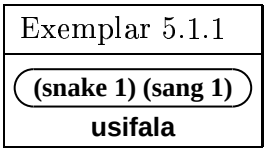
5.1.1 An Initial Observation

Our agent's first learning observation is of another agent using the string '`usifala`' to express the meaning {(snake 1) (sang 1)}.

Observation 5.1.1

`usifala`	{(snake 1) (sang 1)}

A new token is created with this string and meaning, and recorded as the agent's first exemplar. All new exemplars are given a cost of 1.0.



If the agent is now chosen to be the sender in a communicative episode, and is given the meaning {(snake 1) (sang 1)} to express, it will find the exemplar just created as a solution to its search. Any new token with the meaning would cost at least 2.0, and so the exemplar token will be cheaper. The agent will therefore use the string 'usifala' to express the meaning.

If the agent is later selected as a receiver, and happens to be given the string 'usifala' to interpret, it will again find exemplar 5.1.1 as a solution to its search. An empty token with this string would cost 7.0, so the agent will interpret the string as expressing the meaning {(snake 1) (sang 1)}.

There is no guarantee that this is the meaning another agent used this string to express. The observed agent may have been using a random string the other agent just created. But there is some chance that our agent's interpretation might be correct. After all, exemplar 5.1.1 was created because another agent was observed using that meaning and string, and other agents may have made similar observations.

5.1.2 Constructing a Complex Phrase

The agent next observes:

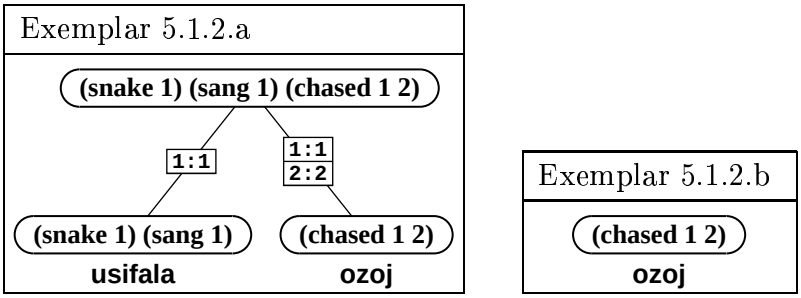
Observation 5.1.2

usifalaozoj	{(snake 1) (sang 1) (chased 1 2)}

The agent has no exemplars with exactly this string and meaning. However its one exemplar covers the the first seven characters of the string, and two of the three formulae of the meaning.

If that exemplar were used to analyze the first part of the string as {(snake 1) (sang 1)}, the remainder, 'ozoj', would be analyzed as contributing the formula (chased 1 2).

The agent creates new token with for this string and meaning, and combines the new token and its token exemplar into a complex phrase, using identity argument maps for both subphrases.



This solution is used to create two new exemplars. One of them corresponds to solution phrase, and one is created from the newly created token. In general, exemplars are created for all subphrases of a learner's solution phrase except those that are unmodified exemplars.

5.1.3 Using a Complex Phrase Exemplar

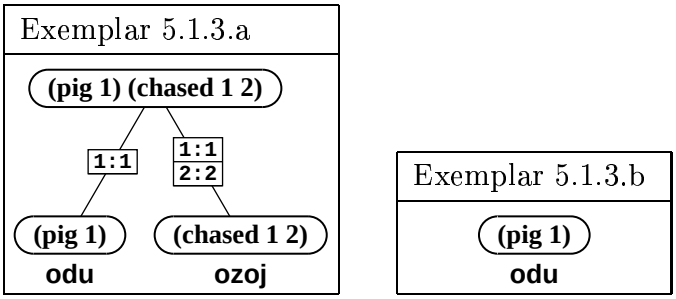
The agent next observes the pair:

Observation 5.1.3 | oduozoj | {(pig 1) (chased 1 2)} |

As with the previous observation, the agent's exemplars cover part of the observed string and meaning. A new token is created with the string 'odu' and the meaning {(pig 1)}.

This time the agent need not create a new complex phrase. Instead, the complex phrase exemplar it just created can be modified by replacing its left subphrase with the new token.

Two new exemplars are created:



The previous two examples illustrate how the acquisition of even a few complex phrase exemplars facilitates the acquisition of singleton tokens. Singleton tokens are extremely useful for communication because they can be inserted into phrases whenever a specific formula is part of a meaning to be expressed.

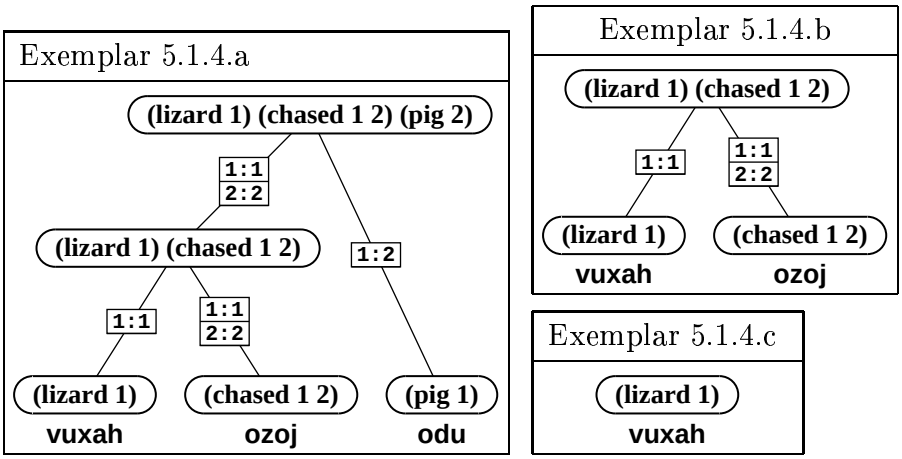
5.1.4 Renaming an Exemplar's Variable

The next observation is:

Observation 5.1.4

vuxahozojodu	{(lizard 1) (chased 1 2) (pig 2)}

A new token with the string '`vuxah`' and meaning {(lizard 1)} is created, and used to replace the left-subphrase of one of the agent's complex phrase exemplars. The agent has a token with the remainder of the observed string, '`odu`', but the variable in that token's formula must be renamed from 1 to 2 using a newly created complex phrase.



5.1.5 Bootstrapping

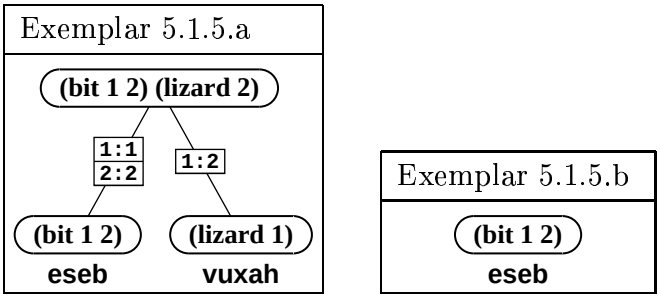
The agent's next observation is:

Observation 5.1.5

esebvuxah	{(bit 1 2) (lizard 2)}

The agent's exemplar 5.1.4.a can be used to combine a newly token with the string '`eseb`' and meaning (bit 1 2) with the exemplar token '`vuxah`'.

The agent has not yet seen a string where a relation token is put first, followed by a property token. However the system emerging from its exemplar set is consistent with interpreting the second token as expressing a property of variable 2.



5.1.6 Creating an Empty Token

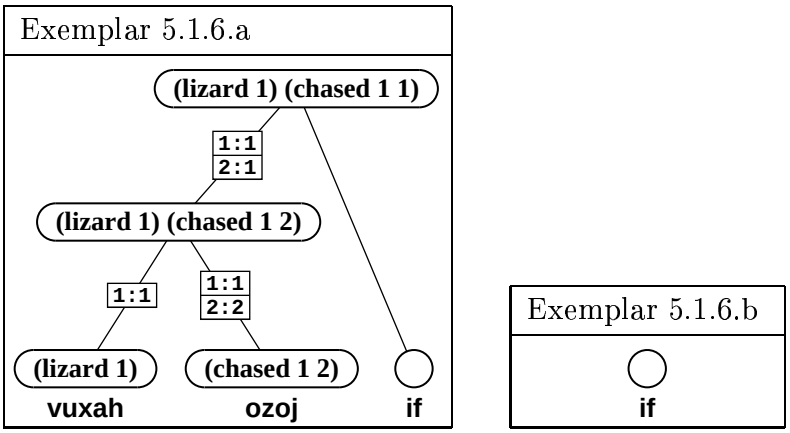
The agent next observes:

Observation 5.1.6

vuxahozojif	{(lizard 1) (chased 1 1)}

The agent's exemplar 5.1.4.b covers all but the last two characters of the observed string. If the the variable 2 were mapped to 1, the meaning of that exemplar would match the observation. However the remainder of the observed string doesn't seem to be contributing anything.

The agent creates an empty token with the string '`if`', and a new complex phrase with the required argument maps.



5.1.7 More of the Same

As its observations continue, the agent accumulates more exemplars, using its existing exemplars to guide its creation of new ones. The rest of the agents

in the population are doing likewise, and in some cases, their observations are of this agent's communicative attempts.

The sequence presented here is fictional, but corresponds in its essential details to what occurs in simulation runs. The main difference between this set of examples, and what occurs in a real simulation, is that the agent in the examples is stupifyingly lucky. The specific sequence of observations described below will never occur so quickly, or in such a convenient sequence. But eventually, agents will make observations, and acquire exemplars, similar to those presented.

Once they have begun to acquire complex phrase exemplars, the agents soon do so at a very rapid rate, and the members of a population begin to succeed occasionally at conveying meanings. The use of complex phrases to analyze observations can simplify, and thus accelerate the agents' acquisition of singleton token exemplars. Soon after they start using complex phrases, a population of agents begin to negotiate a shared set of singleton tokens that one might refer to as the "vocabulary" of their communication system. The existence of a shared vocabulary enables them to come to similar agreement on sets of complex phrase exemplars that encode a productive "grammar" of their communication system.

5.2 Reinforcement & Discouragement

As an agent continues to record learning observations, its exemplar set will accumulate redundant and contradictory elements. The other agents whose observations are observed may be randomly babbling. Even if they aren't, the learner's analyses of its observations are not likely to have much resemblance to the phrases that were used to produce them (other than having the same strings and meanings). The phrases used by the sender and by the learner may have different meanings assigned to different constituents, and in fact might not even agree as to where the boundaries of the constituents are, or how they are combined. The search algorithm makes random choices when it assigns meanings to new tokens, and when it chooses how to combine the constituents of new complex phrases.

From this mess, the agents must select a subset of their exemplars to guide their communication. This is done by decreasing the costs of some exemplars and increasing the costs of others.

5.2.1 Reinforcing Shared Exemplars

Decreasing an exemplar's cost is called **reinforcing** the exemplar. When an exemplar is reinforced, its cost is reduced according to the formula:

$$C_A = C_B - r\,C_B \tag{1}$$

where C_B is the exemplar's cost before it is reinforced, r is the value of the "Reinforcement Rate" parameter, 0.1, and C_A is the new cost assigned to the exemplar.

An exemplar is reinforced when it is used in the phrase an agent constructs to record a learning observation. For example, the agent's first complex phrase exemplar, 5.1.2.a, would have been reinforced when it was used to create the solution phrase for learning observations 5.1.3 and 5.1.4, so its cost after the observations described in Section 5.1 would be 0.81.

The more often an exemplar is part of a learner's solution phrase, the more likely it is that other agents share similar exemplars. The lower costs of reinforced exemplars will result in their being used more often in communicative situations, as well as to record subsequent learning observations. The other agents are also reinforcing exemplars that are used to analyze their learning observations, so the process of reinforcement tends to guide the population towards similar exemplars with similar costs.

5.2.2 Discouraging Inconsistent Exemplars

Increasing an exemplar's cost is called **discouraging** the exemplar. When an exemplar is discouraged, its cost is modified according to the formula:

$$C_A = C_B + d \tag{2}$$

where C_B is the exemplar's cost before it is reinforced, d is the value of the "Discouragement Rate" parameter, 0.1, and C_A is the new cost assigned to the exemplar.

As it records a learning observation, an agent may discover that its current set of exemplars is inconsistent with the observation. To locate such inconsistencies, and to discourage the exemplars responsible for them, an agent performs two searches of its exemplars when it makes a learning observation. In addition to finding the cheapest phrase with the observed string and meaning, the agent also searches its exemplars for any cheaper phrases whose strings match the observation, but whose meanings do not.

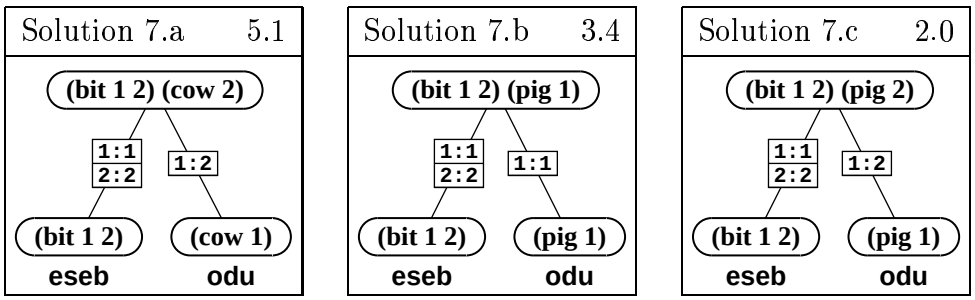


Figure 7: Solution phrases for inconsistent learning observations. Solution 7.a is the cheapest phrase that the learner of Section 5.1 can construct to match Observation 5.2.1. Solution 7.b is the cheapest phrase for Observation 5.2.2. Solution 7.c is the cheapest phrase with the string '`esebodu`', independent of meaning. The cost of each phrase, shown at its upper right, reflects adjustments to exemplar costs due to reinforcement.

For example, suppose that the agent of the previous section were to make one of the following learning observations:

Observation 5.2.1

`esebodu`	{(bit 1 2) (cow 2)}

Observation 5.2.2

`esebodu`	{(bit 1 2) (pig 1)}

To match learning observation 5.2.1, the agent must create a new token pairing '`odu`' with {(cow 1)}, and use it in solution phrase 7.a. To match learning observation 5.2.2, the agent must create a new complex phrase, 7.b, which contains an argument map that inverts the variables of its left subphrase. Given the string '`esebodu`' alone, the agent would use exemplars 5.1.5.a, and 5.1.3.b to create solution 7.c.

If the agent were acting as receiver, and given only the string '`esebodu`', it would use solution 7.c to interpret string as {(bit 1 2) (pig 2)}. If the actual meaning expressed were that of learning observation 5.2.1 or 5.2.2, the agent would interpret the string incorrectly. To reduce the likelihood that the inconsistency will result in its misinterpretation of subsequent communicative attempts, the agent determines which of the exemplars used to construct solution 7.c are responsible for its having the wrong meaning, and those exemplars are discouraged.

If the learning observation had been 5.2.1, the incorrect meaning would be blamed on the use of the token exemplar 5.1.3.b, which pairs the string

'odu' with the meaning {(cow 1)}. If the learning observation had been 5.2.2, the complex phrase exemplar 5.1.5.a would be blamed, because both tokens are consistent with the correct meaning but are not renamed correctly by the complex phrase's argument maps.

The increased cost of discouraged exemplars makes them less likely to be used. This improves the agent's accuracy in interpreting strings generated by others, as well as making its communicative behavior more reliable as a source of learning data for the other agents.

5.2.3 Competition

Some of an agent's exemplars will be repeatedly reinforced, and rarely if ever discouraged. Such exemplars will used more and more often, and will form the basis of the agent's communication system. Other exemplars will be used rarely or never, usually because there are cheaper alternatives.

Most of an agent's exemplars will be alternately reinforced and discouraged, and their costs will fluctuate around the cost of new exemplars. For this to occur, the exemplar has to be cheap enough to be used at all, but there must be some other exemplars, also in active use, that are inconsistent with it. The reinforcement and discouragement therefore implements a competition among groups of exemplars.

Eventually, one of a group of competing exemplars will begin to be reinforced more often, and discouraged less often, than the rest. This could occur purely by chance, or because the winner is more compatible with other of the agent's exemplars. The costs of the other competitors will begin to be increased more often.

The equations for updating exemplar costs, the values of the parameters in those equations, and the cost assigned to new exemplars, were chosen to keep the cost of competing exemplars high.

If an exemplar is being reinforced at the same rate it is being discouraged, its cost, C, will change over time at the rate:

$$\frac{d}{dt}C = k(d - rC) \tag{3}$$

Where k is a measure of the rate at which the exemplar is used to analyze learning observations, and r and d are the values of the reinforcement and discouragement parameters, respectively. For the exemplar's cost to remain constant, it must be the case that:

$$C = d/r. \tag{4}$$

The values of the parameters r and d are equal, and so the cost of an exemplar that is being discouraged as often as it is being reinforced will rise to 1.0, at which point new exemplar may be used instead. If an exemplar is being discouraged more often than it is being reinforced, its cost will continue rising. If the cost rises above the cost of creating new structure (1.5), the agent will begin creating new complex phrases to record observations it otherwise might have used the exemplar for. This makes it possible for discouraged exemplars to be replaced by others more consistent with the emerging system.

5.2.4 Pruning

Exemplars are removed, or **pruned**, from an agent's set if they haven't been used in a solution phrase during the last two hundred times when the agent was chosen as either sender, receiver, or learner. Exemplars in active use tend to be used at a rate of once per thirty or so chances, so the pruned exemplars are unlikely to be used ever again. In general, unused exemplars are that way because their costs are higher than alternatives, and so pruning exemplars tends to have little effect on the emergence of communication systems. In any case, pruning is necessitated by limitations in computer memory.

6 Computational Simulations

A computational simulation of the negotiation process is begun by creating data structures to represent a population of agents and their (initially empty) sets of exemplars.

The simulation proceeds as a sequence of **rounds**, each of which involves two agents chosen at random from the population. One of them is given a formula set as a meaning to express, and finds the cheapest phrase that it can construct with the meaning. The string from that phrase is given to the other agent. In nine out of ten rounds, the other agent also receives the same formula set that the sender was given, and applies the learning algorithms described in Section 5 to create new exemplars, and modify the costs of existing ones. The remaining rounds are used to assess the accuracy of the emerging system. The second agent is given only the string from the

sender's phrase, and its interpretation of that string is compared with the meaning that the sender was given to express.

6.1 Measuring Communicative Accuracy

A numerical measure of the accuracy of the communication between a sender and receiver is computed as:

$$\textbf{Communicative Accuracy} = \frac{1}{2}\left(c/s + c/r\right) \tag{5}$$

Where:

s = the number of elements in the sender's formula set.
r = the number of elements in the receiver's formula set.
c = the number of formulae common to both sets.

If the formulae of the sender and receiver are equivalent, $s = r = c$, and the communicative accuracy value is 1.0. If the two formula sets differ, the quantity c/s is the fraction of the sender's meaning that got through to the receiver, and the quantity c/r is the fraction of the receiver's interpreted meaning that the sender actually expressed.[2] The communicative accuracy measure is the average of these two values.

6.2 An Example Simulation

Figure 8 tracks the progress of a population of ten agents as negotiate a communication system.

As shown in the top plot of Figure 8, the communicative accuracy of the population begins at zero, and stays there for the first several hundred rounds. It is still below 0.01 at round 2500, but then begins to rise steadily. Soon after round 10,000 the accuracy value begins rising sharply, reaching 0.80 near round 26,000 and at the end of the simulation has reached a value of 0.95.

The center plot in Figure 8 records the average number of exemplars per agent. For the first few two thousand or so rounds of the simulation, each agent acquires about one exemplar per round. After round 2000, the agents begin pruning unused exemplars, and the rate of exemplar acquisition begins

[2]These measures are based on the "recall" and "precision" values used to assess information retrieval systems (Kent, et al., 1955).

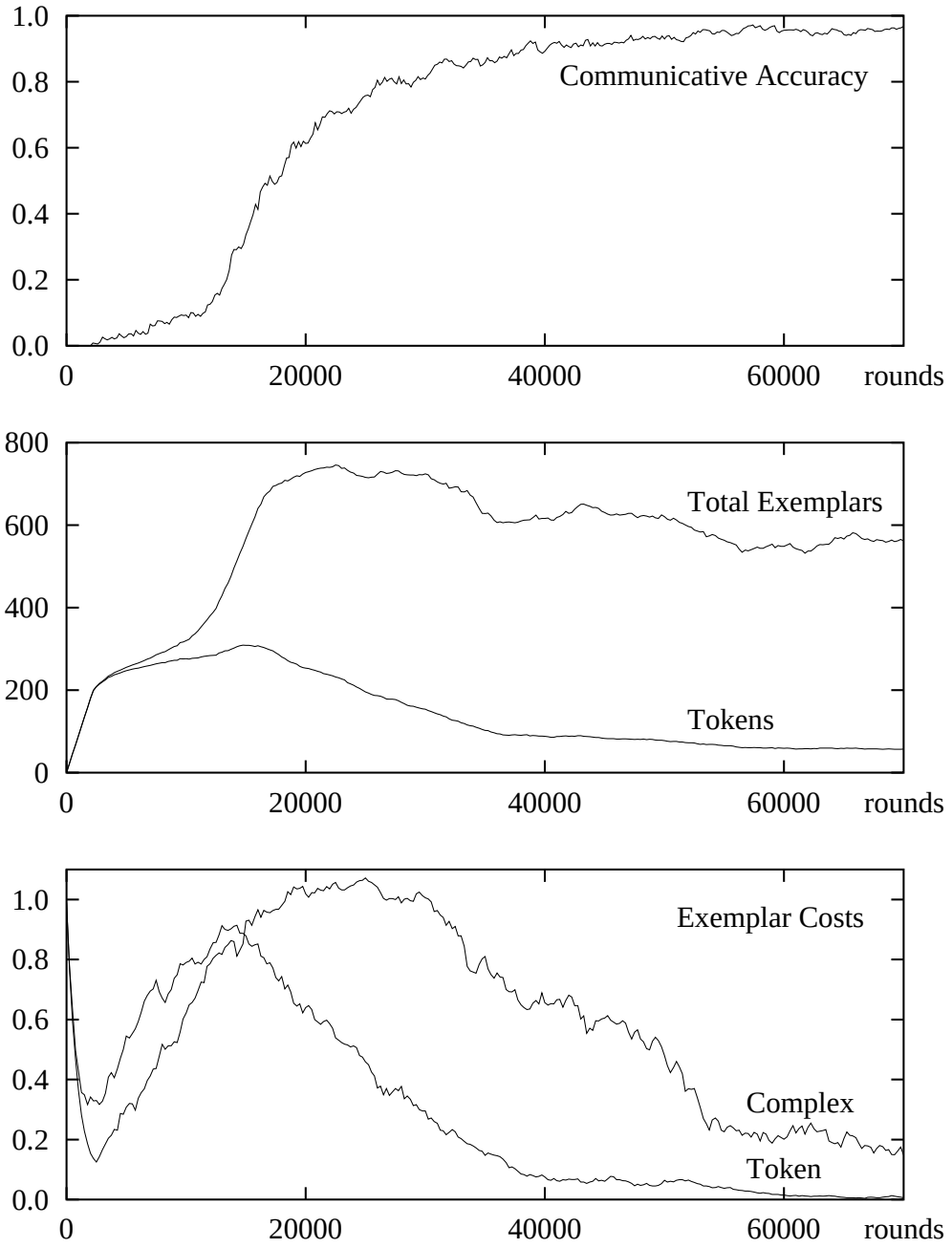


Figure 8: A population of ten agents negotiates a shared communication system. The top plot tracks the average communicative accuracy among the agents, as measured by Equation 5. The center plot shows the average total size of the agents' exemplar sets, and the average number of token exemplars in each set. The bottom plot shows the average cost of complex phrase exemplars and token exemplars used in senders' solution phrases.

to level off. Near round 3000, the agents begin to acquire and use complex phrase exemplars. This is followed by a period of rapid accumulation of exemplars, and a sharp rise in the population's communication accuracy. At round 22,500 the agents have an average of 745 exemplars each, of which 232 are tokens, and the communicative accuracy has reached 0.70. Soon afterwards, as the agents begin to come to agreement on a shared set of exemplars, the number of token exemplars drops fairly rapidly, followed later and more slowly by the number of complex phrase exemplars. At the end of the simulation, the agents have an average of 561 exemplars each, of which 57 are tokens.

The bottom graph in Figure 8 tracks the average cost of exemplars used by senders in the phrases they use to express meanings. Before round 2500, the few exemplars that are being used are almost always reinforced afterwards, and their costs decrease rapidly. Once the agents begin to accumulate exemplars, however, inconsistencies accumulate as well, and many learning observations result in discouragement of exemplars. The average costs of exemplars in active use rises, until by round 20,000 the average cost of complex phrase exemplars being used is about equal to the cost of new exemplars. The costs of token exemplars in active use is a bit less, and begins to drop near round 15,000, suggesting that the agents have begun to come to agreement on them. The cost of complex phrase exemplars continues to remain high until near round 30,000, when it begins to decrease slowly, as it continues to do for the rest of the simulation.

The progress of the simulation shown in Figure 8 is fairly typical. Simulations differ in the final value of the communicative accuracy reached, how long it takes to do so, and the numbers of exemplars possessed by the agents at the end of the simulation. Most populations reach a communicative accuracy value above 0.90, though some do not get much higher than that. A few have reached values above 0.98.

The most variability observed among runs involves the number of rounds that elapse until the initial spurt in the number of complex phrase exemplars. For populations of the same size, this may vary by as much as a factor of two. Once it occurs, most simulations proceed as the example does, and the differences at the start of the runs have little effect on the time taken to reach the maximum accuracy value. The number of rounds taken by a negotiation to reach a communicative accuracy of 0.90 scales approximately linearly with the size of the population.

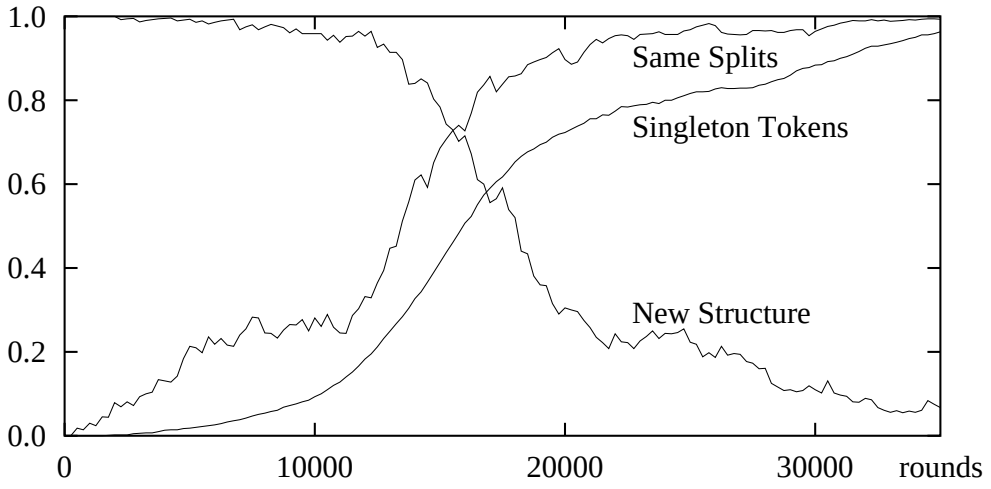


Figure 9: The emergence of agreement in the simulation shown in Figure 8.

6.3 The Emergence of Agreement

Most aspects of a population's communication system are established in a relatively short period early in the negotiation, during which the average communicative accuracy rises most rapidly. In the example simulation, this occurs from rounds 10,000 — 20,000.

Figure 9 presents additional measurements taken during the first part of simulation shown in Figure 8. The plot labeled "New Structure" tracks the average fraction of rounds in which an agent needed to create either a new complex phrase, or a new token, to analyze a learning observation. This value begins to drop soon after round 10,000, right when the communicative accuracy of the population begins to increase. This suggests that the increase in communicative accuracy is triggered by the accumulation of similar exemplars by all of the agents.

The plot labeled "Same Splits" in Figure 9 records how often the sender and receiver in a testing round break a string into constituents at the same places. This value is fairly low at first, because most learning analyses are random, and few if any of the randomly chosen constituent boundaries are aligned. This value also increases sharply soon after round 10,000, suggesting that the agents have at least come to agreement on the basic units of their communication system, if not what they ought to mean.

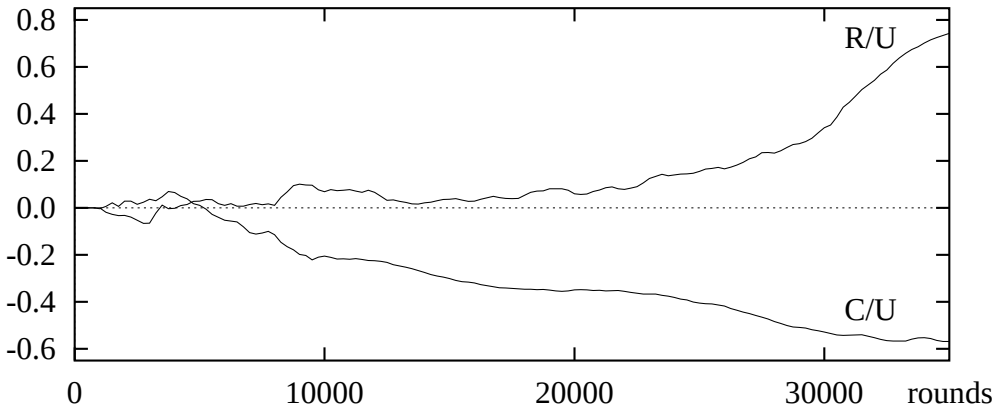


Figure 10: Correlation between the rates exemplars are used and their costs (C/U); and between the rates exemplars are used and the rate at which they are reinforced (R/U), in the simulation shown in Figure 8.

The plot labeled "Singleton Tokens" in Figure 9 tracks the average fraction of each agent's token exemplars that contain a single formula. As mentioned in Section 5, the use of singleton tokens, and agreement on them, is important for the emergence of the systems in general. This value also shows a sharp increase, lagging slightly behind the changes in the other two.

By round 20,000 the agents are using singleton tokens a large majority of the time, they usually don't need to create new structure to analyze learning observations, and the phrases they construct almost always match up on constituent boundaries. The communicative accuracy is fairly high at this point, but improves for several tens of thousands of rounds as competition among exemplars continues to weed out inconsistencies.

The effects of competition can be seen in the values plotted in Figure 10. The plot labeled 'C/U' tracks the correlation between the cost of an exemplar and the number of times in which it was used in a solution phrase, over the previous 100 rounds. The plot labeled 'R/U' tracks the correlation between between the number of times times an exemplar was used, and the number of times it was reinforced, over the previous 100 rounds.

The C/U correlation begins to decline steadily almost as soon as the first exemplars are acquired; this reflects the fact that agents choose the cheapest combination of exemplars that satisfy their search requirements. The R/U

correlation remains near zero, however, until round 20,000. This is consistent with the bottom plot of Figure 8 showing that exemplar costs increase until soon after round 20,000. Before this occurs, many exemplar are discouraged more often than they are reinforced, and so their costs remain high enough for alternatives to be explored. The R/U correlation begins rising sharply near round 30,000 and reaches a value of 0.75 at round 35,000.

As the agents begin to come to agreement on the basic sequences of characters their system uses, and what some of the singleton tokens mean, the costs of exemplars consistent with the emerging system start declining. The costs of their competitors remain high, and many of them are pruned away. Eventually the cheapest solution phrases constructed by learners are almost always consistent with exemplars used by other members of the population, and the agents use, and reinforce, such exemplars almost exclusively.

7 Negotiated Systems

In this section, I describe some of the communication systems that have emerged in simulation runs. I focus on properties of the systems that are seen relatively often in the systems, and those that have superficial similarities to syntactic phenomena in human languages.

Every simulation yields a different communication system and its own set of challenges. The process of understanding how a negotiated system works is a sort of cartoon version of linguistic anthropology, and some of the methods of that discipline can be applied. A set of related meanings can be given to the agents to obtain the phrases used to express them, and sets of strings can be given to the agents to see how they are interpreted. Of course my "subjects" are as patient and cooperative as I need them to be, and I can directly examine their computational mechanisms and exemplar sets.

Usually the agents have come to agreement on a few dozen tokens that they use to express individual formulae. The agents may also share a few tokens with more complex meanings, and some with no meaning at all. Each agent's exemplar set also includes many complex phrase exemplars, but few if any of these exemplars are identical with those of other agents. The agents' success at communication comes not from their possessing identical sets of exemplars, but from regularities in the structures of phrases constructed from their exemplars.

After introducing some terminology, several example systems are de-

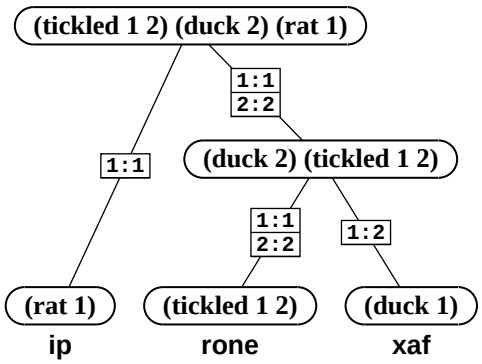


Figure 11: A phrase from System 7.2.

scribed. I will use the number of the subsection in which a system is discussed to refer to that system. Following the discussion of individual systems, I describe some general properties shared by most of the communication systems I have analyzed.

7.1 Introducing Formulae

A phrase from System 7.2 is shown in Figure 11. Before discussing this system in detail, I will use this phrase as an example to illustrate some terminology.

In the analysis of the phrases and exemplars used in negotiated systems, it is useful to determine the origin of the formulae in a phrase's meaning. Sometimes a token with the formula occurs as a subphrase of the phrase being considered, and the variables of the formula are not modified by any intervening argument maps. This occurs for the formulae (rat 1) and (tickled 1 2) in Figure 11. Alternatively, the variables occuring in a token's formula set may be renamed by argument maps to yield formulae in the meaning of the top-level phrase. This occurs for the formula (duck 2) in the example. The formula starts out as (duck 1) in the token whose string is '`xaf`', and its variable is renamed by the complex phrase above it.

I will say that a token **introduces a formula** f_P into the meaning of a phrase that contains the token if the token's formula set contains a formula f_T with the same predicate as f_P, such the argument maps applied to the token, and to any nodes above it in the phrase, rename the variables of f_T to yield f_P. Thus the token '`ip`' introduces the formula (rat 1), the token '`rone`' introduces tickled 1 2, and the token 'xaf' introduces (duck 2) into the

meaning of the phrase in Figure 11.

I will also say that a token **introduces a variable** into the meaning of a phrase if that variable occurs in a formula the token introduces into the phrase. It is possible for a single formula in the meaning of a phrase to be introduced by more than one token, though this rarely occurs in negotiated systems. On the other hand, it is common for a variable to be introduced by several tokens.

I will say that a formula or variable is **introduced below** a node in a phrase, if the formula or variable is introduced into the given phrase by the node itself, or by tokens that are constituents of the node.

7.2 Partitioning

Figure 12 presents another phrase from System 7.2. Note that this phrase has the same the same top-level structure as the phrase in Figure 11. Specifically, the argument maps applied to both subphrases are identical, as are the argument maps applied to the constituents of their right-subphrases. This suggests that similar exemplars were used to construct the two phrases.

The phrase in Figure 12 includes constituent phrases that combine the formulae of pairs of singleton tokens. These phrases occur in the same positions occupied by singleton tokens with the same variables in the simpler phrase in Figure 11. As a result, the tokens in Figure 12 are ordered such that the tokens expressing properties of the same variable are next to each other. Compare this phrase with the one shown in Figure 13, where no such ordering occurs.

I will say that a phrase **partitions** two variables if there is a node in the phrase such that all predicates involving one of the variables are introduced below that node, and no one-argument predicate involving the other variable is introduced below that node. A phrase will be said to be **partitioned** if it partitions each pair of variables used in its formula set.

If a phrase is partitioned, property tokens introducing its variables occur in separate sequences, sometimes with relation tokens or empty tokens among them. A partitioned phrase cannot have a sequence of three property tokens with the first and third involving one variable and the middle one involving a different variable. Such a sequence is seen in the last three tokens of the non-partitioned phrase in Figure 13.

Figure 14 presents more data from the simulation run described in Section 6.2. Whenever an agent was given a meaning to express that contained

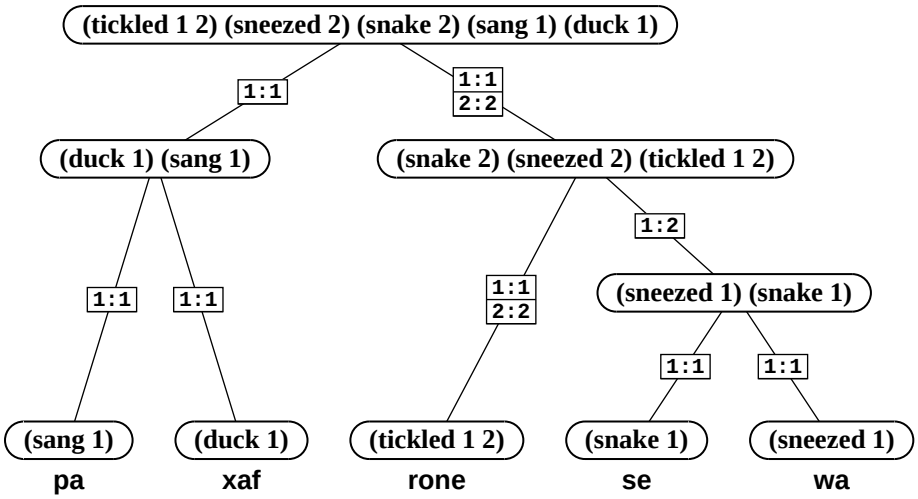


Figure 12: A partitioned phrase, from System 7.2.

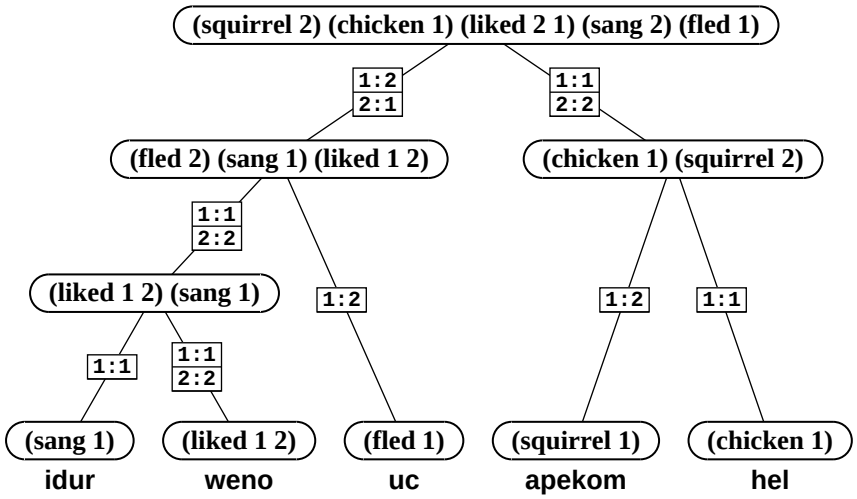


Figure 13: A non-partitioned phrase, used early in a simulation run.

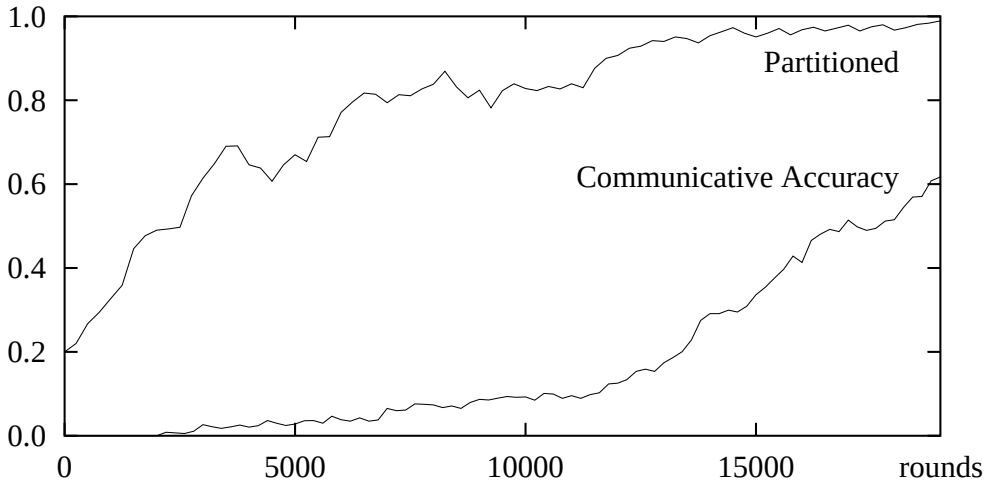


Figure 14: Another record of the simulation shown in Figure 8 of Section 6.2. The plot labeled "Partitioned" tracks the fraction of times that a sender used a partitioned phrase when expressing complex meanings. The Communicative Accuracy plot is the same as in the top graph in Figure 8.

at least two properties of two different variables (as is true for the meanings of the phrases in Figures 12 and 13), the phrase constructed by that agent was examined to see if it partitioned its variables. The plot labeled "Partitioned" in Figure 14 tracks the fraction of phrases that did. The probability that a randomly constructed phrase will be partitioned is approximately 0.2, and the agents start using partitioned phrases at about that frequency. However the fraction of partitioned phrases rises quite rapidly, and reaches 1.0 by round 20,000.

All simulation runs that have been analyzed have had similar results, yielding communication systems that rely almost exclusively on partitioned phrases. One consequence of partitioning is that regularities in a communication system can sometimes be summarized by giving the order in which variables in a phrase's meaning are introduced by singleton tokens. For example, the phrases in Figure 11 and 12 both exhibit the order '1 R 2', where 'R' indicates the position of the token introducing the a relation predicate applied to the two variables. The causes of partitioning are discussed below.

7.3 Empty Tokens as Delimiters

Two phrases from system 7.3 are shown in Figure 15. Phrases in this system often put the relation token first, followed by tokens expressing properties of its arguments.

As described in Section 5.1.6, empty tokens appear early in simulation runs as a result of mismatches of various kinds between the phrase that one agent used to generate a string, and the one a learner created to analyze it. Many of the exemplars that introduce these empty tokens ultimately disappear from the negotiated systems as the agents begin to analyze each other's utterances in consistent ways. But in most of the simulations observed, a few empty tokens remain, sometimes acquiring specific communicative functions.

In phrase 15.a, the empty token '`ajvaf`' occurs between two tokens, the first of which introduces a property of variable 2, and the second introduces a property of variable 1. The token '`ajvaf`' is also used in phrase 15.b, again serving to separate tokens introducing different variables.

The two phrases shown in Figure 16, also from System 7.3, illustrate how the agents that negotiated this system express a pair of simple meanings. In phrase 16.a, the right hand token is interpreted as expressing a property of variable 1. The use of the empty token '`ajvaf`' in 16.b changes the interpretation of the token to its left, such that it expresses a property of variable 2. The exemplar used to construct 16.a is slightly cheaper than that used in 16.b. The two exemplars therefore implement a default rule for the interpretation of a token that occurs to the right of a relation token: It is treated as introducing a property of variable 1 unless it is followed by '`ajvaf`'.

Most exemplar phrases containing the token '`ajvaf`' have the structure of 16.b. This phrase applies the argument map $\boxed{1:2}$ to the right subphrase of the node above the relation token. Agents with similar exemplars will use one of them to analyze a sequence beginning with a relation token, and ending with '`ajvaf`'.

Both exemplars may be used in a single phrase. For example, phrases 15.a and 15.b use '`ajvaf`' to signal the non-default interpretation of the token to its left, however the top-level node both phrases uses the default structure of phrase 16.a to map the rightmost token's argument to variable 1.

To see how the two exemplars interact, agents from the population that negotiated this system were given a string that contained two property tokens between a relation token and the empty token '`ajvaf`'. The phrase most agents used to interpret this string is shown in Figure 17. The first property

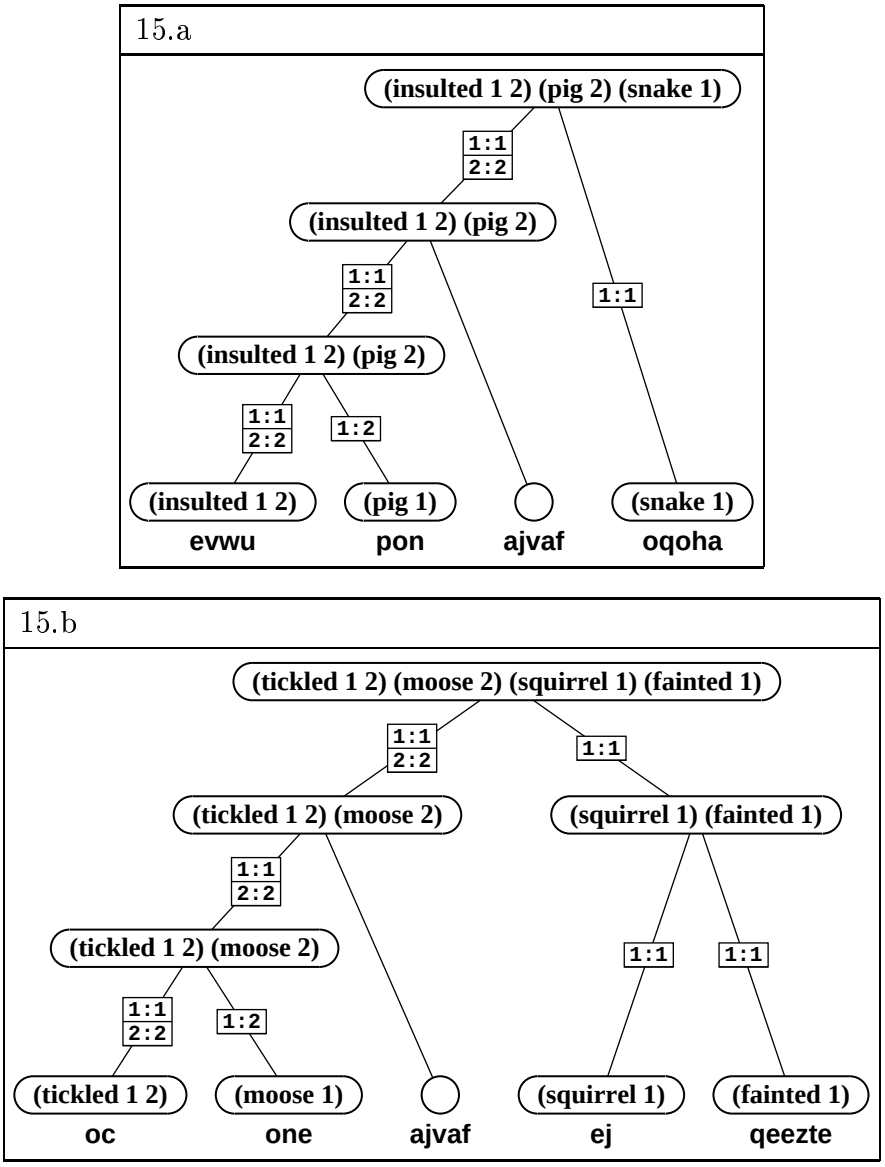


Figure 15: Phrases from System 7.3.

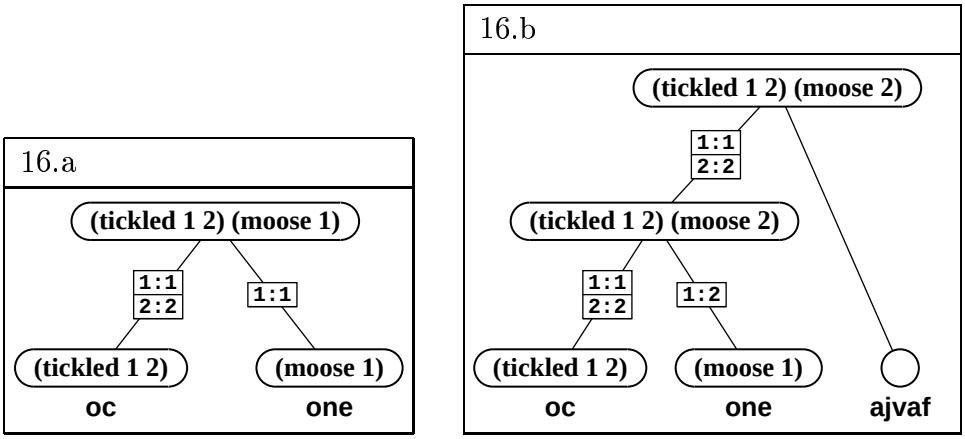


Figure 16: Two more phrases from System 7.3.

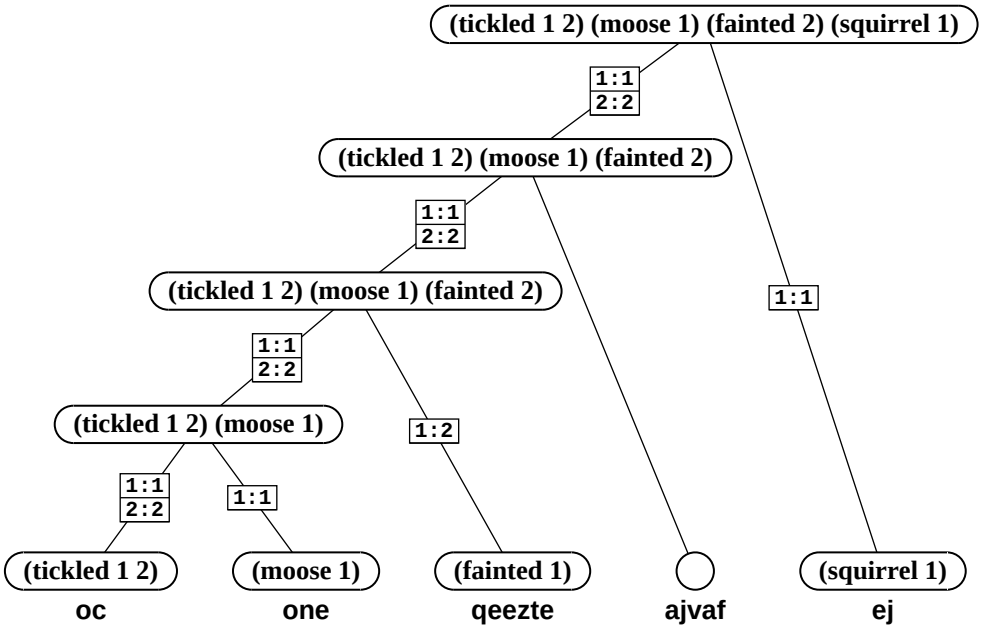


Figure 17: A solution phrase for the string '`oconeqeezteajvafej`'.

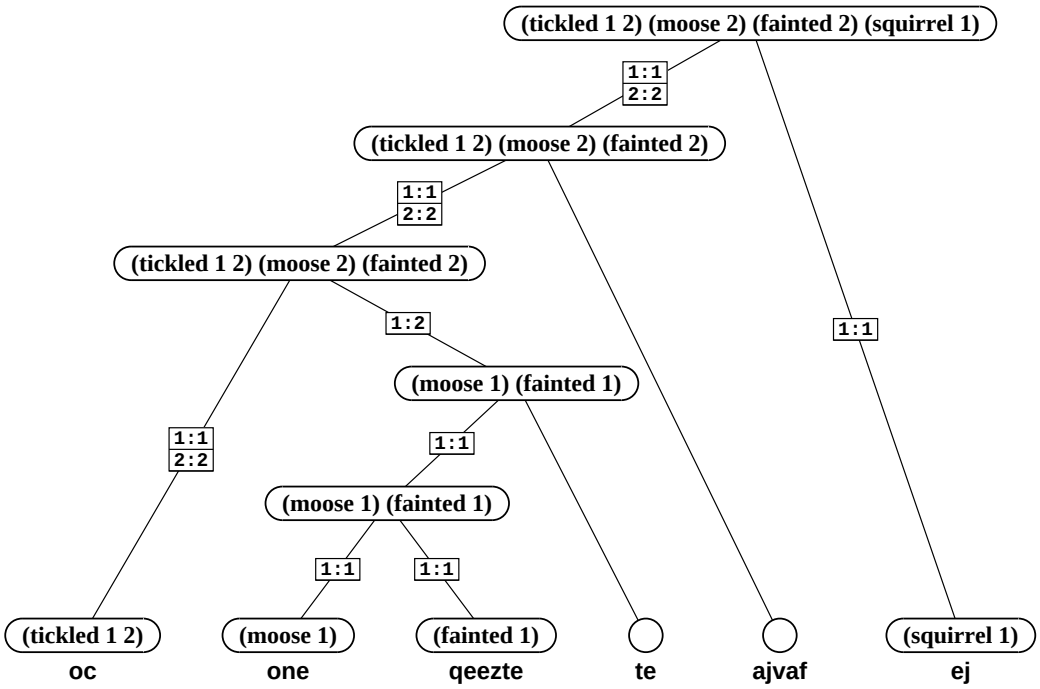


Figure 18: Use of the empty token '`te`' to delimit a sequence of property tokens to which an identity argument is applied.

token, '`one`' is assigned the default interpretation, and its argument is not modified by the argument map above it. However the next property token, '`qeezte`' does have its argument renamed from 1 to 2, by the exemplar containing the empty token. Note that the phrase shown in Figure 17 is not partitioned. This phrase would not be used by the agents to communicate the meaning shown (they would use a phrase with the structure of 15.b), but it is the cheapest phrase they can find to interpret the string.

To express a meaning in which two or more tokens express properties of argument 1, the agents would use a phrase with the structure shown in Figure 18. This phrase uses an additional empty token, '`te`'. The argument map on the right subphrase of the phrase that '`te`' is attached to applies the identity argument map to the formula of the phrase below it. Therefore the token '`te`' can serve to signal the end of a sequence of tokens introducing properties of the same argument. By combining the two formula in this way,

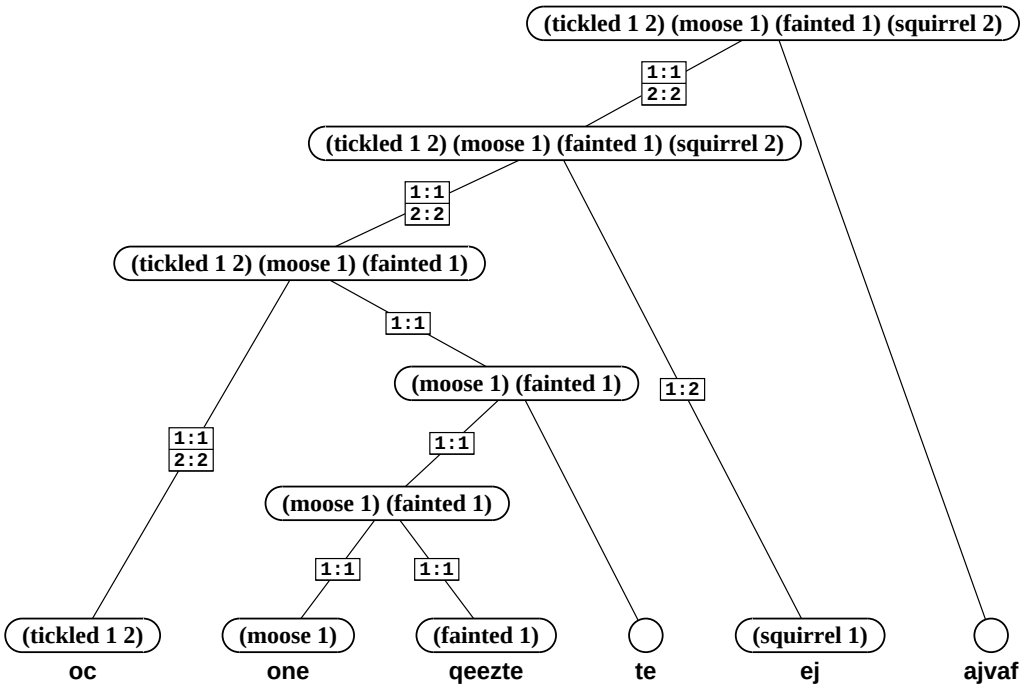


Figure 19: Using '`te`' and '`ajvaf`' to yield an alternative interpretation from the one shown in 18.

is is then possible for the '`ajvaf`' exemplar to apply to both tokens' formulae, and they are mapped together to variable 2 in the meaning of the top-level phrase.

Figure 19 shows another way that the tokens '`te`' and '`ajvaf`' can interact. In Figure 18 variables are introduced in the order 'R 2 1'. The same sequence of property tokens in Figure 19 is interpreted as introducing variables in the order 'R 1 2'. In general, it seems that the token '`ajvaf`' behaves somewhat like a case ending or particle, indicating that the phrase to its left expresses properties of variable 2.

7.4 Marking Reflexive Predicates

In system 7.4, the standard order in which arguments are introduced is '2 1 R', as shown in Figure 20.a. However the empty token '`ojo`' is part of an exemplar containing a non-invertible argument map. If it is used after

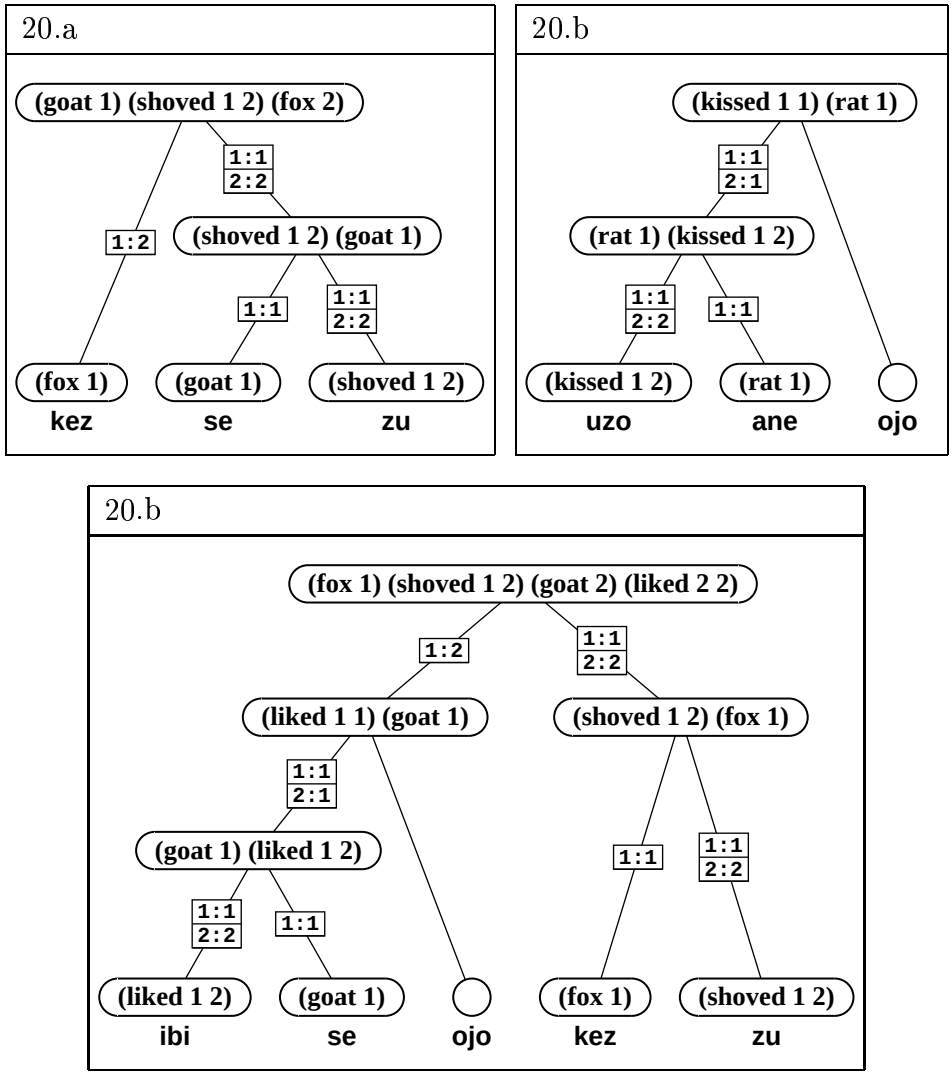


Figure 20: Phrases from System 7.4, illustrating the use of the empty token '`ojo`' as a marker of reflexive predicates.

a phrase containing a relation token and a property token, the relation is interpreted as involving a single argument, as shown in phrase 20.b. In phrase 20.b, both patterns are seen: The left subphrase uses '`ojo`' to create a reflexive predicate whose argument is mapped to variable 2. The top-level phrase uses the standard ordering.

All of the agents in this population possess a number of exemplars containing the empty token '`ojo`', and in all such exemplars, that token is the right subphrase of a phrase whose argument map collapses the two variables of the meaning of its left subphrase.

Therefore if an agent is given a string containing the sequence '`ojo`', it will almost always use one of those exemplars in its interpretation or analysis of that string, and the resulting phrase will contain the collapsing argument map. An agent will likewise almost use exemplar containing the token '`ojo`' to express such meanings, as such exemplars contain the argument map needed to create a reflexive predicate out of one with two arguments.

Not all systems use empty tokens to signal reflexive predicates. In some systems, the have negotiated additional tokens specifically for the reflexive forms of all relations. Other systems use characteristic word orders to signal reflexive predicates. Figure 20 illustrates a reason why the use of an empty token to signal reflexives might have been used in this system. Recall that in this system, if a relation is applied to two different variables, they are introduced before the relation. If the empty token '`ojo`' were missing from the phrase in Figure 20.c, token '`se`' might be interpreted as expressing variable 2 of the relation token '`zu`'. The occurrence of '`ojo`' in the string blocks this interpretation.

7.5 Marking Inverting Maps

A related use of empty tokens is illustrated in Figure 21. In this system, if the empty token '`la`' appears after a relation token, it signals that an inverting argument map is applied to the variables in the relation, as shown in phrase 21.b. This phrase is used to replace the relation token in 21.a, to yield 21.c, in which the roles played by the referents of variables 1 and 2 are the opposite from that in 21.a.

In this system, the empty token '`la`' almost always occurs as the right subphrase of a phrase whose left subphrase is a relation token. The interpretation of the argument role of constituents based on position is fairly rigid in this system, based on the low cost of exemplars with the structure of

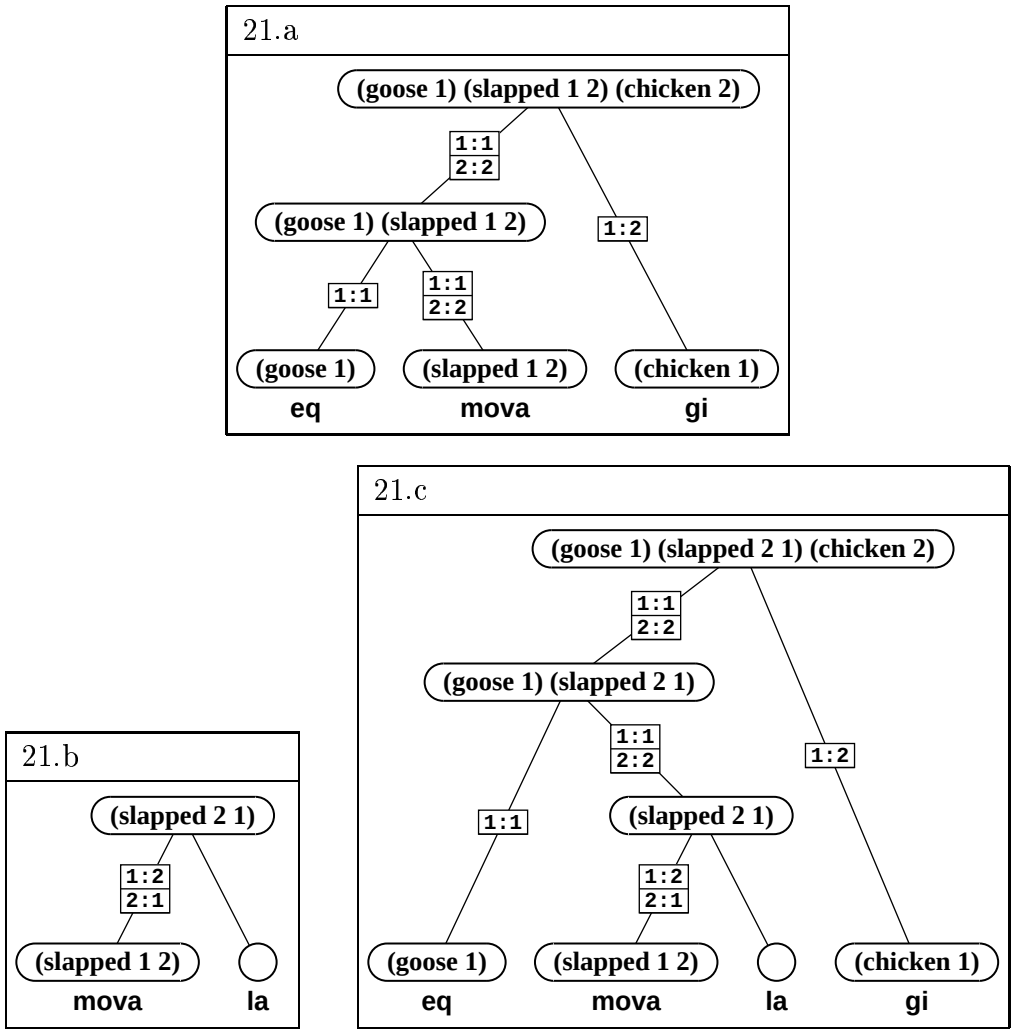


Figure 21: Phrases from System 7.5, illustrating the use of the empty token '`la`' as a marker of inverting argument maps.

phrase 21.a. If the empty token 'textttla' were to attach higher in a phrase, such that it mapped the arguments of property tokens, as well as relation tokens, it would rename the variable 1 in the first property token to variable 2, leaving no way to express properties of variable 1. This token therefore resembles the use of inflectional morphology that verbs in some languages take to indicate the passive voice.

7.6 Overruling the Default Mapping

In System 7.6, the empty token '`uy`' has two different functions, shown in Figure 22. If it occurs before a relation token that is followed by a property token, an inverting argument map is applied to the variables of the phrase. If the empty token occurs after the property token, an identity map is applied to the phrase. Note that in both phrases, an inverting argument map is applied to the variables of the relation token.

Phrase 22.c illustrates the cheapest interpretation most agents assign to a sequence of a relation token and a property token. Note that this phrase uses an argument map that collapses variable 2 to 1. This phrase is the solution that the agents would use if given the meaning {(kissed 1 1) (squirrel 1)} to express. In this system, the default interpretation of relation token followed by a property token is that the relation is reflexive. The empty token '`uy`' is therefore needed to overrule the default interpretation.

The agents are only convey meanings that include reflexive predicates. So in this system, the default interpretation does not correspond to the most common case. On the other hand, this should not necessarily be surprising, because the factors influencing the properties of a communication system do not exert there influences independently. In this case, it seems the token '`uy`' might have first acquired the function of delimiting the set of arguments to a relation. If it were used to do so in almost every phrase where there were multiple arguments, it could have acquired the function of signaling the non-reflexive interpretation of the relation token.

7.7 Equivalent Empty Tokens

In System 7.7, the empty tokens '`laxo`' and `ah`' seem to be more or less equivalent. The phrases shown in Figure 23 illustrate that either token, put before a property token followed by a relation token, maps the variable 1

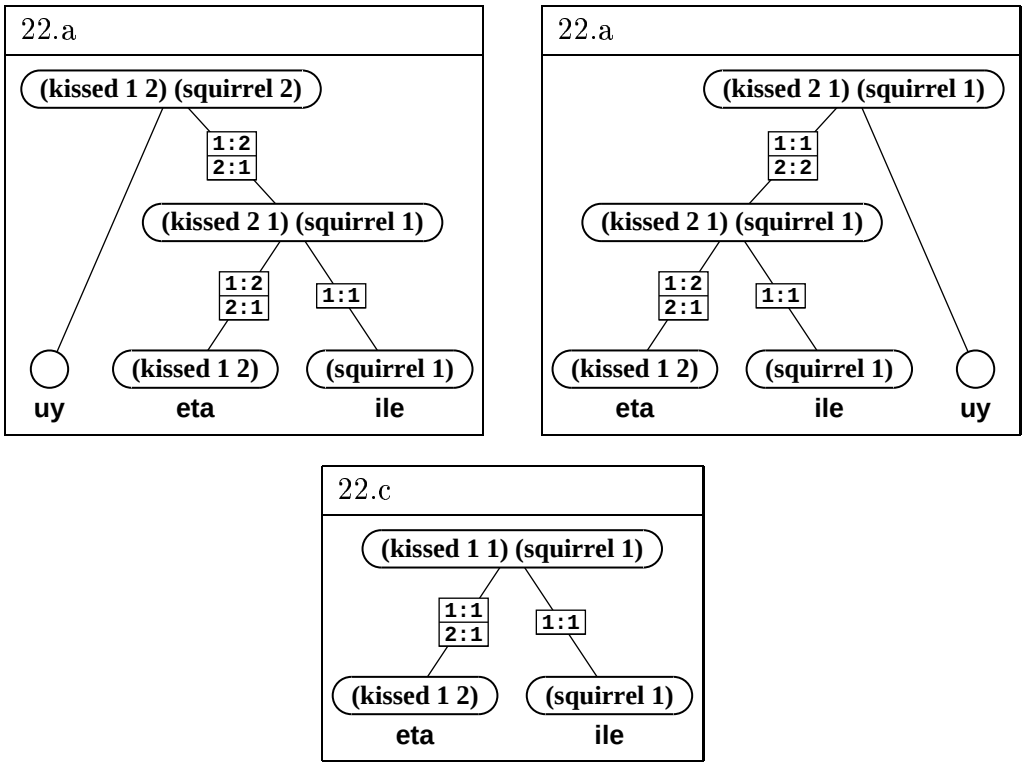


Figure 22: Phrases from System 7.6 illustrating how the empty token 'uy' is needed to overrule the default interpretation of a relation token followed by a property token.

to 2. If either empty token is after a relation token followed by a property token, an identity argument map is applied to the property token.

In most negotiated systems, the empty tokens only occur in specific exemplars, and each empty token's string can serve to indicate particular structural properties of phrases containing it. However in this system, the agents have a pair of exemplar empty tokens, and can use either to replace the other.

The two empty tokens are not always interchangeable. The standard ordering of a two argument relation is shown in Figure 24. Although the structure of the exemplar this phrase is based on is consistent with those shown in Figure 24, it contains the token '`laxo`' at the beginning, and the empty token '`ah`' at the end. The agents don't create phrases with the empty tokens in the other order (although they have no problem interpreting such phrases correctly).

The pattern seen in Figure 24 is used to express a more complicated meaning in Figure 25. In the right subphrase, the empty tokens '`ah`', and '`laxo`' appear to be combined into the empty token '`ahlaxo`', used to indicate the start of a sequence describing a second relation that one of the participants is involved in. The structure of the subordinated relation is otherwise the same, and even includes the sequence '`laxo` ... `ah`', with the additional '`ah`' apparently serving as a marker of subordination.

7.8 Properties of Negotiated Systems

7.8.1 Agreement on Singleton Tokens

As mentioned above, the agents in a simulation tend to come to agreement relatively quickly on a shared set of singleton tokens. In some cases, they use as few as 32 tokens — one for each of the possible predicates. More often, they settle on from 40–70 exemplars, with few that express multiple formula meanings, a few empty tokens, and a few that express reflexive or inverted senses of other relation tokens. In a few of the systems, the agents have a token for almost every possible argument pattern of a relation, and therefore do not need to use argument maps to modify them.

7.8.2 Causes of Partitioning

There is nothing in the model that requires the agents to use only partitioned phrases, and they don't use them at the start of simulation runs, as the plot in

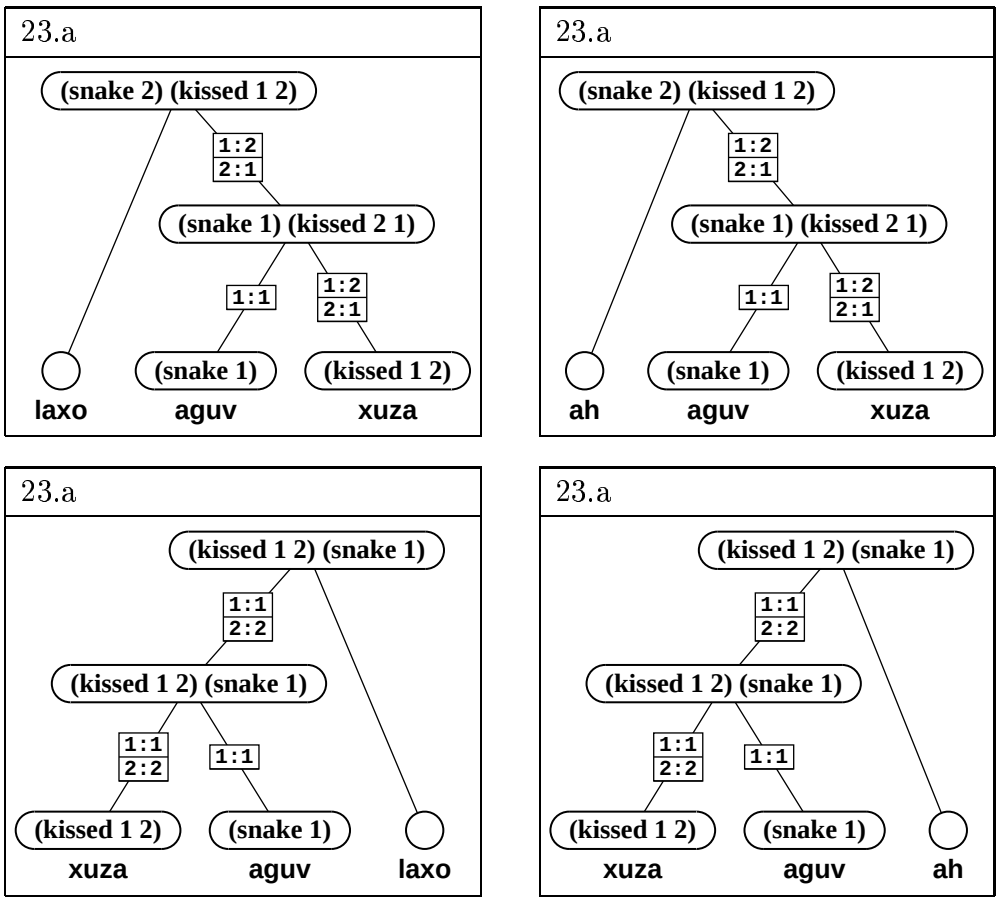


Figure 23: Phrases from System 7.7 illustrating the use of two equivalent empty tokens.

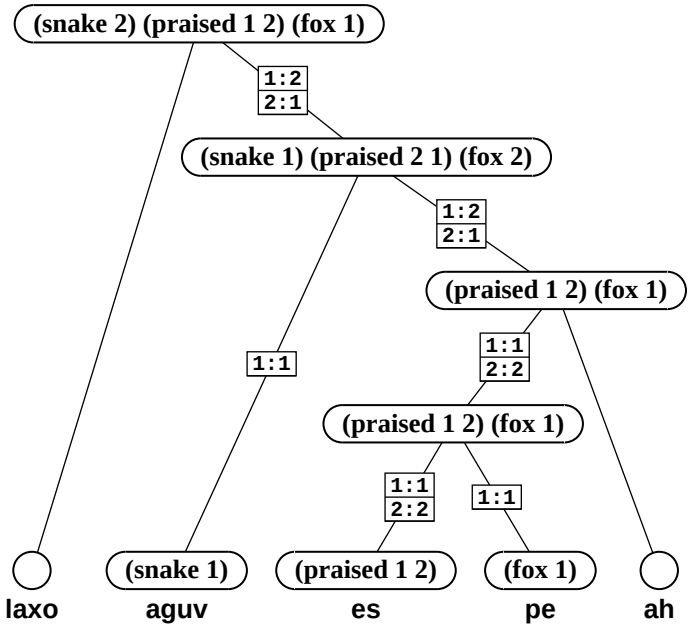


Figure 24: A phrase from System 7.7 illustrating the use of empty tokens to delimit the arguments to a relation.

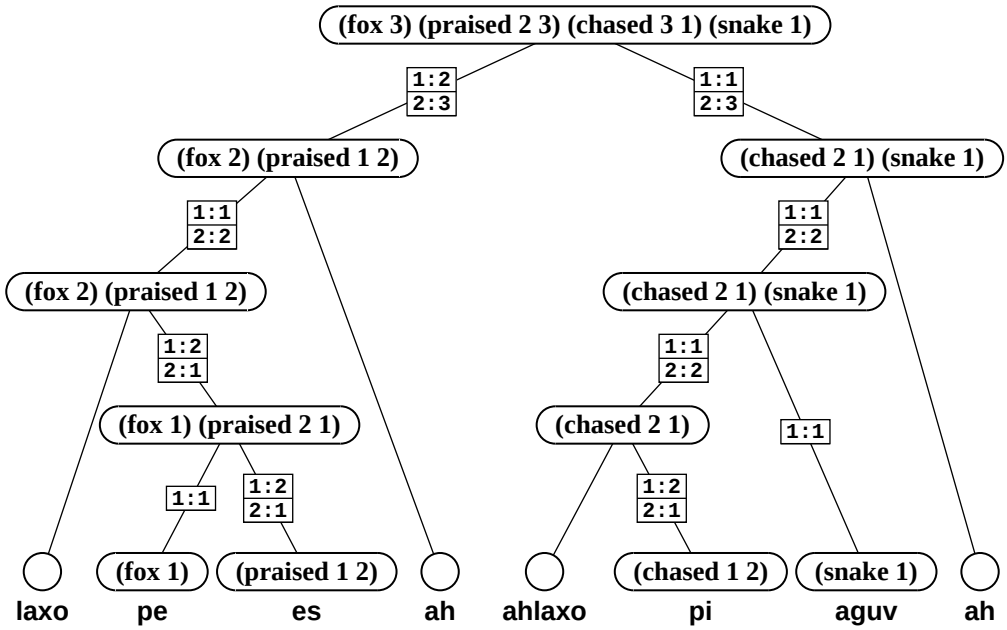


Figure 25: A phrase from System 7.7 illustrating the use of empty tokens to delimit a subordinated relation.

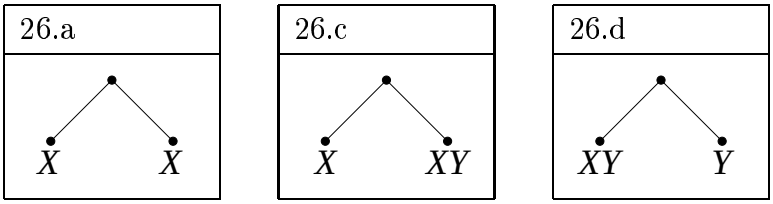


Figure 26: Partitioning Exemplars.

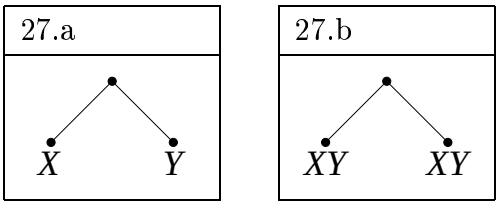


Figure 27: Non-Partitioning Exemplars.

Figure 14 illustrates. However in all runs observed, the agents quickly come to rely almost exclusively on partitioned phrases. This occurs as a consequence of the competition among exemplars induced by their reinforcement and discouragement during learning rounds in the early part of a negotiation. Certain exemplars and sets of exemplars tend to be used and reinforced often relative to their alternatives, and the structures of those exemplars begins to impose partitioning on phrases created from them. Once the trend is established, it is exploited and amplified as new exemplars are created and used.

Exemplars with the structures shown in Figure 26 can be used to construct partitioned phrases. In these diagrams, the letters X and Y are used to designate the variable introduced at or below the indicated node labeled with the letter. Both variables are introduced below nodes labeled XY.

Exemplars with structure 26.a are acquired quickly by all agents. Such exemplars can be used to combine pairs of phrases whose formula sets involve the same variable, and so they are used and reinforced often. Note that the structure of 26.a is seen in the nodes used to combine the first and last pair of property tokens in Figure 12.

The use of exemplars with the structures in Figure 26 does not guarantee that the resulting phrases will be partitioned, but the structures of small phrases containing such exemplars begins the trend towards partitioning. If

an agent has acquired exemplars with structure 26.a, and they have acquired relatively low costs, the agent can combine phrases whose formula sets contain different variables with exemplars whose structures are shown in 26.b and 26.c, as well as the structures corresponding to their mirror-images.

Exemplars with the structures shown in Figure 27 tend to be used in non-partitioned phrases. If the structure 27.a is used, the phrase that contains it will not be partitioned. The structure 27.b is used in non-partitioned phrases to combine phrases containing 27.a, though it can occur in partitioned phrases as well.

Exemplars with the structure 27.a are likely to appear later in a simulation than 26.a, and will probably be used and reinforced less often. In addition, exemplars whose structure is the mirror-image of 27.a will also occur, the use of exemplars from one of the two sets will often cause one or more exemplars from the other set to be discouraged. Until one or the other pattern is established, the costs of exemplars in both sets will remain relatively high. On the other hand, the symmetry of exemplars with the structure of 26.a makes them immune to such competition.

Given that a learner is not able to observe the structure of the phrase used by the sender whose string and meaning is being analyzed, it is often the case that the learner's phrase differs from the sender's. The learner may therefore create, and reinforce, exemplars whose structures are quite different from those used by the sender. However the structures of some exemplars are such that their use in a sender's phrase increases the chance that similar exemplars will be used by learners.

Consider for example, phrases whose meanings involving three formulae: a relation and two property tokens, one for each argument of the relation. (E.g., {(tickled 1 2) (duck 2) (rat 1)}.)

Three tokens introducing a relation and two different variables can occur in six different orders. For each, there are two phrase structures that can be used to combine the tokens into a complex phrase. There are therefore twelve phrase structures that can be used to express the meanings being considered. Some of them are shown in Figures 28 and 29.

Figure 28 shows the two phrase structures consistent with the ordering '1 R 2'. The structures consistent with the ordering '2 R 1' are the mirror-images of those shown. Figure 29 shows the structures consistent with the ordering 'R 1 2'. The mirror images of these structures correspond to the ordering '2 1 R'. The structure of the two phrases with the ordering 'R 2 1' and the two with '1 2 R' are similar to those shown in Figure 29.

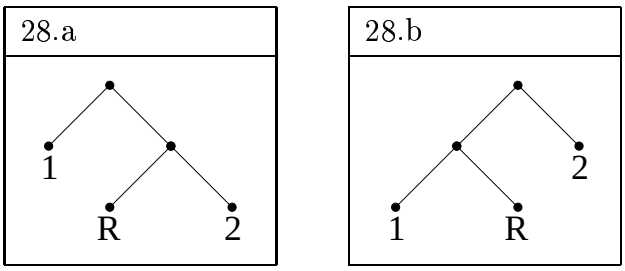


Figure 28: The two phrase structures consistent with the ordering '1 R 2'.

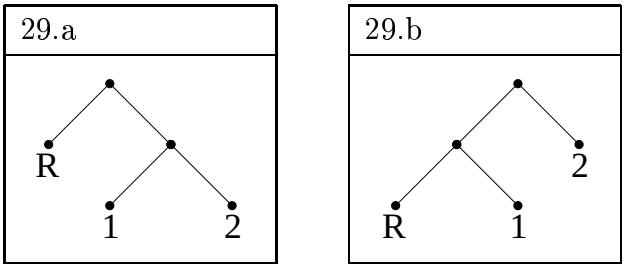


Figure 29: The two phrase structures consistent with the ordering 'R 1 2'.

Note that both of the phrases shown in Figure 28 can be constructed from the "partitioning" exemplars in Figure 26. So if a sender were to use the structure 28.a in its phrase, the learner would reinforce such exemplars, whether its had the same structure used by the sender, or the structure of 28.b.

On the other hand, if the sender used a phrase with the structure of 27, built from the non-partitioning exemplars in Figure 27, the learner might use the structure 27 to record its observation. If so, it would reinforce the partitioning exemplars instead of exemplars corresponding to those the sender used.

The partitioning exemplars in Figure 26 are therefore more likely to be used more often by the learners when they later express meanings, and such exemplars, and new ones created from them, will be used with increasing frequency.

7.8.3 Partial Compositionality

Also partial regularity?

7.8.4 Strict Ordering Versus Empty Tokens

As a very rough generalization, the use of empty tokens in a system is inversely related to the strictness with which the system uses ordering to indicate how the variables in property tokens are mapped. The strictness of ordering varies widely. In some systems, the agents will always use a particular ordering for a two argument predicate, in other systems, the ordering is much more free.

The few systems observed with no empty tokens at all use one of the two orderings: '1 R 2' or '2 R 1' exclusively. It seems that the agents require some way to determine the boundaries of partitioned sequences of tokens introducing properties of more than one variable. If this isn't done by an empty token, a relation token is the only available solution.

However the distinction between systems that use ordering versus those that use empty tokens isn't especially useful for the systems that emerge in the simulations. In most systems, a particular ordering may be used most of the time, but other orderings are also used (and understood); and almost all systems use a number of empty tokens as well.

7.8.5 Promoting Versus Inverting Systems

A sharper distinction has to do with how variable 2 is introduced into the meaning of a phrase. Most of the systems examined in detail use one of two methods.

A **promoting** system uses exemplars with the structures shown in Figure 30. Both of the simple exemplars, 30.a and 30.b can be used to combine a property token with a relation token. One of them applies the identity argument map to the relation token's formula, the other exemplar renames variable 1 to 2. These exemplars can be combined to express properties of both arguments of a relation, as shown in 30.c.

An **inverting** system uses exemplars with the structures shown in Figure 31. In these phrases, an inverting argument map is applied to the formula of a relation. The variable introduced below the other subphrase is therefore mapped to variable 2 in the original relation, which has been renamed to 1 by the inverting map. If this same exemplar (or one with similar structure) is used again, the variables of the relation are renamed back to what they were to start with, but the property token's variable is mapped to 2. The variable introduced by the other subphrase of the top-level phrase is not renamed.

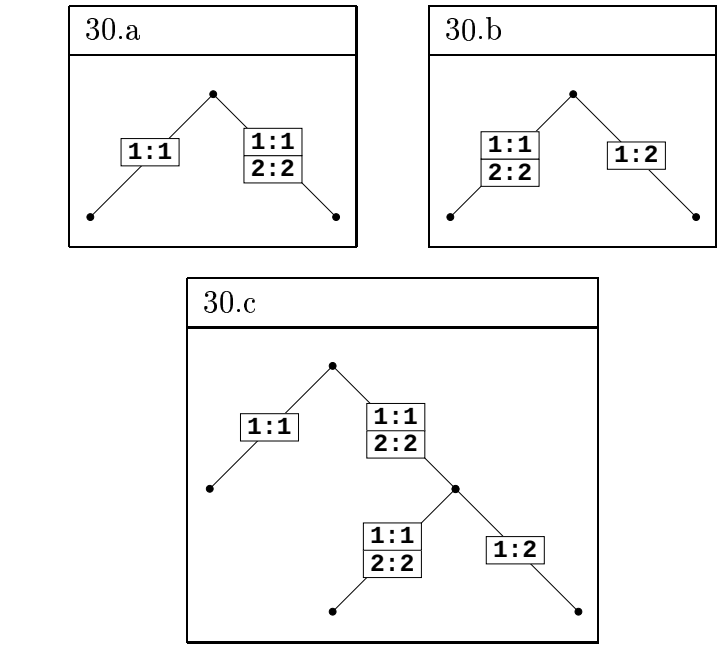


Figure 30: Phrase patterns in "Promoting" systems.

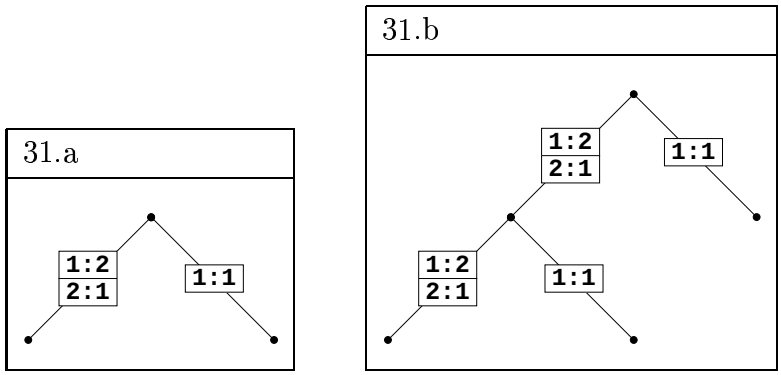


Figure 31: Phrase patterns in "Inverting" systems.

Of those discussed in this section, Systems 7.2, 7.3, 7.4 and 7.5 use the promoting pattern of exemplars. Systems 7.6, and 7.7 use the inverting pattern.

In general, systems that use the promoting pattern must come to agreement on the ordering of property tokens that express a particular argument of a relation. In the examples shown in Figure 30, the ordering leads to the pattern '1 R 2'. Unless one or the other ordering is chosen (or empty tokens are used to signal participant roles), the agent's will incorrectly map the variables.

Inverting systems are in a sense simpler than promoting systems, because they use only one basic pattern of argument maps, shown in 31.a. Where promoting systems must use either ordering or empty tokens to signal how arguments are renamed, in inverting systems, the mapping of each variable can be determined from its depth in the phrase. Inverting systems still use empty tokens, often to delimit the arguments of multiple relations, or to indicate when a non-default argument map is applied to a subphrase.

8 Discussion

In this section I discuss two of the reasons for the agents' success at negotiation communication systems, and what the results presented in this chapter might say about the emergence of human language.

8.1 Meanings and Learnability

According to the framework described in Section 2, syntactic structure is used to represent analyses of mappings between signals and meanings. Such analysis are neither independent of, nor reducible to, the other domains.

This approach qualitatively changes the problem of learning a system. In the model of Gold (1967), the learner is given strings generated by a grammar, and must hypothesize the rules of the grammar from that data alone. Gold shows that this task is impossible if the grammar includes context-free rules.

In the present model, the learner's task is to hypothesize a mapping between strings and meanings, given observations of example mappings. If the domain of meanings can be characterized structurally, the goal of a learner in this model and in Gold's model are formally identical. However the data available to the learner are qualitatively different in the two approaches: In

the task of learning mappings between signals and meanings, positive examples can provide negative evidence.

Given an observation of a meaning M_O and a string S_O used to express that meaning, the learner can use its current hypotheses about the system to interpret S_O as conveying a meaning M_H. If M_H is not equivalent to M_O, the learner has discovered that there is a problem with its hypotheses. Depending on how the hypotheses are represented, the learner may be able to track down and modify those that are to blame. A similar comparison can be made between the string S_O and the string the learner's hypothesized system would use to express M_O.

Hamburger and Wexler (1973) demonstrated that certain classes of transformational grammars that are unlearnable according to Gold's model can be acquired if the learner is given a representation of the meaning of each string. In their model, learners represent their hypotheses about the grammar being learned as a set of rules, and a rule may be removed if it is used in the derivation of a phrase whose meaning does not match a learning observation.[3] Hamburger and Wexler use techniques from the theory of stochastic processes to show that such a learner will acquire the correct set of rules after a bounded number of observations.

Although the model described in this chapter differs from that of Hamburger and Wexler in its representations of syntactic structure and meanings, the results of the computational simulations are consistent with their findings. It might be possible to extend their result show that some classes of grammars can not only be acquired, but can be developed in the first place, provided that learners are given meanings as well as strings.

8.2 The Learning Dynamic

Although each negotiation produces a different communication system, they exhibit a number of regularities and general properties — including the tendency to use a small set of singleton tokens, partitioning of phrases, the use of empty tokens to signal argument maps, the use of either the "promoting" or "inverting" pattern for expressing two argument predicates.

These regularities are consequences of the mechanisms implementing the agents communicative and learning abilities, however nothing in the imple-

[3]The method described in Section 5.2.2 for locating exemplars to discourage is similar to Hamburger and Wexler's algorithm for removing rules.

mentation of the agents require that they be satisfied. The agents can and do produce phrases that fail to satisy them, especially in the early rounds of a negotiation. It would be little use to characterize the agent's learning mechanisms in terms of the properties of the communication systems they give rise to, as such characterizations reverse the actual direction of influence, and provide little help in understanding how communication systems could occur in the first place.

The regularities emerge as dynamical consequences of the processes of learning and negotiation among the agents. Certain kinds of exemplars tend to occur early in negotiations, or tend to be reinforced often relative to alteratives. Sets of exemplars may mutually reinforce each other, or may cause other exemplars to be discouraged and ultimately removed from a system. Some exemplar structures are differentially better at being learned than others, and will tend to accumulate in the population.

The results presented here therefore suggest that the learning dynamic might be responsible for some of the syntactic phenomena seen in human languages. Exploring this possibility requires a different characterization of the problem of language acquisition. Rather than focusing on whether and how languages with specific formal properties can be acquired, attention should be directed at understanding the influences that learning mechanisms can have on the languages that emerge from their use.

8.3 Conclusion

Agents in the computational simulations negotiate communication systems that enable them to accurately convey 2.3×10^{13} meanings after each agent has made fewer than ten thousand learning observations. Each agent acquires several hundred exemplars, of which a few dozen are singleton tokens identical to those of other agents in the population.

The agents express meanings by combining their singleton tokens into complex phrases, using the order of phrases, as well as the presence and position of empty tokens, to indicate configurations of predicate arguments. Empty tokens are also used to signal the boundaries of constituents, the presence of specific argument maps, and details of the structure of phrases containing them.

Systems emerge as the result of a learning mechanism that involves no induction or rules, or noticing of similarities, or setting of parameter values, and is subject to very few internal constraints. The agents learn to commu-

nicate by recording their observations of other agents communicating, and using those records to guide their own attempts.

These results support a simple conclusion: Perhaps language was invented long ago, and is still learned by children today, as a result of trying to be understood, trying to understand, and trying to get better at doing both.

References

[1] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] L. W. Barsalou. On the indistinguishability of exemplar memory and abstraction in category representation. In T. K.Srull and R.S. Wyer, editors, *Advances in Social Cognition*. Lawrence Erlbaum, Hillsdale, NJ, 1989.

[3] John Batali. Computational simulations of the emergence of grammar. In James Hurford, Michael Studdert-Kennedy, and Chris Knight, editors, *Approaches to the Evolution of Language, Social and Cognitive Bases*, pages 405–426. Cambridge University Press, 1998.

[4] A. Black and P. Taylor. Festival speech synthesis system: system documentation. Human Communication Research Centre Technical Report HCRC/TR-83, 1997.

[5] Rens Bod. *Beyond Grammar: An Experience-Based Theory of Language*. CSLI Publications, Center for the Study of Language and Information, Stanford University, 1998.

[6] David Canning. Learning language conventions in common interest signaling games. Unpublished manuscript, Columbia University, 1992.

[7] D. J. Chalmers, R. M. French, and D. R. Hofstadter. High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental and Theoretical Artificial Intelligence*, 4:185–211, 1992.

[8] Noam Chomsky. *Reflections on Language*. Pantheon Books, New York, 1975.

[9] Vincent P. Crawford and John Sobel. Strategic information transmission. *Econometrica*, 50(6):1431–1451, November 1982.

[10] B. Falkenhainer, K. D. Forbus, and D. Gentner. The structure-mapping engine: algorithm and examples. *Artificial Intelligence*, 41:1–63, 1989.

[11] Dedre Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7:155–170, 1983.

[12] E. M. Gold. Language identification in the limit. *Information and Control*, 16:447–475, 1967.

[13] Henry Hamburger and Kenneth Wexler. Identifiability of a class of transformational grammars. In K. J. J. Hintikka, J. M. E. Moravcsik, and P. Suppes, editors, *Approaches to Natural Language*, pages 153–1566. Reidel, Dordrecht, 1973.

[14] Kristian Hammond. Case-based planning: A framework for planning from experience. *Cognitive Science*, 14(3), 1990.

[15] J. R. Hurford. Biological evolution of the Saussurean sign as a component of the language acquisition device. *Lingua*, 77:187–222, 1989.

[16] Edwin Hutchins and Brian Hazlehurst. Learning in the cultural process. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II: SFI Studies in the Sciences of Complexity, Volume X*, pages 689–706, 1991.

[17] A. Kent, M. Berry, F. U. Leuhrs, and J. W. Perry. Machine literature searching viii: Operational criteria for designing i nformation retrieval systems. *American Documentation*, 6(2):93–101, 1955.

[18] David K. Lewis. *Convention: A Philosophical Study.* Harvard University Press, 1969.

[19] Brian MacWhinney and Elizabeth Bates. Functionalism and the competition model. In B. MacWhinney and E. Bates, editors, *The Crosslinguistic Study of Sentence Processing*, pages 3–73. Cambridge University Press, 1989.

[20] D. L. Medin and M. M. Schaffer. Context theory of classification learning. *Psychological Review*, 85:207–238, 1978.

[21] Melanie Mitchell. *Analogy-Making as Perception: A Computer Model.* The MIT Press, Cambridge, MA, 1993.

[22] Richard Montague. *Formal Philosophy.* Yale University Press, New Haven, 1974. Richard Thomason, editor.

[23] Michael Oliphant. Communication as altruistic behavior. Ph.D. Dissertation, Department of Cognitive Science, U.C.S.D., 1997.

[24] F. Harvey Pough, John B. Heiser, and William M. McFarland. *Vertebrate Life.* Prentice-Hall, fourth edition, 1996.

[25] D. Premack. The codes of man and beast. *Behavioral and Brain Sciences*, 6:125–167, 1983.

[26] J. M. Spence. Job market signaling. *Quarterly Journal of Economics*, 87:296–332, 1973.

[27] Luc Steels. Emergent adaptive lexicons. In P. Maes, editor, *Proceedings of the Simulation of Adaptive Behavior Conference.* MIT Press, 1996.

[28] Roger K. R. Thompson. Natural and relational concepts in animals. In Herbert L. Roitblat and Jean-Arcady Meyer, editors, *Comparitive Approaches to Cognitive Science*, pages 175–224. MIT Press/Bradford Books, Cambridge, MA, 1995.