The Multiple Word Guessing Game

Joachim De Beule

Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussel, Belgium,
joachim@arti.vub.ac.be,
http://arti.vub.ac.be/~joachim

Abstract. In this paper I bring together a number of insights in the field of semiotic dynamics. Inspired by recent advances in the understanding of the naming game and by previous attempts to define stages in the formation of lexical languages, we define two classes of language game problems called the single and multiple word guessing games. By being more complex than naming games, but not as complex as language games involving grammar, these problems form a next and tractable step in the understanding of the emergence of language.
I show that a variety of previously proposed setups can be casted as instances of the guessing game, particularly those that investigate cross-situational learning from single word utterances. It is also shown how the problem of solving a multiple word guessing game can effectively be reduced to solving a single word guessing game. The effectiveness of the proposed technique is verified on a large number of problems of varying complexity. Insights gained from the conducted experiments are then used for defining a learning algorithm that outperforms several other algorithms as proposed in literature.

1 Introduction

The past years have brought tremendous progress in the field of semiotic dynamics [19, 12]. We have reached the point where first and previously seemingly unrelated results can be brought together in a concise and foundational way. For example, much of earlier research can be described as finding appropriate problem definitions and possible accompanying solutions (see e.g. [13, 20, 7, 24]). Nowadays, the concept and methodology of language games have been picked up by numerous researchers from a variety of disciplines, in particular from statistical physics, and complex systems [1, 2, 28, 9, 8] and to a lesser extent from theoretical biology (see e.g. [11].) This has lead to an increased and profound understanding of what has become known as the naming game, a particular type of language game. Not only is there now a broad consensus about what exactly the naming game is, there also are numerous experimental and theoretical results that have brought order into the previously incoherent amalgam of intuitions about learning strategies for solving it.

So what's next? Although there already is a noticeable interaction between the frontiers of semiotic dynamics and linguistics (and in particular construction grammar, cognitive linguistics and evolutionary linguistics, see e.g. [6, 4, 25, 3, 15]), the remaining road to the understanding of languages of the complexity of human language is still long and remains to be demystified. In this paper, I aim to take a next step by first defining the guessing game as a class of language games slightly more complex than the naming game. I will show that a variety of previously proposed setups can be casted as an instance of it, particularly those that investigate cross-situational learning from single word utterances [17, 26, 27, 10, 18, 5]. Next, I will introduce the multiple word guessing game and show how the problem of solving it can be reduced to that of solving the simpler guessing game. The approach will be verified by combining it with three different CSL algorithms that have been proposed in the literature. Insights gained from this will then be used to define a variation on one of the algorithms that outperforms all three of them.

Most of the insights brought together in this paper were already present in some form in previous works, and were the result of a fruitful interaction between the fields of semiotic dynamics, statistical physics and complex adaptive systems.

2 The Multiple Word Guessing Game

Let me set the stage by describing a general language game [21]. It consists of a population of agents bootstrapping a language from scratch by engaging in repeated peer-to-peer interactions. We'll restrict ourselves to the mean field limit and ignore issues of topology.[1] Both agents involved in an interaction observe some scene. Then, one of them, the speaker agent, describes the scene to the other, or possibly a part of it. The game succeeds if the hearer agent agrees or is able to determine what was described. After an interaction, agents may change their internal state to become better at the game.

Obviously, the nature of the scenes and how they are observed greatly determines the complexity of the game, as well as the sort of linguistic constructs the agents may use . In the naming game, scenes contain a number of uniquely identifiable objects, like people, but unlike white cue billiard balls. And speaker agents may utter only one (holistic) word per interaction, like a proper name. Because all objects are uniquely identifiable, the speaker can reveal the object being described, e.g. by pointing to it. Hence, there is *no referential uncertainty*, and the dynamics of a naming game involving n objects reduces to the combination of n independent naming games about only 1 object[1]. This coincides with what is called the first stage in lexical communication systems by Steels in [23] and with the observational game in [26]. Currently there is a still growing consensus to call it the naming game. It is already well understood.

Several paths can be taken to increase the complexity of the naming game that all reduce to basically one type of game: the *guessing game.* In the guessing game, pointing is not possible. Hence, the hearer can merely guess what the speaker is naming and there is referential uncertainty. The same situation occurs when utterances may refer to specific *aspects* of objects, like their color or size,

[1] In other words: at all times, all agents have equal chances of interacting.

because then pointing does not reveal the aspect described. In both cases, a game involving n objects or aspects cannot be reduced to n independent 1-object games. What essentially happens is that some sort of cross-situational learning (CSL) is needed to learn the meaning of words. Several variations on the guessing game have been proposed, like the guessing game and the selfish game in [26], but all boil down to essentially the same game, merely differing in the amount of referential uncertainty (sometimes called the degree of joint attention) that is introduced (e.g. the number of objects per scene or the number of aspects per object.) An additional increase in complexity arises when utterances may consist of several words. This *multiple word guessing game* (MWGG) corresponds to Steels' third stage of lexical communication systems. No grammar is allowed, the meaning of an utterance is the simple union of the meanings of its containing words (e.g. there is no additional syntax like word order that adds meaning.)

We will formally define the MWGG in terms of some set $\mathcal{C}$ of abstract *categories*. These can be thought of as representing primitive parts of meaning. An interaction at time t involves a number of meanings called the game's *context* C_t. Each meaning is described by a collection of categories from $\mathcal{C}$. Because we want to allow multiple occurrences of the same category in a single meaning, meanings will be *bags* of categories rather sets.[2] Similarly, utterances will be bags of words. Let us define $B(S)$ as the set of all bags containing elements from the set S. Then meanings are elements of $B(\mathcal{C})$ and, if $\mathcal{W}$ is the set of all possible words, utterances are elements of $B(\mathcal{W})$. For convenience, let $\mathcal{M}$ stand for $B(\mathcal{C})$.

Crucial in the guessing game is how contexts and meanings are determined. We'll assume that meanings are generated according to a probability distribution $o : \mathcal{M} \to [0,1]^3$. Put differently: $o(x)$ is the probability of generating the meaning x. Furthermore, we'll assume that the number of meanings in the context at time t is determined by a probability function $s : \aleph_0 \to [0,1]$. Finally, we'll define an agent at time t as a tuple $\langle W_t, P_t, I_t, ...\rangle$, with W_t the set of words encountered by the agent so far, $P_t : B(\mathcal{W}) \times \mathcal{W} \to B(\mathcal{W})$ the agent's production function and $I_t : B(\mathcal{W}) \times \mathcal{P}(\mathcal{M}) \to \mathcal{M}$ the agent's interpretation function.[4]

---

[2] A bag is like a set but may contain duplicate elements. In this paper, squared brackets are used for denoting bags as in $[1, 2, 3]$.

The union of two bags $U_1 = [e_{11}, ..., u_{1k}]$ and $U_2 = [e_{21}, ..., u_{2l}]$ is again a bag $U = U_1 \cup U_2 = [e_{11}, ..., u_{1k}, e_{21}, ..., e_{2l}]$. In other words: even if for some i and j, $e_{1i} = e_{2j}$, they still are *both* part of the utterance's meaning. In the following, when the union of two bags is taken we mean the operation explained here. Hence we always have $|U| = |U_1| + |U_2|$.

The intersection, equality and difference of two bags are assumed to be defined in a similar fashion (e.g. $[1, 1, 2, 2] \cap [1, 2, 2]] = [1, 2, 2]$.)

Furthermore, we'll use the symbol ζ to denote the function that transforms a bag into a set (e.g. $\zeta([1, 2, 2, 3]) = \{1, 2, 3\}$.) For convenience, we'll also define ζ^{-1} as a function that transforms a set into a bag: $\zeta^{-1}(\{1, 2, 3\}) = [1, 2, 3]$.) Finally, $\#$ is a function that gives the number of occurrences of an element e in a bag B ($\#(1, [1, 2, 2, 3]) = 1$).

[3] Here not the bag $[0, 1]$ is meant but all real numbers between 0 and 1

[4] With $\mathcal{P}(S)$ the powerset of S is meant, i.e. the set of all subsets of S.

The Multiple Word Guessing Game (MWGG). *Every time step t, a speaker agent $s_t = \langle W_s, P_s, I_s \ldots\rangle$ and a hearer agent $h_t = \langle W_h, P_h, I_h, \ldots\rangle$ are randomly selected from a population of agents $\mathcal{A}$. Let C_t be the current context containing a number of meanings determined by s, with each meaning generated according to o*[5]*. Let τ_t be a random meaning from C_t (the* topic *meaning). The* outcome *o_t of the interaction is 1 if $I_h(P_s(\tau_t, W_t), C_t) = \tau_t$ and 0 otherwise. The goal is to maximize the outcome. After an interaction at time t, the speaker s_t may update his internal state by making use of C_t, τ_t, o_t and $u_t = P_s(\tau_t, W_t)$ (u_t is called the utterance of the game). The hearer h_t my only use C_t, u_t and o_t.*

The above definition is still rather general and can serve as a template for many types of games. A particular instance of a MWGG is fully specified by the number of categories $|\mathcal{C}|$, the number of words $|\mathcal{W}|$, the number of allowed words per utterance, the population size $|\mathcal{A}|$, and the probability functions o and s.

If only single word utterances are allowed and the context never contains less than three meanings, the MWGG reduces to a guessing game. If in this case the context size is less than three, it reduces to a naming game. Other reductions are also possible, for example if meanings always contain 1 category then the problem again reduces to a single word guessing game etc. Figure 1 summarizes some of the definitions in a diagrammatic fashion.

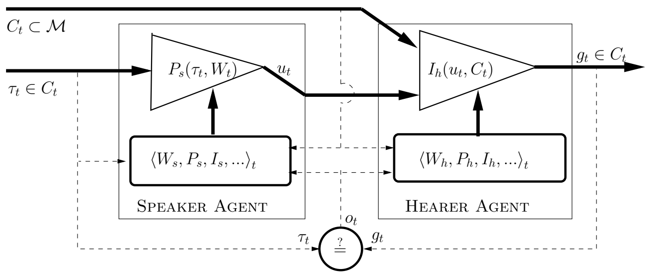

Fig. 1. Schematic representation of a language game at time t (solid flow lines) and learning (dashed flow lines). C_t is the game's context containing the topic meaning τ_t. It is transformed to an utterance u_t by the speaker's production function P_s. The hearer's interpretation function I_h transforms the utterance back into one of the meanings g_t in the context. The game succeeds if this guess equals the topic. In this case the outcome o_t equals 1, otherwise it is 0. (see text for more details.)

[5] Although it is assumed that no two meanings in the context are identical, otherwise the game could fail even when the hearer correctly parsed the speaker's utterance .

3 Problem analysis

In this section an analysis of the MWGG is performed that will allow the formulation of production and interpretation functions based on scored word/meaning mappings. The use of such scored mappings has almost become standard procedure in the field of language emergence modelling, and the most well known function of such scores is to introduce a preference mechanism through re-enforcement and lateral-inhibition. For example, in the naming game, what essentially happens is that several agents are proposing different names for naming an object. All agents can of course simply remember all proposed names and that would solve the problem. However, the resulting language would be very redundant. Moreover, this strategy is not very efficient for optimizing the outcome. If agents re-enforce successfully used names while inhibiting competing ones, the probability of having a successful interaction increases more rapidly.

Therefore, we'll further specify an agent as $\langle W_t, P_t, I_t, \phi_t, ...\rangle$, with $\phi_t : W_t \rightarrow [0, 1]$ a function imposing a preference among the words in W_t.

As explained, one crucial aspect of the naming game is the absence of referential uncertainty. This time however, we have to cope with it and next to preference scores, additional probability scores are required reflecting the correctness of a word/meaning mapping. A probability distribution $\sigma_t : \mathcal{W} \times \mathcal{M} \rightarrow [0, 1]$ linking words to meanings can be used for this. Given such a distribution, the meaning $m_t(w)$ of a word w can be defined as the one with the highest associated probability score:[6]

$$m_t(w) = \text{argmax}_m \sigma_t(w, m)$$

Hence, an agent's state is now fully specified as $\langle W_t, P_t, I_t, \phi_t, \sigma_t\rangle$. In the following we'll assume that a specific agent is given, i.e. that the sets W_t and the functions σ_t and ϕ_t are known.

3.1 The Production Function

The production function P_t transforms a meaning into an utterance. Every word w in an utterance contributes a number of categories $m_t(w)$ with a certain probability $\sigma_t(w, m_t(w))$.

Similarly, both words in a two-word utterance $u = [w_1, w_2]$ contribute to the meaning of the whole. We'll extend the definition of the meaning function m_t to the domain of utterances by letting the meaning of an utterance be the union of the meanings of its containing words:

$$m_t([w_1, ..., w_k]) \equiv \bigcup_{w_i} m_t(w_i)$$

Thus, given a meaning m and a set of words W_t, the set of utterances partially covering m is given by:

$$c_p(m, W_t) = \{u : (\zeta(u) \subset W_t) \wedge (m_t(u) \subset m)\} \quad (1)$$

[6] If several meanings have equal probability then one is selected at random, making m_t a stochastic function.

It contains all utterances u made of words from W_t and with meaning a part of m. The empty utterance [] is also assumed to be an element of $c_p(m, W_t)$.

A score $\psi(u, m) \in [0, 1]$ can be given to each utterance $u = [w_1, ..., w_k]$ as follows:

$$\psi([w_1, ..., w_k], m) \equiv \frac{1}{|m|} \sum_{w_i} \sigma(w_i, m_t(w_i))\phi(w_i)|m_t(w_i)| \tag{2}$$

Every word in the utterance contributes to this score. Words with higher confidence and probability values contribute more, as well as those contributing more meaning.

The production function P_t transforming the topic meaning τ_t into an utterance u_t can now be defined as follows. First the utterance u_p^* (partially) covering τ_t and with the highest score is determined:

$$u_p^*(\tau_t, W_t) = \text{argmax}_{u \in c_p(\tau_t, W_t)} \psi_t(u, \tau_t).$$

Now if this utterance covers the entire topic ($m_t(u_p^*) = \tau_t$) then $u_t = P_t(\tau_t, W_t) \equiv u_p^*(\tau_t, W_t)$.

Otherwise a new word w^* is added contributing the uncovered categories in the topic ($u_t = u_p^* \cup [w^*]$ with $m_{t+1}(w^*) = \tau_t \setminus m_t(u_p^*)$.)[7] In both cases we obviously have $m_{t+1}(u_t) = \tau_t$.

To conclude this section we mention that finding the utterance with maximal score can be casted as a general informed search problem (see e.g. [14]), allowing it to be solved efficiently by a standard A^* algorithm.

3.2 The Interpretation Function

The interpretation function I_t needs to select one of a number of meanings from a context C_t based on the utterance u_t.

This time, for every meaning $m \in C_t$ at least as big as the utterance ($|m| \geq u_t$), the set of utterances made only of words from u_t and partially covering m can be determined:[8]

$$c_i(m, u_t) = \{u : (u \subset u_t) \wedge (m_t(u) \subset m) \wedge (m_t(u) = m \text{ if } u = u_t)\} \tag{3}$$

From these, let $u_i^*(m, u_t)$ be the highest scoring utterance:

$$u_i^*(m, u_t) = \text{argmax}_{u \in c_i(m, u_t)} \psi_t(u, m).$$

$u_i^*(m, u_t)$ is the part of u_t that optimally describes the meaning m. The interpretation function can now be defined as follows:

$$I_t(u_t, C_t) = \text{argmax}_{m \in C_t} \psi_t(c_i(m, u_t), m).$$

[7] This amounts to the following speaker agent state updates: $W_{t+1} = W_t \cup \{w^*\}$, $\sigma_{t+1} = \sigma_t$ except for $\sigma_{t+1}(w, \tau_t \setminus m_t(u_p^*)) = 1.0$ and $\phi_{t+1} = \phi_t$ except for $\phi_{t+1}(w^*) = 1.0$.

[8] The difference between c_p and c_i lies in the fact that for $c_p : \zeta(u) \subset W_t$ while for c_i we have: $u \subset u_t$.

In words, it returns that meaning m from the context that is best described by the utterance u_t in the sense that the agent would be most confident about m if it had to describe one of the meanings from the context itself but using only words from u_t. If there is a complete parse in the sense that all words in u_t can be used for fully describing some meaning m then it will also be the value of I_t.[9]

Note that again formulating the problem as a standard informed search problem allows for incremental parsing by considering the words in the same order as they occur in the utterance. It could be made even more efficient by using a chart (as in *chart parsing*), but we suspect that this would not lead to a great increase in performance, something that needs to be analyzed further.

3.3 Preference Scores

In the previous sections, we have fully defined the production and interpretation functions in terms of the set W_t of known words and the probability and preference functions σ_t and ϕ_t. But we still need to define how these functions can be initialized and how they are updated after an interaction.

From the naming game, we known how the preference function should be updated: according to some enforcement plus lateral inhibition scheme [22]. Hence, after every interaction, regardless of the outcome of the game, for the hearer agent we have: $W_{t+1} = W_t \cup \zeta(u_t)$ and

$$\phi_{t+1}(w) = \begin{cases} \theta + (1-\theta)\phi_t(w) & \text{if } w \in u_t, \\ (1-\theta)\phi_t(w) & \text{if } w \notin u_t \text{ but} \\ & m_t(w) = m_t(w^*) \text{ for some } w^* \in u_t \\ \phi_t(w) & \text{otherwise,} \end{cases}$$

with $\theta \in [0, 1]$ a learning parameter.

Speaker agents do not obtain much information from an interaction: it is not because some other agent understood you that he also would have produced the same utterance. On the other hand, if the interaction fails, the speaker can conclude that he did something wrong. In order to avoid suboptimal stable states (e.g. no state changes even in case of continued failure), speakers do demote preference scores of used words in this case. (see also [8] for a discussion on when to update.) For the same reason that only hearer agents apply lateral inhibition, only hearers update the probability function σ_t. How that can be done is dealt with in the next section.

3.4 Probability Scores

Updating the probability function σ_t requires a set of meanings M_t^* for learning the meaning of u_t. If the hearer's guess $g_t = I_t(u_t, C_t)$ was correct ($g_t = \tau_t$), he

[9] Unless the problem is ill-defined and several meanings have a ψ value of 1, but we excluded this possibility, see footnote 5.

should of course use it ($M^*_t = \{g_t\}$). Otherwise, we propose to use the set of second best meanings in C_t:[10]

$$M^*_t = \text{argmax}_{m \in C_t \setminus \{I_t(u_t, C_t)\}} \psi_t(c_i(m, u_t), m).$$

The optimal parse of u_t for a meaning $m^*_t \in M^*$ is given by $u_{m^*_t} = u^*_i(m^*_t, u_t)$.[11] Although, the parse might be optimal, it is not necessarily complete. Indeed, the only things guaranteed are that $u_{m^*_t}$ is a part of u_t and that it is compatible with m^*_t (i.e. $u_{m^*_t} \subset u_t$ and $m_t(u_{m^*_t}) \subset m^*_t$). But it could just as well be empty.

In case it is not empty but instead covers the entire utterance ($u_{m^*_t} = u_t$),[12] the agent can proceed with strengthening the word meaning associations between the words w in u_t and their meanings $m_t(w)$.

If however at least part of the utterance was left unused ($|u_{m^*_t}| < |u_t|$) then some of the words in u_t should still be mapped onto the part of m^*_t not covered by $u_{m^*_t}$.

To formalize all this, we'll define the *assignment function* ω. Let $\pi(B, n)$ be the bag of partitions of n nonempty sub-bags of the bag B (with $n \leq |B|$). For example:

$$\begin{aligned} \pi([1,2,2],1) &= [[[1,2,2]]], \\ \pi([1,2,2],2) &= [[[1,2],[2]],[[1,2],[2]],[[1],[2,2]]], \\ \pi([1,2,2],3) &= [[[1],[2],[2]]]. \end{aligned}$$

Furthermore, let

$$\overline{\pi}(B, n) = \bigcup_{P \in \pi(B,n)} P.$$

For example:

$$\begin{aligned} \overline{\pi}([1,2,2],1) &= [[1,2,2]], \\ \overline{\pi}([1,2,2],2) &= [[1,2],[2],[1,2],[2],[1],[2,2]], \\ \overline{\pi}([1,2,2],3) &= [[1],[2],[2]]. \end{aligned}$$

Then:

$$\omega(w, u_{m^*_t}) = \begin{cases} [m_t(w)] & \text{if } w \in u_{m^*_t} \\ \overline{\pi}(m^*_t \setminus m_t(u_{m^*}), |u_t| - |u_m|) & \text{otherwise.} \end{cases} \quad (4)$$

[10] the *argmax* function is supposed to return the set of all meanings maximizing its body argument here.

[11] It might seem contradictory to call $u^*_i(m^*_t, u_t)$ a *parse* since it is an utterance itself. It is an optimal parse however in the sense that it determines that part of the utterance u_t that can optimally be used to obtain m^*_t while 'parsing' u_t. It is as if how an utterance u_t is parsed depends on a target meaning m and the 'parse' of u_t given m is $m_t(u^*_i(m, u_t)) \subset m$.

[12] Then, necessarily also $|M^*_t| = 1$ and $m_t(u_{m^*_t}) \equiv m^*_t$, see last conjunct in equation (3)

Hence, for every meaning $m_t^* \in M_t^*$ and utterance u_t, the optimal parse $u_{m_t^*} = u_i^*(m_t^*, u_t)$ can be determined and from it for every word w a bag $\omega(w, u_{m_t^*})$ most probably containing the word's meaning. This is done such that that known and successful words are assigned their proper meaning only, while unknown and unsuccessful words are assigned all possible word meanings left uncovered after optimally parsing the utterance. This information can be used to update the probability function σ_t by cross situational learning.

4 Cross Situational Learning

Cross situational learning has to do with learning the meaning of a word w^* from subsequent examples E_t of possible meanings. In our case, the words are given by the utterance u_t and the examples by $E_t = \cup_{m_t^* \in M_t^*} \omega(w^*, u_{m_t^*})$.

CSL has been approached in a variety of ways in literature. Most approaches belong to one of three categories, here dubbed the *elimination* approach (see [18] and in some sense also [16]), the *enumeration* or *Bayesian* approach [26, 10, 17] and the *adaptive* approach [5]. I'll briefly discuss all three of them in the following.

4.1 Elimination Algorithm

One straightforward CSL algorithm consists of simply eliminating all inconsistent meanings across examples (situations) in the hope that in the end only one, the *'correct'* one, will remain. It can be implemented by keeping an initially empty set $M_t(w^*)$ of possible meanings for w^*, removing elements from it not present in the example E_t:

$$M_{t+1}(w^*) = \begin{cases} M_t \cap \zeta(E_t) & \text{if } M_t \cap \zeta(E_t) \neq \emptyset, \\ \zeta(E_t) & \text{otherwise.} \end{cases}$$

Hence we have:

$$\sigma_{t>0}(w^*, m) = \begin{cases} \frac{1}{|M_t(w^*)|} & \text{if } m \in M_t(w^*), \\ 0 & \text{otherwise.} \end{cases}$$

4.2 Enumeration Algorithm

When acquiring an existing and stable language, the elimination algorithm suffices. However, when bootstrapping a language, it is not expected to perform so well because eliminating all inconsistent meanings quickly leads to empty meaning sets and hence loss of information. Therefore, instead of throwing away all inconsistent meanings, the enumeration algorithm keeps track of the number of times a certain meaning has occurred. Frequently occurring meanings have a higher chance of being correct, but less frequently occurring meanings also have nonzero probability.

We can model this by letting $M_t(w^*)$ be a bag instead of a set:

$$M_{t+1}(w^*) = M_t \cup E_t$$

and

$$\sigma_{t>0}(w^*, m) = \frac{\#(m, M_t(w^*))}{|M_t(w^*)|}$$

4.3 Adaptive Algorithm

One problem with the enumeration algorithm is that it looses its adaptivity as time proceeds. In [5], an algorithm was proposed that solves this problem. If the word w^* is observed for the first time at time t, we have:

$$\sigma_{t+1}(w^*, m) = \begin{cases} \frac{\#(m, M'_t)}{|M'_t|} & \text{if } m \in M'_t, \\ 0 & \text{otherwise.} \end{cases}$$

If the word was already known then the total (posteriori) probability of all meanings consistent with the example E_t is increased while that of all inconsistent meanings is decreased. So far, we could just as well have been describing the enumeration algorithm. However, this time it is done as follows. Let γ be the total a-priori probability of all meanings in E_t ($\gamma = \Sigma_{m \in \zeta(E_t)} \sigma_t(w^*, m)$). This will be increased to $\gamma' = (1-\alpha)\gamma + \alpha$[13], but in such a way that if E_t is in accordance with the current state (i.e. γ large) then the relative differences between the probability scores of consistent meanings is kept. If however E_t is not in accordance with the current state (γ small) than differences are flattened. This is controlled by an interpolation factor $\beta(\gamma) = \sqrt{1-(1-\gamma)^2}$. In sum this amounts to, for all $m \in \zeta(E_t)$:

$$\sigma_{t+1}(w^*, m) = \beta(\gamma)\sigma_t(w^*, m)\frac{\gamma'}{\gamma} + (1-\beta(\gamma))\frac{\gamma'}{|\zeta(E_t)|}$$

In the extreme cases one gets:

- $\gamma = 1$:

$$\sigma_{t+1}(w^*, m) \equiv \sigma_t(w^*, m)$$

- $\gamma = 0$:

$$\sigma_{t+1}(w^*, m) = \begin{cases} \alpha \frac{1}{|\zeta(E_t)|} & \text{if } m \in M'_t, \\ (1-\alpha)\sigma_t(w^*, m) & \text{otherwise,} \end{cases}$$

Note that if there is only one consistent meaning (i.e. $\zeta(E_t) = \{m^*\}$) then we have:

$$\sigma_{t+1}(w^*, m^*) = (1-\alpha)\sigma_t(w^*, m^*) + \alpha,$$
$$\sigma_{t+1}(w^*, m \neq m^*) = (1-\alpha)\sigma_t(w^*, m).$$

[13] $\alpha \in [0, 1]$ is a learning parameter

The a-priori probability of inconsistent meanings is obviously given by $\delta = 1 - \gamma$. This needs to be decreased to $\delta' = 1 - \gamma'$ (with $m \notin E_t$):

$$\sigma_{t+1}(w^*, m) = \sigma_t(w^*, m)\frac{\delta'}{\delta}$$

In the next section, the performance of all three algorithms will be compared on a number of multiple word guessing games.

5 Comparison of Algorithms

5.1 Problem Space and Presentation of Results

Before presenting results, it is first explained which particular set of problems we used for testing. Recall that a MWGG is specified by the number of categories $|\mathcal{C}|$, the number of words $|\mathcal{W}|$, the number of allowed words per utterance, the population size $|\mathcal{A}|$, and the probability functions o and s.

In the following, we'll assume that the number of words is infinite. Furthermore, the maximum number of allowed words per utterance is equal to the number of categories in the topic, but may be less. The population size is taken to be 5. We ran experiments for varying number of categories n_c between 1 and 8. The meanings contained in the context were randomly selected from $\{m : (m \in B(\mathcal{C})) \wedge (l_{\text{min}} \leq |m| \leq l_{\text{max}})\}$ (but never drawing the same meaning twice at the same time.) The context size s_c as well as the parameters l_{max} and l_{min} were kept fixed during the course of an experiment, but different experiments were done for varying values of s_c (ranging from 1 to n_c), l_{min} (from 1 to $n_c - 1$) and l_{max} (from $l_{\text{min}} + 1$ to n_c). Hence, in total 546 problems of varying complexity were used for testing.

All three CSL algorithms were then used in combination with the proposed general approach for tackling the MWGG to solve each problem for 32 times with different random seed values, amounting to 17472 runs per strategy.

Every run, the evolution of the outcome o_t was tracked as a running average $\overline{o}_t$ with averaging interval 100 time steps and initial value 0 ($\overline{o}_0 = 0$ and $\overline{o}_{t+1} = 0.99\overline{o}_t + 0.01o_t$. This value was checked every 10 interactions (at times $t = 0, 10, 20, ...$). Whenever it was equal or greater to 99%, the time was recorded.

Next, per problem and per strategy, the average was taken over all 32 runs. Finally, only those results were kept that had a non-zero standard deviation. The remaining times are the ones that will be presented in the next sections as 'convergence times' (y-axes of graphs.)

Because different problems are of different complexity, they generate different values for the convergence time. In order to allow transparent presentation of all results at once, we defined the complexity of a problem defined by n_c, l_{min}, l_{max} and s_c as:

$$223.67 + 17.17n_p + 37.32s_c + 10.70l_{\text{min}} + 24.71l_{\text{max}}.$$

This equation was obtained from a multi-variate linear regression of the convergence times of the adaptive strategy. In other words, using this as x-value when plotting convergence times for different problems results in that the time needed for the adaptive strategy to solve a problem will be optimally (according to the least square error) and linearly correlated with the problem's complexity and hence this value serves as a measure for the complexity of the problem.

5.2 Results

Figure 2 shows differences in convergence times for the three different algorithms. The adaptive and elimination strategies managed to solve all problems,

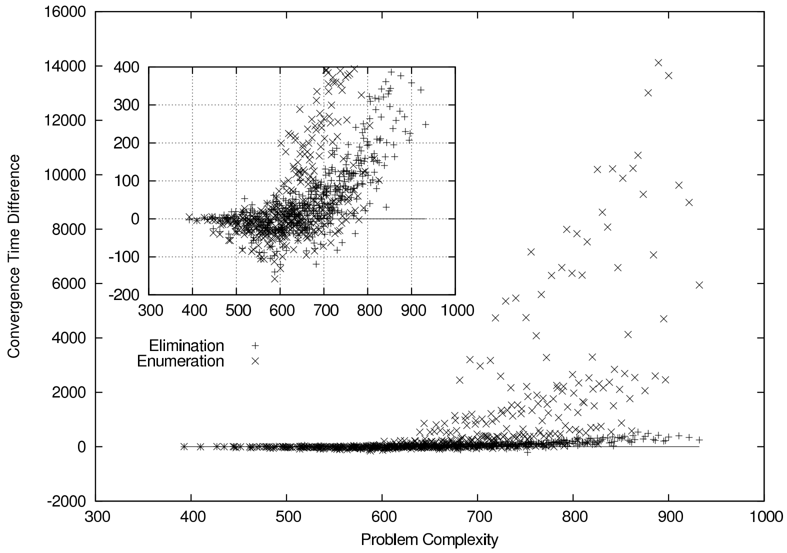

Fig. 2. Differences in convergence times of the elimination and enumeration strategies compared to the adaptive strategy. A positive value means that the strategy requires more iterations than the adaptive one, a negative value means less that iterations are needed.

the enumeration algorithm sometimes failed to solve more complex problems. The adaptive strategy clearly performs best.[14]

The differences between the different strategies can be understood in terms of flexibility. As explained, the elimination strategy remains extremely flexible independent of the learning history and new information may sometimes even result in the complete forgetting of possibly still useful information. On the other

[14] It should be mentioned that the elimination strategy is mostly applied for learning a stable and existing language rather then for bootstrapping a language from scratch. In this case all three algorithms manage to solve all problems but the enumeration algorithm still performs significantly worse while the elimination algorithm performs slightly better than the adaptive algorithm.

side of the spectrum lies the enumeration algorithm which becomes less and less flexible as time proceeds. Although this needn't per se be a bad strategy (think of simulated annealing or other mechanisms that initially promote exploration while gradually preferring consolidation), the way in which the enumeration algorithm does it clearly is not optimal.

These insights can be used to devise yet a better algorithm from the adaptive one by letting the agents vary their learning parameter α according to an estimate of the current communicative success. As can be seen in Figure 3, this indeed leads to the best performance.

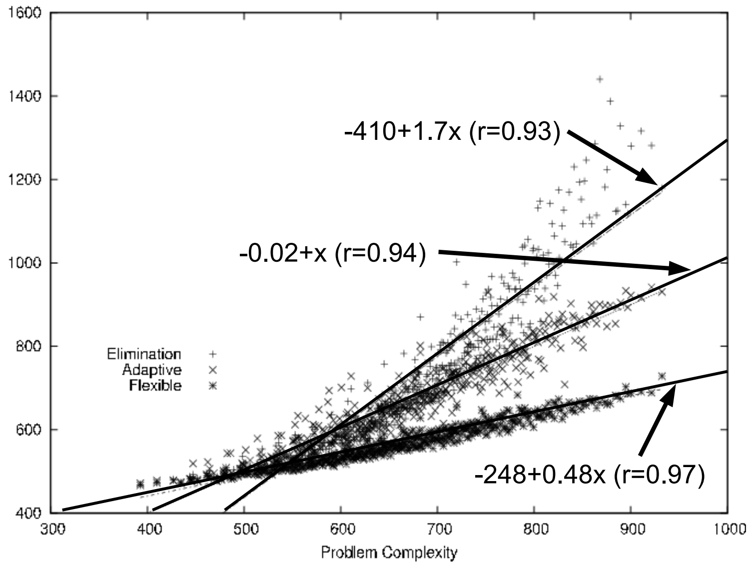

Fig. 3. Convergence times for problems of different complexity for the elimination, the adaptive and a flexible cross-situational learning strategy together with linear regression lines approximating the available data quite well (see text for more details.)

6 Conclusion

In this paper we have attempted to bring together a number of insights in the field of semiotic dynamics. Inspired by recent advances in the understanding of the naming game and by previous attempts to define stages in the formation of lexical languages, we have defined two classes of language game problems called the single and multiple word guessing games. The fundamental differences with the naming game are that there is referential uncertainty (which is why they are called guessing games) and, in case of the MWGG, that utterances may contain several words. By being more complex than naming games, but not being as complex as language games involving grammar, they form a next and tractable step in the understanding of the emergence of language.

It was shown that a variety of previously proposed setups can be casted as instances of the guessing game, particularly those that investigate cross-situational learning from single word utterances. It was also shown how the problem of solving a multiple word guessing game can effectively be reduced to solving a single word guessing game. The effectiveness of the proposed technique was verified by combining it with three different CSL algorithms that have been proposed in the literature. Insights gained from the conducted experiments were then used to define a variation on one of the algorithms that outperforms all three of them.

7 Acknowledgements

This paper was partly supported by the EU ComplexDis project. I would like to thank Ann Nowé and Joris Bleys from the Vrije Universiteit Brussel (VUB) for providing me with useful feedback on previous versions of this paper.

References

1. A. Baronchelli, M. Felici, E. Caglioti, V. Loreto, and L. Steels. Sharp transition towards shared vocabularies in multi-agent systems. *J. Stat. Mech.*, P06014, 2006.
2. Andrea Baronchelli, Luca Dall'Asta, Alain Barrat, and Vittorio Loreto. Strategies for fast convergence in semiotic dynamics. In Luis M. Rocha et al., editor, *Artificial Life X*, pages 480–485. MIT Press, 2006.
3. William Croft. Language structure in its human context: new directions for the language sciences in the twenty-first century. In Patrick Hogan, editor, *Cambridge Encyclopedia of the Language Sciences*. Cambridge University Press, Cambridge, 2007.
4. Joachim De Beule and Benjamin K. Bergen. On the emergence of compositionality. In *Proceedings of the 6th International Conference on the Evolution of Language*, pages 35–42, 2006.
5. Joachim De Beule, Bart De Vylder, and Tony Belpaeme. A cross-situational learning algorithm for damping homonymy in the guessing game. In Luis M. Rocha et al., editor, *Artificial Life X*, pages 466–472. MIT Press, 2006.
6. Joachim De Beule and Luc Steels. Hierarchy in fluid construction grammar. In U. Furbach, editor, *Proceedings of KI-2005*, volume 3698 of *Lecture Notes in AI*, pages 1–15, Berlin, 2005. Springer-Verlag.
7. Edwin D. De Jong. *Autonomous Formation of Concepts and Communication*. PhD thesis, Vrije Universiteit Brussel AI-lab, 2000.
8. Bart De Vylder. *The Evolution of Conventions in Multi-Agent Systems*. PhD thesis, VUB Artificial Intelligence Lab, 2007.
9. Bart De Vylder and Karl Tuyls. How to reach linguistic consensus: A proof of convergence for the naming game. *Journal of Theoretical Biology*, 242(4):818–831, October 2006.
10. Federico Divina and Paul Vogt. A hybrid model for learning word-meaning mappings. In P. Vogt and et al., editors, *Symbol Grounding and Beyond: Proceedings of the Third International Workshop on the Emergence and Evolution of Linguistic Communication*, pages 1–15. Springer Berlin/Heidelberg, 2006.

11. Tom Lenaerts, Bart Jansen, Karl Tuyls, and Bart De Vylder. The evolutionary language game: An orthogonal approach. *Journal of Theoretical Biology*, 235(4):566–582, August 2005.
12. Vittorio Loreto and Luc Steels. Social dynamics: Emergence of language. *Nature Physics*, 3:758–760, November 2007.
13. Michael Oliphant and John Batali. Learning and the emergence of coordinated communication. *The newsletter of the Center of Research in Language*, 11(1), 1997.
14. S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approch*. Prentice Hall, Inc., Upper Saddle River, New Jersey 07458, 1995.
15. Stephen Luke Shead. *Radical frame semantics and biblical Hebrew: exploring lexical semantics*. PhD thesis, University of Sydney, Department of Hebrew, Biblical and Jewish Studies, July 2007. online at http://hdl.handle.net/2123/2016.
16. J.M. Siskind. A computational ctudy of cross-situational learning techniques for learning word-to-meaning mappings. *Cognition*, 61:39–91, 1996.
17. Andrew D.M. Smith. Establishing communication systems without explicit meaning transmission. In J. Kelemen and P. Sosik, editors, *Proceedings of the European Conference on Artificial Life (ECAL01), Prague*, pages 381–390, Berlin, 2001. Springer.
18. K. Smith, A. Smith, R. Blythe, and P. Vogt. Cross-situational learning, a mathematical approach. In P. Vogt, Y. Sugita, E. Tuci, and C. Nehaniv, editors, *Proceedings of the Emergence and Evolution of Linguistic Communication (EELCIII*. Springer, 2006.
19. L. Steels. Semiotic dynamics for embodied agents. *IEEE Intelligent Systems*, 21(3):32–38, May/June 2006.
20. L. Steels and A. McIntyre. Spatially distributed naming games. *Advances in complex systems*, 1(4), January 1999.
21. Luc Steels. Emergent adaptive lexicons. In P. Maes, editor, *Proceedings of the Simulation of Adaptive Behaviour Conference*. The MIT Press, Cambridge, Ma., 1996.
22. Luc Steels. Grounding symbols through evolutionary language games. In Angelo Cangelosi and Domenico Parisi, editors, *Simulating the evolution of language*, pages 211–226. Springer-Verlag New York, Inc., New York, NY, USA, 2002.
23. Luc Steels. The emergence and evolution of linguistic structure: from lexical to rammatical communication systems. *Connection Science*, 17(3-4):213–230, 2005.
24. Luc Steels and Paul Vogt. Grounding adaptive language games in robotic agents. In Phil Husbands and Inman Harvey, editors, *Proceedings of the Fourth European Conference on Artificial Life (ECAL'97), Complex Adaptive Systems*, Cambridge, MA, 1997. The MIT Press.
25. Remi Van Trijp. The emergence of semantic roles in fluid construction grammar. In *Accepted for the 7th evolution of language conference (Evolang 7)*, 2008.
26. Paul Vogt and Hans Coumans. Investigating social interaction strategies for bootstrapping lexicon development. *Journal of Artificial Societies and Social Simulation*, 6(1), Januari 2003.
27. Paul Vogt and A.D.M Smith. Learning colour words is slow: a cross-situational learning account. *Behavioral and Brain Sciences*, 28(4):509–510, 2005.
28. Bart De Vylder. Coordinated communication, a dynamical systems perspective. In *Proceedings of the European Conference on Complex Systems (ECCS06)*, 2006.