Modeling Learning of Derivation Morphology in a Multi-Agent Simulation.

Olga Pustylnikov
University of Bielefeld,
Universitaetsstr. 25, 33615 Bielefeld, Germany
olga.pustylnikov@uni-bielefeld.de

***Abstract*—In this paper we simulate language acquisition by focussing on the emergence of derivation morphology. We assume that modeling the learning behavior of humans can help to enhance methods in language processing and retrieval. That is, when we understand how humans learn, we can design systems applying the same techniques in language processing. Children, acquiring the first language, observe the adults' speaking before learning how to express themselves. Learning is a gradual process of acquiring single sounds (phonology), words (lexis), and more complex constructions (morphology, syntax) [1]. A newly learned material is acquired and recognized by already existing knowledge and similarities on each linguistic level contribute to the recognition of new words [2]. Despite these observations common simulation models do not consider morphological and phonological information within automatic learning processes. Here, we extend the scope of these models focusing on derivational morphology as a means of language comprehension and production.**

I. Introduction

How do children acquire language? The most striking fact in language learning is the ability of humans to infer grammatical relations that are not explicitly learned. Sometimes this fact is called "the poverty of the stimulus" that children can easily overcome compared to non-humans (animal or artificial subjects). There are various explanations for this ability in the literature. [4] explains this ability of human beings by an innate universal grammar which is instantiated with specific parameters of a particular language. By contrast, a functionalist perspective assumes a dynamic grammar construction based on language use [1]. In this second approach learning has a greater importance for language acquisition than innateness [5]. We argue for the last approach and claim together with [2] that regularities are not explicitly learned but rather automatically induced from the input language of the communication partner.

In this paper we concentrate on the acquisition of the derivation rules for the main parts of speech (POS)[1] as a dependent variable and the input language as an independent variable in a child-adult simulation game. The input languages are varied - we test the model on two natural languages (English and German) and on a randomly generated word set. In principal, any language can be input to the system. If a language does not use suffixation to derive new words, more training will be required to learn the correct word-to-POS mappings (since no regularities within the word can be detected).

The theoretical background of the presented system is described in section II. The general scenario of the simulation is explained in section III. The experimental procedure is described in section IV. Results are presented and discussed in section V. Finally, we give some conclusions in section VI.

II. Theoretical Background

In the literature two mechanisms of morphological processing are outlined (*dual route model*): the full word route and the parsing route. Both mechanisms are supposed to dependent on word frequency. A highly frequent word has a greater accessibility in the mental lexicon and can be easily recognized as a whole (full word route) without decomposition into stem and affixes. Processing a low frequent word, decomposition, in turn, might bring additional support and facilitate the recognition task (parsing route). A suffix, for example, that is often used to derive one word (class) from another (e.g., *ease* $>$ *eas-y*, i.e., adjective from verb) can facilitate the processing. In general, each peace of information that can be identified within and beyond the word is utilized in processing.

In languages that make use of derivational morphology, derivational suffixes are used to derive, for example, an adjective from a noun. In some languages (like in German), there are more than one suffixes forming the same word class, which are supposed to compete during the evolution of language. Productive suffixes are those that are used more frequently in word formation than other [9]. However, this assumption is not uncontroversial, since some suffixes reflect different semantic functions in language, that is, they are restricted to different semantic domains of the same word class, and thus, they are not in competition. For example, two derivations *'Gleich-ung'* and *'Gleich-nis'* of the stem *'gleich-'* produce different semantic meanings in German. Furthermore, some derivation suffixes can be fossilized, that is, they are no longer used for production, although, many words formed by these suffixes might exist.

In this paper, we neither make semantic restrictions to the input language nor claim for a real competition of suffixes.

This work is supported by the SFB 673 Alignment in Communication (http://www.sfb673.org/) funded by the German Research Foundation (DFG).

The system presented here is programmed by Roman Pustylnikov as part of his diploma thesis in Information Technology.

[1]The terms "POS" and "word class" are used interchangeably throughout the paper.

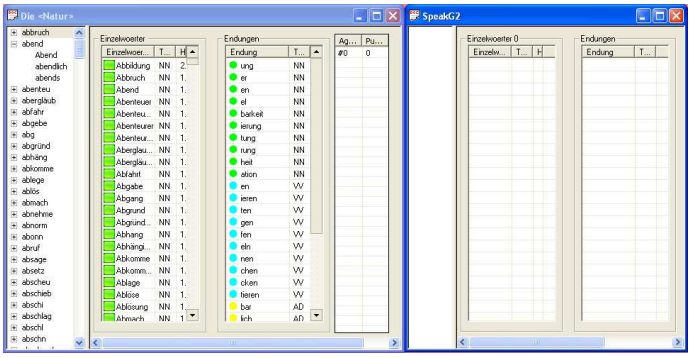

Figure 1. The adult (A) dialogue box (left) and the empty child (C) dialogue box (right).

However, the presented model allows to test these assumptions for a particular language by examining the identified suffixes.

In an *Evolutionary Game Theoretic* framework [3] child-agents C communicate with an adult-agent A and with each other. There are word forms F and meanings M, but the meaning space is restricted to the main word classes. At the beginning, the adult is selected as a sender S, and it utters a word to a randomly selected receiver-child R. The receiver tries to guess the word class of the uttered word and if successful, it becomes the sender. The game is motivated by the research in first language acquisition where children are faced with the problem to segment the speech of the adult and to induce regularities facilitating the interpretation. The segmentation stressed in this game concerns the word level, and utilizes the regularities within the word to guess the word class. In evolutionary game theoretic terms, the population is the number of possible suffixes frequently occuring in words of a concrete word class. Successful suffixes, that improve the recognition are *replicated*, i.e., reused in future conversations. We look which suffixes actually win the race in the natural as well as in the random scenarios.

The difference of this model and the common evolutionary game theoretic models lies in the ability of the agents to induce the word class of the word via decomposition.[2] That is, agents segment words into derivation suffixes and stems. This mechanism is complementary to the full word recognition in language processing according to the *dual route model* (see the previous section).

III. Game Setting

In this section we describe the implementation of the game. The agent architecture is described in section III-A. The decomposition algorithm the agents use is presented in section III-B. Finally, section III-C summarizes the overall game scenario.

A. Agent Architecture

The snapshot of the system in Fig. 1 illustrates the different types of agents at the beginning of the game.

[2]Cf. [10] for a simulation of compositionality.

1) Adult agents: The adult window consists of four parts. The first column containing a tree view shows the **derivation network**. It lists all possible word derivations of a single word with the corresponding stem, e.g., the sub-tree of the word "Abend" (evening) is *abend* > *Abend* > *abendlich* > *abends*. The derivation network is a subset of the **lexicon** containing all single words of the adult with the corresponding word classes and frequencies that are listed in the second column. The third column lists the suffixes according to their priority (See section III-B for a description on how these priorities are obtained) and their word class. The last column maintains the statistics about the communication scores of the agents.

In order to load the derivation network and the lexicon two kinds of data are required. The derivation network is constructed from a text file where each line corresponds to a single "word family"[3], for example:

```
Abhang 0 0 abhängen 1 0 abhängig 2 0 Abhängigkeit 0 0
```

A word is followed by two numbers, the first representing the word class (0 = noun, 1 = verb, 2 = adjective) and the second its frequency. Frequencies are attributed to the use of this word during the game, thus, they are initially set to 0 in the derivation network of the adult.

We have created the German derivation network manually using www.canoo.net and the English network by consulting a dictionary. At first glance, it seems rather awkward to create such a word-family file each time for a language. The initial German file contains 421 entries, i.e., 421 word families. However, since for English we have a much smaller amount of word families (only 49 entries), we selected a random subset of the German derivation network containing 49 entries (see the results for all the three networks in section V).

The second text file is loaded to construct the lexicon. This file contains a previously tagged text corpus (tagged with word class information, e.g., by TreeTagger[4]). This output file is loaded when creating the adult, and all the words that are nouns, verbs or adjectives (+ adverbs) are stored in the lexicon together with their frequencies of occurrence. These frequencies are later used to compute the probability with which a word is uttered by the agents.

In this version of the system, we restrict the word classes to three main classes: verbs, nouns and adjectives. Adverbs are subsumed to the class of adjectives constituting a more general classs of attributes. This simplification is made since derivation processes are mostly observable for these main word classes. However, the model can be easily extended to other word classes or integrated as a module into a more sophisticated decomposition system.

2) Child agents: In contrast to the adult, the child agents have a three column window (Fig. 1, on the right) . All fields are empty at the beginning. In analogy to the adult agent, the child's first sub column is reserved for the derivation network

[3]A *word family* is a group of words which share a common stem and have a common origin. Words within a family can be spread over different word classes [11].

[4]http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html.

and the second for the lexicon. The third column will contain the suffixes induced by analyzing single words. All three parts are filled during the game.

B. Decomposition Algorithm

Each agent (adult or child) is endowed with an internal decomposition mechanism that contains no built in or learned information about a particular language. Thus, in principal any language can be tested by the system. The algorithm utilizes two sorts of information: the lexicon of single words with the corresponding word classes and the derivation network. For the adult these sources are loaded from the files, children construct them on the fly by inducing the regularities from single words. Obviously, the availability of the derivation network which might be hardly obtainable for a particular language is not required (see the small size of the English and the reduced German networks). In this section we describe two basic algorithms used here to obtain the suffixes from the derivation network and from single words.[5]

1) Suffix Induction from the Derivation Network: A derivation network of an agent consists of sub trees representing a common stem (if available) and the possible derivations. For the word "anerkennen" (accept), for instance, the tree would consist of the stem "anerkenn-" and the derivations "anerkennen" (*accept* as a verb) and "Anerkennung" (*acceptance* as a noun). The algorithm does the following:

i. For each derivation tree D (e.g., *anerkennen*, *Anerkennung*) a common stem $St = l_1 \cdots l_n$, where $l_1, ..., l_n$ are letters, is identified by comparing the words letter by letter. Thus $St \bigcup D$ is the common part of all words in D (that is *anerkenn-* in the above example).[6]
ii. Capitals are converted to lowercase letters and umlauts are converted to ASCII. When the stem is identified, the resulting suffixes are added to the suffix collection of the corresponding word class.
iii. Finally, the best (i.e., most frequent) suffixes are merged with the best from the lexicon.

2) Suffix Induction from Lexicon: To find the significant suffixes from the collected set of words four filtering steps are performed. At the beginning all the words are sorted by word class $c \in C$ and each group is analyzed separately. Formally, a lexicon $\mathbb{L} = \bigcup_c L(c)$ with $L(c)$ as a set of words of class c is given. That is, a lexicon $\mathbb{L} = \{\langle w, c\rangle\} | w \in W, c \in C\}$ consists of pairs $\langle w, c\rangle$ of words and classes. Each word $w = l_1 l_2 \cdots l_n$ is represented as a letter string [14]. A suffix s is a substring $s = l_j \cdots l_n$ of w, with $j \leq n - 7$. [7]

[5]The algorithms presented in the following sections use similar techniques as in [12], [13]. We do not compare the proposed method to these approaches since we are primarily interested in derivation morphology here. However, future work should evaluate the performance of the decomposition mechanism introduced in this paper contrasting it with other methods in this area.

[6]Of course, the morpheme analysis of 'anerkennen' presented here is not complete since pre-, in- and circum-fixes are not considered. The example is chosen to illustrate the analysis of suffixes.

[7]The maximal suffix length of 8 was found heuristically, and can be changed on demand.

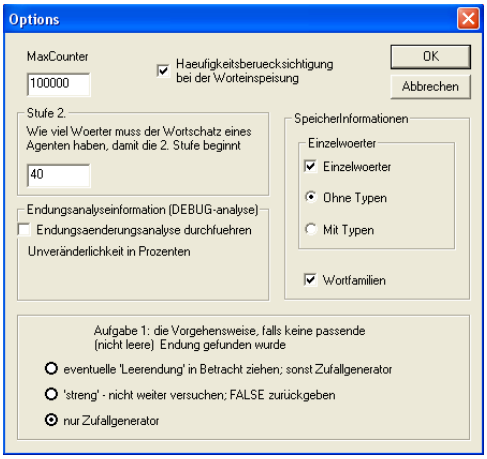

Figure 2. The options-dialogue that allows to vary the parameters: number of iterations, frequency of words uttered to the agents, the amount of feedback, output options etc.

For each word class c every word from the lexicon is analyzed and the relevant suffixes are filtered out by the following algorithm.

i. For each word all suffixes of the maximal length of 8 are extracted (e.g., for the word *apple:* $e > le > ple, ...$). The occurrences of all suffixes in all other words from the lexicon are counted. Suffixes with a frequency of more than 20% are stored in the $group\_list_w$.
ii. If two suffixes in the resulting $group\_list_w$ have similar frequencies (with respect to a similarity threshold), and one of them contains the other, the shorter is removed from the list (e.g., if *'isch'* and *'ch'* have the same frequency, then *'ch'* is removed).
iii. The $group\_list_w$ is added to the $list\_from\_all\_words$ which contains all suffixes from the other words, and the procedure of step 2. is repeated using a more sensitive similarity threshold.
iv. For each suffix s a list is constructed that contains all suffixes that include this suffix (e.g., the list of $s =$ *'ch'* might contain *'ich', 'ach', etc.*) and which contains no suffixes that have a common part larger than s (e.g. *'klich'* and *'rlich'* are removed from the list of *'ch'*, since they are already in the list of *'lich'*). All suffixes are ranked according to the number of different suffixes in their lists.

The highest number of different suffixes in this list gives the best representative suffix for the word class c. The most productive suffixes are those that are included in many different combinations. The filtering steps ensure that redundant suffixes are removed so that they do not influence the final result.[8]

C. Game Procedure

The overall game scenario (Fig. 3) represents communicative acts among adult-child and child-child. The adult speaks in turn to each of the children. He randomly selects a word

[8]See Fig. 6 in the Appendix for the complete algorithm.

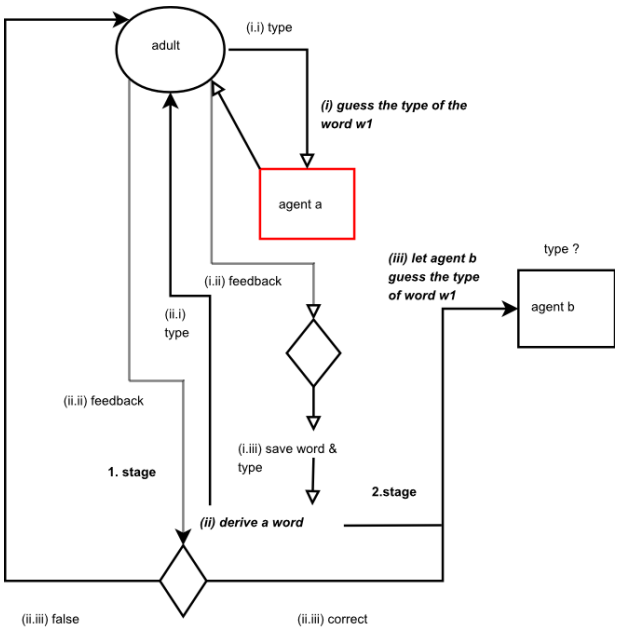

Figure 3. Overview Game Procedure.

from his derivation network (1. stage) or from his lexicon (2. stage) and asks a child to guess the word class of this word. Optionally the frequency of the word is considered, that is, words with a higher frequency are more likely to be selected (see the option "`Haeufigkeitsberuecksichtigung bei der Worteinspeisung`" in Fig. 2). Varying this parameter allows for testing the role of word frequency in correlation with the emergence of productive derivation rules according to the *dual route model* (see, e.g., [9]).

A single game run comprises a predefined number of iterations (see the option "`MaxCounter`" in Fig. 2). Iterations are slightly different according to the current stage (see below) of the game. Each child agent has to solve several communicative tasks during the iteration, and it gets scores for correctly guessing or producing words (see Fig. 3). In the following, we will refer to the single steps (i-iii) (see Fig. 3) to describe the stages of the game.

1. Stage: A game round in this stage can be shortly summarized as follows:

1) A randomly selected receiver (child) has to guess the word class of the word produced by the adult by using its derivation network (i).
2) If 1. was successful, the child derives a new word from another word that is present in its lexicon to a randomly chosen word class, e.g., shine > shiny (ii). The word formation is checked by the adult (ii.i-ii.ii).
3) If 2. was successful, the child utters the word acquired in 1. to its neighbor (iii).

2. Stage: After the predefined number of words in the derivation network (option "`Stufe 2`" in Fig. 2) of one of the agents is reached, all the agents enter the second stage. In this stage children perceive the words from the adult's lexicon. The lexicon contains more words than the derivation network, thus, for some words the possible derivations may be unknown even to the adult. A usual round in this stage can be described as follows:

1) The child has to guess the word class of the word received from the adult's lexicon **or** derivation network (i). The correctness of the answer is no longer controlled.
2) The child derives a new word from an existing word from its lexicon to a randomly chosen word class. Again, the answer is not validated by the adult (ii).
3) The child utters the word from 1) to its neighbor (iii).

Note: the success of the guess is validated according to the speaker's knowledge about the word class. That is, if in 1. the speaker classified "*funny*" as a noun, and now it utters it to the next agent, both will get a score, if the other recognizes *funny* as a noun too. Multiple forms, like the English *ease* as noun or verb, can be learned and differentiated by the agents.

In sum, in the second stage it is not obvious that children just inherit the knowledge learned from the adult. Rather, the communicative dynamics can force them to diverge from the learned behavior. In our experiments we test how much knowledge is needed to produce the correct derivations in a particular language in the second stage.

Word Class	English 49	German 49	German 421
NN	er	ung	ung
	tion	er	er
	le	en	en
VV	ss	en	en
	ry	ieren	ieren
	er	gen	ten
AA	al	er	bar
	tive	lich	lich
	ity	ig	ig

Table I
BEST SUFFIXES INDUCED VIA DECOMPOSITION FOR THE DERIVATION NETWORKS OF THE SIZE: 49 FOR ENGLISH, 49 FOR GERMAN AND 421 FOR GERMAN. THE CORRECT SUFFIXES ARE UNDERLINED.

IV. EXPERIMENTATION

We test the system with respect to the following aspects: the decomposability of the language into stem and potential suffixes and the communicative success of the agents. The decomposability is evaluated by means of the *F-Score* that indicates the correctness of decomposed suffixes to suffixes of a particular language. The *F-Score* averages over *precision* and *recall*, that are applied in information retrieval to measure the goodness of a classification. Precision $\frac{\#\{correctly\ identified \cap identified\}}{\#\{identified\}} \in [0,1]$ relates the number of correctly identified suffixes to the total number of suffixes found by the system as representing a word class. Recall $\frac{\#\{correctly\ identified \cap identified\}}{\#\{correct\}} \in [0,1]$ relates the correctly identified suffixes to the total number of correct suffixes considered for a language. Then, the *F-Score* can be calculated as follows: $\frac{2}{\frac{1}{recall_i}+\frac{1}{precision_i}} \in [0,1]$. An *F-Score* of 1 indicates a perfect match of real suffixes identified by the system, an *F-Score* of 0 shows a complete mismatch, respectively.

language	#deriv.net	F-Score	random baseline
German	49	0.77	0.66
English	49	0.55	0.66
German	421	0.88	0.66

Table II
SUFFIX F-SCORES - SUFFIXES IDENTIFIED BY THE ADULT ANALYZING THE WHOLE DATA VERSUS REAL SUFFIXES OF A LANGUAGE.

To evaluate the communicative success of the game, we calculate the suffix ratio over all agents after a game run. A random language, used here for comparison, has primarily no structure, i.e., no productive suffixes in contrast to English and German. We ask, whether for all the three languages productive rules emerge during communication.

Suffix Ratio

$$SR_{Ra1} = \frac{\sum_{a1 \neq a2, a1, a2 \in A} \sum_{c}^{C} \sum_{r=1}^{R} \sigma(s(a1, c, r), s(a2, c, r)) * \frac{1}{r}}{\sum_{r=1}^{R} \frac{1}{r} * C * N} \in [0, 1) \quad (1)$$

The suffix ratio SR_{Ra1} is computed with respect to a particular agent $a1$ (adult or child). Suffixes stored for each word class C are ranked by the rank order R according to their success in the game. The suffixes s on the rank r of an agent (adult or child) $a1$ are compared to the suffix of an agent (adult or child) $a2$ on the respective rank. $\sigma(s(a1, c, r), s(a2, c, r))$ is a binary predicate that takes the value of 1 when two suffixes are equal and 0 otherwise. N is the number of agents.

The rank r is used as a weighting factor giving to deviations on lower ranks a smaller value. This is because the first suffix is more likely used for production of new words than other. So, in our experiments we calculate SR_1 and consider SR_3 as a complementary information resource, i.e., we evaluate the number of successfully replicated suffixes on the first and on the first three ranks. In the denominator of Equation 1 the sum $\sum_{r=1}^{R} \frac{1}{r}$ gives the maximal value the sum $\sum_{r=1}^{R} \sigma(s(a1, c, r), s(a2, c, r)) * \frac{1}{r}$ can take when we assume that all suffixes coincide.

V. RESULTS AND DISCUSSION

Tab. I and the F-Scores in Tab. II show the three best suffixes for each word class identified by the system. The underlined suffixes are those that really exist in a language.[9] For German the size of the input derivation network (49 vs. 421) does not play a crucial role for the success of decomposition. The network of 421 word families enhances the list solely by one more suffix (F-Score 0.77 vs. 0.88). In English, only a small number of real suffixes is replicated. The F-Score of 0.55 is even below the baseline of randomly assigning suffixes to word

[9]The system does not make a distinction between derivational and inflectional suffixes. This is because the word forms uttered to the agents appear in their infinitive form, so the identified suffixes are those that help to transform one word class to another, or more specifically, into its base form. This explains, why the German '-en' suffix is identified although it is grammatically classified as an inflectional infinitive suffix. This simplification allows us to test the decomposition algorithm in a minimal setting, however, it can be easily extended to learn more sophisticated inflectional relations.

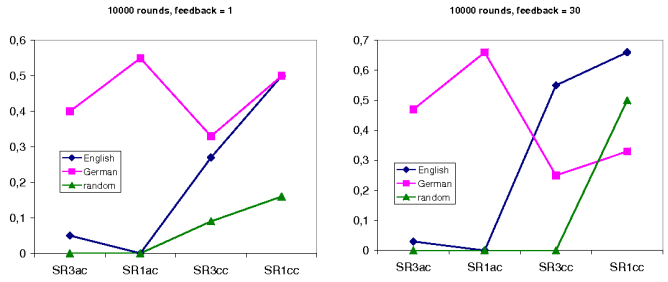

Figure 4. Suffix Ratio: SR_1, SR_3 were calculated for adult-children SR_{3ac}, SR_{1ac} and among children SR_{3cc}, SR_{1cc}.

classes. This fact indicates that German words contain more information about the word class than English using productive suffixes for word formation.

However, this fact for English does not mean that the communication among the agents fails. Looking at suffixes after a communication round of the game for English, German and a random word set we get the average suffix ratios (SR) displayed in Fig. 4. We run the game up to 10 times over 10000 iterations and varied the amount of feedback given by the adult from 1 to 30. We observe that German has a high agreement among adult and children for both variants of feedback. This allows us to conclude that the amount of feedback does not influence the induction of correct derivation suffixes significantly. Children show less agreement but they all agree with the adult (perhaps on different suffixes).

English exhibits almost no agreement between adult and children but a high agreement among children. This is an interesting finding which is probably attributed to the overall use of productive suffixes in English. Children decompose words using the same mechanism as the adult, and processing more words should normally lead to a convergence with the adult, if productive suffixes are present in the language. In the case of English, however, the resulting suffixes of the children are different from those of the adult which points to lower predictability of word classes based on suffixes. Thus, children learn the words as a whole, by the full word route and not that much via parsing. This leads to the overall success of the game, although, the derivation rules might be different from the rules of the adult.

All in all, a language in terms of an agreement on common suffixes emerges in all languages tested here. Even a completely random input produces a convergence of children's suffixes which are different from the random pseudo suffixes of the adult. Thus, structure emerges via communication, although, it can look differently from the structure induced by the adult.

VI. CONCLUSION

In this paper we presented a system modeling the learning of derivation morphology. We found that structural status of the input words influences the newly emergent language.

Two languages, English and German, were compared by the amount of productive suffixes used for word formation.

Although, substantial differences were observed, the evolutionary game dynamics let the agents converge on a common language. The communication success was observed in all cases, even for the random language. If the input language does not exhibit a clear structuring, children converge in the long run on common suffixes. However, the resulting language might be different from that of the adult.

The explicit learning modeled in the first stage does not play a crucial role for the emergence of a structured input. It rather can reenforce the children to learn correct word to word class relations. The structure of the lexical input determines to a large extent the persistence of the given language. If no structural regularities can be induced from the language of the adult, the language is less stable, and in most cases, children have to negotiate other rules facilitating the communication.

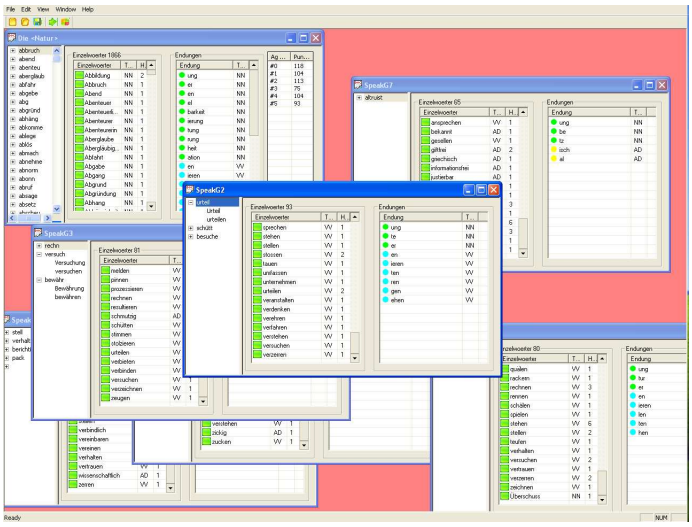

Figure 5. The Figure shows the system at the beginning of the game. The left topmost window represents the adult, the other windows represent children. The small amount of common suffixes (third column) and the very few word families (first column) show that a common language has not been developed yet.

REFERENCES

[1] M. Tomasello, *Constructing a Language : A Usage-Based Theory of Language Acquisition*. Harvard University Press, March 2005.

[2] J. L. Bybee, *Morphology as lexical organization*. Academic Press, 1988, ch. 7, pp. 119–141.

[3] J. M. Smith, *Evolution and the Theory of Games*. Cambridge University Press, 1982.

[4] N. Chomsky, *Aspects of the Theory of Syntax*. MIT Press, 1965.

[5] P. D. Ford and T. Voegtlin, "Learning word meaning and grammatical constructions from narrated video events," in *Proceedings of the HLT-NAACL 2003 workshop on Learning word meaning from non-linguistic data*. Morristown, NJ, USA: Association for Computational Linguistics, 2003, pp. 38–45.

[6] M. Taft and K. Forster, "Lexical storage and retrieval of prefixed words," *Journal of Verbal Learning and Verbal Behavior*, pp. 638–647, 2006.

[7] A. Domínguez, O. Cuetos, and J. Segui, "Morphological processing in word recognition: a review with particular reference to spanish data," *Psicológica*, vol. 21, p. 375401, 2000.

[8] H. Clahsen, I. Sonnenstuhl, and J. P. Blevins, "Derivational morphology in the german mental lexicon: a dual mechanism account," in *In: H. Baayen & R. Schreuder (Eds.), Morphological structure in language processing, Mouton de Gruyter*. 2006, 2003, pp. 125–155.

[9] H. Baayen, "Quantitative aspects of morphological productivity," *Yearbook of Morphology 1991*, pp. 109–150, 1992.

[10] P. Vogt, "The emergence of compositional structures in perceptually grounded language games," *Artificial Intelligence*, vol. 167, no. 1-2, pp. 206–242, September 2005.

[11] C. Römer and B. Matzke, *Lexikologie des Deutschen*, 2nd ed. Gunter Narr Verlag, 2005.

[12] Z. Harris, "Morpheme boundaries within words: Report on a computer test," in *Transformations and Discourse Analysis Papers 73*. Department of Linguistics, University of Pennsylvania, 1967.

[13] J. Goldsmith, "Unsupervised learning of the morphology of a natural language." *Computational Linguistics*, vol. 27, no. 2, pp. 153–198, 2001.

[14] E. Ukkonen, "On-line construction of suffix trees," *Algorithmica*, vol. 14, no. 3, pp. 249–260, 1995.

APPENDIX

for all word classes c **do**
  for all word $\langle w, c\rangle$ **do**
    { 1. filtering step}
    for all $s \in w$ {suffix} **do**
      $N(s)$ = number of words of class c containing s
      if $N(s) \in L(c) \geq 20\%$ of $|L(c)|$ **then**
        $group\_list_w$.push_back(s)
      end if
    end for
    { 2. filtering step}
    sim_val = 0.1
    for all $group\_list_w$ **do**
      $s_1, s_2 \in group\_list_w$, $s_k \neq s_l$, $s_1 \in s_2$,
      $Sim(s_1, s_2)$ {compares the frequencies of s_1 and s_2 in $L(c)$}
      if $Sim(s_k,s_l) \leq$ sim_val **then**
        delete s_1 {'ich', 'ch' $\rightarrow$ the shorter, i.e., 'ch', is removed}
      end if
    end for
    {3. filtering step}
    sim_val = 0.0000001
    for all $group\_list_w$ **do**
      $list\_from\_all\_words$.push_back(group_list$_w$)
    end for
    for all $s_1, s_2 \in list\_from\_all\_words$, $s_1 \neq s_2$, $s_1 \in s_2$, **do**
      $Sim(s_1, s_2)$
      if $Sim(s_1,s_2) \leq$ sim_val **then**
        delete s_1 {'ich', 'ch' $\rightarrow$ 'ch' is removed}
      end if
    end for{4. filtering step} {create suffix trees for each s_1}
    for all $s_1, s_2 \in list\_from\_all\_words$, $s_1 \neq s_2$ **do**
      if $s_1 \in s_2$ **then**
        suffixtree(s_1).add(s_2)
      end if
    end for
    for all $s \in list\_from\_all\_words$ **do**
      for all $s_1, s_2 \in$ suffixtree (s), $s_1 \neq s_2$ **do**
        if $(s_1 \cap s_2) \rightarrow s$ **then**
          mark(s_1)
          mark(s_2)
        end if
      end for
    end for
    for all $s \in list\_from\_all\_words$ **do**
      $Pr(s) = |A(s) \setminus M(s)|$
      $\{A(s)$: suffix tree of s, $M(s)$: marked suffixes of $s\}$
    end for
  end for
end for

Figure 6. Suffix induction from single words in four filtering steps. For each word and each word class: 1) collect all suffixes of a word, 2) filter out suffixes of a similar frequency (according to a similarity threshold) and remove the shorter of the mutually inclusive suffixes. 3) do the same as in 2) for all suffixes of all words with a reduced similarity threshold. 4) construct a suffix tree for each suffix retained after filtering 1-3. Remove all marked suffixes from each suffix tree that contain a larger common part than the considered suffix, i.e., remove 'lich' and 'klich' from the tree of 'ch', since they are already present in 'lich'. $Pr(s)$ is the number the different suffixes, the suffix s is present in. Rank all suffixes according to their $Pr(s)$ values.