An informal discovery procedure for two-level rules

Kimmo Koskenniemi
Department of Modern Languages, University of Helsinki, Finland

ABSTRACT

The paper shows how a certain kind of underlying representations (or deep forms) of words can be constructed in a straightforward manner through aligning the surface forms of the morphs of the word forms. The inventory of morphophonemes follows directly from this alignment. Furthermore, the two-level rules which govern the different realisations of such morphophonemes follow fairly directly from the previous steps. The alignment and rules are based upon an approximate general metric among phonemes, e.g., articulatory features, that determines which alternations are likely or possible. This enables us to summarise contexts for the different realisations.

Keywords: morphological analysis, discovery procedure, two-level rules, two-level morphology, IPA

1 INTRODUCTION

The orientation of this paper is linguistic rather than statistical, and the general framework is not taken from machine learning. The aim of the procedure that this work details is to assist rather than to replace the linguist. The scheme makes use of the common knowledge that human linguists have. The procedure is intended to make a part of such knowledge operational. In order to use the procedure, the linguist must select examples which contain only regular (morpho)phonological alternations. The alternations must be of the types for which the procedure has general models, e.g., assimilations, agreements or phonotactic constraints. A good choice of examples is essential for getting good and general rules.

Journal of Language Modelling Vol 1, No 1 (2013), pp. 155–188

Linguists have a special talent to cope with *regularities* and *exceptions*. A human linguist is able to consider factors beyond plain frequencies and, e.g., recognise the fact that certain morphophonological phenomena are closed, and no new words will follow them, and that some others are productive and readily extend to new words. Using such knowledge, the linguist can sometimes make sense of phenomena which might remain fuzzy for present machine learning and statistical methods.

The aims of the procedure sketched here are (1) to partition a set of example word forms into stems and inflectional morphemes by aligning (character by character) the stems and inflectional morphs, (2) to establish morphophonemic representations for the stems and affixes, and (3) to deduce a set of two-level rules which express the general context conditions (according to which the morphophonemes are realised in the examples and in any similar words). Through such steps, lexicon representations and rules for an inflectional class can be established. Applying the procedure to all productive inflectional classes is needed in order to describe the morphology of a language.

General (but approximate) linguistic knowledge about phonology guides the mechanical procedure presented in this paper. In particular, knowledge of the kinds of phonological alternations that are common in the languages of the world, and the kinds of phonological contexts that such alternations typically occur in, are used. Possible alternations can be, e.g., *assimilations* (where adjacent sounds become more similar to each other), *dissimilations* (where similar sounds become more distinct from each other), *metathesis* (where two sounds are swapped), *phonotactic constraints* (where, e.g., a certain type of syllable structure is enforced), agreements or *harmonies* (where, e.g., some vowels in the affixes become more similar to those in the word root).

Two-level morphology is used here because, in that framework, individual rules can be kept quite *independent* of each other.[1] The original form of the two-level morphology (Koskenniemi, 1983) is mathematically simple because each rule written by the linguist is a con-

[1] See, e.g., Karttunen (1993) for an overall introduction to two-level morphology.

straint which must be satisfied separately. On the other hand, this simplicity makes it more difficult to simulate generative phonology using two-level rules in cases where some phonemes can be affected by several unrelated rules.

A more complex form of the two-level formalism which uses rule conflict resolution mechanisms is more suitable for using similar underlying representations as the generative phonology. Rule conflicts can be resolved by pre-processing the individual rules using the whole grammar and by copying parts of rules into some other rules which makes the rules more complex, see Karttunen *et al.* (1987) for that approach.

The present approach uses *reduced versions* of the two-level grammars which neither use any conflict resolution mechanisms nor need them. In two-level grammars, right-arrow conflicts only arise when the same correspondence occurs in separate rules. This does not happen in the proposed procedure which produces a single rule for each realisation of a morphophoneme (even if such a rule may have several context parts). Left-arrow conflicts arise when different realisations of a morphophoneme have overlapping contexts. The conflict resolution mechanism of two-level compilers recognises the special case where one context is a proper subset of another, and resolves it by prioritising the smaller context. The grammars produced by the proposed procedure avoid left-arrow conflicts because the corresponding morphophonemes will be distinct from each other.

Rewrite rules must usually be applied in a specific order. Applying a rule changes the string, and the remaining rules depend on the earlier ones. The number of different orderings grows according to the factorial of the number of rules, e.g., five rules could be applied in 120 distinct orders but 20 rules in as many as 2,432,902,008,176,640,000. Phonological grammars using rewrite rules need a rule ordering, even if not all rules are equally sensitive to such. Discovering complex rewrite rule grammars is probably difficult because one must discover both the rules and their ordering. The reduced two-level grammars, on the other hand, avoid the rule ordering altogether because all rules are applied in parallel.

The procedure presented here treats the underlying forms of morphemes and the morphophonemes in an extremely concrete manner. *Morphophonemes* are represented just as the *combinations of the corre-*

sponding letters (or phonemes) which we can observe in the surface forms. On the one hand, such an interpretation of morphophonemes is crude, but on the other hand, it is a fact that anybody can observe. *Lexical representations* are sequences of *lexical characters*, each of which is either a letter or a morphophoneme.

Using a systematically selected set of surface forms, the procedure creates lexicon entries and rules for such words. The entries contain morphophonemes deduced by the procedure. The deduced rules are simple two-level rules. The resulting rules can be compiled using the existing two-level compilers, such as the open source HFST-TWOLC (Silfverberg and Lindén, 2009) or the proprietary Xerox LEXC.[2] This paper describes the plan for the actual procedure, including some feasibility estimates according to which the approach appears to be tractable. Unfortunately, an implementation is not yet available (but cf. Section 14).

This work differs from unsupervised discovery of morphology where the lexical units and rules would be induced from raw corpus data. The approach is based on the observation that many linguists find it difficult to express their intuitions as formal rules, whereas they are comfortable with providing concrete examples. This work makes use of this kind of human supervision. When implemented, the procedure would be a useful tool for a linguist. The informal procedure as discussed below, may even guide the linguist in designing rules and grammars even in the absence of an implementation. Theoretically oriented linguists might also discuss the merits and the shortcomings of the very concrete and objective interpretation for morphophonemes presented here.

This paper argues for the utility of phoneme by phoneme (or character by character) alignment of word forms and discusses its tractability. In particular, it presents a way how both the stem parts and the affix parts should and can be aligned. The alignment appears to be possible for a wide array of different types of languages. Such an alignment can be directly utilised in the two-level framework (but less so in the rewriting framework). Once a proper alignment is determined,

[2] See `www.stanford.edu/~laurik/.book2software/twolc.pdf` and `http://www.cis.upenn.edu/~cis639/docs/twolc.html` for the documentation of the two-level grammar formalism.

the lexical representations of stems and affixes are algorithmically and uniquely determined.

It is argued that, in this framework, the deduction of the necessary morphophonological two-level rules is easier than using the rewrite rule framework. In particular, it is shown that the *deduction of rules for one morphophoneme is entirely independent* of the rules for other morphophonemes. In principle, one may discover rules for various kinds of morphophonemic alternations provided that the alternations are (morpho)phonological in their nature, and do not involve suppletion or otherwise introduce, delete or change larger units. Specific procedures for each type of conditioning are needed, but can be designed.

The main claim of this paper is that the framework presented has a better potential to succeed in discovering regularities in languages which have an elaborate morphophonology. Other approaches appear to perform best when applied to English or other languages with fairly simple morphology. In particular, as compared to the machine learning oriented approaches, the framework proposed here appears to be able to cope with more demanding phenomena such as interdigitation of Semitic languages, phenomena based on syllable structures, agreements, metathesis of non-contiguous segments, and the like. This paper argues for the validity of this claim. Deeper understanding can be reached when an implementation is available and has been applied to a number of different languages.

2 PAST WORK

Most of the recent work on the theme of morpho(phono)logical discovery methods has been done within the framework of unsupervised machine learning. These start from large unannotated corpora and other raw language resources and they try to describe the inflection of words in an economic manner without human intervention. The present paper operates differently, and it aims to discover more precise rules than what the unsupervised approaches can.

Gildea and Jurafsky (1995) report experiments with discovering finite-state transducers from large sets of examples. They extend and improve the OSTIA algorithm (Oncina *et al.*, 1993) by aligning the examples using an edit distance based on the phonological binary features of phonemes and use training data of tens of thousands of words in the general framework of Mitchell (1982).

Theron and Cloete (1997) studied a problem where the starting point was a list of pairs consisting of an inflected word form and the corresponding base form. Their work was based on edit distances when aligning the inflected and the base form with each other. Their algorithm deduced the context parts of the two-level rules by starting from the full lists of correct contexts and then by truncating them as long as the rules still accounted for the correct forms.

Yarowsky and Wicentowski (2000) present a method for deducing stem alternation rules and morphological analysis applicable for languages similar to the Indo-European languages which are nowadays spoken in Europe. The method makes use of initial tables of endings, an unannotated corpus and a collection of candidate noun, adjective and verb roots. Roots of word forms and rules are deduced using frequency statistics. The method of identifying roots is trained and combines several models. Most of the irregular English words were learnt by the procedure.

Linguistica algorithm (Goldsmith, 2006) and Morfessor (Creutz and Lagus, 2004) represent word forms using sets of substrings and they utilise criteria such as the minimum description length (MDL). The end results of such processes are sets of strings which are similar to linguistic morphs (but not always the same). Concatenations of such sets model the word forms of the language. Goldwater and Johnson (2004) build on top of Goldsmiths Linguistica and reduce the sets further by introducing phonological rules.

The PhD dissertation of Chan (2008) on the induction of morphology and lexical categories includes a fairly comprehensive survey on previous work on machine learning of morphology. A few studies have a more linguistic conception of the phonological rules to be found. They usually assume that the procedure has access both to the underlying and the surface form of example words.

Johnson (1984) presented a discovery procedure for ordered rewrite rules in the framework of generative phonology. He starts from a given table of forms of a set of lexemes and assumes that the morphemes have been segmented, corresponding phonemes identified, and that the phonemes are represented using their distinctive features. Johnson claimed that his procedure can cope with the rule ordering by considering the contexts of the rules. The last rules in the cascade have contexts which are apparent in the surface forms, whereas the rules early in the sequence tend to use contexts which are less visible in the surface forms. The data on which the method was tested involved six rules for Japanese. He notes that the rules, underlying representations, and the rule orderings are not strongly determined by the data. Lots of computation was required and it resulted in multiple solutions that consisted of different rule orderings and different underlying representations. The goals of Johnson were otherwise similar to those of the present paper.

Touretzky *et al.* (1990) study rules needed in order to generate phonetic realisations out of underlying (morpho)phonemic representations. The rules are learnt step by step from input examples where both the underlying and surface forms are given. The learning occurs by approximating the contexts both from specific examples (which may be too narrow) and generalisations (which may be too broad) as context conditions.

Oflazer and Nirenburg (1999) and Oflazer *et al.* (2001) present a method for bootstrapping morphological analysers by combining human elicitation and machine learning. Human informants provide the examples used by the machine learning process to deduce rewrite rules necessary for accounting for the data. The method has been applied, e.g., to Polish, where alternations both in the stems and in the endings were captured. This interactive approach is relevant for the procedure presented in this paper, and maybe the best parts of these two could be combined in future.

A recent work by Hulden *et al.* (2011) studies the learning of dialectal morphologies using parallel corpora. The authors end up using parallel rewrite rules for describing the mapping between the standard Basque and one of its dialects. They use the FOMA rewrite rule system (Hulden, 2009) which is an open source replacement for Xerox XFST (Beesley and Karttunen, 2003).[3]

The present paper elaborates an earlier work, see Koskenniemi (1991) where a version of the procedure was sketched. The general

[3] Both FOMA and XFST support certain forms of parallel rewrite rules which avoid the rule ordering problems. On the other hand, such parallel replace rules have to be compiled together as a single unit which may become computationally heavy if there are many rules.

idea of first aligning and then deducing was presented there. The present paper formalises the alignment in order to make the potentially intractable task feasible. In addition, a distance metric is presented so that choosing among alternative solutions is better defined. Furthermore, this paper tells more explicitly how languages with infixation, prefixation, etc., can be handled.

3 SIMPLIFIED TWO-LEVEL MORPHOPHONOLOGY

Koskenniemi (1983, 1984) detail the origins of the two-level morphology, Karttunen (1993) presents a brief introduction, and Karttunen and Beesley (2001) presents a history of the two-level formalism.

In all versions of the two-level formalism, the *morphophonological rules* are based on just two representations of word forms: the *lexical representation* and the *surface representation*. There are no intermediate representations between these two. The lexical representation is a string of characters and serves as the underlying representation of *morphemes*.

In the present simplified formalism, the lexical characters of the morphemes may be:

- *phonemes* (or letters) which, in our simplified formalism, always correspond to themselves in the surface representation;
- *morphophonemes* which correspond to two or more alternative phonemes (or letters) in the surface representation; and
- *auxiliary symbols* which always correspond to zero in the surface representation and may be used either as boundary symbols between morphemes or as markers of grammatical categories.

We denote morphophonemes by the alternative letters they represent, e.g., **aä** represents a morphophoneme which can be realised either as **a** or **ä**.[4] Lexical representations are given here with spaces between the characters in order to make the morphophonemes explicit, e.g., **t a l o s s aä**. Surface representations are strings of letters and zeroes (**0**).

[4] Mathematically, these kinds of morphophonemes are tuples rather than sets. In the general case, the morphophoneme is a sequence of the surface letters that the morphophoneme is realised as. Thus, a morphophoneme for an **e** in the base form and alternating with an **i** in its inflected forms is different from an **i** in the base form alternating with an **e**. The shorter notation is used simply for brevity.

Zeroes act as place holders for morphophonemes which are deleted from surface forms. Within the two-level rules, zeroes are, however, treated as characters (rather than epsilons or null strings).

Let us consider a simplified example taken from the Finnish inflection of nouns. Word forms are constructed by affixing endings after the stem, e.g., **piha a** 'yard' + 'partitive', **piho i lla** 'yard' + 'plural' + 'at' and **piho j a** 'yard' + 'plural' + 'partitive'. A linguist would notice that the stem-final vowel is either **a** or **o** in the surface forms, and therefore we interpret it as a morphophoneme **ao** in the lexical form. Similarly, we notice that the plural morph is either **i** or **j** on the surface. Thus, we have the lexical representation **p i h ao** for 'yard' and **ij** for 'plural'. Now we can represent the correspondence of the (somewhat simplified) lexical and surface forms (where **ao** and **ij** are single and indivisible symbols):

```
p i h ao a    p i h ao ij l l a     p i h ao ij a
p i h a  a    p i h o  i  l l a     p i h o  j  a
```

Two-level rules specify how lexical and surface representations may correspond to each other. We need one rule for **ao**, and another for **ij**. Studying these and other examples, the linguist would notice that **ao:o** occurs before the plural morpheme **ij** and that the plural morpheme itself realises as **ij:j** if and only if it ends up between vowels on the surface. A linguist might write the two-level grammar as:

```
Alphabet a d e f g h i j k l m n o p r s t u v y ä ö
         ao:a ij:i ;
Vowel = a e i o u y ä ö ;
Rules
"ao in plural"      ao:o <=> _ ij: ;
"ij between vowels" ij:j <=> :Vowel _ :Vowel ;
```

The alphabet lists all letters and morphophonemes. In addition, it gives the default correspondences of the morphophonemes, here **ao:a** and **ij:i**. The first rule tells two things: first (the right arrow component), that **ao:o** can occur only if it is immediately followed by a lexical **ij**, and secondly (the left arrow component), that when a lexical **ao** occurs in this context, **o** is the only possible corresponding surface character. Similarly, the second rule says that **ij:j** may occur only between surface vowels and that in such a place **ij:i** is forbidden. The pair before the double arrow is called the *centre* of the rule. The *context part* is (or possibly several of such are) on the right of the double arrow. A context part consists of the *left context* and a *right context* separated from each other by an underscore. Two-level grammars usually consist of double arrow <=> rules which combine the requirements of right arrow => and left arrow <= rules.

Two-level rules are usually compiled into finite-state transducers (FSTs). A compiled rule FST accepts all correct examples, i.e. strings of character pairs like: **p:p i:i h:h ao:a** or **p:p i:i h:h ao:o ij:i l:l l:l a:a**. According to a common practice, we represent pairs with identical components with a single character: **p i h ao:a** and **p i h ao:o ij:i l l a**.

The linguist collects a set of examples before even starting to design any rules. The examples specify the task for the rules. This specification is in a form that can be communicated to and understood by other linguists. Furthermore, the generated two-level rules can be checked against these examples in order to verify their correctness – as described in Section 13.

The rules the linguist writes depend on the lexical representations one chooses. The experience of the author, when supervising linguists and students, indicates that when the lexical representations have been carefully designed, the writing of the two-level rules is easy and straightforward. Using a sufficiently large set of morphophonemes, one can keep the rules simple and independent of each other.

4 CHOOSING A PARADIGM OF SELECTED LEXEMES AND FORMS

For the discovery procedure, we collect a table of word forms, i.e. surface forms of inflected words. On each row of the table, we have the same lexeme (in different forms) and on each column, we have the same form (of different lexemes). To be more precise, on a row, there are (possibly slightly different) stems of the same lexeme. One must not include lexemes with *suppletion*, i.e. words where stems represent different lexical units, such as **good** and **bett(-er)**. It is not reasonable even to attempt to model suppletion using two-level rules (whereas rewrite rules can freely be used for such). Two-level rules should be used only for *natural* morphophonological alternations.

In order to describe the steps of the discovery procedure, we use an example given in Table 1, where six Finnish nouns are inflected in five different forms: singular nominative, partitive, and inessive, and plural partitive and inessive (the columns).

	SgNom	SgPtv	SgIne	PlPtv	PlIne
‘house’	talo	taloa	talossa	taloja	taloissa
‘crack’	särö	säröä	särössä	säröjä	säröissä
‘cross’	risti	ristiä	ristissä	ristejä	risteissä
‘notch’	lovi	lovea	lovessa	lovia	lovissa
‘fish’	kala	kalaa	kalassa	kaloja	kaloissa
‘dog’	koira	koiraa	koirassa	koiria	koirissa

Table 1: Example raw word forms

5 LENGTHS OF THE STEMS AND AFFIXES

Alignment consists of augmenting the word forms with zeroes where necessary, and inserting boundary symbols which separate the stems from the affixes. Zeroes are sometimes needed in order to make the surface forms of the corresponding stems and affixes equal in length. After this step, the stems which belong to the same lexeme should have the same number of characters (letters plus possible zeroes), and the same goes for affixes which belong to the same grammatical form.

After adding some zeroes, the lengths of the word forms in the table can be expressed as a sum of the length of the (constant length) stems of the lexeme m_i and the (constant length) affix parts of the grammatical form n_j. Initially, the procedure has the lengths of the raw forms as given in Table 2.

Next, the procedure decomposes the raw lengths of the word forms into a sum of the lengths of the stem and affixes. It may arrive at several solutions and one of these should outperform the other ones during the subsequent steps. The procedure tentatively assumes that the stems are (at least) as long as the singular nominative forms. Then, the affixes are at least as long as the difference between the inflected and nominative forms. In this way, the procedure arrives at the decomposition of lengths in Table 3. The decomposition is accurate in all other places except for the plural forms of **lovi** and **koira** (which

Table 2: Lengths of the raw word forms

	SgNom	SgPtv	SgIne	PlPtv	PlIne
talo	4	5	7	6	8
särö	4	5	7	6	8
risti	5	6	8	7	9
lovi	4	5	7	5	7
kala	4	5	7	6	8
koira	5	6	8	6	8

Table 3: Lengths of the raw word forms tentatively decomposed into the lengths of the stems and affixes

stem		SgNom	SgPtv	SgIne	PlPtv	PlIne
		0	1	3	2	4
talo	4	4+0=4	4+1=5	4+3=7	4+2=6	4+4=8
särö	4	4+0=4	4+1=5	4+3=7	4+2=6	4+4=8
risti	5	5+0=5	5+1=6	5+3=8	5+2=7	5+4=9
lovi	4	4+0=4	4+1=5	4+3=7	4+2=6>5	4+4=8>7
kala	4	4+0=4	4+1=5	4+3=7	4+2=6	4+4=8
koira	5	5+0=5	5+1=6	5+3=8	5+2=7>6	5+4=9>8

are one character too short). Thus, we insert a zero character into those word forms. The zero could be inserted anywhere in the word form and the procedure must usually evaluate several possibilities.

Furthermore, one must allow for more zeroes than the minimum amount to be added if there are more substantial (but regular) alternations within the stem. The calculation gives one or more hypotheses for the lengths of stems and affixes. The procedure proceeds first with the above assumption for the lengths and backtracks only if necessary.

The procedure must also be prepared to consider alternative partitions. One could, e.g., have shorter stems and longer affixes. Not too many alternatives exist, and the procedure can enumerate the decompositions of the lengths without problems and choose the best (or the only possible) alternative during the next steps of the procedure.

Up to now, the procedure has made no assumptions about the position of the stems. They might be at the beginning, end, or somewhere in the middle of the word forms. The stem might even be noncontiguous, i.e. interrupted by inflectional parts. One could claim that establishing the (tentative) lengths of morphemes as the first step is not necessary. It could be solved later, or as a part of the whole task using, e.g., dynamic programming. Such computation, however, appears to be more complex and less disciplined. Establishing the lengths first makes the following steps more tractable.

6 POSITIONS OF THE ZEROES AND PARTITIONING THE WORD FORMS

In the previous step, the procedure made an educated guess about the desired lengths of the stems and affix parts. If the guess produces poor solutions, then the procedure backtracks and modifies the guess. For the time being, however, the procedure sticks to the assumption made in the previous section and adds some zeroes as necessary so that the lengths of the word forms meet the lengths required by the partition in the Table 3. The procedure adds the required amount of zeroes in all different permutations. Thus, from now on, the procedure has full tables of our example words (with zeroes). Each table conforms with the lengths but has the zeroes in random positions of the word forms. There may be many such tables and it is not practical to enumerate them before filtering out the clearly impossible ones.

A human linguist would perhaps immediately see the positions where the zeroes are best added, e.g., **koir0issa**, because in this way the letters in the first four positions of the stem would be identical in all stems of the lexeme. The linguist would exclude other positions for the zero because they would lead to an unnatural correspondence of letters. However, a computer procedure can manage with many possible versions where a correct number of zeroes are added but perhaps not in the correct places.

The partitioning of the word forms into stems and affixes is done by adding a fixed number of boundary symbols (+) into the word forms. If we have only suffixes (as in our example) or only prefixes, one boundary will be sufficient. If we have both prefixes and suffixes, two boundary symbols are needed. More than two may be needed for Semitic languages with both prefixes and suffixes and even interdigitation (where vowel affixes are inserted inside the word root). In fact, the interdigitation does not cause any additional problems for the two-level or rewrite rules cf., e.g., Kataja and Koskenniemi (1988). The problem with interdigitation is how to build the lexical or underly-

Table 4: Example word forms with zeroes and boundaries in bad positions

SgNom	SgPtv	SgIne	PlPtv	PlIne
talo +	talo + a	talo + ssa	talo + ja	talo + issa
särö +	särö + ä	särö + ssä	särö + jä	särö + issä
risti +	risti + ä	risti + ssä	riste + jä	riste + issä
lov**i** +	lov**e** + a	lov**e** + ssa	lov**i** + a0	lo0**v** + issa
kala +	kala + a	kala + ssa	kalo + ja	kalo + issa
k**o**ira +	k**o**ira + a	k**o**ira + ssa	0**k**oir + ia	0**k**oir + issa

ing representation out of morpheme-like elements. That is not, however, within the scope of rule discovery. The added boundary symbols split the word form into even- and odd-numbered segments. The stem of the lexeme consists of either the odd-numbered or even-numbered segments. The remaining segments represent the inflectional affixes which belong to the grammatical form.

On the basis of the calculation of the lengths in the previous steps (and knowing at which end the affixes are located), we can insert the boundary symbols at uniquely determined positions in the word forms which have been augmented with zeros as necessary. This is now done for all alternative tables. If our assumption on the positions of the affixes is wrong, then the following steps will produce poor results, and we must backtrack and revise our assumption.

In our example, the procedure adds exactly one boundary symbol to each of these four word forms and one of the alternative tables for our example could now look as shown in Table 4. There, some corresponding letters are quite incompatible with each other, e.g., **i-e-e-i-v** in the fourth position of the stems for **lovi**, and **o-k** in the second position of the stems for **koira**. We consider vowels to be incompatible with consonants (except with semivowels) and *vice-versa*. We exclude all tables which violate this coarse constraint. Therefore Table 4 (and other tables containing equally poor correspondences) will be excluded.

Among the possibilities, the procedure also produces Table 5, where the characters in the corresponding positions of the same stems are reasonably congruent with each other. The same holds for characters in the corresponding positions of the suffixes. A human linguist would probably like this table best.

SgNom	SgPtv	SgIne	PlPtv	PlIne
talo +	talo + a	talo + ssa	talo + ja	talo + issa
särö +	särö + ä	särö + ssä	särö + jä	särö + issä
risti +	risti + ä	risti + ssä	riste + jä	riste + issä
lovi +	love + a	love + ssa	lov0 + ia	lov0 + issa
kala +	kala + a	kala + ssa	kalo + ja	kalo + issa
koira +	koira + a	koira + ssa	koir0 + ia	koir0 + issa

Table 5:
Example word forms with added zeroes and boundaries in good positions

The simple example we are studying would have only a few thousand different possibilities for adding the two required zeroes. The coarse checking of impossible tables would be no problem at all. With larger examples, some planning for the efficient exclusion of the impossible tables is needed. One option is to use a kind of branch-and-bound algorithm to prune the search space more efficiently. Adding a zero to a wrong place often causes an impossible correspondence so that one can exclude a whole class of tables before even creating them. Another alternative would be to represent the set of tables as finite-state networks which would remain quite reasonable in size and be straightforward to construct. Impossible paths (representing tables) would be excluded by a sequence of XFST or FOMA rules which would filter out strings (i.e. tables) by checking the compatibility of each character position. The PhD dissertation of Grzegorz Kondrak gives, among other things, a survey of the various methods which have been used in aligning words – see Kondrak (2002).

7 MORPHOPHONEMES AND THE REPRESENTATION OF MORPHEMES

Now that we have processed the initial matrix of word forms into a reasonably small set of tentative tables differing from each other in the positioning of zeroes, the next step of the procedure is to rank the remaining tables according to the morphophonemes that they imply. The different stems (for each lexeme) and affixes (for each grammatical form) in the Table 5 are now of equal length. Stems can be extracted and aligned as in the Table 6, where the bottom row indicates the morphophonemic representation that follows from the aligned

Table 6: Stems of the lexemes i.e. word forms with boundaries and zeroes but the affix part removed

t a l o	s ä r ö	r i s t i	l o v i	k a l a	k o i r a
t a l o	s ä r ö	r i s t i	l o v e	k a l a	k o i r a
t a l o	s ä r ö	r i s t i	l o v e	k a l a	k o i r a
t a l o	s ä r ö	r i s t e	l o v 0	k a l o	k o i r 0
t a l o	s ä r ö	r i s t e	l o v 0	k a l o	k o i r 0
t a l o	**s ä r ö**	**r i s t ie**	**l o v ie0**	**k a l ao**	**k o i r a0**

Table 7: Affixes of the grammatical forms

SgPtv	SgIne	PlPtv	PlIne
a	s s a	j a	i s s a
ä	s s ä	j ä	i s s ä
ä	s s ä	j ä	i s s ä
a	s s a	i a	i s s a
a	s s a	j a	i s s a
a	s s a	i a	i s s a
aä	**s s aä**	**ij aä**	**i s s aä**

stems. Most positions in the series of stems contain the same character. The vowels at the end alternate a bit in some stems. The relations between corresponding letters in the endings also look regular (see the Table 7 where we, again, have included in the last row the morphophonemic representations of the affixes).

The criterion for ranking the alternative tables is based on the quality of morphophonemes that each table implies. We denote the morphophonemes by indicating the characters they represent, e.g., **ie**, **ie0**, **ao**, **a0**, **aä** and **ij**.[5] According to common linguistic knowledge, similar phonemes are more likely to alternate with each other and radically different ones may not alternate with each other.

In Table 8, we see the (coarse) phonemic characterisations for the Finnish vowels according to the features used in the IPA (International Phonetic Alphabet). In addition to the named features, we have associated (somewhat *ad hoc*) numerical values with the features for the purposes of the discovery procedure. We can approximate vowels in the languages of the world according to the IPA using three

[5] Technically, e.g., **ie** stands for the tuple (i,i,i,e,e) and **ie0** for $(i,e,e,0,0)$, cf. footnote in Section 3.

Table 8:
Phonological features and numerical approximations of Finnish vowels

Letter	IPA	Height	Backness	Rounding
ä	/æ/	(near-)open 1	front 1	unrounded 0
a	/ɑ/	open 1	back 5	unrounded 0
e	/e/	close-mid 5	front 1	unrounded 0
ö	/ø/	close-mid 5	front 1	rounded 1
o	/o/	close-mid 5	back 5	rounded 1
i	/i/	close 7	front 1	unrounded 0
y	/y/	close 7	front 1	rounded 1
u	/u/	close 7	back 5	rounded 1
j	/j/	semivowel 9	front 1	unrounded 0

Table 9:
Distinctions in the Finnish vowel system

	Front		Back	
Height	unrounded	rounded	unrounded	rounded
Close	/i/ i	/y/ y		/u/ u
Close-mid	/e/ e	/ø/ ö		/o/ o
Open	/æ/ ä		/ɑ/ a	

digits (as in Table 8): one for the tongue height with a scale from 1 (low or open) to 7 (high or close), a second for the backness with a scale from 1 (front) to 5 (back), and third for rounding with 0 (unrounded) and 1 (rounded). The values represent just an ordinal scale, not any physical dimensions. A tongue height of 5, for instance, is higher than 1 but not necessarily five times as high.[6] Phonemes in most languages employ only a part of the possible heights and backness values.

There is no opposition between open and near-open vowels in Finnish, so the difference between them is ignored and the value 1 used for both. The Finnish vowel system is often represented as in Table 9. According to the Tables 8 and 9, **ie** makes a perfect morphophoneme for our purposes, as these two vowels differ by one feature only, the height of the tongue (and even only by one step).

[6] Furthermore, the front vowels differ in their backness: **/i/** and **/y/** are most front, **/e/** and **/ø/** a bit less front and **/æ/** even more to the back. These differences play no role in the present discussion. See *http://www.langsci.ucl.ac.uk/ipa/* for more information on the IPA alphabet.

Table 10: Phonological features of common Finnish consonants

Letter	IPA	Place		Manner	Voicing	
m	/m/	bilabial	1	nasal	voiced	1
p	/p/	bilabial	1	plosive	unvoiced	0
v	/v/	labiodental	2	fricative	voiced	1
t	/t/	alveolar	4	plosive	unvoiced	0
d	/d/	alveolar	4	plosive	voiced	1
s	/s/	alveolar	4	fricative	unvoiced	0
r	/r/	alveolar	4	trill	voiced	1
l	/l/	alveolar	4	lateral approximant	voiced	1
j	/j/	palatal	7	approximant	voiced (semivowel)	1
ng	/ŋ/	velar	8	nasal	voiced	1
k	/k/	velar	8	plosive	unvoiced	0
h	/h/	pharyngeal	10	fricative	unvoiced	0

Using this table, the components of **aä** differ only by one feature: **ä** being front and the **a** being back. In **ao** there are two minimally different values: **o** is rounded and one step more close than **a** which is unrounded. In Finnish, there is no back vowel more like **a** than **o**. Note that the semivowel **j** is given a characterisation both as a vowel (in Table 8) and as a consonant (in Table 10).

Consonants have more possible feature values than vowels. The place of articulation corresponds to the backness of vowels, but the different manners of articulation are less related to each other and do not form a continuum or an ordinal scale. Voicing is a binary feature and can be represented in the same way as the rounding of vowels. The features of some Finnish consonants are given in Table 10. The similarity between **i** and **j** requires a bit of linguistic knowledge: palatal consonants are pronounced roughly at the same place as front vowels, and that a semivowel is like a vowel but pronounced with some friction. The numerical values for the backness of vowels and the place of articulation for consonants seem to be on different scales. No vowels are articulated as front as some consonants. When comparing **j** with vowels, we may treat it as: tongue height 9 (more closed than any vowel), backness 1 (i.e. front), unrounded 0, as in Table 8.

Morpho- phoneme	Heights	Backnesses	Roundings	Penalty
aä	1,1	5,1	0,0	1
ij	7,9	1,1	0,0	1
ie	7,5	1,1	0,0	1
ie0	7,5,–	1,1,–	0,0,–	3
ao	1,5	5,5	0,1	2
a0	1,–	5,–	0,–	2

Table 11:
Penalties for differences of phonemes in morphophonemes implied by Table 5

SgNom	SgPtv	SgIne	PlPtv	PlIne
talo +	talo + a	talo + ssa	talo + ja	talo + issa
särö +	särö + ä	särö + ssä	särö + jä	särö + issä
risti +	risti + ä	risti + ssä	riste + jä	riste + issä
lovi +	love + a	love + ssa	lovi + 0a	lovi + 0ssa
kala +	kala + a	kala + ssa	kalo + ja	kalo + issa
koira +	koira + a	koira + ssa	koiri + 0a	koiri + 0ssa

Table 12:
Example word forms with added zeroes and boundaries in alternative (almost good) positions

Tables organised according to articulatory features reflect the closeness of phonemes. Phonological or morphophonological alternations typically modify just one or sometimes two features of a sound such as the voicing of a stop or the backness of a vowel. For the purposes of the discovery procedures, no perfect metric is required. A rough approximation will be sufficient if it is capable of excluding linguistically infeasible alternations.

In Table 11, we list the numerical characteristics of the vowels in the morphophonemes and use an ad hoc formula for computing a penalty. For the vowel height, we use four levels: (near-)open, close-mid, close, and semivowel. One-level difference in height, a different backness, or rounding counts as 1 each; a bigger difference in height, or if a zero belongs to the morphophoneme that corresponds to a deletion, then it counts as 2. The total penalty of the morphophonemes in Table 11 is 10.

Some other aligned tables may have survived when we filtered out the impossible alignments. In fact, a fairly good alternative would be the one given in Table 12. This one differs from our earlier good table by inserting the zeroes one position later than in our earlier good alter-

Table 13: Penalties for differences of phonemes in morphophonemes implied by Table 12

Morpho- phoneme	Heights	Backnesses	Roundings	Penalty
aä	1,1	5,1	0,0	1
j0	7,–	1,–	0,–	2
i0	7,–	1,–	0,–	2
ie	7,5	1,1	0,0	1
ao	1,5	5,5	0,1	2
ai	1,7	5,1	0,0	3

native. The alignment is almost as good as the earlier one. If we add up the penalties, as in Table 13, we get a total penalty of 11. With these weightings, the procedure would choose the "right" solution, but in general, we cannot select the best one on the basis of the morphophonemes alone. The "almost good" solution could be used for deducing the rules and would account for the facts. Linguistically, however, **ai** is not a very attractive alternation because the components are from the extreme ends of both the height and backness scales. Maybe the penalty for such ought to be even higher than 3.

8 HOW A LINGUIST COULD FIND THE RULES FOR MORPHOPHONEMES

The procedure has used the suitability of induced morphophonemes in order to guide the selection of paradigm tables for the next step where two-level rules are deduced. The procedure continues with the best alternative (which was presented in Table 5). The morphophonemes established in the preceding steps already define how they may be realised on the surface level. The rules must specify in what kinds of contexts each of the alternatives can occur. We first discuss how a human linguist could approach the discovery of the two-level rules needed.

We drop the pluses from our table as they are not needed in the example we are studying (and not in many other cases either). The alphabet of the two-level grammar is already defined through the letters occurring in the example and as a consequence of the morphophonemes implied by the alignment:

```
Alphabet a k l o r s t v ä ö  aä:a aä:ä ij:i ij:j
  ie:i ie:e ie0:i ie0:e ie0:0 ao:a ao:o a0:a a0:0 ;
```

Table 14 presents the facts about the contexts where the morphophonemes occur. The pair in focus (one in each word form) is marked with bold face. Different realisations of a morphophoneme are given in separate columns. For the convenience of the reader, the columns have been arranged so that the realisation with incoherent surrounding contexts is always in the leftmost column, whereas the realisations which have more regular contexts are listed in the other columns.

The upper half of the middle column shows that certain stem-final vowel morphophoneme realisations (**ie:e**, **ie0:0**, **ao:o** and **a0:0**) may only occur before a plural affix which starts with **i:** or **ij:**. Looking at the other columns, we see that such contexts do not occur with the other realisations of the morphophonemes (i.e. **ie:i**, **ie0:0**, **ao:a**, **a0:a** or **ie0:i**). Thus those realisations of stem-final vowels occur (**ie:e**, **ie0:0**, **ao:o** and **a0:0**) only in this context, and this context is the only alternative. As two-level rules:

```
″ie″   ie:e <=> _ [i: | ij:] ;
″ie0″ ie0:0 <=> _ [i: | ij:] ;
″ao″   ao:o <=> _ [i: | ij:] ;
″a0″   a0:0 <=> _ [i: | ij:] ;
```

In the rightmost column, we have one more realisation for which we (as linguists) find a simple formulation: **ie0** is realised as **i** at the end of a word form. As a two-level rule:

```
″ie0″ ie0:0 <=> _ .#.: ;
```

When we look (as linguists) at the distribution of the plural **ij:j** alternative, we note that the stem has to end in a surface vowel. If the end vowel disappears on the surface, then **ij:j** may not occur, thus:

```
″ij″ ij:j <=> [:o | :ö | :e ] _ ;
```

The last morphophoneme to account for is **aä**. Whereas the earlier cases depended only on the immediate context, here the realisation depends on all vowels of stem to the left. The backness of the vowels is decisive, and we note that in the middle column there is at least

Table 14: Realisations of morphophonemes and their contexts

r i s t **ie:i**		
r i s t **ie:i** aä:ä	r i s t **ie:e** ij:j aä:ä	
r i s t **ie:i** s s aä:ä	r i s t **ie:e** i s s aä:ä	
l o v **ie0:e** aä:a	l o v **ie0:0** ij:i aä:a	l o v **ie0:i**
l o v **ie0:e** s s aä:a	l o v **ie0:0** i s s aä:a	
k a l **ao:a**		
k a l **ao:a** aä:a	k a l **ao:o** ij:j aä:a	
k a l **ao:a** s s aä:a	k a l **ao:o** i s s aä:a	
k o i r **a0:a**		
k o i r **a0:a** aä:a	k o i r **a0:0** ij:i aä:a	
k o i r **a0:a** s s aä:a	k o i r **a0:0** i s s aä:a	
l o v ie0:0 **ij:i** aä:a	t a l o **ij:j** aä:a	
k o i r a0:0 **ij:i** aä:a	s ä r ö **ij:j** aä:ä	
	r i s t ie:e **ij:j** aä:ä	
	k a l ao:o **ij:j** aä:a	
s ä r ö **aä:ä**	t a l o **aä:a**	
s ä r ö s s **aä:ä**	t a l o s s **aä:a**	
s ä r ö ij:j **aä:ä**	t a l o ij:j **aä:a**	
s ä r ö i s s **aä:ä**	t a l o i s s **aä:a**	
r i s t ie:i **aä:ä**	l o v ie0:e **aä:a**	
r i s t ie:i s s **aä:ä**	l o v ie0:e s s **aä:a**	
r i s t ie:e ij:j **aä:ä**	l o v ie0:0 ij:i **aä:a**	
r i s t ie:e i s s **aä:ä**	l o v ie0:0 i s s **aä:a**	
	k a l ao:a **aä:a**	
	k a l ao:a s s **aä:a**	
	k a l ao:o ij:j **aä:a**	
	k a l ao:o i s s **aä:a**	
	k o i r a0:a **aä:a**	
	k o i r a0:a s s **aä:a**	
	k o i r a0:0 ij:i **aä:a**	
	k o i r a0:0 i s s **aä:a**	

one back vowel somewhere in the stem whereas in the left column no back vowels occur, thus:[7]

```
"aä" aä:a <=> [:a | :o | :u] ?:?* _ ;
```

It would be difficult even for the linguist to generalise the contexts in the leftmost column. Therefore, we say that the realisations in that column are the *default realisations* of those morphophonemes. As they are the only remaining alternatives of their morphophonemes, no rules are needed for them.

The linguist who knows some facts about Finnish notices that the inflectional class of **risti** is productive and contains plenty of nouns, whereas that of **lovi** is a closed class containing fewer words. The steps above resulted in different stem-final morphophonemes for the lexical representations these words. The establishment of the lexical representations and the design of the corresponding rules was not affected by the existence of two apparently-overlapping inflectional classes. When one builds a lexicon, one must decide, for each such an ambiguous noun, to which class it belongs in order to build an appropriate lexicon entry. That decision may be made by human informants (e.g., by crowdsourcing), or by using evidence, e.g., from corpora or Internet search engines.

9 PROCEDURE FOR FINDING SHORT CONTEXT

We have now seen how a human linguist might discover the two-level rules according to the envisaged procedure. The formal procedure handles each morphophoneme separately. The procedure could start, e.g., with **ie** and try to find a phonologically natural characterisation of contexts such that **ie:i** in our data occurs in contexts of that type but **ie:e** does not. If a satisfactory result is not reached, the procedure tries to find a natural set of contexts where the other alternative **ie:e** may occur but **ie:i** may not.

Many morphophonological alternations are conditioned by an immediate context consisting of just one or a few phonemes. Thus, the procedure tries to find as short a context as possible which still dis-

[7] In a slightly larger and thus more realistic example we would also have forms like **s i n ie0:e aä:ä** and **k a s t ie:i aä:a** which would rule out attempts to explain the outcome of **aä** on the basis of the stem-final vowel alone.

criminates the desired realisations from all other realisations of the morphophoneme (cf., e.g., Theron and Cloete, 1997). The procedure starts with the full contexts for **ie:i** where both the left and right contexts are present, and we have added a word boundary symbol **#:0** at the beginning and end of the word forms:

```
#:0 r i s t _ #:0
#:0 r i s t _ aä:a #:0
#:0 r i s t _ s s aä:ä #:0
```

This disjunction of the full contexts clearly separates the occurrences of **ie:i** from the other realisation (**ie:e**). The procedure drops characters from the outer ends as long as the disjunction still separates the occurrences. If possible, the longer side of the context is truncated before the shorter side. When processing the above contexts, the procedure will erase the left context altogether but one character must be left to the right context – resulting in a context **_ [#: | aä:ä | s]**. The procedure does not accept this result, as it contains both vowels and consonants and, therefore, is not acceptable on the same arguments which were mentioned when the some morphophonemes were excluded as unnatural in Section 7.

The procedure tests the other alternative **ie:e**, and shortens the contexts until there is just one character left in the right context, i.e. **_ [ij:j | i]**. This context is acceptable on the same grounds as the morphophoneme **ij** itself. Some cleaning and generalisation may still be needed as explained in the next section.

For morphophoneme realisations **ie0:e**, **ao:a**, and **a0:a**, the procedure fails to find a natural context which would discriminate them from the other alternatives. For the other realisations of these morphophonemes, the procedure succeeds in the same way as for **ie:e**. For **ie0:0**, the smallest discriminating context clearly becomes **_ #:0**.

The contexts for the realisation **ij:i** would be just one character to the left, **[ie0:0 | a0:0] _**, and for **ij:j** similarly **[o | ö | ie:e | ao:o] _**. If more than one alternative explanation remains, later steps will decide which one is preferred over the others.

10 GENERALISING CONTEXTS

In the previous section, the procedure found two-level contexts consisting of character pairs. In most cases (but not always), having both levels is superfluous. Proper discrimination can usually be achieved using either the lexical or surface context. Furthermore, the contexts can (and ought to) be generalised to use whole classes of phonemes instead of listing only those letters and combinations which happened to be present in the examples. Both the lexical and surface contexts are generalised first, and the choice between them is made thereafter.

The naturalness of contexts can be evaluated according to the phonemes which occur in individual positions of contexts. Morphophonemes occurring in lexical contexts are treated by splitting them into their component letters, e.g., **ao** is treated as if **a** and **o** would occur in that position of the contexts. Table 15 shows how the phonological properties of one-character-long left contexts of the morphophoneme **ij** can be summarised.

contexts for ij:j				
two-level	ao:o ie:e o ö	Height	Back	Round
lexical	a e i o ö	1–7	1–5	0–1
surface	e o ö	5	1–5	0–1
contexts for ij:i				
two-level	ie0:0 a0:0	Height	Back	Round
lexical	0 a e i	1–7	1–5	0
surface	0	–	–	–
ignoring 0		Place	Manner	Voicing
surface	r v	2–4	trill/fricative	1

Table 15: Generalising the one character left context for **ij:j**

Using the table, the procedure observes that the lexical contexts for the two morphophonemes are overlapping and therefore the lexical context is not useful, but the surface contexts are able to discriminate between the two alternants. If we study the surface contexts, we see that surface zeroes probably ought to be ignored, i.e. the context letter would be the one preceding (or, respectively, following) a zero. In this particular case, it would provide a reasonable context for **ij:i**, but it does not matter because we get a better one from the alternative.

The surface contexts in our examples for **ij:j** allow many vowels, and the procedure generalises it to allow all vowels because the context discriminates between the alternatives. This turns out to be fortunate as further examples would bring stems ending in **u** or **y**.

Let us look back at Table 12, our second best table with zeroes and boundaries. That one was only a bit worse when measured on the basis of the morphophonemes it implied. The context condition for the plural **ij** for the best table (after generalisations) was any surface vowel. The plural morpheme in the second best table needs two separate morphophonemes instead of one, **j0** and **i0**, and a rule for each where the context conditions would be no simpler than in the best table. The stem-final morphophonemes **ai** and **ei** for the second best table would be problematic, as both surface realisations of the morphophonemes occur in an identical surface context. Remember that we decided to omit the explicit boundary symbol (+) for brevity. If we keep the boundary symbol, the rules of the best table can use even the surface context, but definitely not the rules of the second best table. With a reasonable penalty formula for rules and their context expressions, the best table would again get a better score than the second best.

The example of Finnish nouns was a very restricted one because it contained no alternating consonants. Finnish is known to be a language with a fairly complex morphophonology, and has plenty of consonantal alterations as well. Consonant gradation weakens voiceless stops **k**, **p** and **t**. A distance metric for consonants would be similar to that of vowels. The place of articulation can be expressed on a scale from 1 to 11 and the voicing with values 0 and 1 as in Table 10. The manner of articulation often varies in the alternations. However, the different manners are in no particular order and it is difficult to say which manners are close to (or far away from) each other.

Let us consider a mini example of the Finnish consonant gradation in Table 16.

Table 16: Examples of consonant gradation of **p** with added zeroes and boundaries

	SgNom	SgEss	SgIne	SgAll
‘twig’	varpu +	varpu + na	varvu + ssa	varvu + lle
‘shield’	kilpi +	kilpe + nä	kilve + ssä	kilve + lle
‘pond’	lampi +	lampe + na	lamme + ssa	lamme + lle
‘stick’	keppi +	keppi + nä	kep0i + ssä	kep0i + lle

The procedure can readily accept the induced morphophonemes **pv**, **pm** and **p0** because they are articulated approximately in the same place: **pv** in places 1–2, **pm** in places 1–1. The procedure aligns and forms the morphophonemes for this mini example without problems. Note that the direction of the change is well motivated: the stop **p** becomes a bit more like the immediately preceding phoneme **r**, **l** or **m**.

The task for the procedure is to find a generalised context which would account for the occurrence of the alternative realisations. Again, left hand context is not useful at all. The right-hand context appears to be quite sufficient. The lexical context for the the weak alternatives **p:v**, **p:m**, and **p:0** is **u s s aä**, **e s s aä**, **i s s aä** or **u l l e**, **e l l e**, **i l l e** which can be generalised as **V: C: C:** (i.e. a closed syllable) without losing any discriminative power.

11 PROCEDURE FOR FINDING HARMONY CONTEXTS

For the morphophoneme **aä**, the mechanism of finding a short context fails, and the procedure *knows* that it has failed (cf. Table 14). The shortening does not progress successfully, and all tentative contexts are unacceptable (having consonants and vowels in the same positions).

Harmony or agreement can be detected using phonological features. The data for this purpose is collected in Table 17, where the vowel context or the whole preceding word form is summarised. On the left, the word forms with different vowel configurations are listed. In the middle, each of these vowels is represented numerically according to Table 8. On the right, there is a summary of the set of vowels in the word form indicating the range of tongue height, backness and rounding for that word form.

The procedure looks at these summary ranges, and tests each of them whether some of these ranges could be used to separate the words where **aä:ä** occurs from those where **aä:a** occurs. Height is not able to discriminate, and neither can rounding. Backness clearly can. A criterion requiring that a word has at least one vowel with backness > 1 will indicate all those contexts or words where **aä:a** may occur.

The procedure thus finds a positive criterion for the occurrences of **aä:a**, but for **aä:ä** there is only a negative criterion. Thus, the pro-

Table 17: Harmony in terms of tongue height, backness and rounding of vowels in the word form

aä:ä		Height	Back	Round
s**ä**r**ö**_, s**ä**r**ö**ss_, s**ä**r**ö**j_	(2,1,0), (4,1,1)	2–4	1	0–1
s**ä**r**öi**ss_	(2,1,0), (4,1,1), (7,1,0)	2–7	1	0–1
r**i**st**i**_	(7,1,0), (7,1,0)	7	1	0
r**i**st**e**j_	(7,1,0), (5,1,0)	5–7	1	0
r**i**st**ei**ss_	(7,1,0), (5,1,0), (7,1,0)	5–7	1	0

aä:a		Height	Back	Round
t**a**l**o**_, t**a**l**o**ss_, t**a**l**o**j_	(1,5,0), (5,5,1)	1–5	5	0–1
t**a**l**oi**ss_	(1,5,0), (5,5,1), (7,1,0)	1–7	1–5	0–1
l**o**v**e**_, l**o**v**e**ss_	(5,5,1), (5,1,0)	5	1–5	0–1
l**o**v**i**_, l**o**v**i**ss_	(5,5,1), (7,1,0)	5–7	1–5	0–1
k**a**l**a**_	(1,5,0), (1,5,0)	1	5	0
k**a**l**o**j_	(1,5,0), (5,5,1)	1–5	5	0–1
k**a**l**oi**ss_	(1,5,0), (5,5,1), (7,1,0)	1–7	1–5	0–1
k**oi**r**a**_, k**oi**r**a**ss_	(5,5,1), (7,1,0)	5–7	1–5	0–1
k**oi**r**i**_, k**oi**r**i**ss_	(5,5,1), (7,1,0), (7,1,0)	5–7	1–5	0–1

cedure classifies **aä:ä** as the default realisation and no rule is written for it. For **aä:a** the criterion can be expressed as:

```
"aä" aä:a <=> [:a | :o | :u] ?:?* _ ;
```

where the procedure generalises the backness even to **u** – for which there is no example in the data.

12 FINDING GRAMMATICAL CONDITIONS

The above methods do not handle morphophonemic alternations which are difficult or impossible to describe using phonological contexts alone. Grammatical conditioning of morphophonemic alternations is a common phenomenon, though. In some languages, certain inflectional forms are characterised by alternations in the stem rather than by overt affixes. The alignment, adding zeroes and segmenting possible affixes, would proceed with no special problems, but no rules would be found.

The discovery procedure sketched above could easily be modified to discover grammatically conditioned regularities. Suppose that we create a special symbol for each grammatical form. We would include an appropriate special symbol at the end of each word form. If all other patterns for contexts fail, then those special symbols would be tried as contexts. If the presence of certain symbols would discriminate the occurrences of different realisations, the procedure would output a rule of the following type:

```
"ae:e"  ae:e <=> _ ?:?* Passive: ;
```

where **Passive** stands for such a special symbol (which is required to be somewhere in the right context).

13 COMPILING AND VERIFYING THE RULES

Even before we have an implementation for the discovery procedure, one can simulate the steps of the procedure manually. The resulting two-level rules may be compiled using the open source two-level rule compiler HFST-TWOLC written by Miikka Silfverberg of the HFST team at the University of Helsinki (Silfverberg and Lindén, 2009).

The table of example word forms after the addition of zeroes and morpheme boundaries can be used for testing the (automatically or manually produced) two-level rules by a trivial conversion script which reads in the aligned word forms, builds the morphophonemes, and outputs strings of character pairs suitable for the HFST-PAIR-TEST program of the HFST suite. The file would have lines such as the following:

```
t a l o
t a l o ij:j aä:a
k a l ao:o ij:j aä:a
```

These lines can be input to the HFST-PAIR-TEST program which reports any violations against the rules it finds in the test data. Violations pinpoint the example word form, the position and the rule where the mistake appears to be.

In addition to testing the obvious positive examples, one may produce a file which systematically contains negative examples, i.e. examples derived from the correct ones but where at least one rule is violated. Such a set of negative examples can be produced out of the positive ones by (1) creating a transducer E which accepts the positive examples, (2) taking its input (i.e. upper) side $E.u$, (3) computing the transducer P which accepts the pair alphabet of the rules, (4) computing $[E.u\ .o.\ P*] - E$ and (5) listing the pair strings it accepts.

14 CONCLUSIONS AND FUTURE TASKS

A concrete implementation of the discovery procedure would, of course, be needed in order to draw any final conclusions. The paper is intended to be a useful specification for implementing the process.[8]

One goal of the above discussion has been to explain the utility of the phoneme-by-phoneme alignment of word forms. If an acceptable alignment is reached, then the establishment of lexical representations is trivial and the induction of rules is fairly simple. Different types of (morpho)phonological phenomena may need specialised functions which can be added in a modular fashion. The two-level grammars that the procedure creates can cope with some phenomena which often cause problems for linguists when they are writing rules:

- *Interactions* do not occur between rules, except that one has to process the realisations of each morphophoneme together so that the morphophoneme leaves just one of its surface realisations without a rule (as the default realisation). The inference of each rule is entirely independent of the form (or existence) of other rules.
- *Epenthesis* (where surface phonemes have no counterpart in the lexical representation) never occurs. Instead, the alignment produces a morphophoneme, e.g., **e0** if a vowel **e** is inserted to resolve a complex consonant cluster. Technically, epenthesis is reduced to normal correspondences. (Rules for epenthesis are tricky in some formalisms.)
- *Overlapping* or inclusive *contexts*. Finnish consonant gradation has one such example with the weak counterparts of **k**. A single **k** is normally deleted in the weak grade (**koko** – **koon**), but in some

[8] There are suitable open source tools available which support the algebra of weighted finite-state transducers and which could be used for implementing the discovery procedure, see, e.g., Lindén *et al.* (2011).

words it alternates with **v** (**puku – puvun**) and between identical vowels preceded by a long vowel it is spelled as an apostrophe (**raaka – raa'an**). The alignment produces three different morphophonemes (**k0**, **kv** and **k'**) for these cases and the rule discovery notices no problems at all.

- All conditions for a certain surface realisation of a morphophoneme are represented using a single rule. In this way, the rest of the sources for rule interactions are avoided. It should be noted that some natural phonological contexts may consist of disjoint parts. The above discussion did not cover such cases. Combinations of contexts can be expressed using multiple context parts in the two-level rules (provided that each context part can be discovered separately), but the discovery of rules needing multiple contexts is not discussed in this paper.

Some simplicity and computational feasibility was gained by adopting the above framework (with many morphophonemes) at a price of losing some linguistic elegance. The procedure moves a part of the complexity of the morphophonology from the rules into the lexical representations. The rules need not bother with all possible realisations of an underlying phoneme because, instead of a phoneme, there is a morphophoneme which specifies exactly what the alternatives are. It should be noted that the above procedure (and the two-level rules in general) appear to work best with *phonemic alphabets*. Conventional orthographies may be quite different from their pronunciation and thus complicate rule discovery. Furthermore, isolating languages like Chinese or Vietnamese have little morphophonology to be discovered by any procedure. In some European languages, the morphophonological rules play a minor role. Thus, the proposed procedure might be most useful in languages with plenty of regular phonological alternations.

The tractability of the procedure is not obvious, even if we assume that the table of examples is well chosen. Some care has been taken in order to restrict the searching sufficiently so that the complexity of the computation would not explode. It seems to be useful to fix the assumptions concerning the stem and affix lengths before one starts finding appropriate places for the zeroes. Similarly, it seems useful to check for rough compatibility before generalising any contexts. These and other similar precautions do not affect the end result, but may help in finding the solutions reasonably fast.

A linguist would probably like to merge similar or related morphophonemes. Good candidates for merging would be morphophonemes which have a default realisation in common. Merging usually requires some revision of the corresponding two-level rules. It appears to be fairly straightforward to check whether such mergers can be done while keeping the revised rules simple and deducible. The criteria used for deducing single rules apply as such for the merging of rules. Such generalisations would make the lexical representations of morphemes and affixes simpler while remaining fully equivalent with the initial version. These tasks and questions may possibly be studied and solved by future work, as well as by the elaboration and tuning of the penalty scores for rules.

REFERENCES

Kenneth R. BEESLEY and Lauri KARTTUNEN (2003), *Finite State Morphology*, Studies in Computational Linguistics, 3, University of Chicago Press, additional info, see: `www.stanford.edu/~laurik/fsmbook/home.html`.

Erwin CHAN (2008), Structures and Distributions in Morphology Learning, a dissertation in Computer and Information Science, University of Pennsylvania.

Mathias CREUTZ and Krista LAGUS (2004), Induction of a Simple Morphology for Highly-Inflecting Languages, in *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology*, pp. 43–51, Association for Computational Linguistics, Stroudsburg, PA, USA.

Daniel GILDEA and Daniel JURAFSKY (1995), Automatic induction of finite state transducers for simple phonological rules, in *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pp. 9–15, Association for Computational Linguistics, Cambridge, Massachusetts.

John GOLDSMITH (2006), An Algorithm for the Unsupervised Learning of Morphology, *Natural Language Engineering*, 12(4):353–371.

Sharon GOLDWATER and Mark JOHNSON (2004), Priors in Bayesian Learning of Phonological Rules, in *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology*, pp. 35–42, Association for Computational Linguistics, Stroudsburg, PA, USA.

Mans HULDEN (2009), Foma: a Finite-State Compiler and Library, in *Proceedings of the Demonstrations Session at EACL 2009*, pp. 29–32, Association for Computational Linguistics, Stroudsburg, PA, USA, `http://www.aclweb.org/anthology/E09-2008`.

Mans HULDEN, Iñaki ALEGRIA, Izaskun ETXEBERRIA, and Montse MARITXALAR (2011), Learning word-level dialectal variation as phonological replacement rules using a limited parallel corpus, in *Proceedings of EMNLP 2011, Conference on Empirical Methods in Natural Language Processing*, DIALECTS'11, Association for Computational Linguistics, Stroudsburg, PA, USA.

Mark JOHNSON (1984), A Discovery Procedure for Certain Phonological Rules, in *Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, pp. 344–347, Association for Computational Linguistics, Stroudsburg, PA, USA, http://www.aclweb.org/anthology/P84-1070.

Lauri KARTTUNEN (1993), Finite-state Constraints, in *Proceedings of the International Conference on Current Issues in Computational Linguistics, June 10–14, 1991. Universiti Sains Malaysia, Penang, Malaysia*, pp. 173–194.

Lauri KARTTUNEN and Kenneth R. BEESLEY (2001), A short history of two-level morphology, http://www.helsinki.fi/esslli/evening/20years/twol-history.pdf.

Lauri KARTTUNEN, Kimmo KOSKENNIEMI, and Ronald M. KAPLAN (1987), A compiler for two-level phonological rules, in M. DALRYMPLE, R. KAPLAN, L. KARTTUNEN, K. KOSKENNIEMI, S. SHAIO, and M. WESCOAT, editors, *Tools for Morphological Analysis*, volume 87-108 of *CSLI Reports*, pp. 1–61, Center for the Study of Language and Information, Stanford University, Palo Alto, California, USA.

Laura KATAJA and Kimmo KOSKENNIEMI (1988), Finite-state Description of Semitic Morphology: A Case Study of Ancient Accadian, in *COLING Budapest: Proceedings of the 12th Conference on Computational Linguistics*, pp. 313–315, Association for Computational Linguistics, Stroudsburg, PA, USA, http://aclweb.org/anthology-new/C/C88/C88-1064.pdf.

Grzegorz KONDRAK (2002), *Algorithms for Language Reconstruction*, Ph.D. thesis, University of Toronto.

Kimmo KOSKENNIEMI (1983), *Two-level Morphology: A General Computational Model for Word-Form Recognition and Production*, number 11 in Publications, University of Helsinki, Department of General Linguistics.

Kimmo KOSKENNIEMI (1984), A General Computational Model for Word-Form Recognition and Production, in *Proceedings of COLING-84, 2–4 July 1984, Stanford University, California*, pp. 178–181, Association for Computational Linguistics, Stroudsburg, PA, USA.

Kimmo KOSKENNIEMI (1991), A Discovery Procedure for Two-level Phonology, in L. CIGNONI and C. PETERS, editors, *Computational Lexicology and Lexicography: Special Issue Dedicated to Bernard Quemada*, volume VI:I, Giardini editori e stampatori in Pisa, Pisa, Italy.

Krister LINDÉN, Erik AXELSON, Sam HARDWICK, Tommi A. PIRINEN, and Miikka SILFVERBERG (2011), HFST – Framework for Compiling and Applying Morphologies, in C. MAHLOW and M. PIOTROWSKI, editors, *Systems and Frameworks for Computational Morphology 2011 (SFCM-2011)*, volume 100 of *Communications in Computer and Information Science*, pp. 67–85, Springer-Verlag.

Tom M. MITCHELL (1982), Generalization as search, *Artificial Intelligence*, 18(2):203–226.

Kemal OFLAZER, Sergei NIRENBURG, and Marjorie MCSHANE (2001), Bootstrapping morphological analyzers by combining human elicitation and machine learning, *Computational Linguistics*, 27(1):59–85.

Kemal OFLAZER and Sergei NIRENBURG (1999), Practical Bootstrapping of Morphological Analyzers, in *Proceedings of Computational Natural Language Learning (CoNLL99). Workshop at EACL'99*, pp. 143–146, Springer-Verlag.

José ONCINA, Pedro GARCÍA, and Enrique VIDAL (1993), Learning subsequential transducers for pattern recognition interpretation tasks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):448–458.

Miikka SILFVERBERG and Krister LINDÉN (2009), Conflict Resolution Using Weighted Rules in HFST-TWOLC, in *Proceedings of the 17th Nordic Conference of Computational Linguistics, NODALIDA 2009*, pp. 174–181, Northern European Association for Language Technology (NEALT), `http://hdl.handle.net/10062/9752`.

Pieter THERON and Ian CLOETE (1997), Automatic Acquisition of Two-Level Morphological Rules, in *Fifth Conference on Applied Natural Language Processing Proceedings of the Conference*, pp. 103–110, Association for Computational Linguistics.

David TOURETZKY, Gillette ELVGREN, and Deirdre W. WHEELER (1990), Phonological rule induction: An architectural solution, in *Proceedings of the 12th Annual Conference of the Cognitive Science Society (COGSCI-90)*, pp. 348–355, Cognitive Science Society.

David YAROWSKY and Richard WICENTOWSKI (2000), Minimally supervised morphological analysis by multimodal alignment, in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics: Hong Kong*, Association for Computational Linguistics.