Smart Paradigms and the Predictability and Complexity of Inflectional Morphology

Grégoire Détrez and Aarne Ranta
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Morphological lexica are often implemented on top of morphological paradigms, corresponding to different ways of building the full inflection table of a word. Computationally precise lexica may use hundreds of paradigms, and it can be hard for a lexicographer to choose among them. To automate this task, this paper introduces the notion of a **smart paradigm**. It is a meta-paradigm, which inspects the base form and tries to infer which low-level paradigm applies. If the result is uncertain, more forms are given for discrimination. The number of forms needed in average is a measure of **predictability** of an inflection system. The overall **complexity** of the system also has to take into account the code size of the paradigms definition itself. This paper evaluates the smart paradigms implemented in the open-source GF Resource Grammar Library. Predictability and complexity are estimated for four different languages: English, French, Swedish, and Finnish. The main result is that predictability does not decrease when the complexity of morphology grows, which means that smart paradigms provide an efficient tool for the manual construction and/or automatically bootstrapping of lexica.

1 Introduction

Paradigms are a cornerstone of grammars in the European tradition. A classical Latin grammar has five paradigms for nouns ("declensions") and four for verbs ("conjugations"). The modern reference on French verbs, *Bescherelle* (Bescherelle, 1997), has 88 paradigms for verbs. Swedish grammars traditionally have, like Latin, five paradigms for nouns and four for verbs, but a modern computational account (Hellberg, 1978), aiming for more precision, has 235 paradigms for Swedish.

Mathematically, a paradigm is a function that produces inflection tables. Its argument is a word string (either a **dictionary form** or a **stem**), and its value is an n-tuple of strings (the word forms):

$$P : \text{String} \rightarrow \text{String}^n$$

We assume that the exponent n is determined by the language and the part of speech. For instance, English verbs might have $n = 5$ (for *sing, sings, sang, sung, singing*), whereas for French verbs in *Bescherelle*, $n = 51$. We assume the tuples to be ordered, so that for instance the French second person singular present subjunctive is always found at position 17. In this way, word-paradigm pairs can be easily converted to morphogical lexica and to transducers that map form descriptions to surface forms and back. A properly designed set of paradigms permits a compact representation of a lexicon and a user-friendly way to extend it.

Different paradigm systems may have different numbers of paradigms. There are two reasons for this. One is that traditional paradigms often in fact require more arguments than one:

$$P : \text{String}^m \rightarrow \text{String}^n$$

Here $m \leq n$ and the set of arguments is a subset of the set of values. Thus the so-called fourth verb conjugation in Swedish actually needs three forms to work properly, for instance *sitta, satt, suttit* for the equivalent of *sit, sat, sat* in English. In Hellberg (1978), as in the French *Bescherelle*, each paradigm is defined to take exactly one argument, and hence each vowel alternation pattern must be a different paradigm.

The other factor that affects the number of paradigms is the nature of the string operations

allowed in the function P. In Hellberg (1978), noun paradigms only permit the concatenation of suffixes to a stem. Thus the paradigms are identified with suffix sets. For instance, the inflection patterns *bil–bilar* ("car–cars") and *nyckel–nycklar* ("key–keys") are traditionally both treated as instances of the second declension, with the plural ending *ar* and the contraction of the unstressed *e* in the case of *nyckel*. But in Hellberg, the word *nyckel* has *nyck* as its "technical stem", to which the paradigm numbered 231 adds the singular ending *el* and the plural ending *lar*.

The notion of paradigm used in this paper allows multiple arguments and powerful string operations. In this way, we will be able to reduce the number of paradigms drastically: in fact, each lexical category (noun, adjective, verb), will have just *one* paradigm but with a variable number of arguments. Paradigms that follow this design will be called **smart paradigms** and are introduced in Section 2. Section 3 defines the notions of **predictability** and **complexity** of smart paradigm systems. Section 4 estimates these figures for four different languages of increasing richness in morphology: English, Swedish, French, and Finnish. We also evaluate the smart paradigms as a data compression method. Section 5 explores some uses of smart paradigms in lexicon building. Section 6 compares smart paradigms with related techniques such as morphology guessers and extraction tools. Section 7 concludes.

2 Smart paradigms

In this paper, we will assume a notion of paradigm that allows multiple arguments and arbitrary computable string operations. As argued in (Kaplan and Kay, 1994) and amply demonstrated in (Beesley and Karttunen, 2003), no generality is lost if the string operators are restricted to ones computable by finite-state transducers. Thus the examples of paradigms that we will show (only informally), can be converted to matching and replacements with regular expressions.

For example, a majority of French verbs can be defined by the following paradigm, which analyzes a variable-size suffix of the infinitive form and dispatches to the *Bescherelle* paradigms (identified by a number and an example verb):

$$\text{mkV} : \text{String} \to \text{String}^{51}$$

$$\text{mkV}(s) =$$

- conj19finir(s), if s ends *ir*
- conj53rendre(s), if s ends *re*
- conj14assiéger(s), if s ends *éger*
- conj11jeter(s), if s ends *eler* or *eter*
- conj10céder(s), if s ends *éder*
- conj07placer(s), if s ends *cer*
- conj08manger(s), if s ends *ger*
- conj16payer(s), if s ends *yer*
- conj06parler(s), if s ends *er*

Notice that the cases must be applied in the given order; for instance, the last case applies only to those verbs ending with *er* that are not matched by the earlier cases.

Also notice that the above paradigm is just like the more traditional ones, in the sense that we cannot be sure if it really applies to a given verb. For instance, the verb *partir* ends with *ir* and would hence receive the same inflection as *finir*; however, its real conjugation is number 26 in *Bescherelle*. That mkV uses 19 rather than number 26 has a good reason: a vast majority of *ir* verbs is inflected in this conjugation, and it is also the productive one, to which new *ir* verbs are added.

Even though there is no mathematical difference between the mkV paradigm and the traditional paradigms like those in *Bescherelle*, there is a reason to call mkV a **smart paradigm**. This name implies two things. First, a smart paradigm implements some "artificial intelligence" to pick the underlying "stupid" paradigm. Second, a smart paradigm uses heuristics (informed guessing) if string matching doesn't decide the matter; the guess is informed by statistics of the distributions of different inflection classes.

One could thus say that smart paradigms are "second-order" or "meta-paradigms", compared to more traditional ones. They implement a lot of linguistic knowledge and intelligence, and thereby enable tasks such as lexicon building to be performed with less expertise than before. For instance, instead of "07" for *foncer* and "06" for *marcher*, the lexicographer can simply write "mkV" for all verbs instead of choosing from 88 numbers.

In fact, just "V", indicating that the word is a verb, will be enough, since the name of the paradigm depends only on the part of speech. This follows the model of many dictionaries and

methods of language teaching, where **characteristic forms** are used instead of paradigm identifiers. For instance, another variant of mkV could use as its second argument the first person plural present indicative to decide whether an *ir* verb is in conjugation 19 or in 26:

$$\text{mkV} : \text{String}^2 \to \text{String}^{51}$$

$$\text{mkV}(s, t) =$$

- conj26partir(s), if for some x, $s = x$+*ir* and $t = x$+*ons*
- conj19finir(s), if s ends with *ir*
- (all the other cases that can be recognized by this extra form)
- mkV(s) otherwise (fall-back to the one-argument paradigm)

In this way, a series of smart paradigms is built for each part of speech, with more and more arguments. The trick is to investigate which new forms have the best discriminating power. For ease of use, the paradigms should be displayed to the user in an easy to understand format, e.g. as a table specifying the possible argument lists:

verb	*parler*
verb	*parler, parlons*
verb	*parler, parlons, parlera, parla, parlé*
noun	*chien*
noun	*chien*, masculine
noun	*chien, chiens*, masculine

Notice that, for French nouns, the gender is listed as one of the pieces of information needed for lexicon building. In many cases, it can be inferred from the dictionary form just like the inflection; for instance, that most nouns ending *e* are feminine. A gender argument in the smart noun paradigm makes it possible to override this default behaviour.

2.1 Paradigms in GF

Smart paradigms as used in this paper have been implemented in the GF programming language (Grammatical Framework, (Ranta, 2011)). GF is a functional programming lnguage enriched with regular expressions. For instance, the following function implements a part of the one-argument French verb paradigm shown above. It uses a case expression to pattern match with the argument s; the pattern _ matches anything, while + divides a string to two pieces, and | expresses alternation. The functions `conj19finir` etc. are defined elsewhere in the library. Function application is expressed without parentheses, by the juxtaposition of the function and the argument.

```
mkV : Str -> V
mkV s = case s of {
  _ + "ir" -> conj19finir s ;
  _ + ("eler"|"eter")
           -> conj11jeter s ;
  _ + "er" -> conj06parler s ;
  }
```

The GF Resource Grammar Library[1] has comprehensive smart paradigms for 18 languages: Amharic, Catalan, Danish, Dutch, English, Finnish, French, German, Hindi, Italian, Nepalese, Norwegian, Romanian, Russian, Spanish, Swedish, Turkish, and Urdu. A few other languages have complete sets of "traditional" inflection paradigms but no smart paradigms.

Six languages in the library have comprehensive morphological dictionaries: Bulgarian (53k lemmas), English (42k), Finnish (42k), French (92k), Swedish (43k), and Turkish (23k). They have been extracted from other high-quality resources via conversions to GF using the paradigm systems. In Section 4, four of them will be used for estimating the strength of the smart paradigms, that is, the predictability of each language.

3 Cost, predictability, and complexity

Given a language $\mathcal{L}$, a lexical category C, and a set P of smart paradigms for C, the **predictability** of the morphology of C in $\mathcal{L}$ by P depends inversely on the average number of arguments needed to generate the correct inflection table for a word. The lower the number, the more predictable the system.

Predictability can be estimated from a lexicon that contains such a set of tables. Formally, a smart paradigm is a family P_m of functions

$$P_m : \text{String}^m \to \text{String}^n$$

where m ranges over some set of integers from 1 to n, but need not contain all those integers. A **lexicon** L is a finite set of inflection tables,

$$L = \{w_i : \text{String}^n \mid i = 1, \ldots, M_L\}$$

[1] Source code and documentation in `http://www.grammaticalframework.org/lib`.

As the n is fixed, this is a lexicon specialized to one part of speech. A **word** is an element of the lexicon, that is, an inflection table of size n.

An **application** of a smart paradigm P_m to a word $w \in L$ is an inflection table resulting from applying P_m to the appropriate subset $\sigma_m(w)$ of the inflection table *w*,

$$P_m[w] = P_m(\sigma_m(w)) : \text{String}^n$$

Thus we assume that all arguments are existing word forms (rather than e.g. stems), or features such as the gender.

An application is **correct** if

$$P_m[w] = w$$

The **cost of a word** w is the minimum number of arguments needed to make the application correct:

$$cost(w) = \underset{m}{\text{argmin}}(P_m[w] = w)$$

For practical applications, it is useful to require P_m to be **monotonic**, in the sense that increasing m preserves correctness.

The **cost of a lexicon** L is the average cost for its words,

$$\text{cost}(L) = \frac{\sum_{i=1}^{M_L} \text{cost}(w_i)}{M_L}$$

where M_L is the number of words in the lexicon, as defined above.

The **predictability** of a lexicon could be defined as a quantity inversely dependent on its cost. For instance, an information-theoretic measure could be defined

$$\text{predict}(L) = \frac{1}{1 + \log \text{cost}(L)}$$

with the intuition that each added argument corresponds to a choice in a decision tree. However, we will not use this measure in this paper, but just the concrete cost.

The **complexity of a paradigm system** is defined as the size of its code in a given coding system, following the idea of Kolmogorov complexity (Solomonoff, 1964). The notion assumes a coding system, which we fix to be GF source code. As the results are relative to the coding system, they are only usable for comparing definitions in the same system. However, using GF source code size rather than e.g. a finite automaton size gives in our view a better approximation of the "cognitive load" of the paradigm system, its "learnability". As a functional programming language, GF permits abstractions comparable to those available for human language learners, who don't need to learn the repetitive details of a finite automaton.

We define the **code complexity** as the size of the abstract syntax tree of the source code. This size is given as the number of nodes in the syntax tree; for instance,

- $\text{size}(f(x_1, \ldots, x_n)) = 1 + \sum_{i=1}^{n} \text{size}(x_i)$
- $\text{size}(s) = 1$, for a string literal s

Using the abstract syntax size makes it possible to ignore programmer-specific variation such as identifier size. Measurements of the GF Resource Grammar Library show that code size measured in this way is in average 20% of the size of source files in bytes. Thus a source file of 1 kB has the code complexity around 200 on the average.

Notice that code complexity is defined in a way that makes it into a straightforward generalization of the cost of a word as expressed in terms of paradigm applications in GF source code. The source code complexity of a paradigm application is

$$\text{size}(P_m[w]) = 1 + m$$

Thus the complexity for a word w is its cost plus one; the addition of one comes from the application node for the function P_m and corresponds to knowing the part of speech of the word.

4 Experimental results

We conducted experiments in four languages (English, Swedish, French and Finnish[2]), presented here in order of morphological richness. We used trusted full form lexica (i.e. lexica giving the complete inflection table of every word) to compute the predictability, as defined above, in terms of the smart paradigms in GF Resource Grammar Library.

We used a simple algorithm for computing the cost c of a lexicon L with a set P_m of smart paradigms:

[2]This choice correspond to the set of language for which both comprehensive smart paradigms and morphological dictionaries were present in GF with the exception of Turkish, which was left out because of time constraints.

- set $c := 0$
- for each word w_i in L,
  - for each m in growing order for which P_m is defined:
    if $P_m[w] = w$, then $c := c + m$, else try with next m
- return c

The average cost is c divided by the size of L.

The procedure presupposes that it is always possible to get the correct inflection table. For this to be true, the smart paradigms must have a "worst case scenario" version that is able to generate all forms. In practice, this was not always the case but we checked that the number of problematic words is so small that it wouldn't be statistically significant. A typical problem word was the equivalent of the verb *be* in each language.

Another source of deviation is that a lexicon may have inflection tables with size deviating from the number n that normally defines a lexical category. Some words may be "defective", i.e. lack some forms (e.g. the singular form in "plurale tantum" words), whereas some words may have several variants for a given form (e.g. *learned* and *learnt* in English). We made no effort to predict defective words, but just ignored them. With variant forms, we treated a prediction as correct if it matched any of the variants.

The above algorithm can also be used for helping to select the optimal sets of characteristic forms; we used it in this way to select the first form of Swedish verbs and the second form of Finnish nouns.

The results are collected in Table 1. The sections below give more details of the experiment in each language.

4.1 English

As gold standard, we used the electronic version of the Oxford Advanced Learner's Dictionary of Current English[3] which contains about 40,000 root forms (about 70,000 word forms).

Nouns. We considered English nouns as having only two forms (singular and plural), excluding the genitive forms which can be considered to be clitics and are completely predictable. About one third of the nouns of the lexicon were not included in the experiment because one of the form was missing. The vast majority of the remaining 15,000 nouns are very regular, with predictable deviations such as *kiss - kisses* and *fly - flies* which can be easily predicted by the smart paradigm. With the average cost of 1.05, this was the most predictable lexicon in our experiment.

Verbs. Verbs are the most interesting category in English because they present the richest morphology. Indeed, as shown by Table 1, the cost for English verbs, 1.21, is similar to what we got for morphologically richer languages.

4.2 Swedish

As gold standard, we used the SALDO lexicon (Borin et al., 2008).

Nouns. The noun inflection tables had 8 forms (singular/plural indefinite/definite nominative/genitive) plus a gender (uter/neuter). Swedish nouns are intrinsically very unpredictable, and there are many examples of homonyms falling under different paradigms (e.g. *val - val* "choice" vs. *val -valar* "whale"). The cost 1.70 is the highest of all the lexica considered. Of course, there may be room for improving the smart paradigm.

Verbs. The verbs had 20 forms, which included past participles. We ran two experiments, by choosing either the infinitive or the present indicative as the base form. In traditional Swedish grammar, the base form of the verb is considered to be the infinitive, e.g. *spela, leka* ("play" in two different senses). But this form doesn't distinguish between the "first" and the "second conjugation". However, the present indicative, here *spelar, leker*, does. Using it gives a predictive power 1.13 as opposed to 1.22 with the infinitive. Some modern dictionaries such as Lexin[4] therefore use the present indicative as the base form.

4.3 French

For French, we used the Morphalou morphological lexicon (Romary et al., 2004). As stated in the documentation[5] the current version of the lexicon (version 2.0) is not complete, and in particular, many entries are missing some or all inflected forms. So for those experiments we only

[3] available in electronic form at `http://www.eecs.qmul.ac.uk/~mpurver/software.html`

[4] `http://lexin.nada.kth.se/lexin/`

[5] `http://www.cnrtl.fr/lexiques/morphalou/LMF-Morphalou.php#body_3.4.11`, accessed 2011-11-04

Table 1: Lexicon size and average cost for the nouns (N) and verbs (V) in four languages, with the percentage of words correctly inferred from one and two forma (i.e. $m = 1$ and $m \leq 2$, respectively).

Lexicon	Forms	Entries	Cost	$m = 1$	$m \leq 2$
Eng N	2	15,029	1.05	95%	100%
Eng V	5	5,692	1.21	84%	95%
Swe N	9	59,225	1.70	46%	92%
Swe V	20	4,789	1.13	97%	97%
Fre N	3	42,390	1.25	76%	99%
Fre V	51	6,851	1.27	92%	94%
Fin N	34	25,365	1.26	87%	97%
Fin V	102	10,355	1.09	96%	99%

included entries where all the necessary forms were presents.

Nouns: Nouns in French have two forms (singular and plural) and an intrinsic gender (masculine or feminine), which we also considered to be a part of the inflection table. Most of the unpredictability comes from the impossibility to guess the gender.

Verbs: The paradigms generate all of the simple (as opposed to compound) tenses given in traditional grammars such as the *Bescherelle*. Also the participles are generated. The auxiliary verb of compound tenses would be impossible to guess from morphological clues, and was left out of consideration.

4.4 Finnish

The Finnish gold standard was the KOTUS lexicon (Kotimaisten Kielten Tutkimuskeskus, 2006). It has around 90,000 entries tagged with part of speech, 50 noun paradigms, and 30 verb paradigms. Some of these paradigms are rather abstract and powerful; for instance, grade alternation would multiply many of the paradigms by a factor of 10 to 20, if it was treated in a concatenative way. For instance, singular nominative-genitive pairs show alternations such as *talo–talon* ("house"), *katto–katon* ("roof"), *kanto–kannon* ("stub"), *rako–raon* ("crack"), and *sato–sadon* ("harvest"). All of these are treated with one and the same paradigm, which makes the KOTUS system relatively abstract.

The total number of forms of Finnish nouns and verbs is a question of definition. Koskenniemi (Koskenniemi, 1983) reports 2000 for nouns and 12,000 for verbs, but most of these forms result by adding particles and possessive suffixes in an agglutinative way. The traditional number and case count for nouns gives 26, whereas for verbs the count is between 100 and 200, depending on how participles are counted. Notice that the definition of predictability used in this paper doesn't depend on the number of forms produced (i.e. not on *n* but only on *m*); therefore we can simply ignore this question. However, the question is interesting if we think about paradigms as a data compression method (Section 4.5).

Nouns. Compound nouns are a problem for morphology prediction in Finnish, because inflection is sensitive to the vowel harmony and number of syllables, which depend on where the compound boundary goes. While many compounds are marked in KOTUS, we had to remove some compounds with unmarked boundaries. Another peculiarity was that adjectives were included in nouns; this is no problem since the inflection patterns are the same, if comparison forms are ignored. The figure 1.26 is better than the one reported in (Ranta, 2008), which is 1.42; the reason is mainly that the current set of paradigms has a better coverage of three-syllable nouns.

Verbs. Even though more numerous in forms than nouns, Finnish verbs are highly predictable (1.09).

4.5 Complexity and data compression

The cost of a lexicon has an effect on learnability. For instance, even though Finnish words have ten or a hundred times more forms than English forms, these forms can be derived from roughly the same number of characteristic forms as in English. But this is of course just a part of the truth: it might still be that the paradigm system itself is much more complex in some languages than oth-

Table 2: Paradigm complexities for nouns and verbs in the four languages, computed as the syntax tree size of GF code.

language	noun	verb	total
English	403	837	991
Swedish	918	1039	1884
French	351	2193	2541
Finnish	4772	3343	6885

ers.

Following the definitions of Section 3, we have counted the the complexity of the smart paradigm definitions for nouns and verbs in the different languages in the GF Resource Grammar Library. Notice that the total complexity of the system is lower than the sum of the parts, because many definitions (such as morphophonological transformations) are reused in different parts of speech. The results are in Table 2.

These figures suggest that Finnish indeed has a more complex morphology than French, and English is the simplest. Of course, the paradigms were not implemented with such comparisons in mind, and it may happen that some of the differences come from different coding styles involved in the collaboratively built library. Measuring code syntax trees rather than source code text neutralizes some of this variation (Section 3).

Finally, we can estimate the power of smart paradigms as a data compression function. In a sense, a paradigm is a function designed for the very purpose of compressing a lexicon, and one can expect better compression than with generic tools such as bzip2. Table 3 shows the compression rates for the same full-form lexica as used in the predictability experiment (Table 1). The sizes are in kilobytes, where the code size for paradigms is calculated as the number of constructors multiplied by 5 (Section 3). The source lexicon size is a simple character count, similar to the full-form lexicon.

Unexpectedly, the compression rate of the paradigms improves as the number of forms in the full-form lexicon increases (see Table 1 for these numbers). For English and French nouns, bzip2 is actually better. But of course, unlike the paradigms, it also gives a global compression over all entries in the lexicon. Combining the two methods by applying bzip2 to the source code gives, for the Finnish verb lexicon, a file of 60 kB, which implies a joint compression rate of 227.

That the compression rates for the code can be higher than the numbers of forms in the full-form lexicon is explained by the fact that the generated forms are longer than the base forms. For instance, the full-form entry of the Finnish verb *uida* (”swim”) is 850 bytes, which means that the average form size is twice the size of the basic form.

5 Smart paradigms in lexicon building

Building a high-quality lexicon needs a lot of manual work. Traditionally, when one is not writing all the forms by hand (which would be almost impossible in languages with rich morphology), sets of paradigms are used that require the lexicographer to specify the base form of the word and an identifier for the paradigm to use. This has several usability problems: one has to remember all the paradigm identifiers and choose correctly from them.

Smart paradigm can make this task easier, even accessible to non-specialist, because of their ability to guess the most probable paradigm from a single base form. As shown by Table 1, this is more often correct than not, except for Swedish nouns. If this information is not enough, only a few more forms are needed, requiring only practical knowledge of the language. Usually (92% to 100% in Table 1), adding a second form ($m = 2$) is enough to cover all words. Then the best practice for lexicon writing might be always to give these two forms instead of just one.

Smart paradigms can also be used for an automatic bootstrapping of a list of base forms into a full form lexicon. As again shown by the last column of Table 1, one form alone can provide an excellent first approximation in most cases. What is more, it is often the case that uncaught words belong to a limited set of “irregular” words, such as the irregular verbs in many languages. All new words can then be safely inferred from the base form by using smart paradigms.

6 Related work

Smart paradigms were used for a study of Finnish morphology in (Ranta, 2008). The present paper can be seen as a generalization of that experiment to more languages and with the notion of code

Table 3: Comparison between using bzip2 and paradigms+lexicon source as a compression method. Sizes in kB.

Lexicon	Fullform	bzip2	fullform/bzip2	Source	fullform/source
Eng N	264	99	2.7	135	2.0
Eng V	245	78	3.2	57	4.4
Swe N	6,243	1,380	4.5	1,207	5.3
Swe V	840	174	4.8	58	15
Fre N	952	277	3.4	450	2.2
Fre V	3,888	811	4.8	98	40
Fin N	11,295	2,165	5.2	343	34
Fin V	13,609	2,297	5.9	123	114

complexity. Also the paradigms for Finnish are improved here (cf. Section 4.4 above).

Even though smart paradigm-like descriptions are common in language text books, there is to our knowledge no computational equivalent to the smart paradigms of GF. Finite state morphology systems often have a function called a **guesser**, which, given a word form, tries to guess either the paradigm this form belongs to or the dictionary form (or both). A typical guesser differs from a smart paradigms in that it does not make it possible to correct the result by giving more forms. Examples of guessers include (Chanod and Tapanainen, 1995) for French, (Hlaváčová, 2001) for Czech, and (Nakov et al., 2003) for German.

Another related domain is the unsupervised learning of morphology where machine learning is used to automatically build a language morphology from corpora (Goldsmith, 2006). The main difference is that with the smart paradigms, the paradigms and the guess heuristics are implemented manually and with a high certainty; in unsupervised learning of morphology the paradigms are induced from the input forms with much lower certainty. Of particular interest are (Chan, 2006) and (Dreyer and Eisner, 2011), dealing with the automatic extraction of paradigms from text and investigate how good these can become. The main contrast is, again, that our work deals with handwritten paradigms that are correct by design, and we try to see how much information we can drop before losing correctness.

Once given, a set of paradigms can be used in automated lexicon extraction from raw data, as in (Forsberg et al., 2006) and (Clément et al., 2004), by a method that tries to collect a sufficient number of forms to determine that a word belongs to a certain paradigm. Smart paradigms can then give the method to actually construct the full inflection tables from the characteristic forms.

7 Conclusion

We have introduced the notion of smart paradigms, which implement the linguistic knowledge involved in inferring the inflection of words. We have used the paradigms to estimate the predictability of nouns and verbs in English, Swedish, French, and Finnish. The main result is that, with the paradigms used, less than two forms in average is always enough. In half of the languages and categories, one form is enough to predict more than 90% of forms correctly. This gives a promise for both manual lexicon building and automatic bootstrapping of lexicon from word lists.

To estimate the overall complexity of inflection systems, we have also measured the size of the source code for the paradigm systems. Unsurprisingly, Finnish is around seven times as complex as English, and around three times as complex as Swedish and French. But this cost is amortized when big lexica are built.

Finally, we looked at smart paradigms as a data compression method. With simple morphologies, such as English nouns, bzip2 gave a better compression of the lexicon than the source code using paradigms. But with Finnish verbs, the compression rate was almost 20 times higher with paradigms than with bzip2.

The general conclusion is that smart paradigms are a good investment when building morphological lexica, as they ease the task of both human lexicographers and automatic bootstrapping

methods. They also suggest a method to assess the complexity and learnability of languages, related to Kolmogorov complexity. The results in the current paper are just preliminary in this respect, since they might still tell more about particular implementations of paradigms than about the languages themselves.

Acknowledgements

We are grateful to the anonymous referees for valuable remarks and questions. The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no FP7-ICT-247914 (the MOLTO project).

References

[Beesley and Karttunen2003] Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications.

[Bescherelle1997] Bescherelle. 1997. *La conjugaison pour tous*. Hatier.

[Borin et al.2008] Lars Borin, Markus Forsberg, and Lennart Lönngren. 2008. Saldo 1.0 (svenskt associationslexikon version 2). *Språkbanken*, 05.

[Chan2006] Erwin Chan. 2006. Learning probabilistic paradigms for morphology in a latent class model. In *Proceedings of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology and Morphology*, SIGPHON '06, pages 69–78, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Chanod and Tapanainen1995] Jean-Pierre Chanod and Pasi Tapanainen. 1995. Creating a tagset, lexicon and guesser for a french tagger. *CoRR*, cmp-lg/9503004.

[Clément et al.2004] Lionel Clément, Benoît Sagot, and Bernard Lang. 2004. Morphology based automatic acquisition of large-coverage lexica. In *Proceedings of LREC-04, Lisboa, Portugal*, pages 1841–1844.

[Dreyer and Eisner2011] Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a dirichlet process mixture model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 616–627, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Forsberg et al.2006] Markus Forsberg, Harald Hammarström, and Aarne Ranta. 2006. Morphological Lexicon Extraction from Raw Text Data. In T. Salakoski, editor, *FinTAL 2006*, volume 4139 of *LNCS/LNAI*.

[Goldsmith2006] John Goldsmith. 2006. An Algorithm for the Unsupervised Learning of Morphology. *Nat. Lang. Eng.*, 12(4):353–371.

[Hellberg1978] Staffan Hellberg. 1978. *The Morphology of Present-Day Swedish*. Almqvist & Wiksell.

[Hlaváčová2001] Jaroslava Hlaváčová. 2001. Morphological guesser of czech words. In Václav Matoušek, Pavel Mautner, Roman Moucek, and Karel Taušer, editors, *Text, Speech and Dialogue*, volume 2166 of *Lecture Notes in Computer Science*, pages 70–75. Springer Berlin / Heidelberg.

[Kaplan and Kay1994] R. Kaplan and M. Kay. 1994. Regular Models of Phonological Rule Systems. *Computational Linguistics*, 20:331–380.

[Koskenniemi1983] Kimmo Koskenniemi. 1983. *Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production*. Ph.D. thesis, University of Helsinki.

[Kotimaisten Kielten Tutkimuskeskus2006] Kotimaisten Kielten Tutkimuskeskus. 2006. KOTUS Wordlist. `http://kaino.kotus.fi/sanat/nykysuomi`.

[Nakov et al.2003] Preslav Nakov, Yury Bonev, and et al. 2003. Guessing morphological classes of unknown german nouns.

[Ranta2008] Aarne Ranta. 2008. How predictable is Finnish morphology? an experiment on lexicon construction. In J. Nivre and M. Dahllöf and B. Megyesi, editor, *Resourceful Language Technology: Festschrift in Honor of Anna Sågvall Hein*, pages 130–148. University of Uppsala. `http://publications.uu.se/abstract.xsql?dbid=8933`.

[Ranta2011] Aarne Ranta. 2011. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford. ISBN-10: 1-57586-626-9 (Paper), 1-57586-627-7 (Cloth).

[Romary et al.2004] Laurent Romary, Susanne Salmon-Alt, and Gil Francopoulo. 2004. Standards going concrete: from LMF to Morphalou. In *The 20th International Conference on Computational Linguistics - COLING 2004*, Genève/Switzerland. coling.

[Solomonoff1964] Ray J. Solomonoff. 1964. A formal theory of inductive inference: Parts 1 and 2. *Information and Control*, 7:1–22 and 224–254.