How Predictable is Finnish Morphology: An Experiment in Lexicon Construction

Aarne Ranta

CLT Seminar, 25 September 2008
Based on the article


Available from series homepage: http://publications.uu.se/abstract.xsql?dbid=8933
**Finnish**

Finnish: a Fenno-Ugric (non-Indo-European) language spoken by 6M people in Finland, Northern Sweden, and North-West Russia.

Related to: Estonian, Sami, Hungarian.

Finnish morphology: extremely complex, extremely regular.

Words can have thousands of forms.

The structure of a Finnish noun

Possible components of a written and spoken word:

<table>
<thead>
<tr>
<th>stem</th>
<th>number</th>
<th>case</th>
<th>possessive</th>
<th>particles</th>
</tr>
</thead>
<tbody>
<tr>
<td>talo +</td>
<td>i +</td>
<td>ssa +</td>
<td>ni +</td>
<td>kin</td>
</tr>
<tr>
<td>&quot;house&quot;</td>
<td>Plural</td>
<td>Inessive</td>
<td>PossessiveSg1</td>
<td>&quot;also&quot;</td>
</tr>
</tbody>
</table>

"also in my houses"

This can be estimated to produce $2 \times 14 \times 6 \times 10 = 1680$ forms.

Lemma: *vesi*

Forms with different stems: *vesi, veden, vettä, vedessä*

Another example: *yö, yön, öitä*
Plan

Implementing and describing morphology
Paradigms and smart paradigms
Finnish nouns
Bootstrapping a lexicon
Other languages
Implementing and describing morphology
Morphological processing

Analysis: given a word (string), find its form description.
Synthesis: given a form description, find the resulting string.
Description = lemma followed by tags

Example of words and form descriptions in Swedish

___ lek +N +Utr +Sg +Indef +Gen
___ leka +V +Pass +Pres
___ leks ___
___ lekes ___/

This tiny example shows that both analysis and synthesis can give many results.
Transducers for morphology

Idea: define a relation between form descriptions and concrete forms,

\[(\text{cat} \mid \text{dog}) (+\text{Pl}:s) \mid (\text{bab} (\text{y}:i) (\emptyset:e) (+\text{Pl}:s))\]

This generates the following relation:

\[\{(\text{cat}+\text{Pl}, \text{cats}), (\text{dog}+\text{Pl}, \text{dogs}), (\text{baby}+\text{Pl}, \text{babies})\}\]

To perform language processing

- to synthesize, apply down the transducer
- to analyse, apply up the transducer
Other formats for a finite morphology

Full-form lexicon: list of all words with their descriptions

lek: lek +N +Utr +Sg +Indef +Nom, leka +V +Act +Imp
leker: leka +V +Act +Pres
lekes: leka +V +Pass +Pres
leks: leka +V +Pass +Pres, lek +N +Utr +Sg +Indef +Gen

Morphological lexicon: list of all lemmas and all their forms

lek N Utr: lek, leks, leken, lekens, lekar, lekars, lekarna, lekarnas
leka V: leka, lekas, leker, leks/lekes, lekte, lektes, lekt, lekts

The forms come in a canonical order, so that it is easy to restore the full description attached to each form.
Analysing with a full-form lexicon

It is easy to compile a full-form lexicon into a **trie** - a prefix tree.

A trie has transitions for each symbol, and it can return a value at any point:

```
  a(2)  r(3)  1: lek +N +Utr +Sg +Indef +Nom
      /       /  2: leka +V +Act +Inf
 l - e - k(1,5) - e - s(4)  3: leka +V +Act +Pres
    \   4: leka +V +Pass +Pres
    s(4)  5: leka +V +Act +Imp
```

N.B. a trie is also a special case of a transducer - an **acyclic deterministic finite automaton**.
Going between the formats

For a finite morphology, it is easy to transform the three formats into each other.

This has sometimes been used for "stealing" a proprietary morphology:

- the transducer is delivered (or made usable over the web) as a binary
- run the transducer on a list of lemmas, to generate all their forms
- this gives a morphological lexicon, which can be compiled into a transducer

There is also a more decent use:

- we are free to specify a morphological lexicon in any way we like
- but we can compile it to a transducer to perform processing tasks
How to define a morphology, 1

Purely finite-state tools: use regular expressions, compile to a transducer

- + linguistic idea: morphological rules are finite-state processes
- + even infinite morphologies automatically become finite-state
- - regular expressions are a low-level language, missing in abstraction and safety (types, data structures)
- - compiling a regular expression into a transducer can be exponential
- - applying the resulting transducer can require backtracking and hence be non-linear in time
How to define a morphology, 2

General programming: use your favourite programming language to define a morphological lexicon (or directly a transducer...)

- + you have a powerful language with data structures and types
- - analysing with an infinite morphology requires more thinking (which can be solved by compiling to a transducer)
Tools for computational morphology

XFST: Xerox Finite State Tool
TwoLC: Xerox Two-Level Morphology Compiler
Zen: linguistic toolkit in OCaml
FM: Functional Morphology library in Haskell
GF: Grammatical Framework

No links: you can easily find these with Google.
Paradigms and smart paradigms
The word and paradigm model

One of the three models in Hockett, "Two models of grammatical description" (Word, 1954).

The traditional model (Greek and Latin grammar).

The most general and powerful: "anything goes".

The other models can be used as auxiliaries when defining a paradigm.

But: there is no precise definition of a paradigm and its application.
Paradigms, mathematically

For each part of speech $C$ ("word class"), associate a finite set $F(C)$ of inflectional features.

An **inflection table** for $C$ is a function of type $F(C) \rightarrow \text{Str}$.

Type $\text{Str}$: lists of strings (which list may be empty).

A **paradigm** for $C$ is a function of type $\text{String} \rightarrow F(C) \rightarrow \text{Str}$.

Thus there are different paradigms for nouns, adjectives, verbs,...
Example: English nouns

\( F(N) = \text{Number}, \text{where Number} = \{\text{Sg, Pl}\} \)

The **worst-case function** needs both forms (using GF notation):

\[
\text{worstN man men} = \\
\text{table} \{ \\
\text{Sg} \Rightarrow \text{man} ; \\
\text{Pl} \Rightarrow \text{men} \\
\}
\]

Regular nouns are defined as follows:

\[
\text{regN dog} = \text{worstN dog (dog + "s")}
\]

We ignore the genitive case for simplicity; it is completely predictable.
Two more paradigms for English nouns

Nouns ending with an s-sound, with plural ending es

sN bus = worstN bus (bus + "es")

Nouns ending with y, plural ies dropping last character

yN fly = worstN fly (init fly + "ies")
Building a lexicon with paradigms

For each entry: just give lemmas with paradigms,

    dog regN
    baby yN
    coach sN
    boy regN
    hero sN
    man men irregN

This can be compiled into a morphological lexicon by applying the paradigms.

Analysis can be performed by compiling the lexicon into a trie.

But how do we select the right paradigm for each word?
Smart paradigms

Use regular expressions to match on the stem and choose the correct paradigm:

```haskell
smartN x = case x of {
  _ + ("a" | "e" | "i" | "o" | "y") + "o" => regN x ;
  _ + ("s" | "sh" | "ch" | "x" | "o") => esN x ;
  _ + ("a" | "e" | "o" | "u") + "y" => regN x ;
  _ + "y" => iesN x ;
  _ => regN x
}
```

(In GF: _ matches anything, + is concatenation, | is union.)
Lexicon with a smart paradigms

Now a lexicon can be written

- embryo smartN
- bus smartN
- boy smartN
- baby smartN
- dog smartN
- man men irregularN
**Overloading**

Functions with different types can have the same name:

\[
\begin{align*}
\text{mkN} &: \text{Str} \to \text{N} = \text{smartN} \\
\text{mkN} &: \text{Str} \to \text{Str} \rightarrow \text{N} = \text{irregN}
\end{align*}
\]

If we know the part of speech, we don't need to mention paradigms:

\[
\text{N:}
\begin{align*}
\text{embryo} \\
\text{bus} \\
\text{boy} \\
\text{baby} \\
\text{dog} \\
\text{man men}
\end{align*}
\]
Finnish nouns
The structure of a Finnish noun: reminder

Possible components of a written and spoken word:

<table>
<thead>
<tr>
<th>stem</th>
<th>number</th>
<th>case</th>
<th>possessive</th>
<th>particles</th>
</tr>
</thead>
<tbody>
<tr>
<td>ves</td>
<td>i</td>
<td>ssä</td>
<td>ni</td>
<td>kin</td>
</tr>
</tbody>
</table>

"water" Plural Inessive Possessive Sg1 "also"

"also in my waters"

This is estimated to lead to $2 \times 14 \times 6 \times 10 = 1680$ forms.

However, the possessive and the particles are (almost) purely concatenative.

Two combinations of number and case don’t exist.

Thus the noun inflection can be defined by a table with 26 forms.
### A Finnish noun inflection table

<table>
<thead>
<tr>
<th>Case</th>
<th>Singular</th>
<th>Plural</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>nominative</td>
<td>vesi</td>
<td>vedet</td>
<td>&quot;water(s)&quot;</td>
</tr>
<tr>
<td>genitive</td>
<td>veden</td>
<td>vesien</td>
<td>&quot;of water(s)&quot;</td>
</tr>
<tr>
<td>partitive</td>
<td>vettä</td>
<td>vesiä</td>
<td>&quot;portion of water(s)&quot;</td>
</tr>
<tr>
<td>essive</td>
<td>vetenä</td>
<td>vesiä</td>
<td>&quot;as water(s)&quot;</td>
</tr>
<tr>
<td>transitive</td>
<td>vedeksi</td>
<td>vesiksi</td>
<td>&quot;to as water(s)&quot;</td>
</tr>
<tr>
<td>inessive</td>
<td>vedessä</td>
<td>vesissä</td>
<td>&quot;in water(s)&quot;</td>
</tr>
<tr>
<td>elative</td>
<td>vedestä</td>
<td>vesistä</td>
<td>&quot;from in water(s)&quot;</td>
</tr>
<tr>
<td>illative</td>
<td>veteen</td>
<td>vesii</td>
<td>&quot;to in water(s)&quot;</td>
</tr>
<tr>
<td>adessive</td>
<td>vedellä</td>
<td>vesillä</td>
<td>&quot;on water(s)&quot;</td>
</tr>
<tr>
<td>ablative</td>
<td>vedeltä</td>
<td>vesiltä</td>
<td>&quot;from on water(s)&quot;</td>
</tr>
<tr>
<td>allative</td>
<td>vedelle</td>
<td>vesille</td>
<td>&quot;to on water(s)&quot;</td>
</tr>
<tr>
<td>abessive</td>
<td>vedettä</td>
<td>vesittä</td>
<td>&quot;without water(s)&quot;</td>
</tr>
<tr>
<td>comitative</td>
<td>-</td>
<td>vesine</td>
<td>&quot;with water(s)&quot;</td>
</tr>
<tr>
<td>instructive</td>
<td>-</td>
<td>vesin</td>
<td>&quot;by means of water(s)&quot;</td>
</tr>
</tbody>
</table>
Finnish paradigms

Nykysuomen Sanakirja ("Dictionary of Contemporary Finnish", NSSI) gives 82 paradigms for nouns and 45 for verbs.

Applying these paradigms is not purely concatenative, though:

- endings obey vowel harmony: choice between a and ä as function of stem

  - stems can undergo consonant gradation: choice between e.g. tt and t as function of ending

\[
\begin{align*}
  \textit{tasku} + \textit{ssa} & = \textit{taskussa} ("in the pocket") \\
  \textit{lätty} + \textit{ssa} & = \textit{lätty} + \textit{ssä} = \textit{lätysä} ("in the pancake")
\end{align*}
\]

Expanding the 82 to purely concatenative paradigms could result in thousands.
Tackling the Finnish complexity: vowel harmony

First of all: separate vowel harmony into a reusable **morphophonemic functions**:

```haskell
vowelHarmony s = case s of {
  _ + ("a" | "o" | "u") + _ => "a" ; -- huppu,hupussa
  _ => "¨ a" -- hyppy,hypysä
}
```

In other words: return a if a, o, or u occurs in the stem, and otherwise return ä.
Tackling the Finnish complexity: consonant gradation

Two more morphophonemic functions:

weakGrade : Str -> Str = \s -> case s of {
    ha + "kk" => ha + "k" ; -- hakku, hakun
    la + "pp" => la + "p" ; -- lappu, lapun
    ka + "tt" => ka + "t" ; -- katto, katon
    ha + "nk" => ha + "ng" ; -- hanko, hangon
    ka + "mp" => ka + "mm" ; -- kampa, kamman
    ra + "nt" => ra + "nn" ; -- ranta, rannan
    ta + "s" + ? => s ; -- tasku, taskun
    ha + "k" => ha ; -- haku, haun
    so + "p" => so + "v" ; -- sopu, sovun
    ro + "t" => ro + "d" ; -- rotu, rodun
    _ => s -- sumu, sumun
}

strongGrade : Str -> Str = -- weakGrade inverted
Tackling the Finnish complexity: the worst-case paradigm

It is in all cases enough to know 10 noun forms to produce all 26 by concatenative processes:

```
mkN_10 ukko ukon ukkoa ukkona ukkoon
    ukkojen ukkoja ukkoina ukoissa ukkoihin =
    let
        a  = last ukkona ;
        uko = init ukon ;
        ukoi = Predef.tk 3 ukoissa ;
    in {s = table {
        NCase Sg Nom => ukko ;
        NCase Sg Gen => ukon ;
        NCase Sg Part => ukkoa ;
        NCase Sg Ess => ukkona ;
        NCase Sg Transl => uko + "ksi" ;
        NCase Sg Iness => uko + "ss" + a ;
        NCase Sg Elat => uko + "st" + a ;
        NCase Sg Illat => ukkoon ;
    } : s}
```
NCase Sg Adess => uko + "ll" + a ;
NCase Sg Ablat => uko + "lt" + a ;
NCase Sg Allat => uko + "lle" ;
NCase Sg Abess => uko + "tt" + a ;
NCase Pl Nom => uko + "t" ;
NCase Pl Gen => ukkojen ;
NCase Pl Part => ukkoja ;
NCase Pl Ess => ukkoina ;
NCase Pl Transl => ukoi + "ksi" ;
NCase Pl Iness => ukoi + "ss" + a ;
NCase Pl Elat => ukoi + "st" + a ;
NCase Pl Illat => ukkoihin ;
NCase Pl Adess => ukoi + "ll" + a ;
NCase Pl Ablat => ukoi + "lt" + a ;
NCase Pl Allat => ukoi + "lle" ;
NCase Pl Abess => ukoi + "tt" + a ;
NComit => init ukkoina + "e" ;
NInstr => ukoi + "n"
A ground paradigm

For words like suo - soita, tie - teitä, yö - öitä.

\[
dSuo : \text{Str} \to \text{NForms} = \lambda \text{suo} \to \\
\text{let} \\
o = \text{last} \ \text{suo} ; \\
a = \text{vowHarmony} \ o ; \\
soi = \text{Predef.tk} \ 2 \ \text{suo} + \ o + \ "i" ; \\
in \text{nForms10} \\
suo (\text{suo} + \ "n") (\text{suo} + \ "t" + \ a) (\text{suo} + \ "n" + \ a) (\text{suo} + \ "h" + \ o + \ "n") \\
(\text{soi} + \ "den") (\text{soi} + \ "t" + \ a) \\
(\text{soi} + \ "n" + \ a) (\text{soi} + \ "ss" + \ a) (\text{soi} + \ "hin") ;
\]
Tackling the Finnish complexity: less paradigms

Using 1-3 arguments, we cut down the 82 paradigms of NSSK to 19.

\[
\begin{align*}
\text{dLujuus} & : \text{(lujuus : Str)} \rightarrow \text{N} \\
\text{dNainen} & : \text{(nainen : Str)} \rightarrow \text{N} \\
\text{dPaluu} & : \text{(paluu : Str)} \rightarrow \text{N} \\
\text{dPuu} & : \text{(puu : Str)} \rightarrow \text{N} \\
\text{dSuo} & : \text{(suo : Str)} \rightarrow \text{N} \\
\text{dKorkea} & : \text{(korkea : Str)} \rightarrow \text{N} \\
\text{dKaunis} & : \text{(kaunis : Str)} \rightarrow \text{N} \\
\text{dLiitin} & : \text{(liitin : Str)} \rightarrow \text{N} \\
\text{dOnneton} & : \text{(onneton : Str)} \rightarrow \text{N} \\
\text{dUkko} & : \text{(ukko,ukon : Str)} \rightarrow \text{N} \\
\text{dSilakka} & : \text{(silakka,silakan,silakoita : Str)} \rightarrow \text{N} \\
\text{dArpi} & : \text{(arpi,arven : Str)} \rightarrow \text{N} \\
\text{dRae} & : \text{(rae,rakeen : Str)} \rightarrow \text{N} \\
\text{dPaatti} & : \text{(paatti,paatin : Str)} \rightarrow \text{N} \\
\text{dTohtori} & : \text{(tohtori : Str)} \rightarrow \text{N} \\
\text{dPiennar} & : \text{(piennar,pientaren : Str)} \rightarrow \text{N}
\end{align*}
\]
dNukke : (nukke,nuken : Str) -> N
dJalas  : (jalas : Str) -> N
dSDP    : (SDP : Str) -> N
A smart paradigm

The following paradigm dispatches to ground paradigms, assuming consonant gradation:

```plaintext
mkN_1 talo = case talo of {
    nai  + "nen" => dNainen ukko ;
    kaun + "is"  => dKaunis ukko ;
    liit + ("i"|"u") + "n"  => dLiitin ukko ;
    rik  + ("as"|"äs") => dRae ukko (strongGrade ...) ;
    luj  + ("uus"|"yys"|"eus"|"eys") => dLujuus ukko ;
    jala + "s" => dJalas ukko ;
    paatt + "i" => dPaatti ukko ukon ;
    ukk  + o@("a"|"o"|"u"|"y"|"ä"|"ö") => dUkko talo (weakGrade ukk + o + "n") ;
    hak  + "e" => dRae talo (strongGrade hak + "een") ;
    _ => dUnix ukko

... 21 cases altogether ...
}
```
Uncertain choices in the smart paradigm

- Ending *i* like *rivi* - *rivin* instead of *kivi* - *kiven*.
- Ending *e* like *perhe* - *perheen* instead of *nukke* - *nuken*.
- Ending *s* like *pakkaus* - *pakkauksen* instead of *rakkaus* - *rakkauden*.
- Ending *a* like *rikka* - *rikoja* instead of *mansikka* - *mansikoita*.
- Grade alternation like *auto* - *oudon* instead of *auto* - *auton*.

These choices are based on statistics on paradigm frequencies.
Correcting uncertain choices

Tradition in Finnish: genitive singular, e.g. kivi - kiven, nukke - nukken, auto - auton.

But actually we get more distinction with the partitive plural:

- kivi - kiviä vs. rivi - rivejä
- nukke - nukkeja vs. perhe - perheitä
- mansikka - mansikoita vs. rikka - rikkoja

We miss auto: auto - autoja, outo - outoja.
Adding forms to noun paradigms

First: nominative singular
Second: partitive plural
Third: genitive singular
Fourth: partitive singular
For those interested to try it out

Inflectional morphology implementations for 15 languages are available from
digitalgrammars.com/gf/lib/resource/

If you have GF installed, go to the resource directory and start GF:

```
% cd GF/lib/resource/
% gf
> import -retain finnish/ParadigmsFin.gf
> cc mkN "rivi"
> cc mkN "kivi" "kiviä"
```
Bootstrapping a lexicon
Lexicon construction

Algorithm:

1. write with a list of nominative singular nouns
2. apply mkN(1) to generate partitive plurals
3. inspect the results, and change wrong partitive plurals
4. apply mkN(2) to generate genitive singulars
5. inspect the results, and change wrong genitive singulars
6. apply mkN(3) to generate the rest of the ten characteristic forms
7. inspect the results, change wrong forms
8. apply mkN(10) to generate correct forms
Phase 1

meri
sade
nainen
kivi
rivi
tohtori
apina
kulki ja
kukka
auto
rakkaus
Phase 2

meri (merejä >> meriä)
sade sateita
nainen naisia
kivi (kivejä >> kiviä)
rivejä
tohtori (tohtoreja >> tohtoreita)
apina (apinoja >> apinoita)
kukkia kukkia
auto autoja
rakkaus rakkausia

For reasons explained in Section 11, it is enough to pay attention to those nouns that end with an i, as well as 3-syllabic nouns ending with an a or ä, to produce a "2-form gold standard". In the above list, five words are manually changed.
The 2-form gold standard is processed with the 2-place noun constructor, to produce a 3-form list; now, the genitive singular is added. In this case, we mostly have to change some 2-syllabic words that don’t have expected consonant gradation, as well as nouns ending with *us* but inflected like *rakkaus* ("love") rather than *pakkaus* ("package").
Phase 3

meri meriä meren
sade sateita sateen
nainen naisia naisen
kivi kiviä kiven
rivi rivejä rivin
tohtori tohtoreita tohtorin
apina apinoita apinan
kulkija kulkijoita kulkikan
kukka kikkia kukan
auto autoja (audon >> auton)
rakkaus rakkausia (rakauksen >> rakkauden)
An extra phase

The partitive singular is deviant for some words in the \( i-e \) paradigm:

meri       meriä       meren (mertä >>> merta)

But this is just a small, limited set of words, which can be treated in a separate lexicon.
How much work is needed

Based on paradigm frequencies, to build a lexicon from 100 lemmas, requires

- check 30 partitive plural forms
- change 15 partitive plural forms
- check 50 genitive singular forms
- change 5 genitive singular forms
- change the whole inflection of 2 words (18 forms)
- altogether, read 80 forms and change 38 of these
How much time is needed

Assumptions:

- processing 100 words in GF: 0.4s
- reading a word form: 5s
- changing a word form: 20s

Lexicon of 100 lemmas: \((5 \times 0.4 + 80 \times 5 + 38 \times 20)\)s = 16 min.

One working day: lexicon of 3,000 lemmas
Evaluation of the smart paradigms

Given: gold standard showing 10 forms of each lemma

For $n = 1, 2, 3, 4$ do:

1. take the subset of $n$ forms for each lemma
2. apply $mkN(n)$ to produce all 10 forms
3. compare with the gold standard with $\text{diff} \mid \text{wc}$
4. obtain the number of lemmas that get wrong
First experiment

100 random nouns from

- *Aino*, a children’s book
- *Duodecim*, a scientific journal in medicine
- *Swadesh*, the 207-word list of "basic words"
- *Dictionary*, a medium-size English-Finnish dictionary

Errors:

<table>
<thead>
<tr>
<th>args</th>
<th>Aino</th>
<th>Duodecim</th>
<th>Swadesh</th>
<th>Dictionary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>16</td>
<td>31</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6</td>
<td>15</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
First experiment: conclusions

- For 80% of nouns, the inflection is correctly inferred from just one form (the nominative singular).
- For 90% of words, it is enough to have one more form (the partitive plural).
- Adding the genitive and partitive singular gets all nouns right, except for a fixed set of nouns that can be given in advance.
Second experiment

*KOTUS*, freely available electronic word list from *Kotimaisten kielten tutkimuskeskus* ("Research Centre for Domestic Languages").

*KOTUS* uses 50 noun paradigms, to annotate the lemmas in the word list.

We implemented the *KOTUS* paradigms in *GF* to create a gold standard of 27,680 nouns, which excluded

- compounds

- *plurale tantum* words
KOTUS results

<table>
<thead>
<tr>
<th>args</th>
<th>KOTUS #</th>
<th>KOTUS %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4993</td>
<td>18.0</td>
</tr>
<tr>
<td>2</td>
<td>1062</td>
<td>3.8</td>
</tr>
<tr>
<td>3</td>
<td>792</td>
<td>2.9</td>
</tr>
<tr>
<td>4</td>
<td>789</td>
<td>2.9</td>
</tr>
</tbody>
</table>

The insignificant drop between 3 and 4 suggests that the singular partitive should rather be treated in an irregularity lexicon.
KOTUS with genitive singular as second form

<table>
<thead>
<tr>
<th>args</th>
<th>KOTUS #</th>
<th>KOTUS %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4993</td>
<td>18.0</td>
</tr>
<tr>
<td>2</td>
<td>3597</td>
<td>13.0</td>
</tr>
<tr>
<td>3</td>
<td>792</td>
<td>2.9</td>
</tr>
<tr>
<td>4</td>
<td>789</td>
<td>2.9</td>
</tr>
</tbody>
</table>

This confirms that the plural partitive is better, by 9 %-units.
How predictable is Finnish morphology?

We compute the **average number of forms** needed to identify the inflection of a Finnish noun in the KOTUS list.

We assume, cautiously, that all words we fail to predict with 3 arguments need 10 forms.

We get

\[
\frac{(792 \times 10 + (1062 - 792) \times 3 + (4993 - 1062) \times 2 + (27680 - 4993))}{27680} = 1.42
\]
Irregularity lexicon

A finite list (a few hundreds) of irregular words.

If we assume that these words only require 1 form, we get 1.16 forms in average.

Which of the figures 1.42 and 1.16 should be used?
The difficult words in KOTUS

Old irregular words, no doubt: *kevät, mies, meri,...*

New load words, where the orthography doesn’t give pronunciation:

- *brie* ("brie cheese") looks like *tie* but sounds like *pii*
- *calvados* ends with an *s*
- *tournedos* ends with an *o*

The latter kind dominates, and is moreover productive.

Hence 1.42 is a more proper figure.

**TODO:** smarter paradigms for loan words than the worst-case function.
New estimate for lexicon writing

Average 1.42 forms needed to identify the inflection of a noun.
Thus 0.42 forms per lemma must be added.
It takes 20 seconds to produce a form.
Hence

- 100 lemmas require 14 minutes (previous estimate: 16 minutes)
- one working day gives 3,480 lemmas (previous: 3,000 lemmas)
Verbs

Verbs have more forms than nouns, but they are more predictable:

<table>
<thead>
<tr>
<th>args</th>
<th>Swadesh</th>
<th>Dictionary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

More than 90% from one form.

Closed set of irregular verbs: nähdä ("see"), seistä ("stand"),...

Load verbs must attach suffixes that clearly identify the paradigm:

- *chattailla* ("to chat on the internet")
- *mailata* ("to send an email")

Selecting the suffix can be an interesting problem for derivational morphology.
Other languages
The GF Resource Grammar Library

Complete inflectional morphology

- types for all forms of open classes (nouns, adjective, verbs)
- worst-case functions for open classes
- smart paradigms
- irregularity lexicon
- lexicon of closed-class words (pronouns, determiners, etc)

In Versions 1.4 and 1.5, we have 15 languages (* = without smart paradigms, 3 languages): *Arabic, *Bulgarian, Catalan, Danish, English, Finnish, French, German, Italian, Hindi, Latin, Norwegian, *Russian, Spanish, Swedish.
Source code for morphology

<table>
<thead>
<tr>
<th>language</th>
<th>lines</th>
<th>remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>2012</td>
<td>unfinished</td>
</tr>
<tr>
<td>Bulgarian</td>
<td>2384</td>
<td>-</td>
</tr>
<tr>
<td>Catalan</td>
<td>7984</td>
<td>generated Besch</td>
</tr>
<tr>
<td>Danish</td>
<td>1267</td>
<td>incl. shared Scand 385</td>
</tr>
<tr>
<td>English</td>
<td>1164</td>
<td>-</td>
</tr>
<tr>
<td>Finnish</td>
<td>1792</td>
<td>no Irreg</td>
</tr>
<tr>
<td>French</td>
<td>2793</td>
<td>incl. shared Romance 514</td>
</tr>
<tr>
<td>German</td>
<td>1271</td>
<td>-</td>
</tr>
<tr>
<td>Italian</td>
<td>7422</td>
<td>generated Besch</td>
</tr>
<tr>
<td>Hindi</td>
<td>496</td>
<td>no Structural, Irreg</td>
</tr>
<tr>
<td>Latin</td>
<td>635</td>
<td>no Structural, Irreg</td>
</tr>
<tr>
<td>Norwegian</td>
<td>1259</td>
<td>incl. shared Scand 385</td>
</tr>
<tr>
<td>Russian</td>
<td>2025</td>
<td>no Irreg</td>
</tr>
<tr>
<td>Spanish</td>
<td>79667</td>
<td>generated Besch, Irreg</td>
</tr>
<tr>
<td>Swedish</td>
<td>1423</td>
<td>incl. shared Scand 385</td>
</tr>
</tbody>
</table>
Modules Besch Irreg Morpho Paradigms Res Common Structural.
## Forms per lemma in Lexicon (rough estimate)

<table>
<thead>
<tr>
<th>language</th>
<th>forms</th>
<th>per lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>1412</td>
<td>4.03</td>
</tr>
<tr>
<td>Bulgarian</td>
<td>625</td>
<td>1.79</td>
</tr>
<tr>
<td>Catalan</td>
<td>560</td>
<td>1.60</td>
</tr>
<tr>
<td>Danish</td>
<td>709</td>
<td>2.03</td>
</tr>
<tr>
<td>English</td>
<td>492</td>
<td>1.41</td>
</tr>
<tr>
<td>Finnish</td>
<td>743</td>
<td>2.12</td>
</tr>
<tr>
<td>French</td>
<td>504</td>
<td>1.44</td>
</tr>
<tr>
<td>German</td>
<td>763</td>
<td>2.18</td>
</tr>
<tr>
<td>Italian</td>
<td>433</td>
<td>1.24</td>
</tr>
<tr>
<td>Norwegian</td>
<td>723</td>
<td>2.07</td>
</tr>
<tr>
<td>Russian</td>
<td>1425</td>
<td>4.07</td>
</tr>
<tr>
<td>Spanish</td>
<td>557</td>
<td>1.59</td>
</tr>
<tr>
<td>Swedish</td>
<td>793</td>
<td>2.27</td>
</tr>
</tbody>
</table>

Method: `wc Lexicon ; let forms = words - 5*384 in (words, words/350)`