[*Proceedings of a Workshop on Machine Translation*, July 1990, UMIST]

TWO LEVEL MORPHOLOGY IN A UNIFICATION BASED FORMALISM

Vito Pirrelli
University of Salford

1 Two level Morphology: theoretical underpinnings

In the literature one traditionally refers to a Rearrangement Component (RRC) as a set of morphonological rules whose intended usage, in the framework of the Lexicalist Hypothesis put forward by Aronoff in the late seventies, is to provide a modular and 'clean' account of those spelling changes which, otherwise, would be bound to be accounted for directly by a Word Formation Component.

An RRC can 'neatly' handle:

- those cases where the same morphological process triggers different spelling changes:
  - English
    * employ + -ee → employee
    * nominate + -ee → nominee
    * immerse + -tion → immersion
    * subvert + -tion → subversion
    * conceive + -tion → conception
  - Italian:
    * interpretare + -zione → interpretazione
    * espellere + -zione → espulsione
    * esplodere + -zione → esplosione
    * riscuotere + -zione → riscossione

- *identical spelling changes* related to different *morphological processes* of the same nature.
  - English Inflection
    * third singular person of the present tense:
      try + s → tries
    * plural nouns
      spy + s → spies

- *identical spelling changes* related to different *morphological processes* of a different nature.
  - English
    * derivation
      love + er → lover

* inflection
  pale + er → paler

— Italian

* composition
  semi + interrato → seminterrato (en. basement)
* derivation
  stori + ico → storico (en. historical)
* inflection
  ciliegi + i → ciliegi (en. cherry trees)

2 Computational characteristics of Koskenniemi's (1983) Two-level Model of morphological processing as concretely embodied in the KIMMO system of Karttunen (1983)

- spelling changes rules that typically go along with affixation and inflection are encoded in a FINITE STATE TRANSDUCER COMPONENT.
- roots and affixes are listed in a dictionary component along with their co-occurence restrictions (e.g. '-tion' is attached only to verbs to form nouns and so on and so forth)

Formally speaking a specific spelling change process amounts to a constraint on the correspondence between a lexical and surface string:

an example of 'Y-change':

```
surface string      S P I E S
                    - - - - -
                    | | | | |
lexical string      S P Y + S
```

Spelling changes

Y-change controls the occurrence of the lexical/surface pairs y/y and y/i: lexical 'y' must correspond to surface 'i' when it occurs before lexical '+s' which will itself come out as surface 'es' because of other constraints.

Each such constraint is implemented as a particular state in a finite-state transducer with two scanning heads that move together along the lexical and the surface strings.

The dictionary

The dictionary component of the KIMMO system is divided into sections called lexicons, which are all ultimately reachable from a distinguished ROOT LEXICON.

Co-occurrence restrictions involve listing a set of continuation lexicons for each entry.

3 Two level Morphology using a PATR-like unification grammar: Bear's Implementation

Spelling changes.

They are expressed through constraints between surface and underlying strings (lexical strings) which are an outgrowth of those of Karttunen. A built-in default mechanism maps a character into itself and a diacritic (e.g. '+') into the empty string. Exceptions are treated by explicit rules.

Co-occurrence restrictions.

Morphosyntactic rules are expressed via rules. Rules state how two dags (feature structures) can be combined to form a third. They can also be interpreted as specifying how a given dag can be decomposed into two others (rules are always binary).

DRAWBACKS:

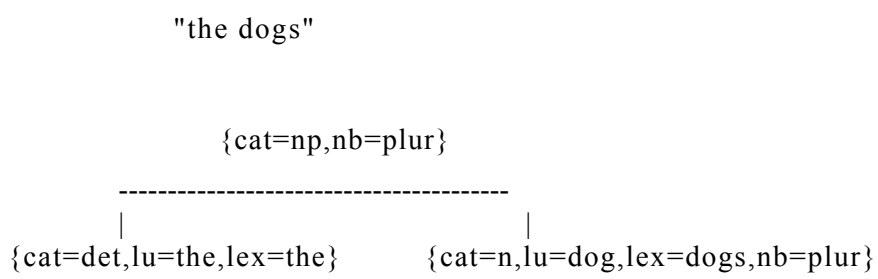
Both 2 and 3 'guess something' (which means they change a character into another character, or a state into another one) choose a possibility and arrange for rejection if the tentative mapping does not hold true.

This amounts to massive backtracking to the last erroneous choice point, and a display of brute force research.

4 Two level Morphology in Eurotra: a sample implementation

4.1 Unification in Eurotra

Eurotra tree notation



Eurotra bracketing notation

{cat=np,nb=plur}[{cat=det,lu=the,lex=the},{cat=n,lu=dog,lex=dogs,nb=plur}].

1. strings are associated with feature-value based informational domain. For each node in a tree structure each feature can get ONE and only ONE value (as in the mathematical definition of 'function'), features as functions.
2. substrings are combined in more complex structures (trees) on the basis of immediate dominance and linear precedence rules.
3. the association process will be non-destructive, so that application of any given rule may create new structure, but will never erase any old information (in our framework that is true only within a generator).

4. set Union:

{a,b,c} U {a.d.f} → {a,b,c,d,f}.

5. Unification with feature bundles. (Set Union).

{lex=the,infl=invariant} U {lex=the,lu=the,cat=det} →
{lex=the,lu=the,cat=det,infl=invariant}.

{lex=the} U {lex=dog,lu=dog,cat=n,nb=sing} → FAILURE

{lex=the,lex=dog} is NOT a well formed feature structure.

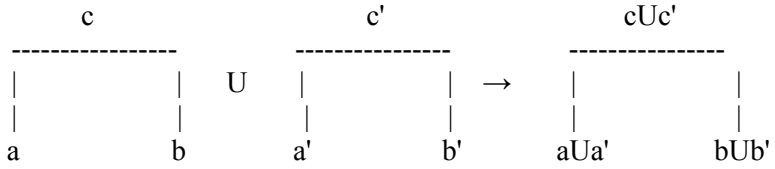
6. Strict Unification (or Mapping). Proper inclusion.

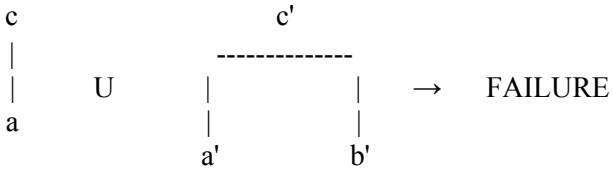
{lex=the} sU {lex=the,lu=the,cat=det} → {lex=det,lu=the,cat=det}.

{lex=dog,nb=sing} sU {lex=dog,lu=dog,cat=n} → FAILURE.

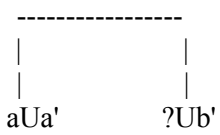
Neither of the two previous sets is a proper subset of the other.

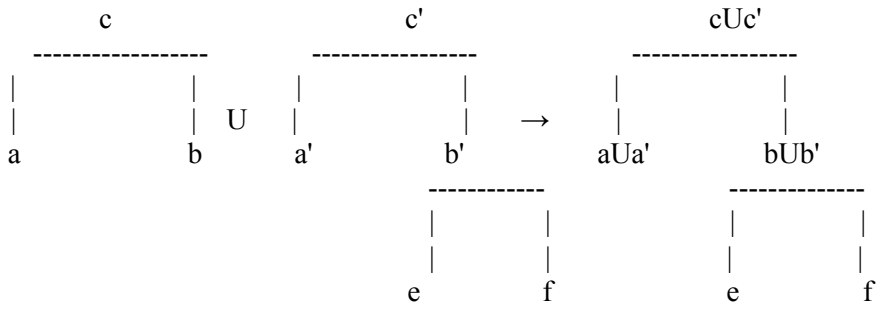
7. Mapping between tree structures





the two structures are not isomorphic:





We can consider a leaf (a simple feature structure) as a degenerate case of a tree (a tree of depth 0).

Unlike Unification between feature structures, mapping between trees is defined via recursive structural correspondences; the recursion is on the depth of the trees involved:

- if one of the two structures is a leaf (feature structure), they unify if the leaf unifies with the top node of the other tree structure.
- if they are of depth 1 they unify when:
  - — they are isomorphic (for each branch in a tree there is one and only one branch in the other tree with the same structural position), and
  - — their leaves unify.
- two structure trees of depth > 1 unify if their sub-structures unify.

4.2 Implementing a 'constraint propagation model' (Barton, Berwick, Ristad 87)

Interaction between spelling changes rules and dictionary.

A flow chart of morphological processing:

```
string  →    ENT    →    EMS   →  ECS   → ERS → IS.

       ----------------------
       Two level Morphology

        |                |
        |                |

        --------------- >
       spelling change rules

ENT dictionary      EMS dictionary

 (characters)       (stems and affixes)
```

Spelling changes rules and dictionary items are not procedurally linked in the way they are in the aforementioned frameworks.

We have to identify 'probable' spelling changes beforehand, waiting for the next level dictionary to confirm or reject the analysis provided by the previous level rules.

Solution: the system does not turn characters into anything else (which corresponds to the change of state in the KIMMO implementation).

Our strategy is to delay the assignment of definite features to ambiguous strings of characters, until ultimately features will be grounded in the dictionary information.

The system does not 'guess' any correspondence (which in case of failure would require backtracking) but inject provisional features (when a particular context shows up), which the lexicon is expected to confirm, unifying with them, or reject if unification between strings and lexicon fails.

All the possible 'changings' are taken into account simultaneously, between ENT and EMS.

Ambiguous analyses are compacted in a single feature bundle.

They will be displayed as different output objects if and only if the lexicon can unify with them (in Eurotra terms, 'consolidate them') in more than one way. Therefore no backtracking is required.

AN EXAMPLE

```
Input to ENT:

a)
                              string
-----------------------------|-----------------------
char char char char char char char char char char char
  I   m    p    r    o    p    r    i    e    t    a


rules at ENT


in_2 * {type=word} [ {type=char,lex=i},{map=n/type=char,lex=m},
({type=char,lex=p};{type=char,lex=b}), *{type=char}].


ieta = {type=word} [*{type=char},{type=char,lex=i},
                    {map=i/type=char,lex=e},
                    {type=char,lex=t},{type=char,lex=a}] .


Input to EMS:

b)

               word
--------------|------------------
i  m  p  r  o  p  r  i  e  t  a
i yes p  r  o  p  r  i yes t  a
    n                  i
    m                  e
```

In b) different values are shown. The first row stands for the lu values of each character. The second one for the lex values of all the characters which have been left unchanged by the RR component.

'yes' under 'm' and 'e' indicates that some changing has taken place. The 'content' of this changing is expressed by the values of third row. 'm' can be mapped onto 'n'; 'e' onto 'i'. Last we have the lex values for the 'affected' characters. As a second step we input b) to ems. The Lexicon of EMS is made out of structure building rules which are character based.

Therefore:

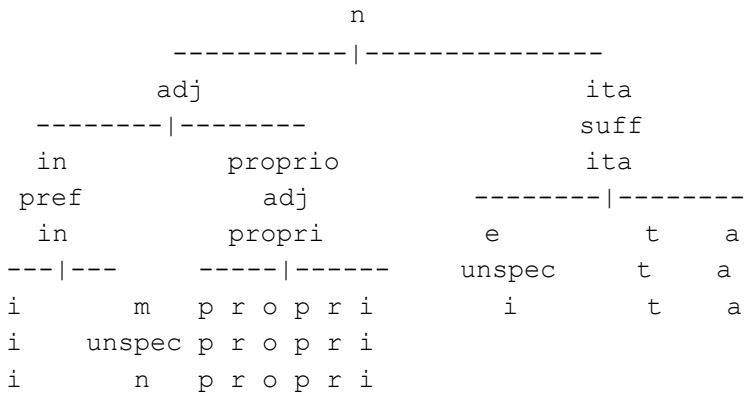
'lex' stands for a superficial character; 'lu' for an unaffected underlying character; 'map' for a potential underlying character.

dictionary at EMS:

```
proprio= {cat=adj, lu=proprio, lex=propri,type=word3}[
{type=char,lu=p},
{type=char,lu=r},
{type=char,lu=o},
{type=char,lu=p},
{type=char,lu=r},
{type=char,lu=i}].


in={cat=pref, type=morph, lu=in, subcat=adj, attcat=adj, lex=in} [
{type=char,lu=i},
{type=char,map=n}].

ita={cat=suff, type=morph, lu=ita, subcat=n, attcat=adj, lex=ita} [
{type=char,map=i},
{type=char,lu=t},
{type=char,lu=a}] .
```

Some of the characters of an EMS dictionary entry unify on the basis of the 'map' value instead of the 'lu' one. This happens only for those characters which can be affected by spelling changes.

Output of EMS:



At EMS 'lex' is assigned the value 'unspec' (mnemonic for unspecified), whenever a change is proposed at ENT (and a 'map' value is introduced).

Morphosyntactic constraints are expressed in the same formalism:

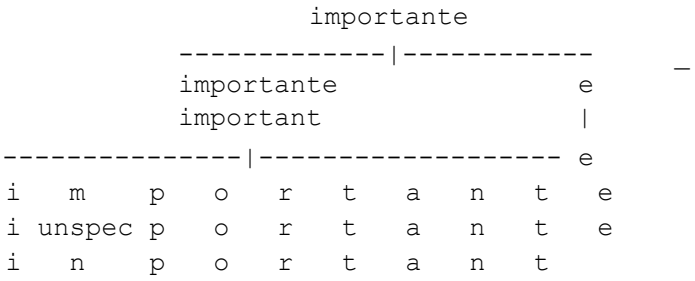
```
inflection= {type=word,cat=Y,number=X,infl=yes,lu=L}
[{cat=Y,type=word,lu=L} ,^{type=char,map=nil,change=yes},
{cat=infl, type=morph, number=X, attcat=Y} ] .

suffixation= {type=word,cat=X, der=yes}
[{cat=Y,type=word} , {cat=suff,type=morph,subcat=X, attcat=Y} ].

prefixation={cat=X,der=yes,type=word}
[{cat=pref,type=morph,subcat=X,attcat=Y},{cat=Y,type=word}].
```

AN AMBIGUOUS CASE

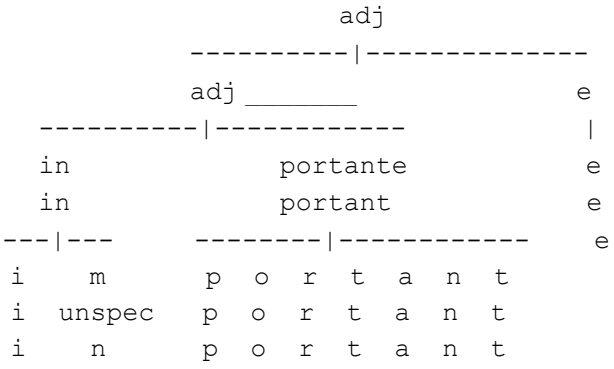
Input to EMS:

```
d)


            word
--------------|-------------
i  m  p  o  r  t  a  n  t  e
i  n  p  o  r  t  a  n  t  e
   m

dictionary entries at EMS:

importante= {cat=adj, lu=importante, lex=important,type=word} [
{type=char,lu=i},
{type=char,lu=m},
{type=char,lu=p},
{type=char,lu=o},
{type=char,lu=r},
{type=char,lu=t},
{type=char,lu=a},
{type=char,lu=n},
{type=char,lu=t}] .


portante= {cat=adj, lu=portante, lex=portant,type=word} [
{type=char,lu=p},
{type=char,lu=o},
{type=char,lu=r},
{type=char,lu=t},
{type=char,lu=a>,
{type=char,lu=n},
{type=char,lu=t}].
```

Output of EMS:



e)



5 Conclusions

This strategy replaces powerful computational machinery (backtracking and brute force search) with a more limited form of inference. Passing from one level to the other the system fleshes out a central skeleton of feature bundles via unification based, context dependent rules.

In a modular architecture like Eurotra other features can be left unspecified until information from other components (e.g. syntactic information like agreement) is provided to fill the unspecified slot.

6 References

Barton, Berwick, Ristad, 1987: Computational Complexity and Natural Language. MIT press. Cambridge Massachusetts.

Bear, J., 1988: Generation and Recognition of inflectional morphology. Proceedings of the Vienna workshop on Knowledge based language processing.

Karttunen, L., 1983: A two level morphology analyzer. Texas Linguistic Forum 22:165-186.

Koskenniemi, K., 1983: Two level morphology: a general computational model for word form recognition and production. University of Helsinki, Publication 11.