Chunking: A full example

En français

This will be a full example of chunking, which we build from the ground up.

We will look at Esperanto <-> English and try to translate the sentence "La libro estas blua" to "The book is blue".

Contents

Overview

First a little overview of how 3-stage transfer normally works:

- Transfer stage: Words are translated using the bidix and categorized and put into chunks (in the .t1x file). Here the tags in the words can also be added, removed or made into 'pointers' that points to the tags in the enclosing chunk.
- Interchunk stage: Chunks are reordered, combined and split and chunk tags changed (in the .t2x file)
- Postchunk stage: The words in the chunks are restored (in the .t3x file)

If we look at how "The blue book is good" goes throgh the system, we have just before transfer:

```
^The<det><def><sp>$ ^blue<adj>$ ^book<n><sg>$ ^be<vbser><pres><p3><sg>$ ^good<adj><sint>$
```

which is transferred to Esperanto and chunked into

```
^det_adj_nom<SN><sg><nom>{^La<det><def><2><3>$ ^blua<adj><2><3>$ ^libro<n><2><3>$}$
^ser<SV><pres><p3><sg>{^esti<vbser><pres>$}$
^adj<SN><sg>{^bona<adj><sg><nom>$}$
```

Here 'det_adj_nom' is the name of the chunk and <SN><sg><nom> the chunk's tags. The content of the chunk is {^La<det><def><2><3>$ ^blua<adj><2><3>$ ^libro<n><2><3>$} where the <2> and <3> are pointers to the chunk's tag (<sg> and <nom> respectively). This allows us to change the values at chunk level later on, if necessary.

In this simple case nothing happens at the interchunk stage. After the postchunk stage it looks like:

```
^La<det><def><sg><nom>$ ^blua<adj><sg><nom>$ ^libro<n><sg><nom>$ ^esti<vbser><pres>$ ^bona<adj><sg><nom>$
```

which becomes "La blua libro estas bona".

Starting from the ground

Now we will try the same sentence Esperanto -> English, but with more or less empty t1x, t2x and t3x files. After tagger disambiguation "La blua libro estas bona" is:

```
^La<det><def><sp>$ ^blua<adj><sg><nom>$ ^libro<n><sg><nom>$
^esti<vbser><pres>$
^bona<adj><sg><nom>$
```

Without any rules the result will just be that each word gets a 'default' chunk:

```
^default{^The<det><def><sp>$}$ ^default{^blue<adj><sg><nom>$}$ ^default{^book<n><sg><nom>$}$
^default{^be<vbser><pres>$}$
^default{^good<adj><sint><sg><nom>$}$
```

Now let's define some rules. To begin with we will define some basic one-word rules à la Apertium New Language Pair HOWTO#Transfer rules.

The determiner

First for the determiner. First we define the category 'c_det' which contains all words marked as <det> and <det><other tags>. Then we define attributes for number and kind of determiner.

```
  <def-cat n="c_det">
     <cat-item tags="det"/>
     <cat-item tags="det.*"/>
  </def-cat>

...
  <def-attr n="a_nbr">
     <attr-item tags="sg"/>
     <attr-item tags="sp"/>
     <attr-item tags="pl"/>
  </def-attr>

  <def-attr n="a_det">
     <attr-item tags="det.def"/>
     <attr-item tags="det.ind"/>
     <attr-item tags="det.pos"/>
     <attr-item tags="det.qnt"/>
  </def-attr>
...
  <rule>
```

```
      <pattern>
        <pattern-item n="c_det"/>
      </pattern>
      <action>
        <out>
          <chunk name="det" case="caseFirstWord">
            <tags>
              <tag><lit-tag v="SN"/></tag>
              <tag><clip pos="1" side="tl" part="a_nbr"/></tag>
            </tags>
            <lu>
              <clip pos="1" side="tl" part="lem"/>
              <clip pos="1" side="tl" part="a_det"/>
              <clip pos="1" side="sl" part="a_nbr" link-to="2"/>
            </lu>
          </chunk>
        </out>
      </action>
    </rule>
```

The rule looks for category c_det (anything starting with <det>) and therefore recognizes the ^La<det><def><sp>$ and outputs

```
^det<SN><sp>{^The<det><def><2>$}$
```

"outside", on the chunk is put first attribute <SN> and then the number attribute. Inside the chunk (in a 'lexical unit') is put the translated lemma (the bidix has la -> the), the determiner attributes (<det><def>) and a pointer to tag 2 (the part="a_nbr" must be there but is ignored in chunking mode).

The `case` attribute of the chunk has the value `caseFirstWord`. If the determiner is the first word in the sentence, the lemma of the chunk will be output as "Det", otherwise, it will be "det". Other possible values of `case` are `variableCase` and `case0therWord`.

> Does anyone know what these other values involve, and how inconsistencies are resolved among `<lu>` case and chunk case? --unhammer 14:13, 16 February 2010 (UTC)

Adjectives

Here we have ^blua<adj><sg><nom>$ and ^bona<adj><sg><nom>$ which we - if they stand alone (that is, we are still at the one word = one chunk level) - would like to get chunked as

```
^adj<SN><sg>{^blue<adj>$}$
^adj<SN><sg>{^good<adj><sint>$}$
```

First we define the category c_adj and then the attributes that can appear on an adjective:

```
  <def-cat n="c_adj">
     <cat-item tags="adj.*"/>
  </def-cat>
...
  <def-attr n="a_adj">
     <attr-item tags="adj"/>
     <attr-item tags="adj.comp"/>
     <attr-item tags="adj.sup"/>
     <attr-item tags="adj.sint"/>
     <attr-item tags="adj.sint.comp"/>
     <attr-item tags="adj.sint.sup"/>
  </def-attr>
```

Here it becomes a rather long list as adjectives can be comparative and superlative (blue, more blue, most blue) and synthetic (see List of symbols#Adjectives). This is taken care of by the bidix so we really don't have to worry about the syntheticness of ^good<adj><sint>$:

```
<e><p><l>blua<s n="adj"/></l><r>blue<s n="adj"/></r></p></e>
<e><p><l>bona<s n="adj"/></l><r>good<s n="adj"/><s n="sint"/></r></p></e>
```

The transfer rule that recognizes adjectives and output chunks called adj with attributes <SN>+number of the adjective (<sg>) and inside the chunk is the lemma (good) with the adjectives attributes (<adj><sint>):

```
  <rule>
     <pattern>
       <pattern-item n="c_adj"/>
     </pattern>
     <action>
       <out>
         <chunk name="adj" case="caseFirstWord">
           <tags>
             <tag><lit-tag v="SN"/></tag>
             <tag><clip pos="1" side="tl" part="a_nbr"/></tag>
           </tags>
           <lu>
             <clip pos="1" side="tl" part="lem"/>
             <clip pos="1" side="tl" part="a_adj"/>
           </lu>
         </chunk>
       </out>
     </action>
  </rule>
```

So before, during and after transfer the adjectives looks like

```
^blua<adj><sg><nom>$
^adj<SN><sg>{^blue<adj>$}$
```

```
^blue<adj>$
```

Where did the <nom> tag go?

You may have noticed that blue in Esperanto actually was ^blua<adj><sg>**<nom>**$. What happened to that?

Well, as it wasn't mentioned in the <out> part of the rule it just disappeared. In Esperanto nouns and adjectives in accusative get an extra -n in the end, so:

- "Mi havas la blua**n** libro**n** (^blua<adj><sg><acc>$ ^libro<n><sg><acc>$) is "I have the blue book" whereas
- "Mi**n** havas la blua libro (^blua<adj><sg><nom>$ ^libro<n><sg><nom>$) is "The blue book has me".

For now we will just ignore the <nom> and <acc> tags.

Nouns

Nouns are more or less the same game as adjectives, apart from the fact that nouns in English are inflected according to number, so its important to keep the <sg> or <pl> on the noun:

```
  <rule>
     <pattern>
       <pattern-item n="c_nom"/>
     </pattern>
     <action>
       <out>
         <chunk name="nom" case="caseFirstWord">
           <tags>
             <tag><lit-tag v="SN"/></tag>
             <tag><clip pos="1" side="tl" part="a_nbr"/></tag>
           </tags>
           <lu>
             <clip pos="1" side="tl" part="lem"/>
             <clip pos="1" side="tl" part="a_nom"/>
             <clip pos="1" side="tl" part="a_nbr" link-to="2"/>
           </lu>
         </chunk>
       </out>
     </action>
  </rule>
```

Before, during and after transfer the nouns looks like

```
^libro<n><sg><nom>$
^nom<SN><sg>{^book<n><2>$}$
^book<n><sg>$
```

Handling of multiwords with inner inflection

The above works fine, but unfortunately the world is not that simple: Sometimes several words in one language corresponds to some words in the other language, and therefore we have to add an extra twist when we output chunks. Please don't worry if you dont understand all in this section, the conclusion is simple and you can use it even if you dont follow the arguments here (who says I understand it myself? ;-).

For example Esperanto 'parlamentano' is 'member of parliament'.

So in the eo.dix is:

```
<e><p><l>parlamentano</l><r>parlamentano<s n="n"/><s n="m"/><s n="sg"/><s n="nom"/></r></p></e>
```

In the bidix is:

```
<e><p><l>parlamentano<s n="n"/><s n="m"/></l><r>member<g><b/>of<b/>parliament</g><s n="n"/></r></p></e>
```

In the en.dix is:

```
<e><i>member</i><par n="house__n"/><p><l><b/>of<b/>parliament</l><r><g><b/>of<b/>parliament</g></r></p></e>
```

Using the above noun rule we get

```
parlamentano
^parlamentano<n><m><sg><nom>$
^nom<SN><sg>{^member# of parliament<n><2>$}$
^member# of parliament<n><sg>$
#member# of parliament
```

compare this with a singleword

```
libro
^libro<n><sg><nom>$
^nom<SN><sg>{^book<n><2>$}$
^book<n><sg>$
book
```

and you will (?) see (at least if youre smarter than me) that the problem is that its really the 'member' which should be inflected, not the whole multiword. So we must take a look at the output for the noun chunk:

```
            <lu>
              <clip pos="1" side="tl" part="lem"/>
              <clip pos="1" side="tl" part="a_nom"/>
              <clip pos="1" side="tl" part="a_nbr" link-to="2"/>
            </lu>
```

The solution is to split the lemma "member# of parliament" up in its two parts, the **head** word (*lemh* - member) and the **queue** of following words (*lemq* - of parliament):

```
            <lu>
              <clip pos="1" side="tl" part="lemh"/>
              <clip pos="1" side="tl" part="a_nom"/>
              <clip pos="1" side="tl" part="a_nbr" link-to="2"/>
              <clip pos="1" side="tl" part="lemq"/>
            </lu>
```

This gives correctly.

```
parlamentano
^parlamentano/parlamentano<n><m><sg><nom>$
^parlamentano<n><m><sg><nom>$
^nom<SN><sg>{^member<n><2># of parliament$}$
^member<n><sg># of parliament$
member of parliament
```

Lifesaving rule of thumb for dummies

Forget all about it the above and just remember:

- write part="lem**h**" instead of part="lem" when you output the lemma
- always add a part="lemq" in the end of all lexical units you output.

WRONG:

```
            <lu>
              <clip pos="1" side="tl" part="lem"/>
              ...
            </lu>
```

RIGHT:

```
          <lu>
            <clip pos="1" side="tl" part="lemh"/>
            ...
            <clip pos="1" side="tl" part="lemq"/>
          </lu>
```

Grouping multiple words into a chunk

Simply add patterns in the same chunk

Grouping multiple words into a chunk is surprisingly simple once you get the grip of singleword chunks: You just make a pattern that matches a sequence of categories:

```
  <rule>
    <pattern>
      <pattern-item n="c_det"/>
      <pattern-item n="c_nom"/>
    </pattern>
```

now the action should still be to output a single chunk, but this time we choose to copy the number tag from the noun (clip pos="2"), of course:

```
    <action>
      <out>
        <chunk name="det_nom" case="caseFirstWord">
          <tags>
            <tag><lit-tag v="SN"/></tag>
            <tag><clip pos="2" side="tl" part="a_nbr"/></tag>
          </tags>
```

and then we output **two** lexical units, the first copies from the det lemma, the second from the noun lemma:

```
          <lu>
            <clip pos="1" side="tl" part="lem"/>
            <clip pos="1" side="tl" part="a_det"/>
            <clip pos="1" side="sl" part="a_nbr" link-to="2"/>
          </lu>
          <b pos="1"/>
          <lu>
            <clip pos="2" side="tl" part="lem"/>
            <clip pos="2" side="tl" part="a_nom"/>
            <clip pos="2" side="sl" part="a_nbr" link-to="2"/>
          </lu>
        </chunk>
      </out>
    </action>
  </rule>
```

Using this rule gives

```
la libro
^la<det><def><sp>$ ^libro<n><sg><nom>$
^det_nom<SN><sg>{^the<det><def><2>$ ^book<n><2>$}$
^the<det><def><sg>$ ^book<n><sg>$
the book
```

det_adj_nom chunk

```
  <rule>
     <pattern>
       <pattern-item n="c_det"/>
       <pattern-item n="c_adj"/>
       <pattern-item n="c_nom"/>
     </pattern>
     <action>
       <out>
         <chunk name="det_adj_nom" case="caseFirstWord">
           <tags>
             <tag><lit-tag v="SN"/></tag>
             <tag><clip pos="3" side="tl" part="a_nbr"/></tag>
           </tags>
           <lu>
             <clip pos="1" side="tl" part="lem"/>
             <clip pos="1" side="tl" part="a_det"/>
             <clip pos="1" side="sl" part="a_nbr" link-to="2"/>
           </lu>
           <b pos="1"/>
           <lu>
             <clip pos="2" side="tl" part="lem"/>
             <clip pos="2" side="tl" part="a_adj"/>
           </lu>
           <b pos="2"/>
           <lu>
             <clip pos="3" side="tl" part="lem"/>
             <clip pos="3" side="tl" part="a_nom"/>
             <clip pos="3" side="sl" part="a_nbr" link-to="2"/>
           </lu>
         </chunk>
       </out>
     </action>
  </rule>
```

Using this rule gives, not surprisingly

```
la blua libro
^la<det><def><sp>$ ^blua<adj><sg><nom>$ ^libro<n><sg><nom>$
^det_adj_nom<SN><sg>{^the<det><def><2>$ ^blue<adj>$ ^book<n><2>$}$
^the<det><def><sg>$ ^blue<adj>$ ^book<n><sg>$
the blue book
```

Word/chunk reordering

Now that "La libro estas bona" -> "The book is good" works, lets look at how chunk reordering works. In Esperanto you make a sentence into a question by putting "Ĉu" in the start of the sentence: "Ĉu la libro estas bona?" In English the verb needs to come first: "Is the book good?".

- Is there a followup to this post? --Jonasfromseier 18:36, 21 April 2013 (UTC)

See also

- Chunking
- Apertium stream format#Chunks
- Preparing to use apertium-transfer-tools
- English and Esperanto