Hybrid Rule-Based – Example-Based MT: Feeding Apertium with Sub-sentential Translation Units

Felipe Sánchez-Martínez[†] Mikel L. Forcada[†,‡] Andy Way[‡]

† Dept. Llenguatges i Sistemes Informàtics — Universitat d'Alacant, Spain
{fsanchez,mlf}@dlsi.ua.es

‡ School of Computing — Dublin City University, Ireland
{mforcada,away}@computing.dcu.ie

13th November 2009
3rd Workshop on Example-Based Machine Translation

Outline

Motivation

Predictability of rule-based MT (RBMT) systems:

- Lexical and structural selection is consistent
- Errors can be attributed to a particular module
- Eases postedition for dissemination

Usually RBMT systems do not benefit from the postedition effort of professional translators

- Incorporating postedition work is not trivial
- Some RBMT may benefit from the translation units found in translation memories (usually whole sentences)

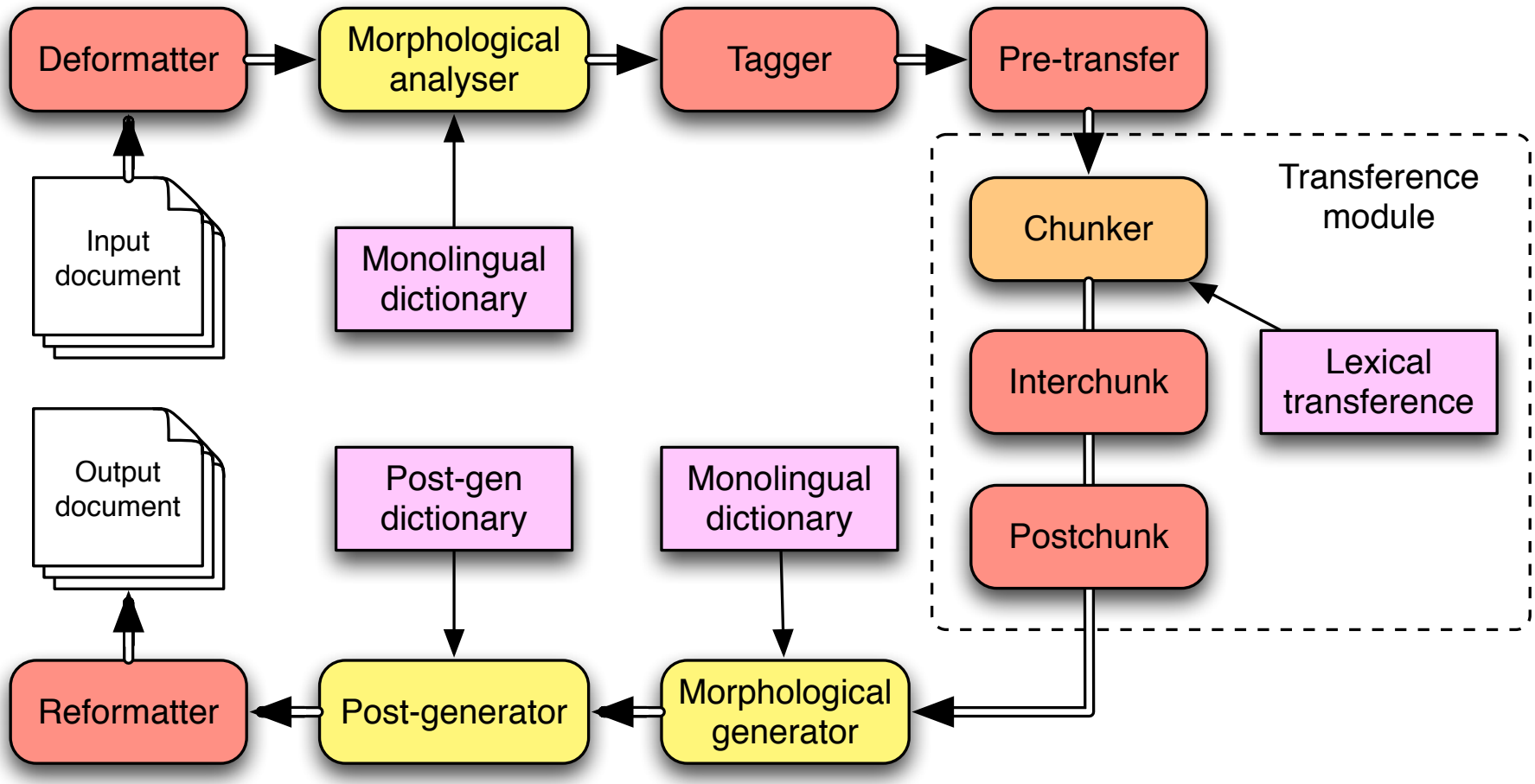
Goal

Integrate sub-sentential translation units into the Apertium free/open-source MT platform

- Sub-sentential translation units are more likely to be re-used than whole sentences

Test the bilingual chunks automatically obtained using the maker-based chunkers and chunk aligners of Matrex

Apertium rule-based MT engine



Example of translation /1

Source text:

Francis' car is broken

De-formatter:

Francis'[ ]car[ ]is broken

Morphological analyser:

^Francis'/Francis<np><ant><m><sg>+'s<gen>$[ ]
^car/car<n><sg>$[ ]
^is/be<vbser><pri><p3><sg>$
^broken/break<vblex><pp>$

Part-of-speech tagger:

^Francis<np><ant><m><sg>$ ^'s<gen>$[ ]
^car<n><sg>$[ ]^be<vbser><pri><p3><sg>$
^break<vblex><pp>$

Apertium: Example of translation /2

Structural transfer (prechunk) + Lexical transfer:

```
^nom<SN><UNDET><m><sg>{^Francis<np><ant><3><4>$}$
^pr<GEN>{}$[ <strong>]
^nom<SN><UNDET><m><sg>{^coche<n><3><4>$}$
[</strong> ] ^be_pp<Vcop><vblex><pri><p3><sg><GD>{
^estar<vblex><3><4><5>$
^romper<vblex><pp><6><5>$}$
```

Structural transfer (interchunk):

```
[<strong>]^nom<SN><PDET><m><sg>{^coche<n><3><4>$}$
[</strong> ]^pr<PREP>{ ^de<pr>$}$
^nom<SN><PDET><m><sg>{ ^Francis<np><ant><3><4>$}$
^be_pp<Vcop><vblex><pri><p3><sg><m>{
^estar<vblex><3><4><5>$
^romper<vblex><pp><6><5>$}$
```

Apertium: Example of translation /3

Structural transfer (postchunk):

[]^el<det><def><m><sg>$ ^coche<n><m><sg>$
[] ^de<pr>$ ^Francis<np><ant><m><sg>$
^estar<vblex><pri><p3><sg>$
^romper<vblex><pp><m><sg>$

Morphological generator and post-generator:

[]el coche[ ]de Francis está roto

De-formatter:

el coche de Francis está roto

Target text:

el coche de Francis está roto

Considerations

Requirements

- Not break the application of structural transfer rules
- Use the longest possible chunks

How

- Introducing chunks delimiters as format information

  `... is [BCH_12_0]the chunk[ECH_12_0] that ...`

- Chunks can be then recognised after the translation

  `... es [BCH_12_0]el segmento[ECH_12_0] que ...`

Side effect

- Format information may be moved around
- Or deleted by some rules (bug)

Translation approach

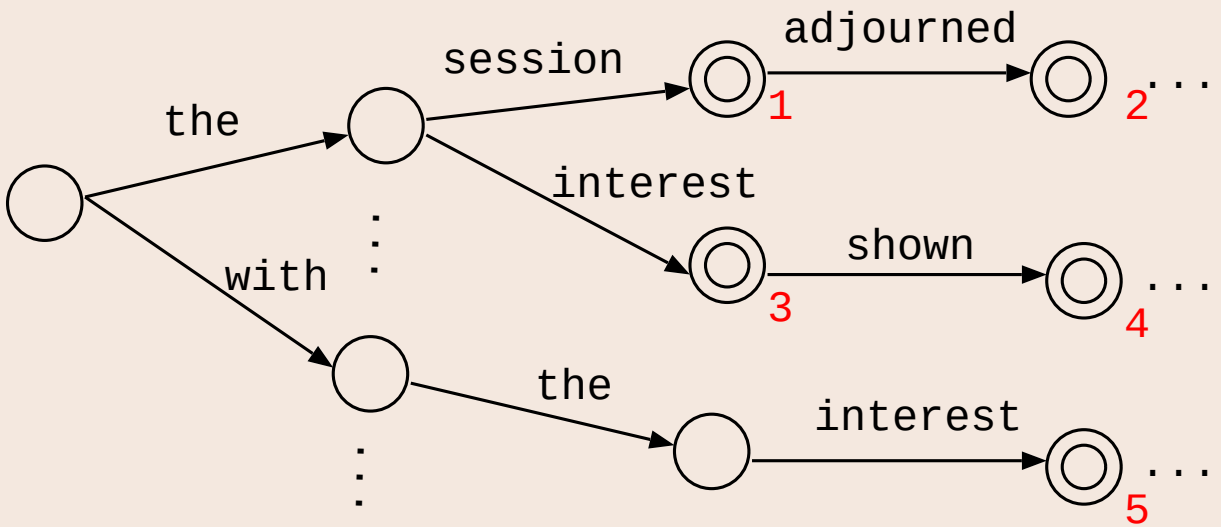
Algorithm

1. apply a dynamic-programming algorithm to compute the best coverage of the input sentence
   - Introduce chunk delimiters as format information
2. translate the input sentence as usual by Apertium
   - Detected chunks are also translated
3. use a language model to choose one of the possible translations for each of the bilingual chunks detected
   - One source-language chunk may have different target-language translations
   - Also consider Apertium translation

Computation of the best coverage /1

Data structure

Store source-language chunks in a trie of strings

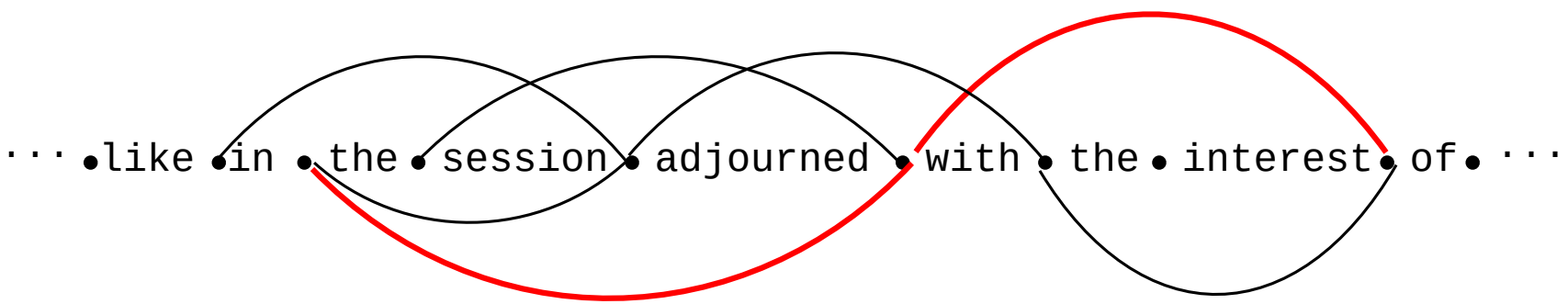


It allows to compute the best coverage in a efficient way

Computation of the best coverage /2

Algorithm

- A set of alive states in the trie is maintained to compute all the possible ways to cover the input sentence
- At each position the best coverage until that position is stored
  - Only the best coverage up to the last / word
- A new search is started at every word
- Is applied to text segments shorter than sentences
  - The best coverage can be retrieved when there are no more alive states



Computation of the best coverage /3

The best coverage

- is the one that uses the least possible number of chunks
  - longest possible chunks
- each not covered word counts like one chunk
- if two coverages use the same number of chunks, the one that uses the most frequent chunks is used

Experimental setup /1

Corpora

- Corpora distributed for the WMT 09 Workshop for MT
- Language pairs: Spanish→English (`es-en`), English→Spanish (`en-es`)
- Linguistic data: `apertium-en-es`; SVN revision 9284

Tools

- Apertium
- Giza++ and Moses to calculate word alignments and lexical probabilities
- SRILM to train 5-gram language models
- Matrex to segment training corpora and to align chunks

Experimental setup /2

Training corpus preprocessing

Max. sentence length: 45 words

Max. word ratio: 1.5 words (mean ration + std. dev.)

Corpus	Sentences	English words	Spanish words
Training	1,187,905	26,983,025	27,951,388
Development	2,050	49,884	52,719
Test	3,027	77,438	80,580

Experimental setup /3

Marker-based bilingual chunks

- Based on the 'Marker Hypothesis'
- Marker words: prepositions, pronouns, determiners, etc.
- Chunks start with a maker word
- Chunks contain at least one non-marker word

Chunks filtering

- There must be at least one word aligned in each side
- Chunks not seen at least θ times are discarded
  - Tested values: $\theta \in [5, 80]$
- Chunks containing punctuation marks and numbers are discarded

Results: BLEU scores

Translation	Apertium		Apertium+chunks		
	dev	test	θ	dev	test
English→Spanish	17.10	18.51	11	17.41	18.94
Spanish→English	17.71	18.81	28	17.91	19.14

- Small improvement, not statistical significant
  - Statistical significance test: bootstrap resampling
- Improvement is larger in the test corpus

Translation	dev	test
English→Spanish	+0.31	+0.43
Spanish→English	+0.20	+0.33

Results: Analysis

Translation	Number of chunks (% words covered)		
	Detected	Finally used	
		All	Apertium
English→Spanish	6,812 (18%)	5,546 (15%)	2,662 (7%)
Spanish→English	6,321 (17%)	5,488 (14%)	2,929 (8%)

- Around half of the chunks finally used are translate the same way as Apertium
- Chunks detected and not used due to chunk delimiters placed in the wrong position

Spanish→English translation

Example

S:	desde hace muchos años un fenómeno misterioso ...
R:	for years , a mysterious phenomenon ...
A:	from does a lot of years a mysterious phenomenon ...
A+C:	for many years a mysterious phenomenon ...
S:	olmert devolvería ... las zonas ocupadas a cambio de la paz
R:	olmert would return ... territories in exchange for peace
A:	olmert it would give back ... zones to change of the peace
A+C:	olmert it would give back ... zones in exchange for peace
S:	pero hay una cosa que nos une :
R:	but there is one thing that connects us :
A:	but there is a thing that joins us :
A+C:	but there is one thing that joins us :

Discussion /1

Novel approach to integrate sub-sentential translation units in Apertium

- Uses of the longest possible chunks and a language model

Approach tested using marker-based chunks

- Small improvement
- Most of the chunks are translated the same way as Apertium

Noise introduced due to how Apertium manages format information

- Some chunks are not applied because chunk delimiters are lost or moved to a wrong position

Discussion /2

Improvement of the computation of the best coverage when two coverages use the same number of chunks:

- Use the bilingual chunk that would produce the most-likely TL translation instead of the most frequent one
- How? Using a language model with gaps

◯ in the session ◯ ◯ with the interest .

Improvement of the bilingual chunks filtering

- Current approach only based on chunks frequency
- Longer chunks are penalised in favour of shorter ones

Thanks for your attention!

Hybrid Rule-Based – Example-Based MT: Feeding Apertium with Sub-sentential Translation Units

Felipe Sánchez-Martínez[†] Mikel L. Forcada[†,‡] Andy Way[‡]

- An open-source implementation is available at `http://sf.net/projects/apertium/files/` package name: `apertium-chunks-mixer`
- More information on the Apertium web page: `http://www.apertium.org`