Today's Topic

• Have partially graded Homework 2 ...

• Assume you have installed tregex as requested last time
• Homework 5 out today
tregex

• Select the PTB directory: TREEBANK_3/parsed/mrg/wsj/
  • Deselect any unwanted files
• The_Wonderful_World_of_Tregex.ppt by Galen Andrew
• Help button
• Browse:
• Adjust Max displayed trees if needed:
Tregex Pattern Syntax and Uses

Using a Tregex pattern, you can find only those trees that match the pattern you’re looking for. The following table shows the symbols that are allowed in the pattern, and below there is more information about using these patterns.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>A &lt; B</td>
<td>A is the only child of A</td>
</tr>
<tr>
<td>A &gt; B</td>
<td>A is the only child of B</td>
</tr>
<tr>
<td>A &lt;&lt; B</td>
<td>A dominates B via an unbroken chain (length &gt; 0) of unary local trees.</td>
</tr>
<tr>
<td>A &gt;&gt;&gt; B</td>
<td>A is dominated by B via an unbroken chain (length &gt; 0) of unary local trees.</td>
</tr>
<tr>
<td>A ++ B</td>
<td>A is a left sister of B (same as S+, for context-free trees)</td>
</tr>
<tr>
<td>A -- B</td>
<td>A is a right sister of B (same as S-, for context-free trees)</td>
</tr>
<tr>
<td>A + B</td>
<td>A is the immediate left sister of B (same as S, for context-free trees)</td>
</tr>
<tr>
<td>A - B</td>
<td>A is the immediate right sister of B (same as S, for context-free trees)</td>
</tr>
<tr>
<td>A .. B</td>
<td>A is a sister of B and precedes B</td>
</tr>
<tr>
<td>A . B</td>
<td>A is a sister of B and follows B</td>
</tr>
<tr>
<td>A .. B</td>
<td>A is a sister of B and immediately precedes B</td>
</tr>
<tr>
<td>A . B</td>
<td>A is a sister of B and immediately follows B</td>
</tr>
<tr>
<td>A &lt;&lt;&lt; B</td>
<td>A dominates B via an unbroken chain of (zero or more) nodes matching description C</td>
</tr>
<tr>
<td>A &lt;&lt;&lt; B</td>
<td>A is dominated by B via an unbroken chain of (zero or more) nodes matching description C</td>
</tr>
<tr>
<td>A + B C</td>
<td>A precedes B via an unbroken chain of (zero or more) nodes matching description C</td>
</tr>
<tr>
<td>A + B C</td>
<td>A follows B via an unbroken chain of (zero or more) nodes matching description C</td>
</tr>
<tr>
<td>A .. B</td>
<td>A is a head of phrase A</td>
</tr>
<tr>
<td>A .. B</td>
<td>A is a head of phrase B</td>
</tr>
<tr>
<td>A .. B</td>
<td>A is the immediate head of phrase A</td>
</tr>
<tr>
<td>A .. B</td>
<td>A is the immediate head of phrase B</td>
</tr>
<tr>
<td>A .. B</td>
<td>A and B are the same node</td>
</tr>
<tr>
<td>A : B</td>
<td>this is a pattern-segmenting operator that places no constraints on the relationship between A and B</td>
</tr>
</tbody>
</table>
tregex

- Help: *tregex expression syntax is non-standard wrt bracketing*

Label descriptions can be literal strings, which much match labels exactly, or regular expressions in regular expression bars: /regex/. Literal string matching proceeds as String equality. In order to prevent ambiguity with other Tregex symbols, only standard "identifiers" are allowed as literals, i.e., strings matching [a-zA-Z][a-zA-Z0-9_]*. If you want to use other symbols, you can do so by using a regular expression instead of a literal string. A disjunctive list of literal strings can be given separated by '|'. The special string '__' (two underscores) can be used to match any node. (WARNING!! Use of the '__' node description may seriously slow down search.) If a label description is preceeded by '@', the label will match any node whose basicCategory matches the description. NB: A single '@' thus scopes over a disjunction specified by '|'.

@NP/VP means things with basic category NP or VP. Label description regular expressions are matched as find(), as in Perl/tgrep; you need to specify ^ or $ to constrain matches.

In a chain of relations, all relations are relative to the first node in the chain. For example, $(s < vp < np)$ means "an S over a VP and also over an NP". If instead what you want is an S above a VP above an NP, you should write "s < (vp < np)".

Nodes can be grouped using parens '(' and ')' as in $s < (np $++ vp)$ to match an S over an NP, where the NP has a VP as a right sister.
tregex

• **Help:** *tregex boolean syntax is also non-standard*

**Boolean relational operators**

Relations can be combined using the '!' operator, and made optional with the '?' operator. Thus \((\text{NP} < \text{NN} \mid < \text{NNS})\) will match an NP node dominating either an NN or an NNS. \((\text{NP} > \text{S} \& \text{S}++ \text{VP})\) matches an NP that is both under an S and has a VP as a right sister.

Relations can be grouped using brackets ']' and '['. So the expression

\[\text{NP} [< \text{NN} \mid < \text{NNS}] \& > \text{S}\]

matches an NP that (1) dominates either an NN or an NNS, and (2) is under an S. Without brackets, \& takes precedence over \|, and equivalent operators are left-associative. Also note that \& is the default combining operator if the operator is omitted in a chain of relations, so that the two patterns are equivalent:

\[(\text{S} < \text{VP} < \text{NP})\]
\[(\text{S} < \text{VP} \& < \text{NP})\]

As another example, \((\text{VP} < \text{VV} \mid < \text{NP} \& \% \text{NP})\) can be written explicitly as \((\text{VP} [< \text{VV} \mid [< \text{NP} \& \% \text{NP} \] \)\)

Relations can be negated with the '!' operator, in which case the expression will match only if there is no node satisfying the relation. For example \((\text{NP} \& \text{NNP})\) matches only NPs not dominating an NNP. Label descriptions can also be negated with '!'': \((\text{NP} !\text{NNP}!\text{NNS})\) matches NPs dominating some node that is not an NNP or an NNS.

Relations can be made optional with the '?' operator. This way the expression will match even if the optional relation is not satisfied. This is useful when used together with node naming (see below).
tregex

• Help

Basic Categories

In order to consider only the "basic category" of a tree label, i.e. to ignore functional tags or other annotations on the label, prefix that node's description with the @ symbol. For example (@NP @/NN.?/) This can only be used for individual nodes; if you want all nodes to use the basic category, it would be more efficient to use a {@link edu.stanford.nlp.trees.TreeNormalizer} to remove functional tags before passing the tree to the TregexPattern.

Segmenting patterns

The ":" operator allows you to segment a pattern into two pieces. This can simplify your pattern writing. For example, the pattern

    S : NP

matches only those S nodes in trees that also have an NP node.
tregex

• $x <, y$, 1st child $y$; $x \leftarrow y$, last child $y$;
• $x \$+ y$, $x$ immediate left sister of $y$

Naming nodes

Nodes can be given names (a.k.a. handles) using '='. A named node will be stored in a map that maps names to nodes so that if a match is found, the node corresponding to the named node can be extracted from the map. For example ($\text{n}\text{p} < \text{n}\text{n}\text{p}=$name) will match an NP dominating an NNP and after a match is found, the map can be queried with the name to retrieve the matched node using TregexMatcher#getNode(Object o) with (String) argument "name" (not "=name"). Note that you are not allowed to name a node that is under the scope of a negation operator (the semantics would be unclear, since you can't store a node that never gets matched to). Trying to do so will cause a ParseException to be thrown. Named nodes can be put within the scope of an optionality operator.

Named nodes that refer back to previous named nodes need not have a node description -- this is known as "backreferencing". In this case, the expression will match only when all instances of the same name get matched to the same tree node. For example: the pattern

```java
(@np \<, (@np $+ (/ / $+ (@np $+ /,/=comma))) \< =comma)
```

matches only an NP dominating exactly the sequence NP , NP ,-- the mother NP cannot have any other daughters. Multiple backreferences are allowed. If the node w/ no node description does not refer to a previously named node, there will be no error, the expression simply will not match anything.

Another way to refer to previously named nodes is with the "link" symbol: '~'. A link is like a backreference, except that instead of having to be equal to the referred node, the current node only has to match the label of the referred to node. A link cannot have a node description, i.e. the '~' symbol must immediately follow a relation symbol.
tregex

- Pattern:
  - (@NP <, (@NP $+ /, / $+ (@NP $+ /,/=comma))) <- =comma)

Key:
- `<`, first child
- `$+`, immediate left sister
- `<-`, last child

From file: /Users/sandiway/research/TREEBANK_3/parsed/mrg/wsj/00/wsj_0003.mrg
tregex

• Help

Variable Groups

If you write a node description using a regular expression, you can assign its matching groups to variable names. If more than one node has a group assigned to the same variable name, then matching will only occur when all such groups capture the same string. This is useful for enforcing coindexation constraints. The syntax is

```regex
/ <regex-stuff> /#<group-number>%<variable-name>
```

For example, the pattern (designed for Penn Treebank trees)

```plaintext
@SBAR < ^WH.*-(\[0-9]+)$#/1%index << (_=empty < (-NONE-/ < ^\*T\*-([0-9]+)$#/1%index))
```

will match only such that the WH- node under the SBAR is coindexed with the trace node that gets the name empty.
Neither Lorillard nor the researchers...
tregex

• Different results from:
  • $@SBAR < /\^WH.*-([0-9]+)$/#1%index << (@NP < (/^-NONE-/ < /\^T\*-(([0-9]+)$/#1%index)\))$
Reason for difference

Example:

WHADVP also possible (not just WHNP)
Treebank Guide

• Parsing guide 1, prsguid1.pdf (318 pages):

Bracketing Guidelines for Treebank II Style
Penn Treebank Project

Principal authors:
Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre
Major contributors:
Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, Britta Schasberger

January 1995

prsguid2.pdf: addendum for the Switchboard corpus
Homework 5

• Usual rules, due next Wednesday (midnight)
• One PDF file to me
Free Relatives

• Taken from Caponigro, I. (proceedings of WECOL 2002).
• Some background:

(1)  
  a.  I appreciate \textsubscript{FR} what you did for me.  
  a’. I appreciate \textsubscript{DP} your help.  
  b. \textsubscript{FR} Who couldn’t sleep enough] felt tired the following morning.  
  b’. \textsubscript{DP} The insomniacs] felt tired the following morning.  
  c. You can’t smoke \textsubscript{FR} where the kids are sleeping.  
  c’. You can’t smoke \textsubscript{DP} there.  
  d. He opened the door \textsubscript{FR} when I was about to knock.  
  d’. He opened the door \textsubscript{DP} then.
Free Relatives

• Free vs. Headed relatives:

(2)  

a. FR: I like \([_{FR} \underline{\text{________}} \text{what you bought}]\).

b. = HR: I like \([_{HR} \underline{\text{the thing(s) which you bought}}] \text{ HEAD}\).

(4)  

a. I like \([_{FR} \underline{\text{what you bought}}]\).

a'. = I like \([_{\text{the thing(s) you bought}}]\).

b. I wonder \([_{\text{wh-Q what you bought}}] \).

b'. = I wonder \([_{\text{which thing(s) you bought}}] \).
9.2.3 Free ("headless") relatives

1. General. A free or headless relative is defined as any relative clause that lacks a head. Free ("headless") relatives are labeled SBAR-NOM.

(PP instead of
(S-NOM (NP-SBJ *))
(VP listening
(PP-CLR to
(SBAR-NOM (WHNP-1 what)
(S (NP-SBJ *T=-155)
(VP is
(PP-LOC-PRD in
(NP his soul))))))))

( (S (SBAR-NOM-SBJ (WHNP-1 what)
(S (NP-SBJ *T=-1)
(VP is
(PP-PRD of
(NP (NP (ADJP much more)
importance)
(PP to
(NP the Colombian economy))
(PP than
(NP (NP the supposed benefits)
(PP of
(NP laundered drug money)))))))))

(VP is
(NP-PRD (NP higher prices)
(PP for
(NP (NP Colombia 's)
legitimate products))))
.
)
Free Relatives

• I asked what he asked

- indirect question interpretation

(S (NP-SBJ I)
  (VP asked
   (SBAR (WHNP-1 what)
    (S (NP-SBJ he)
     (VP asked
      (NP *T*-1))))))

- free-relative interpretation

(S (NP-SBJ I)
  (VP asked
   (SBAR-NOM (WHNP-1 what)
    (S (NP-SBJ he)
     (VP asked
      (NP *T*-1))))))

In the first example (below), the SBAR is bracketed as a clausal complement (in this case, an indirect question) and does not receive the -NOM tag. The sentence can be paraphrased as, “I asked, what did he ask?”

In the second example, the clause is bracketed as a free relative and does receive the -NOM tag. The sentence can be paraphrased as “I asked that which he asked” or “I asked the same question that he asked.”
Free Relatives

SBAR complements of verbs such as *ask*, *tell*, and *know* that can take clausal complements are usually analyzed as clausal complements. Free-relative interpretations of clausal complements of VP happen infrequently, and are either due to a bona-fide free-relative interpretation, or to error.

```
(NP (NP the first)
 (SBAR (WHNP-1 0)
  (S (NP-SBJ *T*-1)
   (VP to
    (VP tell
     (NP him)
    (SBAR (WHNP-154 what)
     (S (NP-SBJ *T*-154)
      (VP is
       (PP-LOC-PRD in
        (NP our minds)))))))))))
```
Homework 5

1. What words or phrases **head** free relatives? List as many as you can think of.

2. Using tregex, devise search strings to build a count/frequency table of the words/phrase that comprises the 775 reported examples

<table>
<thead>
<tr>
<th>Type</th>
<th>Count</th>
<th>Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Ww]hat</td>
<td>500*</td>
<td>65%*</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

*sample numbers only...

3. Submit your tregex search strings and your table