Searching in Discourse-Annnotated Treebanks

Jiří Mírovský
Charles University
Institute of Formal and Applied Linguistics
Prague, November 12, 2018
Searching in Discourse-Annotated Treebanks

Outline

- Prague Dependency Treebank (PDT) & Discourse relations
- Prague Markup Language (PML)
- PML-Tree Query
- PDT and PML-Tree Query
- PDTB and PML-Tree Query
Searching in Discourse-Annotated Treebanks

Outline

- Prague Dependency Treebank (PDT) & Discourse relations
- Prague Markup Language (PML)
- PML-Tree Query
- PDT and PML-Tree Query
- PDTB and PML-Tree Query
Prague Dependency Treebank

- **Czech journalistic texts** from 1990's
- **50 thousand** sentences annotated manually on **several layers**
  - morfological layer (part of speech, case, ...)
  - analytical layer (surface syntax)
  - tectogrammatical layer (deep syntax)
Prague Dependency Treebank
versions and availability

- **PDT 1.0** – published in 2001 (LDC)
- **PDT 2.0** – published in 2006 (LDC)
  - tectogrammatical layer in large scale
- **PDT 2.5** – published in 2011 (Lindat/Clarin, Creative Commons License)
  - multiword expressions ("named entities")
- **PDiT 1.0** – published in 2012 (Lindat/Clarin, ...)
  - discourse relations, bridging anaphora, extended textual coreference
- **PDT 3.0** – published in 2013 (Lindat/Clarin, ...)
- **PDiT 2.0** – published in 2016 (Lindat/Clarin, ...)
  - secondary discourse connectives, further extended coreference
- **PDT 3.5** – published in 2018 (Lindat/Clarin, ...)

ÚFAŁ
Máma [Mom] – Subject
mele [is_mincing] – Predicate
maso [meat] – Object
Máma [Mom] – Actor
mele [is_mincing] – Predicate
maso [meat] – Patient
Asi patnáctkrát jsem jí třísknul o zem, ale sotva jsem ji poškrábal.

[Lit.: I smashed it against the ground about fifteen times but I barely scratched it.]
Občané, kteří v sebeobraně poškodili zdraví útočníka nebo ho dokonce zabili, bývají za své jednání často nespravedlivě stíháni.

[Lit.: Citizens who in self-defence harmed health of the attacker or even killed him, are for their actions often unfairly prosecuted.]
In the whole PDT 3.0 (50th. sentences), there are

- **20,556** discourse relations
  - **6,226** inter-sentential
  - **14,330** intra-sentential
  (plus **83** list structures)
- **95,302** relations of textual coreference
- **23,312** relations of grammatical coreference
- **34,367** bridging relations
 Searching in Discourse-Annotated Treebanks

Outline

- Prague Dependency Treebank (PDT) & Discourse relations
- Prague Markup Language (PML)
- PML-Tree Query
- PDT and PML-Tree Query
- PDTB and PML-Tree Query
PML – Prague Markup Language

PML is a general XML-based format for all kinds of linguistically annotated treebanks.
Prague Markup Language
three components for your data

- **PML-schema**
  - data structure

- **Stylesheet**
  - data appearance

- **Macros**
  - data manipulation
Description of the **structure** of the data

- **types of nodes** in the data (root, node, terminal, non-terminal, ...)
- **relations** among nodes (child relation between non-terminal → non-terminal, non-terminal → terminal, coreference, discourse relations, ...)
- **names** and **types** (and special roles) of **attributes**
- **values** of enumerative attributes
<type name="t-node.type">
  <!-- simplified! -->
  <structure role="#NODE" name="t-node">
    <member as_attribute="1" name="id" role="#ID" required="1">
      <cdata format="ID"/>
    </member>
    <member name="is_generated" type="bool.type"/>
    <member name="t_lemma" required="1">
      <cdata format="any"/>
    </member>
    <member name="functor" required="1">
      <alt type="func.type"/>
    </member>
    <member name="deepord" role="#ORDER" required="1">
      <cdata format="nonNegativeInteger"/>
    </member>
    <member name="discourse" required="0">
      <list ordered="0" type="t-discourse-link.type"/>
    </member>
    ...
  </structure>
</type>
Prague Markup Language
PML-schema

```xml
<type name="t-discourse-link.type"> <!-- simplified! -->
  <structure>
    <member name="target_node.rf" required="0">
      <cdata format="PMLREF"/>
    </member>
    <member name="start_range" required="1"> ... </member>
    <member name="target_range" required="0"> ... </member>
    <member name="discourse_type" type="t-discourse-type.type" required="0"/>
    <member name="a-connectors.rf" required="0">
      <list ordered="0"> <cdata format="PMLREF"/> </list>
    </member>
    <member name="t-connectors.rf" required="0">
      <list ordered="0"> <cdata format="PMLREF"/> </list>
    </member>
    <member name="connective" required="0"> <!-- for searching in PML-TQ only (not in the distributed data) -->
      <cdata format="any"/>
    </member>
    ...
  </structure>
</type>
```
Prague Markup Language
Stylesheet

How to **present** the data to the user

- **attributes** displayed at nodes
- **relations** displayed between nodes
- **shape** of nodes and edges
- **position** of nodes
- ...
A Lorillard spokeswoman said, "This is an old story."
A Lorillard spokeswoman said, "This is an old story."
Prague Markup Language
Macros

Perl code to change the data or their appearance

- run by a key stroke
- annotation of the data
- various possibilities to present the same data
- ...
Prague Markup Language
Macros
Až na jednu jedinou, tu hlavní
RSTR která proklamativně zakotvuje existenci Česko - Slovenské celní unie

a
RSTR která má výpovědní lhůtu jeden rok
connective: a (A_H_A)
range: 0->0 (A_H_A)
Which treebanks we have in PML?

- Prague family of treebanks (PDT, PCEDT, PDTSC, CzEng, ...)
- HamleDT
- Tiger Corpus, BNC, Penn Treebank, Penn Discourse Treebank, ...
Once the data are in PML, you can

- use Tree Editor **TrEd** to open, browse and manually **edit** the data
- use **btred** to **process** the data from the command line – apply perl/btred **scripts** to the data
- use **PML-Tree Query** to **search** in the data
Once the data are in PML, you can

- use Tree Editor **TrEd** to **open**, **browse** and **manually edit** the data
- use **btred** to **process** the data from the command line – **apply** perl/btred **scripts** to the data
- use **PML-Tree Query** to **search** in the data
Searching in Discourse-Annotated Treebanks

Outline

- Prague Dependency Treebank (PDT) & Discourse relations
- Prague Markup Language (PML)
- PML-Tree Query
- PDT and PML-Tree Query
- PDTB and PML-Tree Query
PML-TQ is a powerful open-source user-friendly search tool for all kinds of linguistically annotated treebanks.

PML – Prague Markup Language (XML)
TQ – Tree Query
PML-TQ (2009): Petr Pajas, Jan Štěpánek


http://ufal.mff.cuni.cz/pmltq/

Currently maintained and developed by:

Matyáš Kopp
PML-Tree Query

Client-server architecture

- 3 clients
- 2 backends (servers)
PML-TQ: Servers

2 backends (servers):

- **database** (PostgreSQL, Oracle)
  - suitable for **large**(!?), **static** treebanks
- **Tree Editor TrEd**
  - **small**, **changing** data (up to ~10k trees)
PML-TQ: Clients

3 clients:

- **Web browser** (SVG, CSS, Javascript)
  - portable, limited functionality

- **TrEd**
  - requires installation, full power of TrEd environment

- **command-line** (simple, text-based)
PML-Tree Query in TrEd
PML-Tree Query
outline

- PDT – Prague Dependency Treebank
- Discourse relations in PDT
- PML – Prague Markup Language
- PML-Tree Query
- PDT and PML-Tree Query
- PDTB and PML-Tree Query
A query searching for a single node that:

- is an ACTor
- its semantic part of speech is not noun
- it does not have a substitute t_lemma

Textual form of the query:

```
t-node
  [ functor = "ACT", gram/sempos !~ "^n", t_lemma !~ "^#" ]
```
A result:

Jisté ale je, že je dost levorukých [prodavaček]. (PDT)

[It is, however, certain that many [shop-assistants] are left-handed.]
A query searching (on the analytical layer) for a **Predicate** governing a **Subject** and an **Object** with the surface order **Object** – **Predicate** – **Subject**

**Textual form of the query:**

```
a-node
    [ afun = "Pred", order-follows $o, order-precedes $s, 
      a-node $o =
          [ afun = "Obj" ],
      a-node $s =
          [ afun = "Sb" ] ]
```
A result:

Rozepře prý zinscenoval tisk. (PDT)

[lit. The disputes, Acc-Obj allegedly staged the_press, Nom-Sb]

[The disputes were allegedly staged by the press.]
PML-Tree Query
Non-existence (query)

A query searching (on the analytical layer) for a **Predicate** not governing a Subject

Textual form of the query:

```
a-node
  [ afun = "Pred",  
    0x a-node
      [ afun = "Sb" ] ]
```

---

**Diagram:**

```
  a-node
    [ afun = "Pred" ]
    0x
  a-node
    [ afun = "Sb" ]
```

---

**Diagram:**

```
  child
```

---

**Diagram:**

```
  a-node
    [ afun = "Pred" ]
    0x
  a-node
    [ afun = "Sb" ]
```

---

**Diagram:**

```
  a-node
    [ afun = "Pred" ]
    0x
  a-node
    [ afun = "Sb" ]
```

---
A result:

V tom s vámi nesouhlasím. (PDT)
[lit. In that with you [I] disagree.]
[In that [I] do not agree with you.]
A query searching for a **PREDicate** governing an **ACTor** that is **not** (on the analytical layer) represented by a **Subject**

**Textual form of the query:**

```plaintext
t-node [ functor = "PRED",
    t-node [ functor = "ACT",
        a/lex.rf a-node [ afun != "Sb" ] ] ];
```

```plaintext
t-node
    functor = "PRED"

---

t-node
    functor = "ACT"

---

a-node
    afun != "Sb"
```
A result:

K zákazu je ČR vázána mezinárodními dohodami. (PDT)

[Lit.: To the_ban is ČR bound by international agreements.]}

[The Czech Republic is bound to [implement] the ban by international agreements.]
A query searching for an inter-sentential discourse relation (technically, two nodes representing the two arguments, connected by a discourse arrow)

Textual form of the query:

t-node
[!same-tree-as $t, member discourse
[discourse type = "reason", target_node.rf t-node $t = []]];
Neprošel s ní celnicí. **Tak** si ji **pověsil** ve své hospodě na stěnu. (PDT)

[lit.: **He did not get** with it through_customs. **So** REFL it **hung** in his pub on the_wall.]

[He **could not get through** the customs with it. **So** he **has hanged** it in his pub on the wall.]

(The story is about climbers talking about a perfect prosthesis.)
A query searching for **non-contrastively contextually bound nodes** from which there is no anaphoric reference to the previous context.

Textual form of the query:

```
t-node
[ tfa = "t",
  coref_special !~ ".",
  0x coref_gram.rf t-node [],
  0x coref_text/target_node.rf t-node [],
  0x bridging/target_node.rf t-node [] ];
```
A result:

Na dovolené chceme především odpočívat. (PDT)
[On vacation, we want above all to rest.]

Previous context:
Pojedete do zahraničí s cestovkou? (PDT)
[Will you go abroad with a travel agency?]
A query searching for **inter-sentential discourse relations**; the output filter provides a distribution of discourse types.

**Textual form of the query:**

```
t-node
  [ !same-tree-as $t,
    member discourse $m =
      [ type = "discourse",
        target_node.rf t-node $t = [] ] ];

>> for $m.discourse_type give $1,count() sort by $2 desc
```
## PML-Tree Query
Output filters (result)

<table>
<thead>
<tr>
<th>Discourse type</th>
<th>Number of occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>opp</td>
<td>1,601</td>
</tr>
<tr>
<td>conj</td>
<td>1,255</td>
</tr>
<tr>
<td>reason</td>
<td>902</td>
</tr>
<tr>
<td>confr</td>
<td>272</td>
</tr>
<tr>
<td>conc</td>
<td>236</td>
</tr>
<tr>
<td>preced</td>
<td>215</td>
</tr>
<tr>
<td>grad</td>
<td>184</td>
</tr>
<tr>
<td>restr</td>
<td>149</td>
</tr>
<tr>
<td>explicat</td>
<td>121</td>
</tr>
<tr>
<td>corr</td>
<td>110</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
A query searching for all discourse relations; the output filter gives a distribution of connectives and their intra- and inter-sentential usages.

Textual form of the query:

```
t-node $s := [ member discourse $m := [ type = "discourse", target_node.rf t-node $t := [ ] ] ];
>> give lower($m.connective), if(tree_no($s) = tree_no($t),1,0), if(tree_no($s) = tree_no($t),0,1)
>> for $1 give distinct $1, sum($2), sum($3), sum($2)+sum($3)
>> give $1,$4,$2,"(" & $2 * 100 div $4 & ")", $3, "(" & 100 - ($2 * 100 div $4) & ")"
sort by $2 desc
```
### PML-Tree Query

**Output filters (result #2)**

<table>
<thead>
<tr>
<th>Connective</th>
<th>Total</th>
<th>Intra-sentential</th>
<th>(%)</th>
<th>Inter-sentential</th>
<th>(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a [and]</td>
<td>5,128</td>
<td>4,815</td>
<td>(93%)</td>
<td>313</td>
<td>(7%)</td>
</tr>
<tr>
<td>však [however]</td>
<td>1,356</td>
<td>236</td>
<td>(17%)</td>
<td>1,120</td>
<td>(83%)</td>
</tr>
<tr>
<td>ale [but]</td>
<td>1,134</td>
<td>758</td>
<td>(66%)</td>
<td>376</td>
<td>(34%)</td>
</tr>
<tr>
<td>když [when]</td>
<td>478</td>
<td>478</td>
<td>(100%)</td>
<td>0</td>
<td>(0%)</td>
</tr>
<tr>
<td>protože [because]</td>
<td>469</td>
<td>463</td>
<td>(98%)</td>
<td>6</td>
<td>(2%)</td>
</tr>
<tr>
<td>totiž [actually, in fact]</td>
<td>405</td>
<td>20</td>
<td>(4%)</td>
<td>385</td>
<td>(96%)</td>
</tr>
<tr>
<td>:</td>
<td>353</td>
<td>310</td>
<td>(87%)</td>
<td>43</td>
<td>(13%)</td>
</tr>
<tr>
<td>pokud [if]</td>
<td>342</td>
<td>342</td>
<td>(100%)</td>
<td>0</td>
<td>(0%)</td>
</tr>
<tr>
<td>proto [therefore]</td>
<td>339</td>
<td>32</td>
<td>(9%)</td>
<td>307</td>
<td>(91%)</td>
</tr>
<tr>
<td>aby [to]</td>
<td>276</td>
<td>275</td>
<td>(99%)</td>
<td>1</td>
<td>(1%)</td>
</tr>
</tbody>
</table>
Searching in Discourse-Annotated Treebanks

Outline

- Prague Dependency Treebank (PDT) & Discourse relations
- Prague Markup Language (PML)
- PML-Tree Query
- PDT and PML-Tree Query
- PDTB and PML-Tree Query
Penn Discourse Treebank 2.0 (2008, LDC)

- WSJ part of the Penn Treebank
  - 50 thousand sentences annotated (among others) on the surface syntax layer
- discourse relations annotated on raw texts

We use a combination of both annotations and a transformation to the PML
A query searching for an intra-sentential *discourse relation* with an explicit connective

Textual form of the query:
```plaintext
node
[ same-tree-as $t,
  member discourse
  [ type = "Explicit",
    target_node.rf node $t := [[]] ] ];
```
PML-Tree Query
A simple PDTB example (result)

A result:

*The governor couldn’t make it, so the lieutenant governor welcomed the special guests.* (PDTB-2.0)
A query searching for all explicit intra-sentential discourse relations and – thanks to the output filter – produces a distribution table of the senses of these relations, sorted in the descending order by the number of occurrences.

Textual form of the query:

```plaintext
node [same-tree-as $t, member discourse [ type = "Explicit", target_node.rf node $t := [], member conn [ member sem $s := [] ] ]];

>> for $s.sense give $1, count() sort by $2 desc
```
### PML-Tree Query

A PDTB example with an output filter (result)

<table>
<thead>
<tr>
<th>Sense</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expansion.Conjunction</td>
<td>2 431</td>
</tr>
<tr>
<td>Contingency.Cause.Reason</td>
<td>1 475</td>
</tr>
<tr>
<td>Temporal.Synchrony</td>
<td>1 424</td>
</tr>
<tr>
<td>Temporal.Asynchronous.Succession</td>
<td>1 041</td>
</tr>
<tr>
<td>Comparison.Contrast</td>
<td>923</td>
</tr>
<tr>
<td>Contingency.Condition.Hypothetical</td>
<td>767</td>
</tr>
<tr>
<td>Temporal.Asynchronous.Precedence</td>
<td>731</td>
</tr>
<tr>
<td>Comparison.Contrast.Juxtaposition</td>
<td>591</td>
</tr>
<tr>
<td>Contingency.Cause.Result</td>
<td>444</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
PML-Tree Query
A PDTB example with genres (query)

A query searching for all *senses* annotated at all *discourse relations* in the data and produces distributions of the four *semantic classes* for each individual *genre*.

Textual form of the query:
```
root $r :=
  [ descendant node
    [ member discourse
      [ member conn
        [ member sem $s := [[]]]] ];

>> for $r.genre_ad, match($s.sense, ‘^[^.]\+’) give $1,$2,count() sort by $1,$3 desc
>> give $1,$2, ceil($3 * 100 div sum($3 over $1)) & ‘%'
# PML-Tree Query

A PDTB example with genres (result)

<table>
<thead>
<tr>
<th>Genre</th>
<th>Class</th>
<th>Freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>errata</td>
<td>Comparison</td>
<td>65%</td>
</tr>
<tr>
<td>errata</td>
<td>Contingency</td>
<td>18%</td>
</tr>
<tr>
<td>errata</td>
<td>Temporal</td>
<td>12%</td>
</tr>
<tr>
<td>errata</td>
<td>Expansion</td>
<td>6%</td>
</tr>
<tr>
<td>essay</td>
<td>Expansion</td>
<td>42%</td>
</tr>
<tr>
<td>essay</td>
<td>Contingency</td>
<td>25%</td>
</tr>
<tr>
<td>essay</td>
<td>Comparison</td>
<td>21%</td>
</tr>
<tr>
<td>essay</td>
<td>Temporal</td>
<td>14%</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A query searching for all *explicit discourse relations*; the output filter produces a distribution of *senses* for each *connective*.

Textual form of the query:

```
node
  [member discourse
    [ type = "Explicit",
      member conn $c :=
        [ member sem $s := [ ] ] ] ] ];
  >> give $c,lower($c.head),match($s.sense, '[^\.]+'$')
  >> give distinct $1,$2,concat($3,'/' over $1 sort by $3)
  >> for $2,$3 give $1,$2,count()
  >> for $1,$2,$3 give $1,$2 & '\ '(' & $3 & ' \')', sum($3 over $1) sort by $1,$2
  >> give distinct $1,concat($2,', ' over $1),$3 sort by $1
```
## PML-Tree Query

A PDTB example: Appendix A of the PDTB manual (result)

<table>
<thead>
<tr>
<th>Connective</th>
<th>Senses</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>accordingly</td>
<td>Result (5)</td>
<td>5</td>
</tr>
<tr>
<td>additionally</td>
<td>Conjunction (7), Expectation (2),</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Expectation/Succession (1),</td>
<td></td>
</tr>
<tr>
<td>after</td>
<td>Reason/Succession (50), Specification/Succession (1), Succession (523)</td>
<td>577</td>
</tr>
<tr>
<td>afterward</td>
<td>Precedence (11), Conjunction (1733),</td>
<td>11</td>
</tr>
<tr>
<td>also</td>
<td>Conjunction/Synchrony (2), List (10), Specification (1)</td>
<td>1746</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PML-Tree Query

Query Language Highlights

- **queries** can span **over all layers** of annotation (including annotation dictionaries) and **over all sentences in one document**
- allows **arbitrary logical constraints**
- supports **output filters** (generate custom text output, compute statistics, ...)
- offers **graphical query representation** with **relations** (links) between nodes **depicted as arrows**
- understands **PML data model** (no conversion, no information loss)
PML-Tree Query

Thank you for your attention!