

Nové možnosti vyhledávání ve stromových korpusech

Jan Štěpánek, Petr Pajas

ÚFAL

6. 4. 2009

Co je PML-TQ?

Dotazovací jazyk a vyhledávač
nad vrstevnatými treebanky (ve formátu PML)
s grafickým rozhraním
poháněný SQL databází nebo (B/N/J)TrEdem.

Rozdíly

- ♦ Jiný způsob zápisu podmínky na uzly na tranzitivní hraně
- ♦ Množství nových možností:
 - prohledávání více rovin
 - přímá podpora PML formátu
 - výstupní filtry

Kompatibilita

- ♦ Import dotazu

Základní vlastnosti

- ♦ funguje pro lib. *stromovou* anotaci v PML
- ♦ korespondence dotazu a výsledku (uzly a hrany)

Uzly

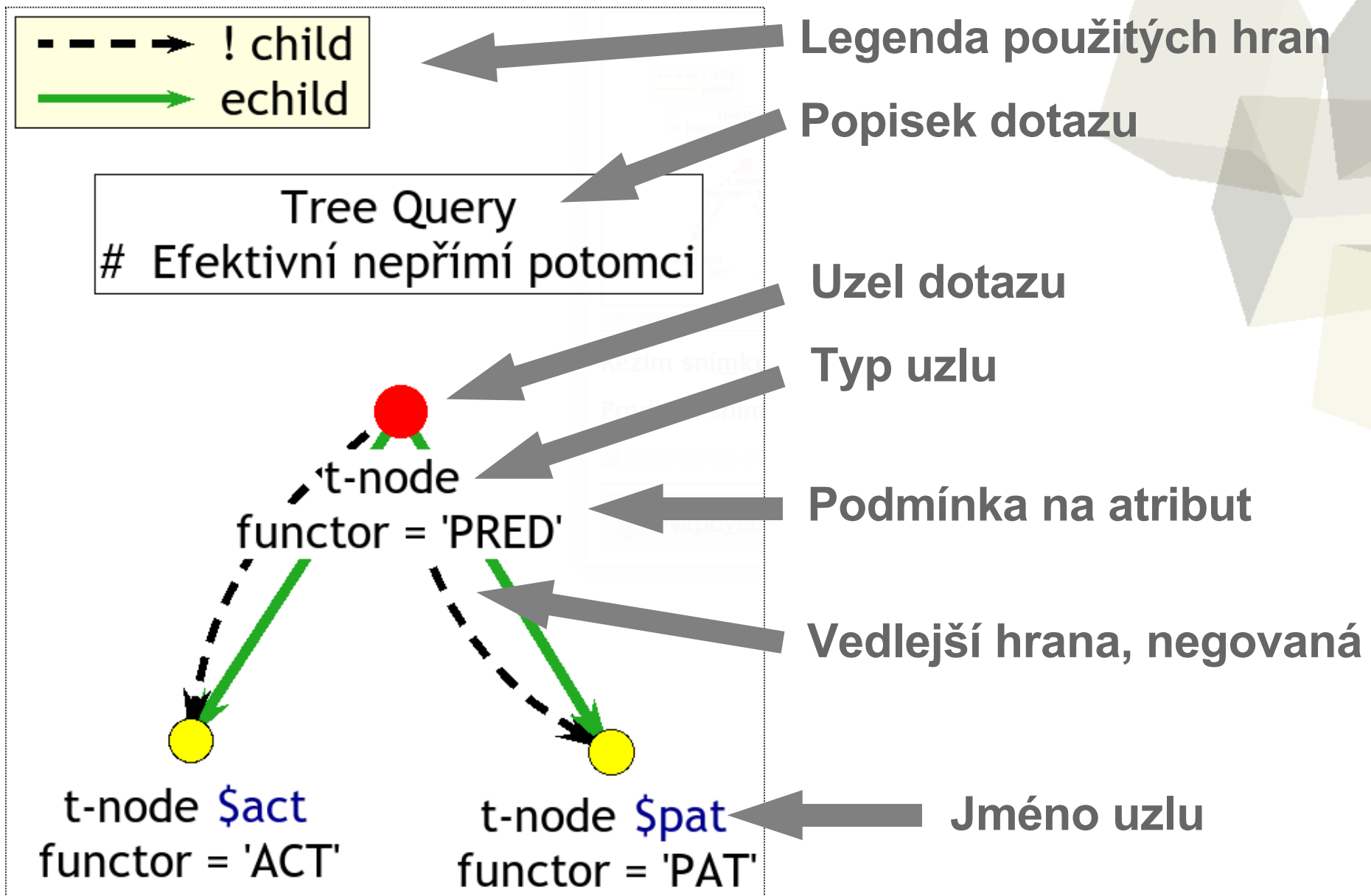
- ♦ různost uzlů
- ♦ typování (rovina)
- ♦ pojmenování

Hrany

- ♦ reprezentují relace mezi uzly
- ♦ vedlejší hrany (šipky)
- ♦ hrany zastupují různé typy relací
 - závislost
 - tranzitivní závislost
 - efektivní závislost
 - libovolnou PML referenci, např.
 - koreferenci
 - odkaz do jiné roviny (vč. PDT Vallexu)

- ♦ na uzly lze klást podmínky
 - hodnota atributu (vč. regulárních výrazů)
 $afun = "Sb"$ $gram/sempos \sim "\wedge n"$
 - vztah k atributům jiných uzlů
 $afun = \$b.afun$
- ♦ logické výrazy (and, or, not)
- ♦ funkce: práce s čísly a řetězci, informace o uzlech
 - některé odpovídají meta-atributům z NetGrapu
- ♦ poddotazy
 - počet výskytů (existenční kvantifikace)
- ♦ výstupní filtry
 - statistické tabulky

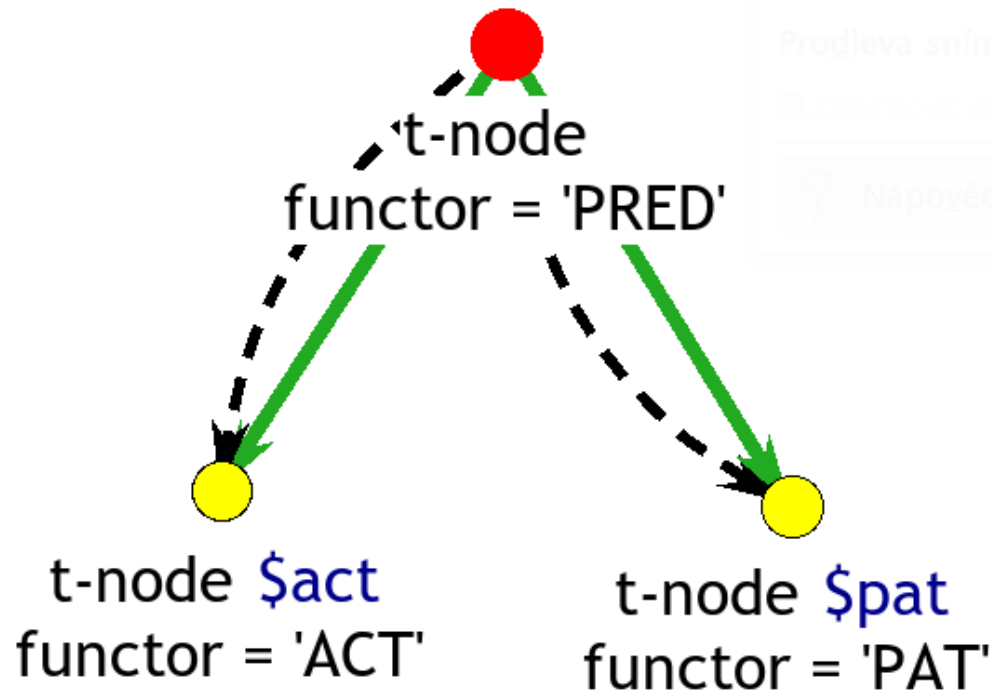
Grafická podoba dotazu



Textová podoba dotazu

---> ! child
--> echild

Tree Query
Efektivní nepřímí potomci



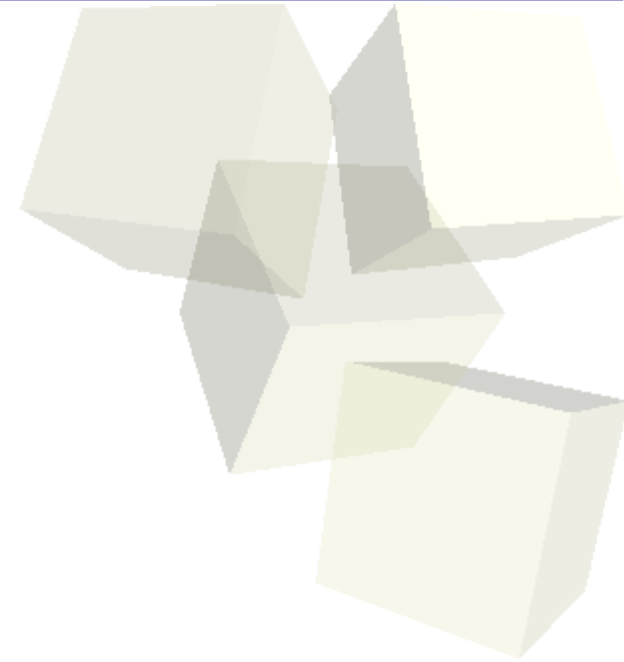
Formální zápis dotazu

```
t-node [  
  functor = 'PRED',
```

```
  echild t-node $act :=  
    [ functor = 'ACT' ],
```

```
  echild t-node $pat :=  
    [ functor = 'PAT' ],
```

```
  !child $act,  
  !child $pat  
]
```

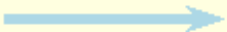






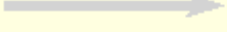



...

Typy relací (šipek)

Standardní


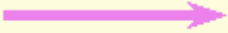

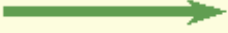



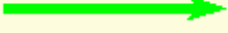
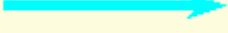
definované v PML-TQ

	ancestor
	child
	depth-first-follows
	depth-first-precedes
	descendant
	order-follows
	order-precedes
	parent

+ zvláštní hrana `member`

Specifické

definované uživatelem či
odvozené ze schématu.
Např. pro PDT 2.0

	<code>a/aux.rf</code>
	<code>a/lex.rf</code>
	<code>a/lex.rf a/aux.rf</code>
	<code>compl</code>
	<code>coref_gram</code>
	<code>coref_text</code>
	<code>echild</code>
	<code>eparent</code>
	<code>val_frame.rf</code>

- ♦ inspirována XPath 2.0 a XQuery 2.0
- ♦ atomické hodnoty: řetězce v uvozovkách, čísla
- ♦ atributy a podatributy (functor, gram/sempos)
- ♦ speciální symboly:

[...]	uzel dotazu, uvnitř podstrom
=	rovnost
~	porovnání s regulárním výrazem
~*	totéž, case-insensitive
+, -, *, div, mod	aritmetické operace
!	negace
and, or	logické spojky
čárka	zkratka za and
\$jméno	proměnná označující uzel
\$číslo	sloupec (výstupní filtry)
?	nepovinný uzel
&	konkatenace řetězců

Orientace šipek v dotazu

- ♦ nemusí nutně odpovídat typické orientaci hran ve výsledku (šipka *parent*)

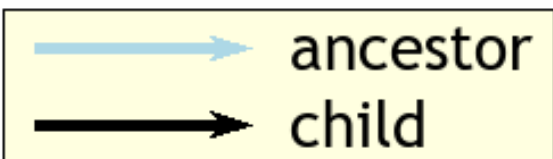
- ♦ směr dolů často jen logicky strukturuje dotaz:

„uzel, který má ...“

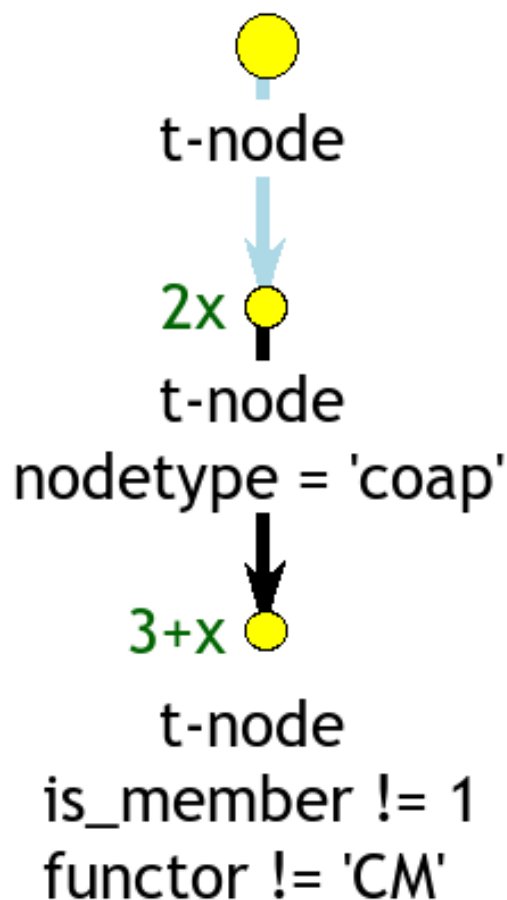
- ♦ **poddotaz**: podstrom, jehož kořen má omezení na počet výskytů (vzhledem k rodiči v dotazu)

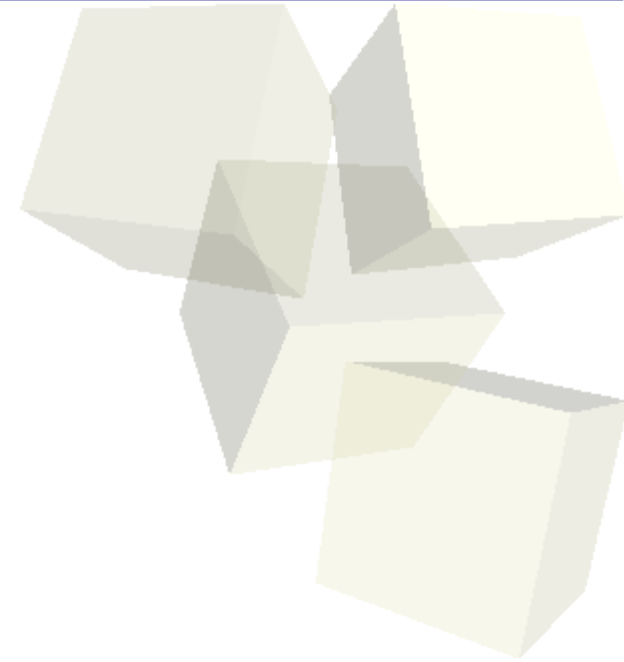
„uzel, který má alespoň 3 ...“

- ♦ uzly poddotazu se jen počítají, nezobrazují se jako výsledky



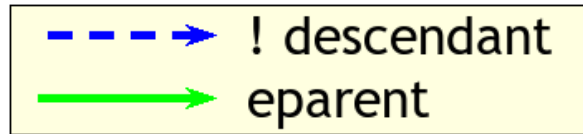
Tree Query



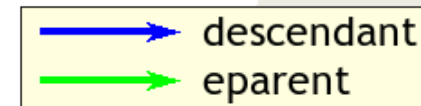
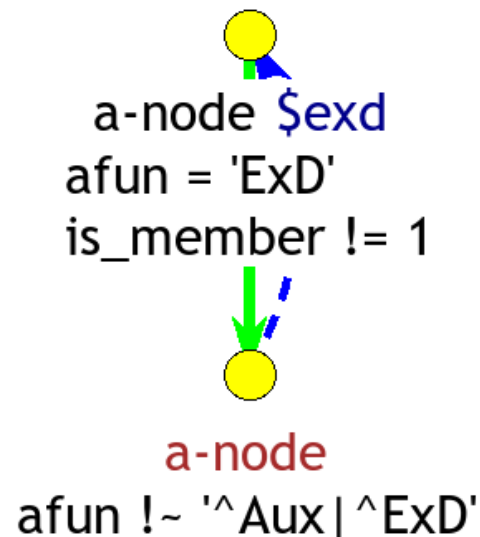


...

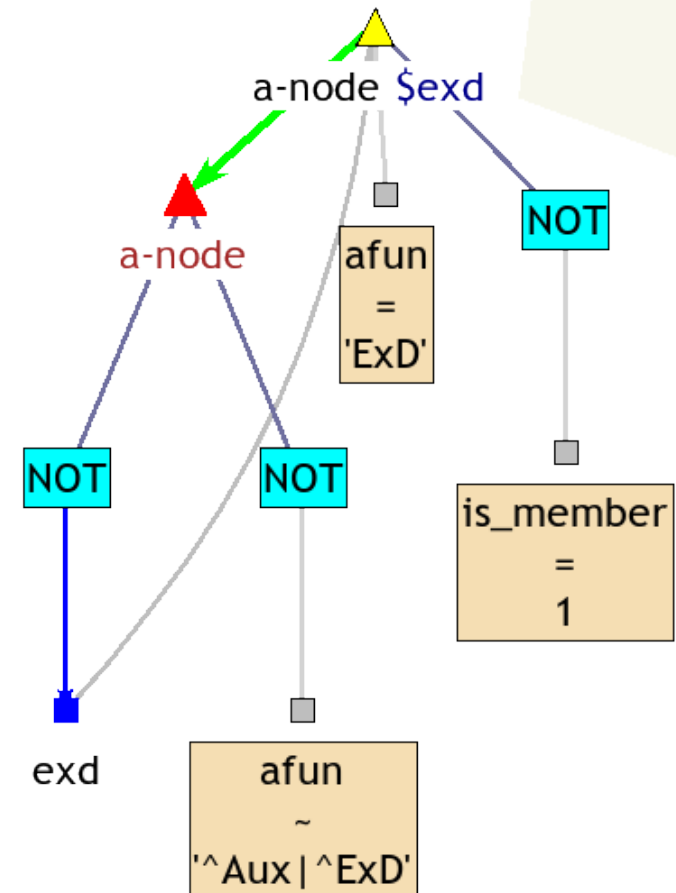
Logické spojky a podmínky lze rozvinout do uzlů

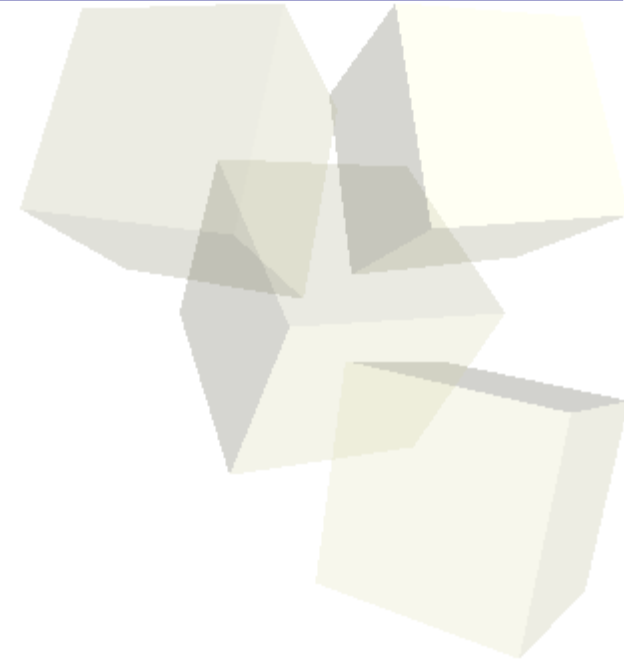


Tree Query
"ExD" jako společné rozvití



Tree Query
"ExD" jako společné rozvití

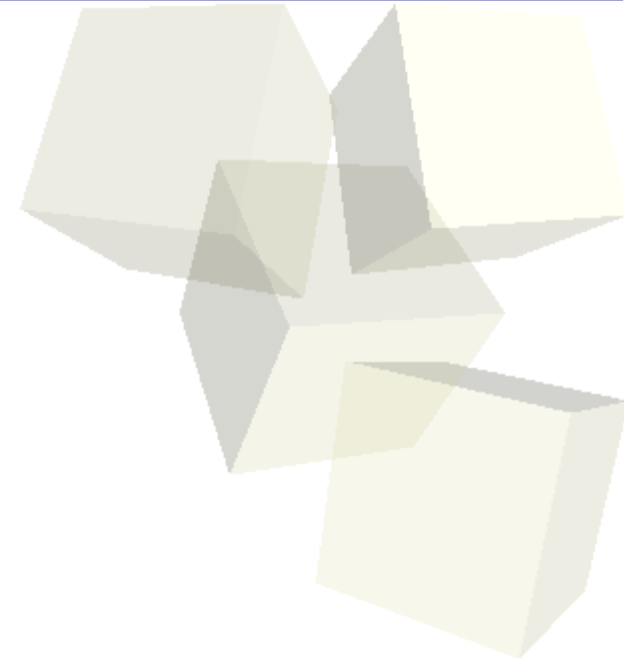




...

- ♦ řetězce
 - substr, lower, upper, replace, tr, length
- ♦ regulární výrazy
 - match, substitute
- ♦ čísla
 - round, ceil, floor, trunc, percent
- ♦ informace o uzlech
 - id, name, file, tree_no, address
 - depth, depth_first_order
 - sons, descendants
 - lbrothers, rbrothers
- ♦ agregační funkce (výstupní filtry)
 - count, max, min, avg, sum, concat

- ♦ textový výstup
 - jen hodnoty určitých atributů a hodnoty z nich vypočtené
 - statistické údaje
 - počty, maxima, minima
 - distribuce, četnosti (agregační funkce)
- ♦ filtr vrací tabulku hodnot (oddělených tabulátorem)
- ♦ filtry lze řetězit
 - 1. filtr zpracuje výsledky dotazu
 - každý další filtr pracuje s výstupem předchozího



...

Dvě implementace vyhledávače

- ◆ Relační SQL databáze (Oracle, PostgreSQL)
 - převod z PML do relačních tabulek
 - dotaz se přeloží na SQL
 - vhodné pro rozsáhlá statická data (treebank)
- ◆ Implementace v Perlu (TrEd, bTrEd, nTrEd, jTrEd)
 - pracuje přímo s PML formátem
 - vyžaduje, aby dotaz byl souvislý graf
 - vhodné pro živá data (anotátoři)

Uživatelská rozhraní

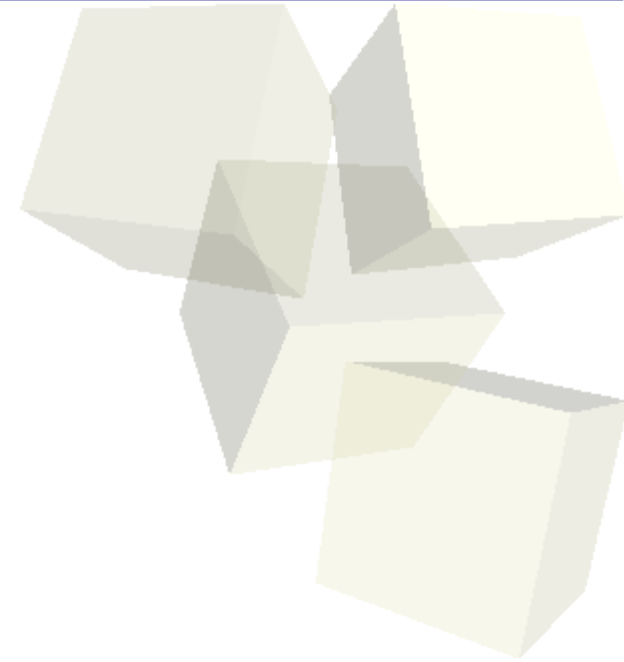
- ◆ TrEd
- ◆ webový formulář (SQL backend)
- ◆ příkazová řádka (příkaz pmltq)
- ◆ API (TrEd, bTrEd...)

Data dostupná v databázi

- ♦ PDT 2.0 (train + PDT Vallex)
- ♦ PEDT (interně)
- ♦ Plánujeme
 - CoNLL 2009 (public domain data, tj. katalánština, čínština, angličtina, němčina, španělština)
 - CzEng (automaticky naparsovaný paralelní korpus)

Přes BTrEd

- ♦ Cokoliv :-)

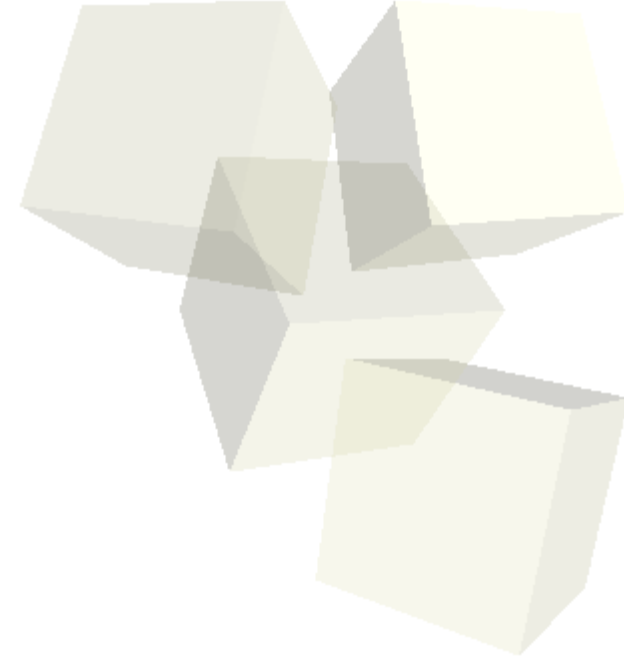


...

Co v PML-TQ nejde?

- ♦ příznak pro ztranzitivnění libovolné relace
 - např. tranzitivní coref_text.rf
 - (komplikované v SQL)
- ♦ otočení libovolné relace (např. otočená a/aux.rf)
 - Perl-backend vyžaduje, aby dotaz měl (po doplnění otočitelných relací) orientovanou kostru
 - lze řešit např. předpočítáním inverzních relací
- ♦ uzly s pod-specifikovaným typem
- ♦ disjunkce kompletních dotazů

- ♦ uživatelem definovaná rozšíření
- ♦ definice nové relace pomocí dotazu
- ♦ různá vylepšení GUI
- ♦ možnost přerušit vyhodnocování dotazu
- ♦ grafické webové rozhraní k PML-TQ (např. upravený NetGraph Client?)
- ♦ dokonalejší plánovač v BTrEdu
- ♦ dotaz jako test (pro dané uzly v datech)



<http://ufal.mff.cuni.cz/~pajas/pmltq/>