LPath+ : A First-Order Complete Language for Linguistic Tree Query

Steven Bird and Catherine Lai

Department of Computer Science
University of Melbourne, AUSTRALIA

Linguistic Data Consortium
Department of Linguistics
University of Pennsylvania, USA

19th Pacific Asia Conference on Language, Information and Computation
require large databases of annotated text and speech
language documentation, linguistic analysis
tasks: collection, curation, annotation, analysis
size: $10^3 - 10^6$ words
annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Linguistic Description

- require large databases of annotated text and speech
- language documentation, linguistic analysis
- tasks: collection, curation, annotation, analysis
- size: $10^3 - 10^6$ words
- annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
require large databases of annotated text and speech
language documentation, linguistic analysis
tasks: collection, curation, annotation, analysis
size: $10^3 - 10^6$ words
annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Linguistic Description

- require large databases of annotated text and speech
- language documentation, linguistic analysis
- tasks: collection, curation, annotation, analysis
- size: \(10^3 - 10^6\) words
- annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
require large databases of annotated text and speech

language documentation, linguistic analysis

tasks: collection, curation, annotation, analysis

size: $10^3 - 10^6$ words

annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Language Technology

- require large databases of annotated text and speech
  - QA, MT, TDT, SLDS, TIDES, ...
  - tasks: collection, curation, annotation, analysis
  - size: $10^6 - 10^9$ words
  - annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Language Technology

- require large databases of annotated text and speech
- QA, MT, TDT, SLDS, TIDES, ...
- tasks: collection, curation, annotation, analysis
- size: $10^6 - 10^9$ words
- annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Language Technology

- require large databases of annotated text and speech
- QA, MT, TDT, SLDS, TIDES, ...
- tasks: collection, curation, annotation, analysis
- size: $10^6$ – $10^9$ words
- annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Language Technology

- require large databases of annotated text and speech
- QA, MT, TDT, SLDS, TIDES, ...
- tasks: collection, curation, annotation, analysis
- size: $10^6 - 10^9$ words
- annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Language Technology

- require large databases of annotated text and speech
- QA, MT, TDT, SLDS, TIDES, ...
- tasks: collection, curation, annotation, analysis
- size: $10^6 - 10^9$ words
- annotations: phonetics, prosody, orthography, syntax, dialog, and gesture
Linguistic Annotation

primary data: immutable, “signal”, supports incoming references

annotations: coding scheme, structure, hierarchy: trees
primary data: immutable, “signal”, supports incoming references
annotations: coding scheme, structure, hierarchy: trees
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus $\neq$ database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
- optimisation, indexing
- three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus \(\neq\) database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
- optimisation, indexing
- three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus $\neq$ database!
- collection of records, schema
  - managed: integrity, quality
  - access: shared, controlled
  - language: query, update
  - optimisation, indexing
  - three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus $\neq$ database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
- optimisation, indexing
- three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus $\neq$ database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
- optimisation, indexing
- three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus \(\neq\) database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
  - optimisation, indexing
  - three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus \( \neq \) database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
- optimisation, indexing
- three-level model
Linguistic Databases

- corpus: source, balance, curation, critical judgement
- corpus \(\neq\) database!
- collection of records, schema
- managed: integrity, quality
- access: shared, controlled
- language: query, update
- optimisation, indexing
- three-level model
Semistructured Databases

- relational model vs XML
- optional, repeatable elements
- tree model
- schema-later
- XPath
Semistructured Databases

- relational model vs XML
- optional, repeatable elements
  - tree model
  - schema-later
- XPath
Semistructured Databases

- relational model vs XML
- optional, repeatable elements
- tree model
  - schema-later
- XPath
Semistructured Databases

- relational model vs XML
- optional, repeatable elements
- tree model
- schema-later
- XPath
Semistructured Databases

- relational model vs XML
- optional, repeatable elements
- tree model
- schema-later
- XPath
Motivations

- benefitting from database theory and technology
- 2005 vs 1965: one-level model: flat files, ad hoc QLs
- general data model: Annotation Graphs
  
  *Bird & Liberman (2001), Speech Communication 33, pp 23-60*
- existing QLs inadequate:
  - relational and semistructured QLs: expressiveness
  - linguistic QLs: efficiency
Motivations

- benefitting from database theory and technology
- 2005 vs 1965: one-level model: flat files, ad hoc QLs
- general data model: *Annotation Graphs*
  
  *Bird & Liberman (2001), Speech Communication 33, pp 23-60*

- existing QLs inadequate:
  - relational and semistructured QLs: expressiveness
  - linguistic QLs: efficiency
Motivations

- benefitting from database theory and technology
- 2005 vs 1965: one-level model: flat files, ad hoc QLs
- general data model: **Annotation Graphs**
  
  *Bird & Liberman (2001), Speech Communication 33, pp 23-60*

- existing QLs inadequate:
  - relational and semistructured QLs: expressiveness
  - linguistic QLs: efficiency
Motivations

- benefitting from database theory and technology
- 2005 vs 1965: one-level model: flat files, ad hoc QLs
- general data model: *Annotation Graphs*
  
  *Bird & Liberman (2001), Speech Communication 33, pp 23-60*

- existing QLs inadequate:
  - relational and semistructured QLs: expressiveness
  - linguistic QLs: efficiency
Motivations

- benefiting from database theory and technology
- 2005 vs 1965: one-level model: flat files, ad hoc QLs
- general data model: Annotation Graphs
  
  *Bird & Liberman (2001), Speech Communication 33, pp 23-60*

- existing QLs inadequate:
  - relational and semistructured QLs: expressiveness
  - linguistic QLs: efficiency
Motivations

- benefitting from database theory and technology
- 2005 vs 1965: one-level model: flat files, ad hoc QLs
- general data model: **Annotation Graphs**

  *Bird & Liberman (2001), Speech Communication 33, pp 23-60*

- existing QLs inadequate:
  - relational and semistructured QLs: expressiveness
  - linguistic QLs: efficiency
First Order Logic over Trees

- universal language for describing structures
- equivalent to SQL
- two linguistic tasks on trees: finding, relating
- both require exactly two free variables
First Order Logic over Trees

- universal language for describing structures
- equivalent to SQL
  - two linguistic tasks on trees: finding, relating
  - both require exactly two free variables
First Order Logic over Trees

- universal language for describing structures
- equivalent to SQL
- two linguistic tasks on trees: finding, relating
  - both require exactly two free variables
First Order Logic over Trees

- universal language for describing structures
- equivalent to SQL
- two linguistic tasks on trees: finding, relating
- both require exactly two free variables
Formulas with Two Free Variables

- binary relations on trees
- \( \{(x, y) \mid \exists z \cdots \} \)
Filters
Axiomatisation of FO^2_{Tree}

- signature: desc, fs
- c.f. Relational tables
- derived relations:
  - \( fs \equiv_{\text{def}} \{ x, z \mid \text{desc}(x, z) \land \neg \exists y (\text{desc}(x, y) \land \text{desc}(y, z)) \} \)
  - \( f \equiv_{\text{def}} \{ w, z \mid \text{fs}(w, z) \lor \exists x, y (\text{desc}(x, w) \land \text{fs}(x, y) \land \text{desc}(y, z)) \} \)
Create a scalable and reusable model for linguistic query and relate it to well-understood semistructured (XML) and relational (SQL) languages.
Acknowledgements

1. **US National Science Foundation:**
   Grant No. 0317826 *Querying Linguistic Databases.*

2. **Co-Principal Investigators:**
   Susan Davidson, Mark Liberman

3. **Research Students:**
   Yi Chen, Scott Cotton, Catherine Lai, Haejoong Lee,
   Kazuaki Maeda, Yifeng Zheng

4. **For more information:**
   [http://www.ldc.upenn.edu/Projects/QLDB/](http://www.ldc.upenn.edu/Projects/QLDB/)
Acknowledgements

1. US National Science Foundation: Grant No. 0317826 *Querying Linguistic Databases*.

2. Co-Principal Investigators: Susan Davidson, Mark Liberman


4. For more information: http://www.ldc.upenn.edu/Projects/QLDB/
Acknowledgements

1. US National Science Foundation: Grant No. 0317826 *Querying Linguistic Databases*.

2. Co-Principal Investigators: Susan Davidson, Mark Liberman


4. For more information: http://www.ldc.upenn.edu/Projects/QLDB/
Acknowledgements

1. US National Science Foundation: Grant No. 0317826 *Querying Linguistic Databases*.

2. Co-Principal Investigators:
   Susan Davidson, Mark Liberman

3. Research Students:
   Yi Chen, Scott Cotton, Catherine Lai, Haejoong Lee, Kazuaki Maeda, Yifeng Zheng

4. For more information:
   [http://www.ldc.upenn.edu/Projects/QLDB/](http://www.ldc.upenn.edu/Projects/QLDB/)
Linguistic Trees

What's special about linguistic trees? Immutability of leaves, precedence.

Steven Bird and Catherine Lai
Linguistic Trees 2: Proper Analyses

CFG Productions

S → NP VP (NP)  
VP → V NP (NP)  
NP → NP PP  
NP → Det Adj* N  
PP → Prep NP

Some Proper Analyses

I saw the old N with NP today  
I V the Adj man PP today  
NP saw NP with a telescope NP  
I VP NP  
I saw NP today
Linguistic Trees 3: Charts

Steven Bird and Catherine Lai

LPath\textsuperscript{+} Linguistic Query Language
Existing Query Languages

- **Linguistic QLs: Issues with efficiency, reusability**
  - CorpusSearch (U Penn); EMU (Macquarie U); Gsearch (U Edinburgh); Linguists’ Search Engine (U Maryland); NetGraph (Charles U, Prague); NXT Search, Q4M, TIGERSearch (U Stuttgart); TGrep2 (MIT); VIQTORYA (Tübingen, Paris);
  - Semistructured QLs: problems with expressiveness (XPath) or efficiency (XQuery)
Existing Query Languages

- **Linguistic QLs**: Issues with efficiency, reusability
  - CorpusSearch (U Penn); EMU (Macquarie U); Gsearch (U Edinburgh); Linguists’ Search Engine (U Maryland); NetGraph (Charles U, Prague); NXT Search, Q4M, TIGERSearch (U Stuttgart); TGrep2 (MIT); VIQTORYA (Tübingen, Paris);

- **Semistructured QLs**: problems with expressiveness (XPath) or efficiency (XQuery)
Existing Query Languages

- Linguistic QLs: Issues with efficiency, reusability
  - CorpusSearch (U Penn); EMU (Macquarie U); Gsearch (U Edinburgh); Linguists’ Search Engine (U Maryland); NetGraph (Charles U, Prague); NXT Search, Q4M, TIGERSearch (U Stuttgart); TGrep2 (MIT); VIQTORYA (Tübingen, Paris);
- Semistructured QLs: problems with expressiveness (XPath) or efficiency (XQuery)
E.g.: NPs whose rightmost child is N

- **TGrep2**: \( NP \leftarrow N \)
- **EMU**: \( \text{end} (\text{Syntax}=NP, \text{Syntax}=N)=1 \)
- **TIGERSearch**
  \[
  [\text{cat}="NP"] \land \#n2:[\text{pos}="N"] \\
  \land (\#n1 >* \#n2) \land (\#n1 >@r \#n3) \\
  \land (\#n2 >* \#n4)
  \]
- **CorpusSearch**
  \[
  \text{node}: \text{NP} \\
  \text{query}: \text{NP iDomsLast1 N}
  \]
- **NXT Search**
  \[
  (\$np \text{ cat}) (\$w \text{ word}) : \\
  (\$np@\text{cat}=="NP") \land (\$w@\text{pos}=="N") \\
  \land (\$np ^1[-1] \$w)
  \]
Sample Queries: Immediate Following

\[ Q_1: \text{Find noun phrases that follow a verb} \quad //\text{V/following::NP} \]
Sample Queries: Immediate Following

$Q_1$: Find noun phrases that follow a verb //V//following::NP

$Q_2$: Find noun phrases that immediately follow a verb

Steven Bird and Catherine Lai

LPath+: Linguistic Query Language
Sample Queries: Subtree Scoping

Q3: Find nouns that follow a verb which is a child of a verb phrase

//VP/V/following::N

Steven Bird and Catherine Lai  LPath+ Linguistic Query Language
Sample Queries: Subtree Scoping

\[ Q_3: \text{Find nouns that follow a verb which is a child of a verb phrase} \]
\[
//VP/V/following::N
\]

\[ Q_4: \text{Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase} \]

Steven Bird and Catherine Lai
**Sample Queries: Edge Alignment**

**Q_5:** Find noun phrases which are the rightmost child of a verb phrase

`//VP/*[last()]self::NP`

---

Steven Bird and Catherine Lai
Sample Queries: Edge Alignment

**Q₅**: Find noun phrases which are the rightmost child of a verb phrase

\[//\text{VP}/[^\text{last()}][\text{self}::\text{NP}]\]

**Q₆**: Find noun phrases which are the rightmost descendants of a verb phrase

---

Steven Bird and Catherine Lai

LPath⁺ Linguistic Query Language
Path language for linguistic trees


Extended with “zero-star height” closures (LPath⁺)
LPath

- Path language for linguistic trees
  

- Extended with “zero-star height” closures (LPath⁺)
**LPath\(^+\) Syntax**

```plaintext
locpath  ::=  abspath | abspath '{' locpath '}\) | locpath '|' locpath
abspath ::= | locstep abspath
locstep ::= axis test ("['fexpr']") (closure)
fexpr  ::=  locpath | fexpr 'and' fexpr | fexpr 'or' | 'not' fexpr | '(' fexpr ')'
axis ::= \'\' | '\\' | '\\*' | '.' | '/' | '//\' | '// | >' | '<-' | '-->' | '<--' | '=' | '<=' | '==>' | '<=='
closure ::= '"?' | '*" | '+'
test ::= p | _ | '^p | p'$
```
### Navigation Axes in LPath

<table>
<thead>
<tr>
<th>Parent Axis</th>
<th>Child Axis</th>
<th>Ancestor Axis</th>
<th>Descendant Axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>\</td>
<td>/</td>
<td>\</td>
<td>//</td>
</tr>
<tr>
<td>\*</td>
<td>//</td>
<td>\*</td>
<td>//*</td>
</tr>
<tr>
<td>*</td>
<td>=&gt;</td>
<td>=&gt;</td>
<td>=&gt;</td>
</tr>
<tr>
<td>--&gt;</td>
<td>&lt;=</td>
<td>&lt;=</td>
<td>&lt;=</td>
</tr>
<tr>
<td>---</td>
<td>==</td>
<td>==</td>
<td>==</td>
</tr>
<tr>
<td>.</td>
<td>self</td>
<td>self</td>
<td>self</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\</td>
<td>\</td>
</tr>
<tr>
<td></td>
<td></td>
<td>\</td>
<td>\</td>
</tr>
<tr>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>=&gt;</td>
<td>=&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;=</td>
<td>&lt;=</td>
</tr>
<tr>
<td></td>
<td></td>
<td>==</td>
<td>==</td>
</tr>
</tbody>
</table>

- Parent: \n- Child: /
Subtree Scoping

- XPath doesn’t support tree queries that circumscribe navigations to remain inside a subtree
- necessity for compositional queries
  - $Q_4$: Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase.
- LPath uses braces to represent subtree scoping
  - $Q_3$: Find nouns that follow a verb which is a child of a verb phrase $//\text{VP}/\text{V} \rightarrow \text{N}$
  - $Q_4$: $//\text{VP} /\{/\text{V} \rightarrow \text{N}\}$.
Subtree Scoping

- XPath doesn’t support tree queries that circumscribe navigations to remain inside a subtree
- Necessity for compositional queries
  - \(Q_4\): Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase.
- LPath uses braces to represent subtree scoping
  - \(Q_3\): Find nouns that follow a verb which is a child of a verb phrase  
    \(//VP/V-->N\)
  - \(Q_4\):  
    \(//VP/\{V-->N\}\).
XPath doesn’t support tree queries that circumscribe navigations to remain inside a subtree

necessity for compositional queries

- $Q_4$: Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase.

LPath uses braces to represent subtree scoping

- $Q_3$: Find nouns that follow a verb which is a child of a verb phrase //VP//V-->N
- $Q_4$: //VP{/V-->N}.
Subtree Scoping

- XPath doesn’t support tree queries that circumscribe navigations to remain inside a subtree
- Necessity for compositional queries
  - $Q_4$: Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase.
- LPath uses braces to represent subtree scoping
  - $Q_3$: Find nouns that follow a verb which is a child of a verb phrase $//VP/V-->N$
  - $Q_4$: $//VP\{/V-->N\}$. 
Subtree Scoping

- XPath doesn’t support tree queries that circumscribe navigations to remain inside a subtree
- Necessity for compositional queries
  - $Q_4$: Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase.
- LPath uses braces to represent subtree scoping
  - $Q_3$: Find nouns that follow a verb which is a child of a verb phrase $//VP/V\rightarrow N$
  - $Q_4$: $//VP/{V\rightarrow N}$. 
Subtree Scoping

- XPath doesn’t support tree queries that circumscribe navigations to remain inside a subtree.
- Necessity for compositional queries:
  - $Q_4$: Within a verb phrase, find nouns that follow a verb which is a child of the verb phrase.
- LPath uses braces to represent subtree scoping:
  - $Q_3$: Find nouns that follow a verb which is a child of a verb phrase: $//VP/V\rightarrow N$
  - $Q_4$: $//VP/\{V\rightarrow N\}$.
Edge Alignment

- XPath doesn’t fully support alignment on the tree
  - $Q_6$: Find noun phrases which are the rightmost descendants of a verb phrase.
- LPath provides the following grep-like syntactic sugar
  - Leftmost descendant of $A$: $^A$
  - Rightmost descendant of $A$: $A$

$Q_6$: $//$VP{///NP$}$
Edge Alignment

- XPath doesn’t fully support alignment on the tree
  - $Q_6$: Find noun phrases which are the rightmost descendants of a verb phrase.
- LPath provides the following grep-like syntactic sugar
  - Leftmost descendant of $A$: $^A$
  - Rightmost descendant of $A$: $A$

- $Q_6$: //VP { //NP $}
Edge Alignment

- XPath doesn’t fully support alignment on the tree
  - $Q_6$: Find noun phrases which are the rightmost descendants of a verb phrase.
- LPath provides the following grep-like syntactic sugar
  - Leftmost descendant of A: $^A$
  - Rightmost descendant of A: $A$
  - $Q_6: //VP { //NP$ }$
Edge Alignment

- XPath doesn’t fully support alignment on the tree
  - $Q_6$: Find noun phrases which are the rightmost descendants of a verb phrase.
- LPath provides the following grep-like syntactic sugar
  - Leftmost descendant of $A$: $^A$
  - Rightmost descendant of $A$: $A$
- $Q_6$: `//VP { //NP $}`
Edge Alignment

- XPath doesn’t fully support alignment on the tree
  - $Q_6$: Find noun phrases which are the rightmost descendants of a verb phrase.

- LPath provides the following grep-like syntactic sugar
  - Leftmost descendant of A: $^A$
  - Rightmost descendant of A: $A$

- $Q_6$: \(\text{//VP { //NP }$\)
Edge Alignment

- XPath doesn’t fully support alignment on the tree
  - \( Q_6 \): Find noun phrases which are the rightmost descendants of a verb phrase.
- LPath provides the following grep-like syntactic sugar
  - Leftmost descendant of A: \(^A\)
  - Rightmost descendant of A: \(A\)$$
- \( Q_6 \): \(/ /VP \{ / /NP \$ \} \)
### Sample Queries in LPath

**Q₁** Find noun phrases that follow a verb

\[//V\rightarrow NP\]

**Q₂** Find noun phrases that are immediately following a verb.

\[//V\rightarrow NP\]

**Q₃** Find nouns that follow a verb which is a child of a verb phrase.

\[//VP//V\rightarrow N\]

**Q₄** Within a verb phrase, find nouns that follow a verb which is a child of a verb phrase.

\[//VP\{//V\rightarrow N\}\]

**Q₅** Find noun phrases which are the rightmost child of a verb phrase.

\[//VP\{//NP\$\}\]

**Q₆** Find noun phrases which are rightmost descendants of a verb phrase.

\[//VP\{//NP\$\}\]
Sample Queries in LPath

\(Q_1\) Find noun phrases that follow a verb
\(/ \text{V} \rightarrow \text{NP}\)

\(Q_2\) Find noun phrases that are immediately following a verb.
\(/ \text{V} \rightarrow \text{NP}\)

\(Q_3\) Find nouns that follow a verb which is a child of a verb phrase.
\(/ \text{VP} / \text{V} \rightarrow \text{N}\)

\(Q_4\) Within a verb phrase, find nouns that follow a verb which is a child of a verb phrase. \(/ \text{VP} / \{\text{V} \rightarrow \text{N}\}\)

\(Q_5\) Find noun phrases which are the rightmost child of a verb phrase.
\(/ \text{VP} / \{\text{NP}\}$\)

\(Q_6\) Find noun phrases which are rightmost descendants of a verb phrase.
\(/ \text{VP} / \{\text{NP}\}$\)

Steven Bird and Catherine Lai

LPath\(^+\) Linguistic Query Language
Sample Queries in LPath

\( Q_1 \)  Find noun phrases that follow a verb
\[ / \ V \rightarrow NP \]

\( Q_2 \)  Find noun phrases that are immediately following a verb.
\[ / \ V \rightarrow NP \]

\( Q_3 \)  Find nouns that follow a verb which is a child of a verb phrase.
\[ / \ VP / \ V \rightarrow N \]

\( Q_4 \)  Within a verb phrase, find nouns that follow a verb which is a child of a verb phrase.
\[ / \ VP \{ / \ V \rightarrow N \} \]

\( Q_5 \)  Find noun phrases which are the rightmost child of a verb phrase.
\[ / \ VP \{ / NP $ \} \]

\( Q_6 \)  Find noun phrases which are rightmost descendants of a verb phrase.
\[ / \ VP \{ / / NP $ \} \]
Sample Queries in LPath

Q₁ Find noun phrases that follow a verb
   //V→→NP

Q₂ Find noun phrases that are immediately following a verb.
   //V→NP

Q₃ Find nouns that follow a verb which is a child of a verb phrase.
   //VP/V→→N

Q₄ Within a verb phrase, find nouns that follow a verb which is a child of a verb phrase. //VP { //V→→N }

Q₅ Find noun phrases which are the rightmost child of a verb phrase.
   //VP { //NP$ }

Q₆ Find noun phrases which are rightmost descendants of a verb phrase.
   //VP { //NP$ }
Sample Queries in LPath

Q₁ Find noun phrases that follow a verb
   \[
   \text{//}_V \rightarrow \text{NP}
   \]

Q₂ Find noun phrases that are immediately following a verb.
   \[
   \text{//}_V \rightarrow \text{NP}
   \]

Q₃ Find nouns that follow a verb which is a child of a verb phrase.
   \[
   \text{//}_{VP} / _V \rightarrow _N
   \]

Q₄ Within a verb phrase, find nouns that follow a verb which is a child of a verb phrase.
   \[
   \text{//}_{VP} \{ / _V \rightarrow _N \}
   \]

Q₅ Find noun phrases which are the rightmost child of a verb phrase.
   \[
   \text{//}_{VP} \{ / _{NP} \$
   \]

Q₆ Find noun phrases which are rightmost descendants of a verb phrase.
   \[
   \text{//}_{VP} \{ / _{NP} \$
   \]
Sample Queries in LPath

Q₁  Find noun phrases that follow a verb
    //V-->NP

Q₂  Find noun phrases that are immediately following a verb.
    //V-->NP

Q₃  Find nouns that follow a verb which is a child of a verb phrase.
    //VP/V-->N

Q₄  Within a verb phrase, find nouns that follow a verb which is a child of a verb phrase.  //VP { //V-->N }

Q₅  Find noun phrases which are the rightmost child of a verb phrase.
    //VP { //NP$ }

Q₆  Find noun phrases which are rightmost descendants of a verb phrase.
    //VP { //NP$ }
Apparent mismatch...

Path queries are **acyclic**, $\text{FO}^2_{\text{Tree}}$ formulas may be cyclic

Scoping syntax leads to cycles

E.g. $\forall N \{ \forall P P \rightarrow N \forall V P \}$
Solution 1: Re-Introduce Variables

\{(w, z) \mid \exists x, y : NP(w) \land PP(x) \land N(y) \land VP(z) \\
\land desc(w, x) \land f(x, y) \land desc(y, z) \\
\land desc(w, y) \land desc(w, y)\}
Solution 2: Factor out Cycles

---

Steven Bird and Catherine Lai

LPath+ Linguistic Query Language
Interval Labeling Scheme

Steven Bird and Catherine Lai
### Evaluation: Storage

<table>
<thead>
<tr>
<th>left</th>
<th>right</th>
<th>depth</th>
<th>id</th>
<th>pid</th>
<th>name</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>root</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>@lex I</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>VP</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>5</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>6</td>
<td>@lex saw</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>3</td>
<td>8</td>
<td>5</td>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>8</td>
<td>NP</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>10</td>
<td>9</td>
<td>Det</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>6</td>
<td>11</td>
<td>10</td>
<td>@lex the</td>
<td></td>
</tr>
</tbody>
</table>

...
## LPath Axes and Label Conditions

<table>
<thead>
<tr>
<th>Vertical Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>child((m, n))</td>
<td>(n.id = m.pid)</td>
</tr>
<tr>
<td>descendent((m, n))</td>
<td>(m.l \geq n.l, m.r \leq n.r, m.d &gt; n.d)</td>
</tr>
<tr>
<td>parent((m, n))</td>
<td>(m.id = n.pid)</td>
</tr>
<tr>
<td>ancestor((m, n))</td>
<td>(m.l \leq n.l, m.r \geq n.r, m.d &lt; n.d)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Horizontal Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>immediate-following((m, n))</td>
<td>(m.l = n.r)</td>
</tr>
<tr>
<td>following((m, n))</td>
<td>(m.l \geq n.r)</td>
</tr>
<tr>
<td>immediate-preceding((m, n))</td>
<td>(m.r = n.l)</td>
</tr>
<tr>
<td>preceding((m, n))</td>
<td>(m.r \leq n.l)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sibling Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>immediate-following-sibling((m, n))</td>
<td>(m.l = n.r, m.pid = n.pid)</td>
</tr>
<tr>
<td>following-sibling((m, n))</td>
<td>(m.l \geq n.r, m.pid = n.pid)</td>
</tr>
<tr>
<td>immediate-preceding-sibling((m, n))</td>
<td>(m.r = n.l, m.pid = n.pid)</td>
</tr>
<tr>
<td>preceding-sibling((m, n))</td>
<td>(m.r \leq n.l, m.pid = n.pid)</td>
</tr>
</tbody>
</table>

Interval labels: \((l,r,d,id,pid)\)

- l - left
- r - right
- d - depth
## Evaluation: Axes and Join Constraints

<table>
<thead>
<tr>
<th>Vertical Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>child($m, n$)</td>
<td>$n.id = m.pid$</td>
</tr>
<tr>
<td>descendent($m, n$)</td>
<td>$m.l \geq n.l, m.r \leq n.r, m.d &gt; n.d$</td>
</tr>
<tr>
<td>parent($m, n$)</td>
<td>$m.id = n.pid$</td>
</tr>
<tr>
<td>ancestor($m, n$)</td>
<td>$m.l \leq n.l, m.r \geq n.r, m.d &lt; n.d$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Horizontal Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>immediate-following($m, n$)</td>
<td>$m.l = n.r$</td>
</tr>
<tr>
<td>following($m, n$)</td>
<td>$m.l \geq n.r$</td>
</tr>
<tr>
<td>immediate-preceding($m, n$)</td>
<td>$m.r = n.l$</td>
</tr>
<tr>
<td>preceding($m, n$)</td>
<td>$m.r \leq n.l$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sibling Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>immediate-following-sibling($m, n$)</td>
<td>$m.l = n.r, m.pid = n.pid$</td>
</tr>
<tr>
<td>following-sibling($m, n$)</td>
<td>$m.l \geq n.r, m.pid = n.pid$</td>
</tr>
<tr>
<td>immediate-preceding-sibling($m, n$)</td>
<td>$m.r = n.l, m.pid = n.pid$</td>
</tr>
<tr>
<td>preceding-sibling($m, n$)</td>
<td>$m.r \leq n.l, m.pid = n.pid$</td>
</tr>
</tbody>
</table>
//VP/V-->N

select T2.* from T T0, T T1, T T2
where T0.tid=T1.tid and T0.id=T1.pid
and T0.tag='VP' and T1.tid=T2.tid
and T1."right"<=T2."left"
and T1.tag='V' and T2.tag='N'

//VP{/V-->N}

select T2.* from T T0, T T1, T T2
where T0.tid=T1.tid and T0.id=T1.pid
    and T0."left"<T1."right" and T0."left"<T2."right"
    and T0.tag='VP' and T1.tid=T2.tid
    and T1."left"<T0."right" and T1."right"<=T2."left"
    and T1.tag='V' and T2."left"<T0."right" and T2.tag='N'
Experiments

- Load test data
- Translate LPath to SQL using yacc
- Compare with two existing linguistic query engines: CorpusSearch, TGrep2
- Evaluate LPath queries on two data sets
  - Wall Street Journal (WSJ)
  - Switchboard (SWB)
Experiments

- Load test data
- Translate LPath to SQL using yacc
  - Compare with two existing linguistic query engines: CorpusSearch, TGrep2
- Evaluate LPath queries on two data sets
  - Wall Street Journal (WSJ)
  - Switchboard (SWB)
Experiments

- Load test data
- Translate LPath to SQL using yacc
- Compare with two existing linguistic query engines: CorpusSearch, TGrep2
- Evaluate LPath queries on two data sets
  - Wall Street Journal (WSJ)
  - Switchboard (SWB)
Experiments

- Load test data
- Translate LPath to SQL using yacc
- Compare with two existing linguistic query engines: CorpusSearch, TGrep2
- Evaluate LPath queries on two data sets
  - Wall Street Journal (WSJ)
  - Switchboard (SWB)
Experiments

- Load test data
- Translate LPath to SQL using yacc
- Compare with two existing linguistic query engines: CorpusSearch, TGrep2
- Evaluate LPath queries on two data sets
  - Wall Street Journal (WSJ)
  - Switchboard (SWB)
Experiments

- Load test data
- Translate LPath to SQL using yacc
- Compare with two existing linguistic query engines: CorpusSearch, TGrep2
- Evaluate LPath queries on two data sets
  - Wall Street Journal (WSJ)
  - Switchboard (SWB)
## Test Data Sets

### Statistics of data sets

<table>
<thead>
<tr>
<th></th>
<th>WSJ</th>
<th>SWB</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File Size</strong></td>
<td>35983kB</td>
<td>35880kB</td>
</tr>
<tr>
<td><strong>Tree Nodes</strong></td>
<td>3484899</td>
<td>3972148</td>
</tr>
<tr>
<td><strong>Unique Tags</strong></td>
<td>1274</td>
<td>715</td>
</tr>
<tr>
<td><strong>Maximum Depth</strong></td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>LPath Query</td>
<td>Size of WSJ Result</td>
<td>Size of SWB result</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Q1 //S//_[@lex=saw]]</td>
<td>153</td>
<td>339</td>
</tr>
<tr>
<td>Q2 //VB-&gt;NP</td>
<td>23618</td>
<td>16557</td>
</tr>
<tr>
<td>Q3 //VP/VB--&gt;NN</td>
<td>63857</td>
<td>32386</td>
</tr>
<tr>
<td>Q4 //VP {/VB--&gt;NN}</td>
<td>46116</td>
<td>25305</td>
</tr>
<tr>
<td>Q5 //VP {/NP$}</td>
<td>29923</td>
<td>22554</td>
</tr>
<tr>
<td>Q6 //VP {/NP$}</td>
<td>215104</td>
<td>112159</td>
</tr>
<tr>
<td>Q7 //VP {/^[VB-&gt;NP-&gt;PP$}]</td>
<td>2831</td>
<td>1963</td>
</tr>
<tr>
<td>Q8 //S[/NP/ADJP]</td>
<td>7832</td>
<td>2900</td>
</tr>
<tr>
<td>Q9 //NP [not (/J)]</td>
<td>211392</td>
<td>109311</td>
</tr>
<tr>
<td>Q10 //NP [-&gt;PP [/IN[@lex=of]]=&gt;VP]</td>
<td>192</td>
<td>31</td>
</tr>
<tr>
<td>Q11 //S{[/<em>[@lex=what]--&gt;</em>[@lex=building]]}</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>
## Queries (2)

<table>
<thead>
<tr>
<th>LPath Query</th>
<th>Size of WSJ Result</th>
<th>Size of SWB result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q₁₂ //__[@lex=rapprochement]</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Q₁₃ //__[@lex=1929]</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>Q₁₄ //ADVP-LOC-CLR</td>
<td>60</td>
<td>0</td>
</tr>
<tr>
<td>Q₁₅ //WHPP</td>
<td>87</td>
<td>20</td>
</tr>
<tr>
<td>Q₁₆ //RRC/PP-TMP</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>Q₁₇ //UCP-PRD/ADJP-PRD</td>
<td>17</td>
<td>4</td>
</tr>
<tr>
<td>Q₁₈ //NP/NP/NP/NP/NP/NP</td>
<td>254</td>
<td>12</td>
</tr>
<tr>
<td>Q₁₉ //VP/VP/VP</td>
<td>8769</td>
<td>6093</td>
</tr>
<tr>
<td>Q₂₀ //PP=&gt;SBAR</td>
<td>640</td>
<td>651</td>
</tr>
<tr>
<td>Q₂₁ //ADVP=&gt;ADJP</td>
<td>15</td>
<td>37</td>
</tr>
<tr>
<td>Q₂₂ //NP=&gt;NP=&gt;NP</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Q₂₃ //VP=&gt;VP</td>
<td>20</td>
<td>72</td>
</tr>
</tbody>
</table>
Query Execution Time

Switchboard

Query Execution Time (s)

<table>
<thead>
<tr>
<th>Query</th>
<th>LPath</th>
<th>Tgrep</th>
<th>CorpusSearch</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.001</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>0.1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>1000</td>
<td>10</td>
<td>100</td>
</tr>
</tbody>
</table>

Linguistic Query Language

Steven Bird and Catherine Lai

LPath+ Linguistic Query Language
Scalability as Data Size (WSJ) Increases

![Graph showing scalability as data size increases]

- **Q6**
- **LPath**
- **Tgrep**
- **CorpusSearch**

Steven Bird and Catherine Lai  
LPath+ Linguistic Query Language
**Discussion**

- **LPath engine works well in most cases**
- **LPath is slower than TGrep2 in low selectivity queries**
  - \( Q_{10} : /NP[->&PP[//IN[@lex=of]]]=>VP \)
  - \( Q_{18} : /NP/NP/NP/NP/NP \)
  - \( Q_{22} : /NP=>NP=>NP \)

### Most frequent tags in data sets

<table>
<thead>
<tr>
<th></th>
<th><strong>WSJ</strong></th>
<th></th>
<th><strong>SWB</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tag</td>
<td>Frequency</td>
<td>Tag</td>
<td>Frequency</td>
</tr>
<tr>
<td>1</td>
<td>NP</td>
<td>292430</td>
<td>-DFL-</td>
<td>193708</td>
</tr>
<tr>
<td>2</td>
<td>VP</td>
<td>180405</td>
<td>VP</td>
<td>185259</td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
<td>163935</td>
<td>NP-SBJ</td>
<td>135867</td>
</tr>
<tr>
<td>4</td>
<td>IN</td>
<td>121903</td>
<td></td>
<td>135753</td>
</tr>
<tr>
<td>5</td>
<td>NNP</td>
<td>114053</td>
<td>,</td>
<td>133528</td>
</tr>
<tr>
<td>6</td>
<td>S</td>
<td>107570</td>
<td>S</td>
<td>132336</td>
</tr>
</tbody>
</table>
Discussion

- LPath engine works well in most cases
- LPath is slower than TGrep2 in low selectivity queries
  - $Q_{10} : //NP[->PP[//IN[@lex=of]]]=>VP$
  - $Q_{18} : //NP/NP/NP/NP/NP$
  - $Q_{22} : //NP=>NP=>NP$

Most frequent tags in data sets

<table>
<thead>
<tr>
<th></th>
<th>WSJ</th>
<th></th>
<th>SWB</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tag</td>
<td>Frequency</td>
<td>Tag</td>
<td>Frequency</td>
</tr>
<tr>
<td>1</td>
<td>NP</td>
<td>292430</td>
<td>-DFL-</td>
<td>193708</td>
</tr>
<tr>
<td>2</td>
<td>VP</td>
<td>180405</td>
<td>VP</td>
<td>185259</td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
<td>163935</td>
<td>NP-SBJ</td>
<td>135867</td>
</tr>
<tr>
<td>4</td>
<td>IN</td>
<td>121903</td>
<td>.</td>
<td>135753</td>
</tr>
<tr>
<td>5</td>
<td>NNP</td>
<td>114053</td>
<td>,</td>
<td>133528</td>
</tr>
<tr>
<td>6</td>
<td>S</td>
<td>107570</td>
<td>S</td>
<td>132336</td>
</tr>
</tbody>
</table>
Discussion

- LPath engine works well in most cases
- LPath is slower than TGrep2 in low selectivity queries
  - $Q_{10}$: //NP[->PP[//IN[@lex=of]]]=>VP
  - $Q_{18}$: //NP/NP/NP/NP/NP
  - $Q_{22}$: //NP=>NP=>NP

Most frequent tags in data sets

<table>
<thead>
<tr>
<th></th>
<th>WSJ</th>
<th>SWB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag</td>
<td>Frequency</td>
<td>Tag</td>
</tr>
<tr>
<td>1</td>
<td>NP</td>
<td>-DFL-</td>
</tr>
<tr>
<td>2</td>
<td>VP</td>
<td>VP</td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
<td>NP-SBJ</td>
</tr>
<tr>
<td>4</td>
<td>IN</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>NNP</td>
<td>,</td>
</tr>
<tr>
<td>6</td>
<td>S</td>
<td>S</td>
</tr>
</tbody>
</table>
Discussion

- LPath engine works well in most cases
- LPath is slower than TGrep2 in low selectivity queries
  - $Q_{10} : \text{//NP[->PP[//IN[@lex=of]]]=>VP]$
  - $Q_{18} : \text{/NP/NP/NP/NP/NP\}
  - $Q_{22} : \text{/NP=>NP=>NP}$

Most frequent tags in data sets

<table>
<thead>
<tr>
<th></th>
<th>WSJ</th>
<th>SWB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NP</td>
<td>-DFL-</td>
</tr>
<tr>
<td>2</td>
<td>VP</td>
<td>VP</td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
<td>NP-SBJ</td>
</tr>
<tr>
<td>4</td>
<td>IN</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>NNP</td>
<td>,</td>
</tr>
<tr>
<td>6</td>
<td>S</td>
<td>S</td>
</tr>
</tbody>
</table>

Steven Bird and Catherine Lai

LPath+ Linguistic Query Language
Discussion

- LPath engine works well in most cases
- LPath is slower than TGrep2 in low selectivity queries
  - $Q_{10}: /\NP[->PP[/IN[@lex=of]]]=\VP$
  - $Q_{18}: /\NP/\NP/\NP/\NP/\NP$
  - $Q_{22}: /\NP=\NP=\NP$

Most frequent tags in data sets

<table>
<thead>
<tr>
<th></th>
<th>WSJ</th>
<th></th>
<th></th>
<th>SWB</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tag</td>
<td>Frequency</td>
<td>Tag</td>
<td>Frequency</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>NP</td>
<td>292430</td>
<td>-DFL-</td>
<td>193708</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>VP</td>
<td>180405</td>
<td>VP</td>
<td>185259</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
<td>163935</td>
<td>NP-SBJ</td>
<td>135867</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>IN</td>
<td>121903</td>
<td>,</td>
<td>135753</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>NNP</td>
<td>114053</td>
<td>,</td>
<td>133528</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>S</td>
<td>107570</td>
<td>S</td>
<td>132336</td>
<td></td>
</tr>
</tbody>
</table>

Steven Bird and Catherine Lai LPath: Linguistic Query Language
Conclusions

- Path languages are very suitable for linguistics
- Proposed LPath language which extends XPath to support immediate precedence, subtree scoping, edge alignment
- Designed a labeling scheme to speed up LPath navigations
- Implemented an LPath interpreter on top of SQL
- General purpose, scalable
Conclusions

- Path languages are very suitable for linguistics
- Proposed LPath language which extends XPath to support immediate precedence, subtree scoping, edge alignment
- Designed a labeling scheme to speed up LPath navigations
- Implemented an LPath interpreter on top of SQL
- General purpose, scalable
Conclusions

- Path languages are very suitable for linguistics
- Proposed LPath language which extends XPath to support immediate precedence, subtree scoping, edge alignment
- Designed a labeling scheme to speed up LPath navigations
- Implemented an LPath interpreter on top of SQL
- General purpose, scalable
Conclusions

• Path languages are very suitable for linguistics
• Proposed LPath language which extends XPath to support immediate precedence, subtree scoping, edge alignment
• Designed a labeling scheme to speed up LPath navigations
• Implemented an LPath interpreter on top of SQL

General purpose, scalable
Conclusions

- Path languages are very suitable for linguistics
- Proposed LPath language which extends XPath to support immediate precedence, subtree scoping, edge alignment
- Designed a labeling scheme to speed up LPath navigations
- Implemented an LPath interpreter on top of SQL
- General purpose, scalable
Ongoing Work: Update Language, Tree Transformations

MOVE //NP/PP RIGHT

Ongoing Work: Update Language, Tree Transformations
Who is going to parse all those trees?
- automatic parsing (web scale)
- semantic web (web as trees)

Who is going to type those queries?
- graphical interpreter
- tree drawings $\rightarrow$ LPath$^+$ $\rightarrow$ SQL
Who is going to parse all those trees?
- automatic parsing (web scale)
- semantic web (web as trees)

Who is going to type those queries?
- graphical interpreter
- tree drawings → LPath⁺ → SQL
Who is going to parse all those trees?
- automatic parsing (web scale)
- semantic web (web as trees)

Who is going to type those queries?
- graphical interpreter
- tree drawings $\rightarrow$ LPath$^+$ $\rightarrow$ SQL
Who is going to parse all those trees?
- automatic parsing (web scale)
- semantic web (web as trees)

Who is going to type those queries?
- graphical interpreter
- tree drawings → LPath$^+$ → SQL
Who is going to parse all those trees?
- automatic parsing (web scale)
- semantic web (web as trees)

Who is going to type those queries?
- graphical interpreter
  - tree drawings $\rightarrow$ LPath$^+$ $\rightarrow$ SQL
Who is going to parse all those trees?
- automatic parsing (web scale)
- semantic web (web as trees)

Who is going to type those queries?
- graphical interpreter
- tree drawings $\rightarrow$ LPath$^+$ $\rightarrow$ SQL