

# Probabilistic Tree Automata and Conditional Random Fields for Trees

Rémi Gilleron - joint work with Marc Tommasi and Florent Jousse

Mostrare project  
Lille university and INRIA Futurs  
[www.grappa.univ-lille3.fr/mostrare](http://www.grappa.univ-lille3.fr/mostrare)

Workshop on Tree Automata - June 8, 2006

# Outline

- 1 Motivations
- 2 Probabilistic Tree Automata
- 3 Conditional Random Fields for trees
- 4 Conclusion and perspectives

# Semantic Integration *Doan & Halevy 2005*

## Context:

- distributed data sources
- according to different schemas
- with ambiguous contents (texts)
- manually or automatically produced

**Objective:** resolve heterogeneities w.r.t. the schemas and their data

## Applications:

- data integration systems
- peer data management
- model management

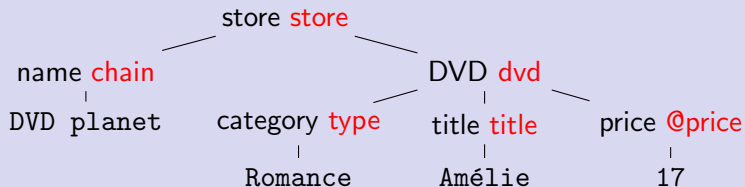
**Focus:** on XML databases, i.e. **trees**

# Tasks in Semantic integration

## Some basic Tasks:

- Information extraction from Web documents
- Schema matching

## in the DVD domain



- XML data transformation

## And different approaches:

- Program-based or Rule-based solutions
- **Learning-based solutions**

# Learning-based approach for Schema Matching

## The problem:

- **Given** a source schema and a target schema; data instances; maybe extra-knowledge: dictionaries, ontologies, matches, constraints
- **Find**
  - ▶ a matching between elements: category  $\rightarrow$  type
  - ▶ or a matching between elements and attributes: price  $\rightarrow$  @price
  - ▶ or complex matchings
- **or find** XML data transformation: elaborate matches into mappings, to enable translation of queries and data across schemas

## Machine Learning for tree structured data:

- Use the existing toolbox of algorithms (*Doan et al, ...*)
- Grammatical inference (*Niehren et al, Raeymaeckers et al*)
- **Probabilistic models and algorithms**

# Probabilistic Models for Trees

XML is a good excuse to do tree automata theory

XML is a good excuse to do machine learning theory and to study probabilistic models for trees

# Outline

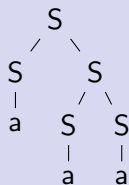
- 1 Motivations
- 2 Probabilistic Tree Automata**
- 3 Conditional Random Fields for trees
- 4 Conclusion and perspectives

# Probabilistic Automata for sequences

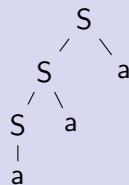
## HMMS = PAS

- Hidden Markov Models and (non deterministic) automata are equivalent
- Deterministic PAS are strictly less expressive than PAS
- PAS are a special case of multiplicity automata (or weighted automata) defining rational power series over  $[0,1]$
- Widely used in speech recognition, natural language processing, information extraction, bio-informatics

$$S \rightarrow SS \frac{1}{6} \mid Sa \frac{1}{3} \mid a \frac{1}{2}$$



$$p(\mathbf{x} = aaa, \mathbf{y}) = \frac{1}{6} \times \frac{1}{2} \times \frac{1}{6} \times \frac{1}{2} \times \frac{1}{2}$$



$$p(\mathbf{x} = aaa, \mathbf{z}) = \frac{1}{3} \times \frac{1}{3} \times \frac{1}{2}$$

- PCFGs as **generative models**: they define joint distributions  $p(\mathbf{x}, \mathbf{y})$
- $p(\mathbf{x}) = \sum_{\mathbf{y}} p(\mathbf{x}, \mathbf{y})$ ;  $p(\mathbf{y} \mid \mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})}$
- $S \rightarrow SS \alpha \mid a 1 - \alpha$  is **consistent** if  $\alpha \leq \frac{1}{2}$

# The three questions for generative models

A set of rules and a real-valued parameter vector  $\Lambda$

## Three problems:

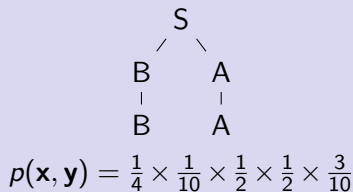
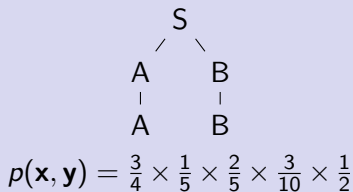
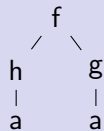
- **Inference:** given an observable  $\mathbf{x}$ , compute  $p(\mathbf{x})$
- **Inference:** given an observable  $\mathbf{x}$ , compute  $\operatorname{argmax}_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$
- **Training:** given a sample set of observables, learn the real-valued parameter vector  $\Lambda$  maximising the likelihood of  $S$ .

## Algorithms

- **forward-backward** or **inside-outside** variables: dynamic programming techniques to memoize results
- **Viterbi** paths or trees
- **Baum-Welch:** iterate Expectation (find the most likely) and Maximisation (find parameters values) until convergence

# Probabilistic Regular Tree Grammars - ranked trees

$$A \rightarrow g(A) \frac{1}{2} \mid h(A) \frac{1}{5} \mid a \frac{3}{10} \qquad S \rightarrow f(A, B) \frac{3}{4} \mid f(B, A) \frac{1}{4}$$
$$B \rightarrow g(B) \frac{2}{5} \mid h(B) \frac{1}{10} \mid a \frac{1}{2}$$



- Regular Tree Grammars (or  $\downarrow$ -TAGs) + real-valued parameter vector
- Derivation tree languages = local languages  $\subset$  regular languages
- Define joint (probability) distributions  $p(\mathbf{x}, \mathbf{y})$
- Algorithms extend from PCFGs to PRTGs

# Probabilistic Regular Tree Grammars - unranked trees

## Unranked PRTGs

- rules of the form:  $r : A \rightarrow f(\text{regexp}(A, f))$   $w(r)$  such that 
$$\sum_{lhs(r)=A} w(r) = 1$$
- every  $\text{regexp}(A, f)$  is defined by a probabilistic model for sequences

## Binary encodings

- Fix a binary encoding
- and consider PRTGs over binary trees

PRTGs are **generative models** for unranked trees

## Questions:

- Compare unranked PRTGs w.r.t. probabilistic models for sequences
- Compare unranked PRTGs with binary PRTGs

# Probabilistic Tree Automata as conditional models

$$S \rightarrow f(A, B) \mid f(B, A); A \rightarrow g(A) \mid h(A) \mid a; B \rightarrow g(B) \mid h(B) \mid a$$

reversing arrows, i.e. bottom-up case:

$$a \rightarrow A \frac{1}{4} \mid B \frac{3}{4};$$
$$h(A) \rightarrow A; g(A) \rightarrow A; h(B) \rightarrow B; g(B) \rightarrow B; f(A, B) \rightarrow S; f(B, A) \rightarrow S$$
$$p(S(A, B) \mid f(a, a)) = \frac{1}{4}; p(S(B, A) \mid f(a, a)) = \frac{3}{4}$$

- If rule probabilities satisfy:  $\sum_{rhs(r)=q} w(r) = 1$ , we define joint distributions
- If rule probabilities satisfy:  $\sum_{lhs(r)=f(q_1, q_2)} w(r) = 1$ , we define **conditional (probability) distributions**  $p(\mathbf{y} \mid \mathbf{x})$
- For an observable tree  $\mathbf{x}$ ,  $f(q_1, q_2) \rightarrow q$   $w$  can be viewed as: if nodes  $x_{n,1}$  and  $x_{n,2}$  are labeled by  $q_1$  and  $q_2$ , if symbol  $x_n$  is  $f$ , then label  $y_n$  by  $q$  with probability  $w$ .

# Probabilistic tree automata - summary

## generative models

- PRTGs define generative models, i.e. distributions  $p(\mathbf{x}, \mathbf{y})$
- In the ranked case, PRTGs are weighted tree automata related to tree series (*Seidl, Kuich, Vogler, ...*). Following (*Denis et al, COLT 2004 & 2006*) for sequences, investigate the special case of probability distributions and learning weighted tree automata
- The unranked case needs to be investigated

But, generative models need to model observables  $\mathbf{x}$

## conditional models

- PTAs can define conditional distributions  $p(\mathbf{y} \mid \mathbf{x})$
- For sequences, PTAs have been extended to MEMMs (*Mc Callum, ICML 2000*) and to CRFs (*Lafferty et al., ICML 2001*)

# Outline

- 1 Motivations
- 2 Probabilistic Tree Automata
- 3 Conditional Random Fields for trees**
- 4 Conclusion and perspectives

# Conditional Random Fields

## In general

- Conditional Model for labeling introduced by (*Lafferty et al., ICML 2001*); a recent survey: “An Introduction to Conditional Random Fields for Relational Learning” (*Sutton and McCallum, 2006*)
- A graphical structure defines dependencies and Markov conditions to simplify conditional distributions

## For sequences

- linear chain CRFs: positions and pairs of consecutive positions
- Extensively used for sequence labeling. Very good results in:
  - ▶ Information Extraction (*Pinto, McCallum et al, SIGIR 2003*) (*Sarawagi and Cohen, NIPS 2004*)
  - ▶ Part-Of-Speech tagging (*Sha and Pereira, HLT/NAACL-03*)
  - ▶ Named Entities *McCallum and Li, CoNLL 2003*

# Conditional Random Fields for trees

## XCRFs *Jousse et al, 2006*

- observable tree  $\mathbf{x}$  and labeling tree  $\mathbf{y}$  define random fields  $X_i$  and  $Y_i$  indexed by nodes
- an XCRF is defined by a set of feature functions  $\{f_k\}$  and a real-valued parameter vector  $\Lambda = \{\lambda_k\}$
- Feature functions are real-valued functions whose parameters are:
  - ▶ labels in the triangle clique  $(n, n.i, n.(i+1))$
  - ▶  $\mathbf{x}$ : the whole observable (input document)
- $$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{n,n.i} \exp \left( \sum_k \lambda_k f_k(y_n, y_{n.i}, y_{n.(i+1)}, \mathbf{x}, n, n.i) \right)$$
  - ▶  $Z(\mathbf{x})$  is a normalization factor (sum over all  $\mathbf{y}$ )
  - ▶ exponential form

# Feature functions

## Schema matching in the DVD domain



## Node feature function:

$$f(y_n, \mathbf{x}) = 1 \text{ if } \begin{cases} y_n = \text{"title"} \text{ and} \\ x_n = \text{"title"} \text{ and} \\ \text{father}(x_n) = \text{"DVD"} \end{cases}$$

# Feature functions

## Schema matching in the DVD domain



## Edge feature function:

$$f(y_n, y_{n,i}, x) = 1 \text{ if } \begin{cases} y_n = \text{"store"} \text{ and} \\ y_{n,i} = \text{"chain"} \text{ and} \\ x_{n,i} = \text{"name"} \text{ and} \\ i = 0 \end{cases}$$

# Feature functions

## Schema matching in the DVD domain



## Triangle feature function:

$$f(y_n, y_{n.i}, y_{n.(i+1)}, x) = 1 \text{ if } \begin{cases} y_n = \text{"dvd"} \text{ and} \\ y_{n.i} = \text{"title"} \text{ and} \\ y_{n.(i+1)} = \text{"@price"} \text{ and} \\ x_{n.(i+1)} = \text{"price"} \end{cases}$$

# PRTGs and XCRFs

## Theorem

*For ranked trees, conditional distributions defined by PRTGs can be defined by XCRFs*

## Sketch

- For every rule  $r : A \rightarrow f(B, C)$  w, define a triangle feature function

$$f_r(y_n, y_{n.i}, y_{n.(i+1)}, \mathbf{x}) = \begin{cases} 1 & \text{if } x_n = f, y_n = A, y_{n.i} = B, y_{n.(i+1)} = C \\ 0 & \text{otherwise} \end{cases}$$

and set  $\lambda_r = \log w$

- Complete the automaton, define features with weight  $-\infty$
- $p_{crf(A)}(\mathbf{y} \mid \mathbf{x}) = p_A(\mathbf{y} \mid \mathbf{x}) = \frac{p_A(\mathbf{x}, \mathbf{y})}{p_A(\mathbf{x})}$

# Inference

## Labeling with XCRFs

**Given** an XCRF and an XML document  $\mathbf{x}$ , **find** the most likely labeling tree

$$\hat{y} = \operatorname{argmax}_y p(y|\mathbf{x})$$

## Algorithm

- The algorithm extends over the **Viterbi algorithm** for HMMs and PCFGs.
- The complexity is in  $O(C \times N \times M^3)$  where :
  - ▶  $C$  is the number of feature functions
  - ▶  $N$  is the number of nodes in the XML tree
  - ▶  $M$  is the number of labels in the XCRF

# Training

## Training XCRFs

- **Given** an XCRF defined by a set of feature functions  $\{f_k\}$  and a sample  $S$  of pairs  $(\mathbf{x}^i, \mathbf{y}^i)$
- **Estimate** the real-valued parameter vector  $\hat{\Lambda} = \operatorname{argmax}_{\Lambda} p(S)$

## Algorithm

- The criterion we used for parameter estimation is the **maximum penalized log-likelihood**.
- A gradient ascent is used: LBFG-S. (*Wallach, CLUK 2003*)
- Complexity is in  $O(C \times N \times M^3 \times G)$  where :
  - ▶  $C$  is the number of feature functions
  - ▶  $N$  is the number of nodes in the XML tree
  - ▶  $M$  is the number of labels in the XCRF
  - ▶  $G$  is the number of gradient steps

# The XCRF System

## Implementation

- The XCRF System has been implemented in JAVA
- An XCRF (labels, feature functions) is defined with an XML file
- Features can be defined over element nodes, attribute nodes and text nodes via XPATH expressions
- Can be found at: <http://treecrf.gforge.inria.fr>

## Experimental results

Preliminary results are promising for:

- structured information extraction (XML challenge)
- schema matching (real estate domain, *Doan*)

# Outline

- 1 Motivations
- 2 Probabilistic Tree Automata
- 3 Conditional Random Fields for trees
- 4 Conclusion and perspectives**

# Conclusion

## PRTGs

PRTGs should be investigated in the unranked case: definitions, expressivity, tree series, inference and training algorithms

## The XCRF System

- Other learning criteria and approximate learning
- New algorithms for large sets of labels ( $M^3$ )
- Combine XCRFs (structural view) with linear chain CRFs (textual view)

## Tree transformations

From schema matching to mappings, i.e. from tree labeling to tree transformation: labels as edit operations

# Conclusion

## PRTGs

PRTGs should be investigated in the unranked case: definitions, expressivity, tree series, inference and training algorithms

## The XCRF System

- Other learning criteria and approximate learning
- New algorithms for large sets of labels ( $M^3$ )
- Combine XCRFs (structural view) with linear chain CRFs (textual view)

## Tree transformations

From schema matching to mappings, i.e. from tree labeling to tree transformation: labels as edit operations

Thank you