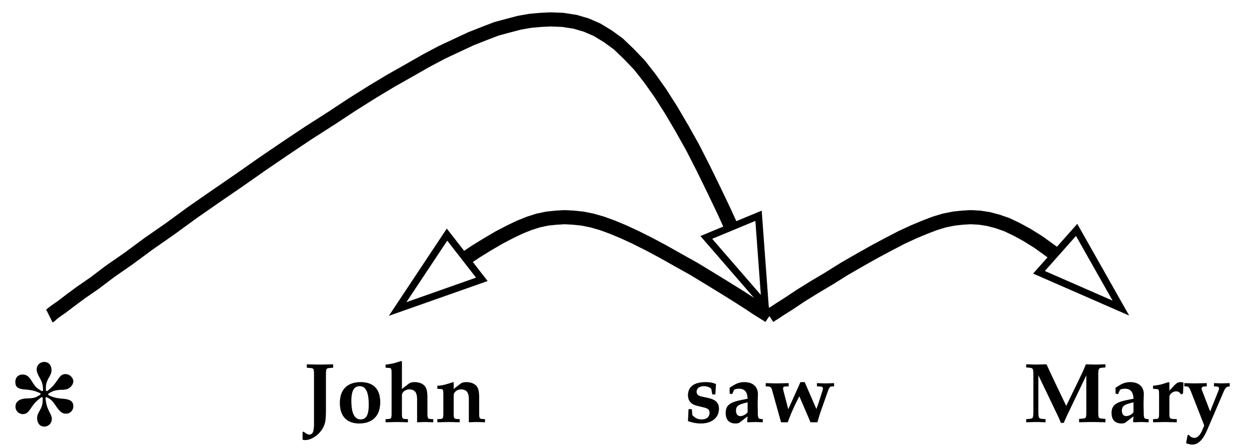
Structured Prediction Models via the Matrix-Tree Theorem

Terry Koo maestro@csail.mit.edu
Amir Globerson gamir@csail.mit.edu
Xavier Carreras carreras@csail.mit.edu
Michael Collins mcollins@csail.mit.edu

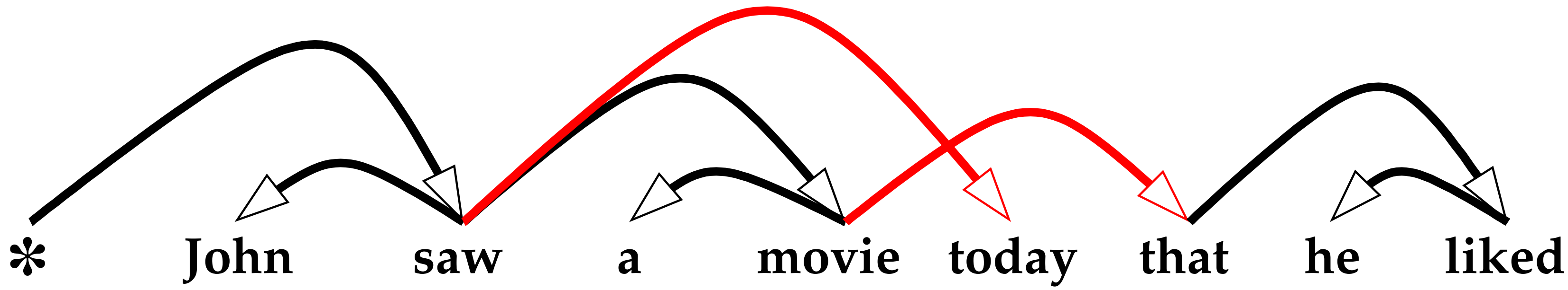
MIT Computer Science and Artificial Intelligence Laboratory

Dependency parsing



- Syntactic structure represented by head-modifier dependencies

Projective vs. non-projective structures



- Non-projective structures allow *crossing dependencies*
- Frequent in languages like Czech, Dutch, etc.
- Non-projective parsing is max-spanning-tree (McDonald et al., 2005)

Contributions of this work

- Fundamental inference algorithms that sum over possible structures:

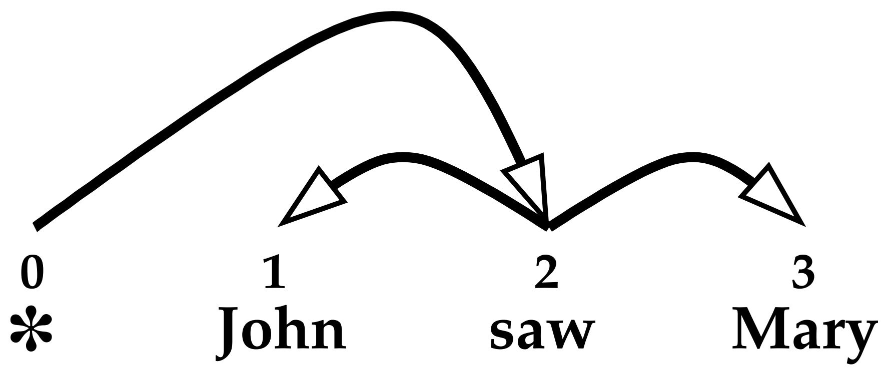
Model type	**Inference Algorithm**
HMM	Forward-Backward
Graphical Model	Belief Propagation
PCFG	Inside-Outside
Projective Dep. Trees	Inside-Outside
Non-projective Dep. Trees	**??**

- This talk:
  - *Inside-outside-style* algorithms for non-projective dependency structures
  - An application: training log-linear and max-margin parsers
  - Independently-developed work: Smith and Smith (2007), McDonald and Satta (2007)

Overview

- Background
- Matrix-Tree-based inference
- Experiments

Edge-factored structured prediction



- A dependency tree y is a set of head-modifier dependencies (McDonald et al., 2005; Eisner, 1996)
  - (h, m) is a dependency with feature vector $\mathbf{f}(\mathbf{x}, h, m)$
  - $\mathcal{Y}(\mathbf{x})$ is the set of all possible trees for sentence $\mathbf{x}$

$$y^* \quad = \quad \operatorname*{argmax}_{y \in \mathcal{Y}(\mathbf{x})} \sum_{(h,m) \in y} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m)$$

Training log-linear dependency parsers

- Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, minimize

$$L(\mathbf{w}) \quad = \quad \frac{C}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} \log P(y_i \mid \mathbf{x}_i; \mathbf{w})$$

Training log-linear dependency parsers

- Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, minimize

$$L(\mathbf{w}) = \frac{C}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} \log P(y_i \,|\, \mathbf{x}_i; \mathbf{w})$$

- Log-linear distribution over trees

$$P(y \,|\, \mathbf{x}; \mathbf{w}) = \frac{1}{Z(\mathbf{x}; \mathbf{w})} \exp\left\{ \sum_{(h,m) \in y} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m) \right\}$$

$$Z(\mathbf{x}; \mathbf{w}) = \sum_{y \in \mathcal{Y}(\mathbf{x})} \exp\left\{ \sum_{(h,m) \in y} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m) \right\}$$

Training log-linear dependency parsers

- Gradient-based optimizers evaluate $L(\mathbf{w})$ and $\frac{\partial L}{\partial \mathbf{w}}$

$$
\begin{aligned}
L(\mathbf{w}) \quad = \quad & \frac{C}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} \sum_{(h,m)\in y_i} \mathbf{w}\cdot\mathbf{f}(\mathbf{x}_i, h, m) \\
& + \sum_{i=1}^{N} \log Z(\mathbf{x}_i; \mathbf{w})
\end{aligned}
$$

- Main difficulty: computation of the *partition functions*

Training log-linear dependency parsers

- Gradient-based optimizers evaluate $L(\mathbf{w})$ and $\frac{\partial L}{\partial \mathbf{w}}$

$$\begin{aligned}\frac{\partial L}{\partial \mathbf{w}} \;&=\; C\mathbf{w} - \sum_{i=1}^{N} \sum_{(h,m)\in y_i} \mathbf{f}(\mathbf{x}_i, h, m) \\ &\quad + \sum_{i=1}^{N} \sum_{h',m'} P(h' \rightarrow m' \,|\, \mathbf{x}; \mathbf{w}) \mathbf{f}(\mathbf{x}_i, h', m')\end{aligned}$$

- The *marginals* are edge-appearance probabilities

$$P(h \rightarrow m \,|\, \mathbf{x}; \mathbf{w}) \;=\; \sum_{y\in\mathcal{Y}(\mathbf{x})\,:\,(h,m)\in y} P(y \,|\, \mathbf{x}; \mathbf{w})$$

Generalized log-linear inference

- Vector $\boldsymbol{\theta}$ with parameter $\theta_{h,m}$ for each dependency

$$P(y \,|\, \mathbf{x}; \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{x}; \boldsymbol{\theta})} \exp\left\{ \sum_{(h,m) \in y} \theta_{h,m} \right\}$$

$$Z(\mathbf{x}; \boldsymbol{\theta}) = \sum_{y \in \mathcal{Y}(\mathbf{x})} \exp\left\{ \sum_{(h,m) \in y} \theta_{h,m} \right\}$$

$$P(h \to m \,|\, \mathbf{x}; \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{x}; \boldsymbol{\theta})} \sum_{y \in \mathcal{Y}(\mathbf{x}) \,:\, (h,m) \in y} \exp\left\{ \sum_{(h,m) \in y} \theta_{h,m} \right\}$$

- E.g., $\theta_{h,m} = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, h, m)$

Applications of log-linear inference

- Generalized inference engine that takes $\boldsymbol{\theta}$ as input
  - Different definitions of $\boldsymbol{\theta}$ can be used for log-linear or max-margin training

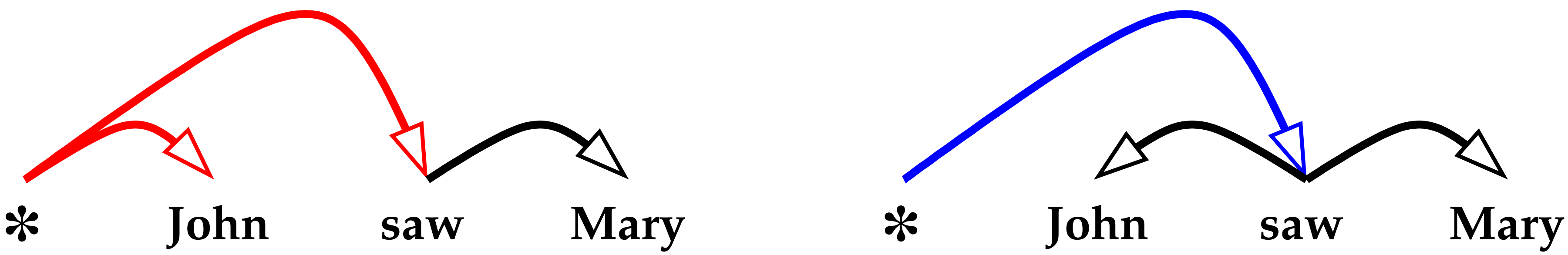
$$\begin{aligned}
\mathbf{w}^*_{LL} &= \operatorname*{argmin}_{\mathbf{w}} \left[\frac{C}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} \log P(y_i \mid \mathbf{x}_i; \mathbf{w}) \right] \\
\mathbf{w}^*_{MM} &= \operatorname*{argmin}_{\mathbf{w}} \left[\frac{C}{2}||\mathbf{w}||^2 + \sum_{i=1}^{N} \max_{y} \left(E_{i,y} - m_{i,y}(\mathbf{w}) \right) \right]
\end{aligned}$$

- Exponentiated-gradient updates for max-margin models
  - Bartlett, Collins, Taskar and McAllester (2004)
  - Globerson, Koo, Carreras and Collins (2007)

Overview

- Background
- Matrix-Tree-based inference
- Experiments

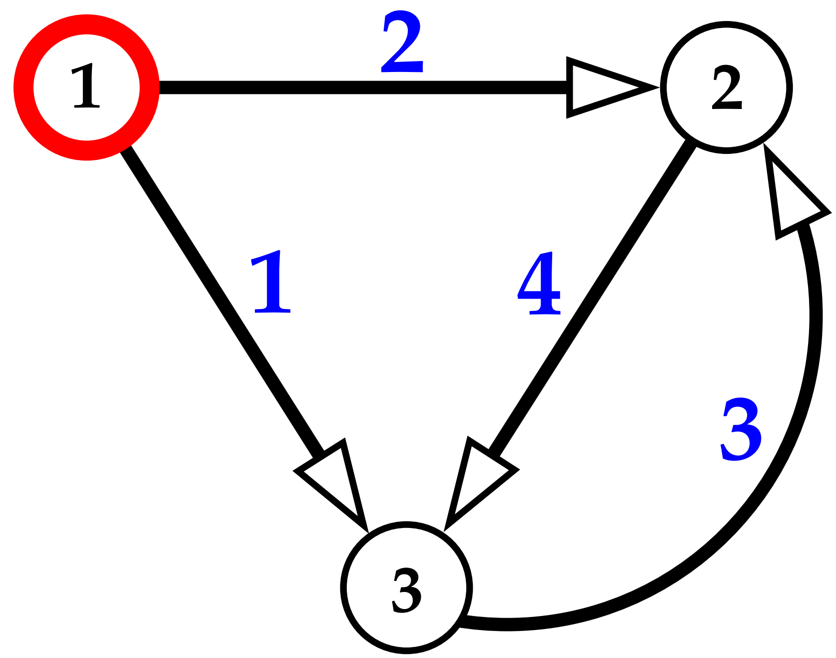
Single-root vs. multi-root structures



- Multi-root structures allow *multiple* edges from $*$
- Single-root structures have *exactly one* edge from $*$
- Independent adaptations of the Matrix-Tree Theorem: Smith and Smith (2007), McDonald and Satta (2007)

Matrix-Tree Theorem (Tutte, 1984)

- Given:
  1. Directed graph G
  2. Edge weights $\boldsymbol{\theta}$
  3. A node r in G

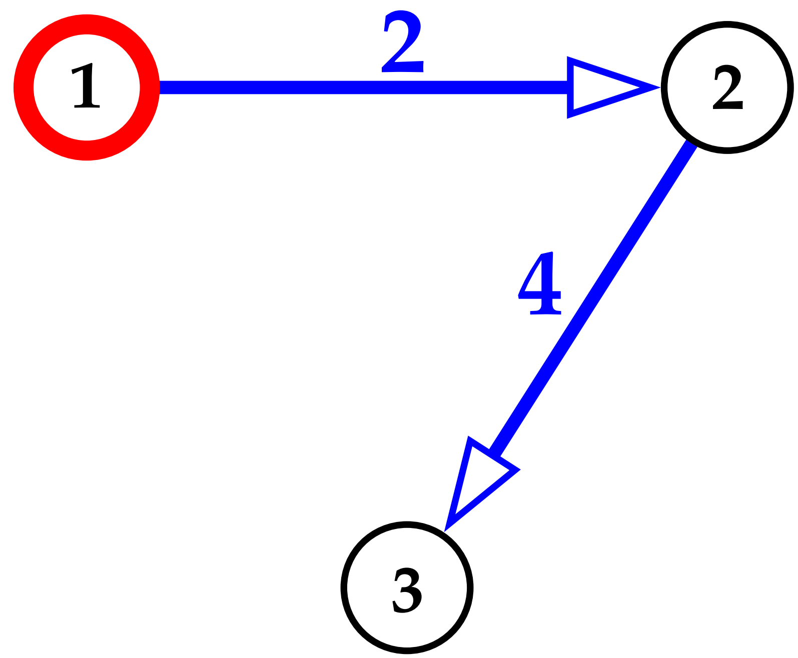


A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of G *rooted at* r

Matrix-Tree Theorem (Tutte, 1984)

- Given:

  1. Directed graph G
  2. Edge weights $\boldsymbol{\theta}$
  3. A node r in G

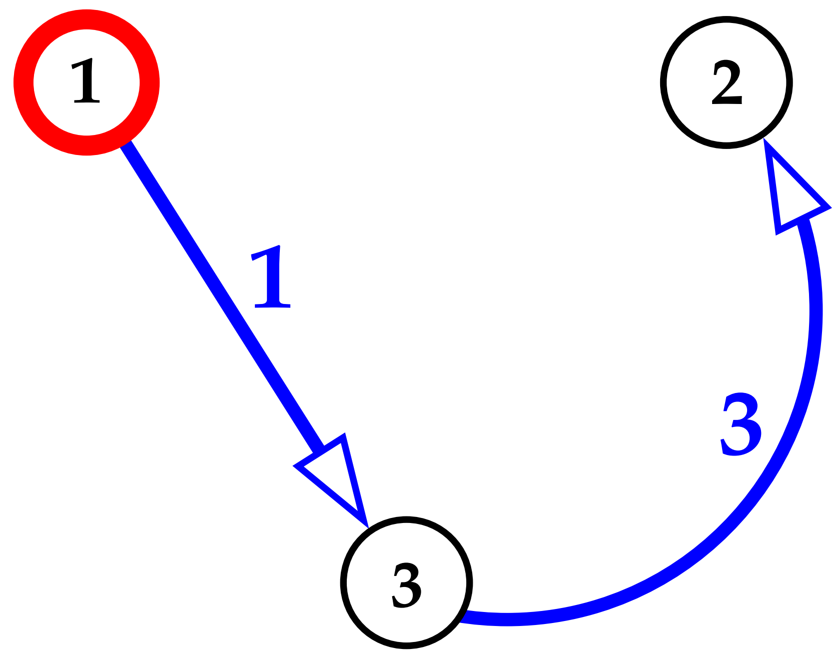


A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of G *rooted at* r

$$\sum \;=\; \exp\{2+4\}$$

Matrix-Tree Theorem (Tutte, 1984)

- Given:
  1. Directed graph G
  2. Edge weights $\boldsymbol{\theta}$
  3. A node r in G



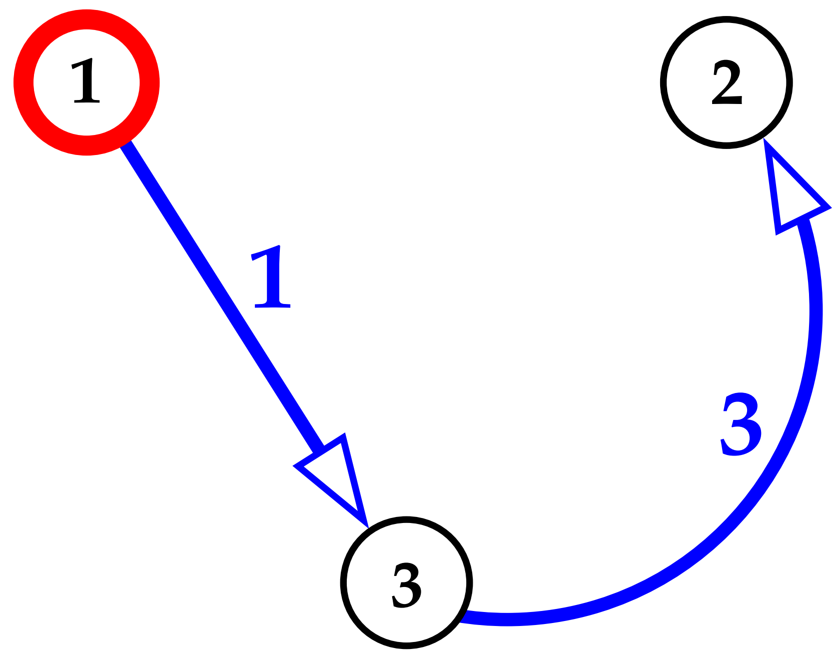
A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of G *rooted at* r

$$\sum \;=\; \exp\{2+4\} + \exp\{1+3\}$$

Matrix-Tree Theorem (Tutte, 1984)

- Given:

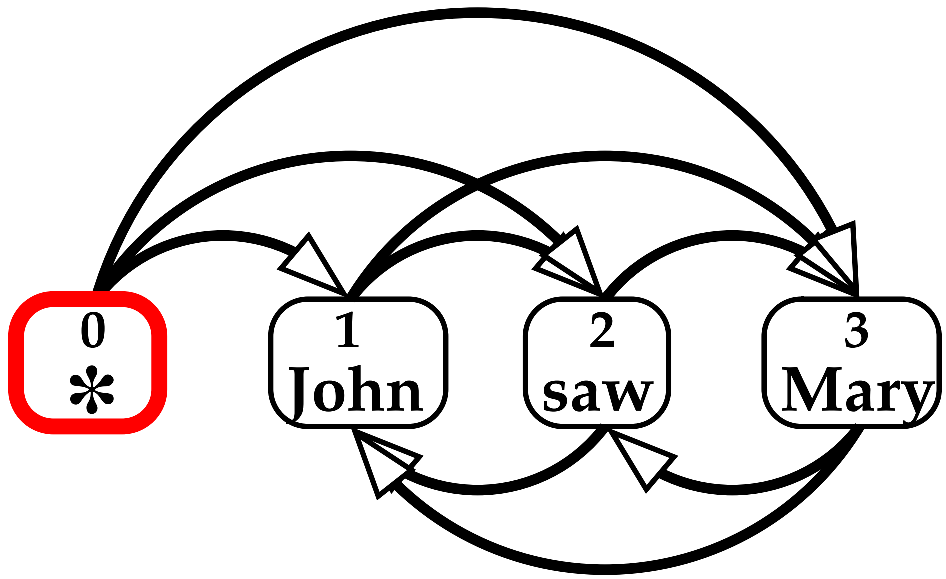
  1. Directed graph G
  2. Edge weights $\boldsymbol{\theta}$
  3. A node r in G



A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of G *rooted at* r

$$\sum \;=\; \exp\{2+4\} + \exp\{1+3\} \;=\; \det(L^{(1)})$$

Multi-root partition function



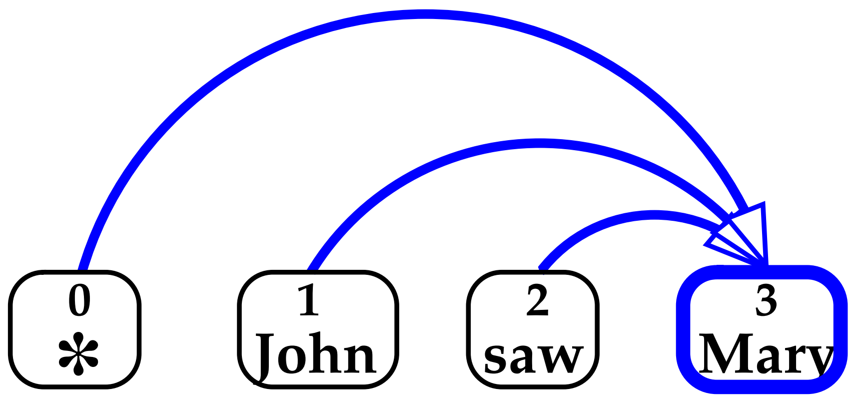
- Edge weights $\boldsymbol{\theta}$, root $r = 0$
- $\det(L^{(0)}) \;=\;$ non-projective *multi-root* partition function

Construction of $L^{(0)}$

- $L^{(0)}$ has a simple construction

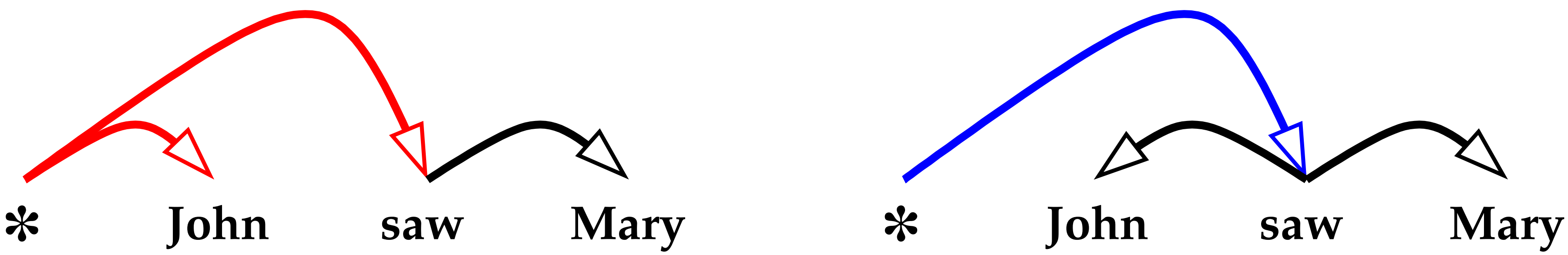
$$\begin{aligned} \text{off-diagonal:} \quad L^{(0)}_{h,m} &= -\exp\{\theta_{h,m}\} \\ \text{on-diagonal:} \quad L^{(0)}_{m,m} &= \sum_{h'=0}^{n} \exp\{\theta_{h,m}\} \end{aligned}$$

- E.g., $L^{(0)}_{3,3}$



- The determinant of $L^{(0)}$ can be evaluated in $O(n^3)$ time

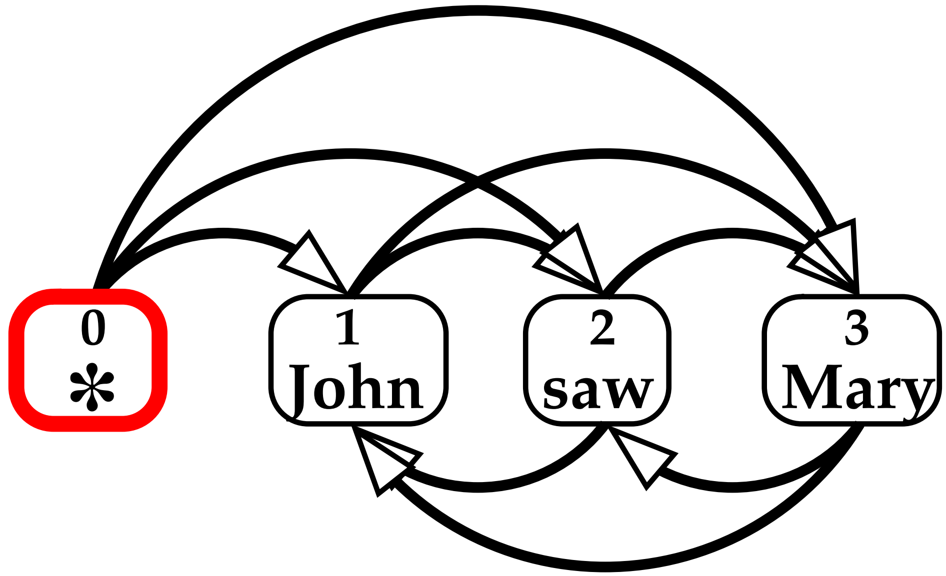
Single-root vs. multi-root structures



- Multi-root structures allow *multiple* edges from *
- Single-root structures have *exactly one* edge from *
- Independent adaptations of the Matrix-Tree Theorem: Smith and Smith (2007), McDonald and Satta (2007)

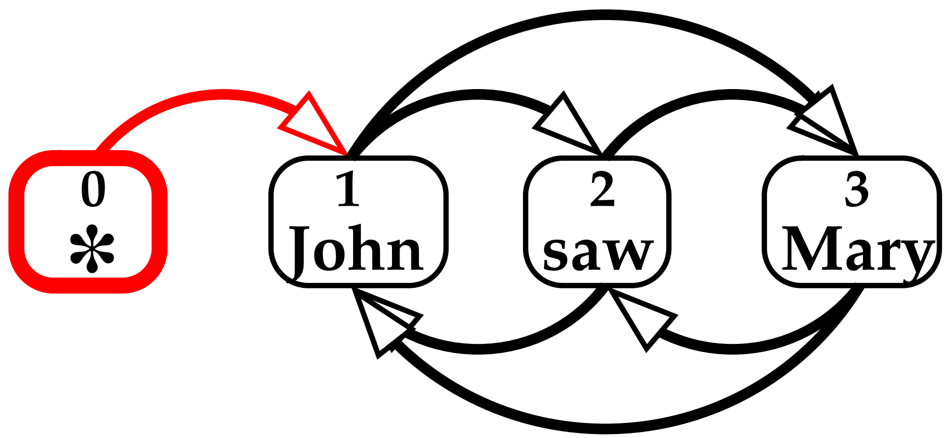
Single-root partition function

- Naïve method for computing the single-root non-projective partition function



Single-root partition function

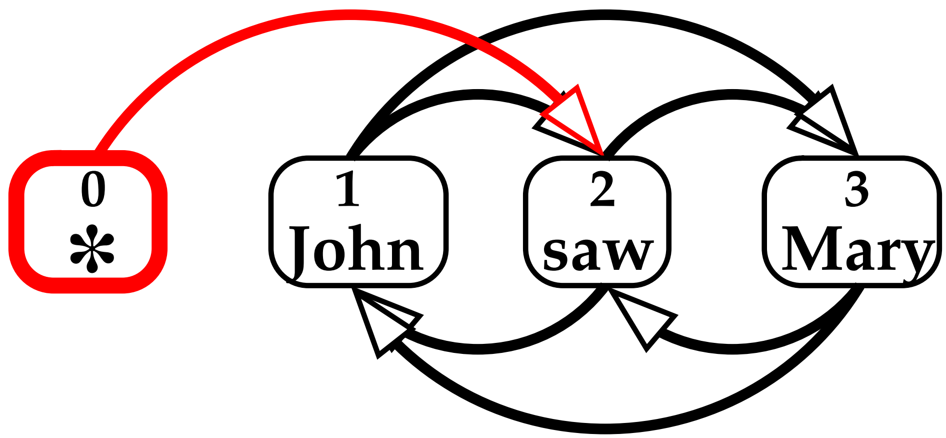
- Naïve method for computing the single-root non-projective partition function



- Exclude all root edges except $(0, 1)$
- Computing n determinants requires $O(n^4)$ time

Single-root partition function

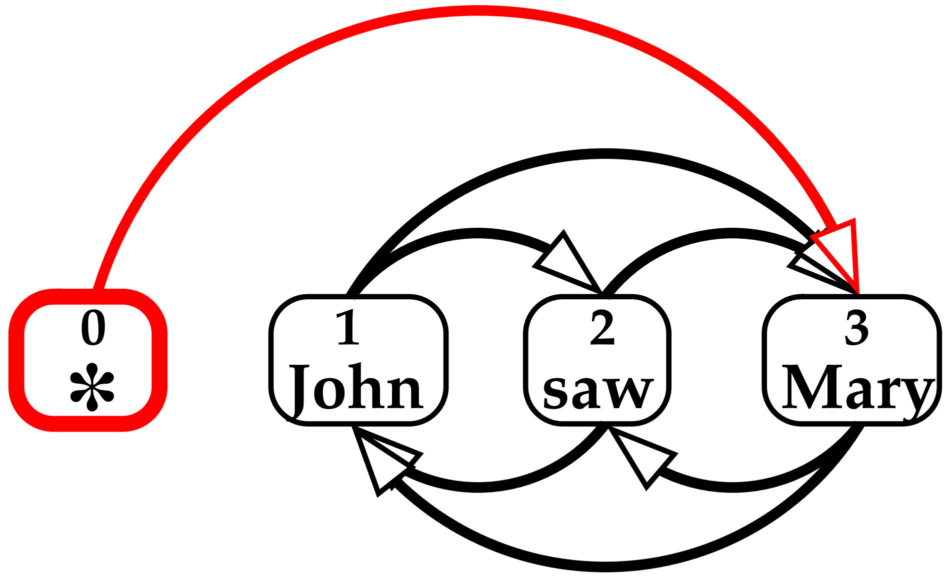
- Naïve method for computing the single-root non-projective partition function



- Exclude all root edges except $(0, 2)$
- Computing n determinants requires $O(n^4)$ time

Single-root partition function

- Naïve method for computing the single-root non-projective partition function



- Exclude all root edges except $(0, 3)$
- Computing n determinants requires $O(n^4)$ time

Single-root partition function

- An alternate matrix $\hat{L}$ can be constructed such that $\det(\hat{L})$ is the single-root partition function

$$\begin{aligned}
\text{first row:} \quad \hat{L}_{1,m} &= \exp\left\{\theta_{0,m}\right\} \\
\text{other rows, on-diagonal:} \quad \hat{L}_{m,m} &= \sum_{h'=1}^{n} \exp\left\{\theta_{h,m}\right\} \\
\text{other rows, off-diagonal:} \quad \hat{L}_{h,m} &= -\exp\left\{\theta_{h,m}\right\}
\end{aligned}$$

- Single-root partition function requires $O(n^3)$ time

Non-projective marginals

- The log-partition generates the marginals

$$P(h \to m \mid \mathbf{x}; \boldsymbol{\theta}) = \frac{\partial \log Z(\mathbf{x}; \boldsymbol{\theta})}{\partial \theta_{h,m}} = \frac{\partial \log \det(\hat{L})}{\partial \theta_{h,m}}$$
$$= \sum_{h',m'} \frac{\partial \log \det(\hat{L})}{\partial \hat{L}_{h',m'}} \frac{\partial \hat{L}_{h',m'}}{\partial \theta_{h,m}}$$

- Derivative of log-determinant: $\dfrac{\partial \log \det(\hat{L})}{\partial \hat{L}} = \left(\hat{L}^{-1}\right)^T$

- Complexity dominated by matrix inverse, $O(n^3)$

Summary of non-projective inference

- Partition function: matrix determinant, $O(n^3)$
- Marginals: matrix inverse, $O(n^3)$
- Single-root inference: $\hat{L}$
- Multi-root inference: $L^{(0)}$

Overview

- Background
- Matrix-Tree-based inference
- Experiments

Log-linear and max-margin training

- Log-linear training

$$\mathbf{w}^*_{LL} \;=\; \operatorname*{argmin}_{\mathbf{w}} \left[\frac{C}{2} ||\mathbf{w}||^2 - \sum_{i=1}^{N} \log P(y_i \,|\, \mathbf{x}_i; \mathbf{w}) \right]$$

- Max-margin training

$$\mathbf{w}^*_{MM} \;=\; \operatorname*{argmin}_{\mathbf{w}} \left[\frac{C}{2} ||\mathbf{w}||^2 + \sum_{i=1}^{N} \max_{y} \left(E_{i,y} - m_{i,y}(\mathbf{w}) \right) \right]$$

Multilingual parsing experiments

- Six languages from CoNLL 2006 shared task
- Training algorithms: averaged perceptron, log-linear models, max-margin models
- Projective models vs. non-projective models
- Single-root models vs. multi-root models

Multilingual parsing experiments

Dutch (4.93%cd)	Projective Training	Non-Projective Training
Perceptron	77.17	78.83
Log-Linear	76.23	79.55
Max-Margin	76.53	79.69

- Non-projective training helps on non-projective languages

Multilingual parsing experiments

Spanish (0.06%cd)	Projective Training	Non-Projective Training
Perceptron	81.19	80.02
Log-Linear	81.75	81.57
Max-Margin	81.71	81.93

- Non-projective training doesn't hurt on projective languages

Multilingual parsing experiments

- Results across all 6 languages (Arabic, Dutch, Japanese, Slovene, Spanish, Turkish)

Perceptron	79.05
Log-Linear	79.71
Max-Margin	79.82

- Log-linear and max-margin parsers show improvement over perceptron-trained parsers
  - Improvements are statistically significant (sign test)

Summary

- Inside-outside-style inference algorithms for non-projective structures
  - Application of the Matrix-Tree Theorem
  - Inference for both multi-root and single-root structures
- Empirical results
  - Non-projective training is good for non-projective languages
  - Log-linear and max-margin parsers outperform perceptron parsers

Thanks!

Thanks for listening!

Thanks!

Challenges for future research

- State-of-the-art performance is obtained by higher-order models (McDonald and Pereira, 2006; Carreras, 2007)
- Higher-order non-projective inference is nontrivial (McDonald and Pereira, 2006; McDonald and Satta, 2007)
- Approximate inference may work well in practice
- Reranking of k-best spanning trees (Hall, 2007)