Lagrangian Relaxation for Natural Language Processing

Alexander Rush

MIT CSAIL
Statistical Natural Language Processing
Los detectives salvajes es la novela que lanzó a Roberto Bolaño a la fama literaria internacional antes de que 2666 estableciera su reputación para siempre.

The Savage Detectives is the novel that launched Roberto Bolaño to international literary fame before 2666 established his reputation forever.
show me flights from New York to LA departing on Thursday

Flights from New York, NY (all airports) to Los Angeles, CA (LAX)

<table>
<thead>
<tr>
<th>Depart</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thu, Jan 30</td>
<td>Mon, Feb 3</td>
</tr>
</tbody>
</table>

Nonstop only

- United       from $1,034
- Alaska       from $1,034
- American     from $1,034
- JetBlue      from $1,034
- Virgin America from $1,034
- Delta        from $1,054

All flights Nonstop and connecting

- Delta        from $488
- AirTran      from $682
- Other airlines from $803
Goal: Predict best output under a statistical model.

\[ y^* = \text{argmax } f(y) \quad y \in \mathcal{Y} \]

Example: Spanish → English Translation

- \( \mathcal{Y} \): Possible English translations
- \( f(y) \): Probability of translation \( y \)
- \( y^* \): Optimal translation under this model
Goal: Predict best output under a statistical model.

\[ y^* = \arg\max_{y \in \mathcal{Y}} f(y) \]

Example: Spanish $\rightarrow$ English Translation

\( \mathcal{Y} \quad \text{Possible English translations} \)

\( f(y) \quad \text{Probability of translation } y \)

\( y^* \quad \text{Optimal translation under this model} \)
John saw a movie today that he liked.
Statistical Model of Syntax

*0 John1 saw2 a3 movie4 today5 that6 he7 liked8
John saw a movie today that he liked.
John saw a movie that he liked today.
Australia is one of the few countries that have diplomatic relations with North Korea.
载入黛妃死因调查资料的两台手提电脑遭窃
Into devi princess death investigation information of two hands mentioned computers in
Diana will be recorded down in the death investigation.
Two laptops with information on the cause of Princess Diana’s death were stolen.
Statistical Model of Speech

How permanent are their records?
Statistical Model of Speech
Transcription: Help peppermint on their records
Transcription: How permanent are their records
Natural Language Systems

**Aim:**
- Rich models of language
- Optimal predictions
- Efficient inference

Better Model $\Rightarrow$ Harder Inference

**Approach:** Lagrangian Relaxation for Natural Language Inference
**Traveling Salesman Problem:** Find the highest-weight tour,

\[ \text{TOUR} \]

\[ \text{TOUR} \subset 1\text{-TREE} \]

**Idea:** Use Lagrange multipliers to encourage best 1-tree to be a tour.
Lagrangian Relaxation allows us to derive novel inference algorithms.

**Method:**
- Define a larger, relaxed set, $\mathcal{Y} \subset \mathcal{Z}$, to optimize over.
- Introduce Lagrange multipliers to encourage original constraints.

**Challenge:** Design a relaxed set that produces optimal solutions efficiently.
Benefits of Lagrangian Relaxation

Simple - Uses standard combinatorial algorithms.
  - Dynamic Programming
  - Shortest Path
  - Spanning Tree Algorithms

Efficient - Comparable to heuristic inference algorithms.

Strong Guarantees - Gives a certificate of optimality when exact.
Overview

1. Lagrangian Relaxation for Dependency Parsing

2. Lagrangian Relaxation for Syntax-Based Translation

3. Future Work
A dependency arc indicates a head-modifier relationship.

A dependency parse is a directed spanning tree over a sentence.
Statistical Models of Syntactic Parsing

Constituency Parsers (1994 -)
- High accuracy model of syntax
- Predict context-free derivations
- Around 20 words per second

Dependency Parsers (2003 -)
- Comparably accurate model
- Predict dependency structures
- Around 20,000 words per second
Statistical Models of Syntactic Parsing

Constituency Parsers (1994 -)

- High accuracy model of syntax
- Predict context-free derivations
- Around 20 words per second

Dependency Parsers (2003 -)

- Comparably accurate model
- Predict dependency structures
- Around 20,000 words per second

Scale of Dependency Parsers

5.2 million books, 500 billion words (Google Books Corpus)
Applications of Dependency Parsing

Many applications rely on fast and accurate dependency parsing:

- Question Answering
- Information Extraction
- Biological Text Processing
- Dialog Systems
- Summarization
- Statistical Machine Translation
- Sentiment Analysis
- Hedge and Negation Detection
Show me flights from NY to LA Thursday evening.

\[ \lambda x. \text{flight}(x) \land \text{from}(x, \text{ny}) \land \text{to}(x, \text{la}) \land \text{during}(x, \text{evening}) \land \text{day}(x, \text{thur}) \]
Abraham Lincoln was born February 12, 1809 in a one-room log cabin on the Sinking Spring Farm in Hardin County, Kentucky (now LaRue County). He is descended from Samuel Lincoln, who arrived in Hingham, Massachusetts, from Norfolk, England, in the 17th century...

* Abraham Lincoln was born February 12, 1809.
This study demonstrates that IL-8 activates CXCR1.

IL-8 activates endothelial cell CXCR1 and CXCR2 through Rho and Rac signaling pathways

This molar ratio of serum RBP to TTR...

No Interaction
Let $\mathcal{Y}$ be all possible directed spanning trees.

\[
\arg\max_{y \in \mathcal{Y}} f(y)
\]
First-Order Model of Dependency Parsing (McDonald et al., 2005)

\[ f(y) = \text{score}(*_0 \rightarrow \text{saw}_2) + \text{score}(\text{saw}_2 \rightarrow \text{movie}_4) + \text{score}(\text{saw}_4 \rightarrow \text{today}_5) \ldots \]

where \( \text{score}(\text{saw}_2 \rightarrow \text{today}_5) = \log p(\text{today}_5 | \text{saw}_2) \)

or \( \text{score}(\text{saw}_2 \rightarrow \text{today}_5) = w \cdot \phi(\text{saw}_2, \text{today}_5) \)
First-Order Model of Dependency Parsing (McDonald et al., 2005)

\[ f(y) = \text{score}(\ast_0 \rightarrow \text{saw}_2) + \text{score}(\text{saw}_2 \rightarrow \text{movie}_4) + \text{score}(\text{saw}_4 \rightarrow \text{today}_5) \ldots \]

where \( \text{score}(\text{saw}_2 \rightarrow \text{today}_5) = \log p(\text{today}_5 | \text{saw}_2) \)

or \( \text{score}(\text{saw}_2 \rightarrow \text{today}_5) = w \cdot \phi(\text{saw}_2, \text{today}_5) \)

Inference Problem: \( \mathcal{Y} \) - all directed spanning trees.

\[ \arg\max_{y \in \mathcal{Y}} f(y) \]

Directed Spanning Tree
Second-Order Model  
(McDonald et al., 2005)

\[ f(y) = \text{score}(*_0 \rightarrow [\text{NULL}] \text{saw}_2) + \text{score}(\text{saw}_2 \rightarrow [\text{NULL}] \text{movie}_4) + \]
\[ \text{score}(\text{saw}_4 \rightarrow [\text{movie}_4] \text{today}_5) + \ldots \]

where \( \text{score}(\text{saw}_4 \rightarrow [\text{movie}_4] \text{today}_5) = \log p(\text{today}_5 | \text{saw}_2, \text{movie}_4) \)

or \( \text{score}(\text{saw}_4 \rightarrow [\text{movie}_4] \text{today}_5) = w \cdot \phi(\text{saw}_2, \text{movie}_4, \text{today}_5) \)
Second-Order Model (McDonald et al., 2005)

\[ f(y) = score(*_0 \rightarrow [\text{NULL}] \text{saw}_2) + score(\text{saw}_2 \rightarrow [\text{NULL}] \text{movie}_4) + \\
\quad score(\text{saw}_4 \rightarrow [\text{movie}_4] \text{today}_5) + \ldots \]

where \( score(\text{saw}_4 \rightarrow [\text{movie}_4] \text{today}_5) = \log p(\text{today}_5 | \text{saw}_2, \text{movie}_4) \)

or \( score(\text{saw}_4 \rightarrow [\text{movie}_4] \text{today}_5) = w \cdot \phi(\text{saw}_2, \text{movie}_4, \text{today}_5) \)

**Inference Problem:** \( \mathcal{Y} \) - all directed spanning trees.

\[
\arg\max_{y \in \mathcal{Y}} f(y)
\]

NP-Hard
Define $\mathcal{Z}$ as all directed subgraphs, such as

$\mathcal{Y} \subset \mathcal{Z}$
Define $\mathcal{Z}$ as all directed subgraphs, such as

\[
\begin{align*}
\ast_0 & \quad \text{John}_1 \quad \text{saw}_2 \quad a_3 \quad \text{movie}_4 \quad \text{today}_5 \quad \text{that}_6 \quad \text{he}_7 \quad \text{liked}_8 \\
\ast_0 & \quad \text{John}_1 \quad \text{saw}_2 \quad a_3 \quad \text{movie}_4 \quad \text{today}_5 \quad \text{that}_6 \quad \text{he}_7 \quad \text{liked}_8
\end{align*}
\]

\[
\mathcal{Y} \subset \mathcal{Z}
\]
Individual Inference Algorithm

\[
\text{argmax } f(z) \\
\quad \text{where } z \in \mathcal{Z}
\]

Use dynamic programming to find arcs on each side of each word.

\[
\text{score}(\text{saw}_2 \rightarrow [\text{NULL}] \ a_3) + \text{score}(\text{saw}_2 \rightarrow [a_3] \ \text{today}_5)
\]
Reformulation of the Inference Problem

\[
\max_{y \in \mathcal{Y}} f(y)
\]

Equivalent to

\[
\max_{z \in \mathcal{Z}, \ y \in \mathcal{Y}} f(z)
\]

such that for all arcs \(i, j\),

\[
Z_{ij} = Y_{ij}
\]

where \(Y_{ij} = 1\) if arc \((i, j)\) appears in parse \(y\).
Deriving the Algorithm

**Inference Problem:**

\[
\max_{z \in Z, y \in \mathcal{Y}} f(z) \text{ such that } z_{ij} = y_{ij} \text{ for all arcs } i, j
\]

**Relaxed Problem:**

\[
L(\lambda) = \max_{z \in Z, y \in \mathcal{Y}} f(z) - \sum_{i,j} \lambda_{ij} (z_{ij} - y_{ij})
\]

\[
= \max_{z \in Z} \left( f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \right) + \max_{y \in \mathcal{Y}} \left( \sum_{i,j} \lambda_{ij} y_{ij} \right)
\]

*Individual Inference*  
*Max Directed Spanning Tree*
Deriving the Algorithm

$$L(\lambda) = \max_{z \in Z} \left( f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \right) + \max_{y \in Y} \left( \sum_{i,j} \lambda_{ij} y_{ij} \right)$$

Weak Duality: For any $\lambda$, $L(\lambda)$ upper bounds the optimal score, i.e.

$$L(\lambda) \geq \max_{y \in Y} f(y)$$

Furthermore if structures are the same, then this upper bound will be tight.
**Idea:** Iteratively minimize the upper bound $L(\lambda)$.

\[
L(\lambda^{(t)}) = \max_{z \in Z} \left( f(z) - \sum_{i,j} \lambda_{ij}^{(t)} z_{ij} \right) + \max_{y \in Y} \left( \sum_{i,j} \lambda_{ij}^{(t)} y_{ij} \right)
\]

For $t$ from 1 to $T$,

1. Compute $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$. 
For $t$ from 1 to $T$,

1. **Compute** $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. **Update multipliers** 
   
   $$
   \lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t(z_{ij}^{(t)} - y_{ij}^{(t)}). 
   $$

---

**Individual Inference**

$$
z^{(t)} \leftarrow \arg\max_{z \in \mathbb{Z}} f(z) - \sum_{i,j} \lambda_{ij} z_{ij}
$$

**Max Directed Spanning Tree**

$$
y^{(t)} \leftarrow \arg\max_{y \in \mathcal{Y}} \sum_{i,j} \lambda_{ij} y_{ij}
$$

---

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda_{ij}^{(1)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw$_2$ → that$_6$</td>
<td>0</td>
</tr>
<tr>
<td>liked$_8$ → he$_7$</td>
<td>0</td>
</tr>
<tr>
<td>movie$_4$ → that$_6$</td>
<td>0</td>
</tr>
<tr>
<td>liked$_8$ → John$_1$</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
For $t$ from 1 to $T$,

1. **Compute** $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda_{ij}^{(t+1)} ← \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

\[
\begin{align*}
z^{(t)} &\leftarrow \text{argmax}_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \\
y^{(t)} &\leftarrow \text{argmax}_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij}
\end{align*}
\]
For $t$ from 1 to $T$,

1. Compute $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

\[ z^{(t)} \leftarrow \text{argmax}_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \]

\[ y^{(t)} \leftarrow \text{argmax}_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij} \]
For $t$ from 1 to $T$,

1. Compute $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda^{(t+1)}_{ij} \leftarrow \lambda^{(t)}_{ij} - \alpha_t (z^{(t)}_{ij} - y^{(t)}_{ij})$.

**Individual Inference**

\[
z^{(t)} \leftarrow \arg\max_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij}
\]

**Max Directed Spanning Tree**

\[
y^{(t)} \leftarrow \arg\max_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij}
\]

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda^{(t+1)}_{ij}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw$_2$ → that$_6$</td>
<td>1</td>
</tr>
<tr>
<td>liked$_8$ → he$_7$</td>
<td>1</td>
</tr>
<tr>
<td>movie$_4$ → that$_6$</td>
<td>-1</td>
</tr>
<tr>
<td>liked$_8$ → John$_1$</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

Alexander Rush (MIT CSAIL)  
Lagrangian Relaxation for NLP
For $t$ from 1 to $T$,

1. Compute $z(t)$ and $y(t)$ for $\lambda(t)$.
2. If $z(t)$ and $y(t)$ are the same, return $z(t)$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

### Individual Inference

\[ z(t) \leftarrow \text{argmax}_{z \in Z} \, f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \]

### Max Directed Spanning Tree

\[ y(t) \leftarrow \text{argmax}_{y \in Y} \, \sum_{i,j} \lambda_{ij} y_{ij} \]

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda_{ij}^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw$_2$ → that$_6$</td>
<td>1</td>
</tr>
<tr>
<td>liked$_8$ → he$_7$</td>
<td>1</td>
</tr>
<tr>
<td>movie$_4$ → that$_6$</td>
<td>-1</td>
</tr>
<tr>
<td>liked$_8$ → John$_1$</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
For $t$ from 1 to $T$,

1. **Compute** $z(t)$ and $y(t)$ for $\lambda(t)$.
2. If $z(t)$ and $y(t)$ are the same, return $z(t)$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

**Individual Inference**

\[
z(t) \leftarrow \text{argmax}_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij}
\]

**Max Directed Spanning Tree**

\[
y(t) \leftarrow \text{argmax}_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij}
\]
For $t$ from 1 to $T$,

1. Compute $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

**Individual Inference**

\[
\begin{align*}
z^{(t)} &\leftarrow \arg\max_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \\
y^{(t)} &\leftarrow \arg\max_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij}
\end{align*}
\]

**Max Directed Spanning Tree**

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda_{ij}^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw_2 $\rightarrow$ that_6</td>
<td>1</td>
</tr>
<tr>
<td>liked_8 $\rightarrow$ he_7</td>
<td>1</td>
</tr>
<tr>
<td>movie_4 $\rightarrow$ that_6</td>
<td>-1</td>
</tr>
<tr>
<td>liked_8 $\rightarrow$ John_1</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
For $t$ from 1 to $T$,

1. Compute $z(t)$ and $y(t)$ for $\lambda(t)$.
2. If $z(t)$ and $y(t)$ are the same, return $z(t)$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

**Individual Inference**

\[
z(t) \leftarrow \text{argmax}_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij}
\]

**Max Directed Spanning Tree**

\[
y(t) \leftarrow \text{argmax}_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij}
\]

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda_{ij}^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw$_2 \rightarrow$ that$_6$</td>
<td>2</td>
</tr>
<tr>
<td>liked$_8 \rightarrow$ he$_7$</td>
<td>1</td>
</tr>
<tr>
<td>movie$_4 \rightarrow$ that$_6$</td>
<td>-2</td>
</tr>
<tr>
<td>liked$_8 \rightarrow$ John$_1$</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
For $t$ from 1 to $T$,

1. Compute $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda^{(t+1)}_{ij} \leftarrow \lambda^{(t)}_{ij} - \alpha_t (z^{(t)}_{ij} - y^{(t)}_{ij})$.

**Individual Inference**

\[
z^{(t)} \leftarrow \arg\max_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij}z_{ij}
\]

**Max Directed Spanning Tree**

\[
y^{(t)} \leftarrow \arg\max_{y \in Y} \sum_{i,j} \lambda_{ij}y_{ij}
\]

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda^{(3)}_{ij}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw$_2$ → that$_6$</td>
<td>2</td>
</tr>
<tr>
<td>liked$_8$ → he$_7$</td>
<td>1</td>
</tr>
<tr>
<td>movie$_4$ → that$_6$</td>
<td>-2</td>
</tr>
<tr>
<td>liked$_8$ → John$_1$</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
For $t$ from 1 to $T$,

1. **Compute** $z^{(t)}$ and $y^{(t)}$ for $\lambda^{(t)}$.
2. If $z^{(t)}$ and $y^{(t)}$ are the same, return $z^{(t)}$.
3. Update multipliers $\lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)})$.

**Individual Inference**

\[
\begin{align*}
  &z^{(t)} \leftarrow \arg\max_{z \in Z} f(z) - \sum_{i,j} \lambda_{ij} z_{ij} \\
  &y^{(t)} \leftarrow \arg\max_{y \in Y} \sum_{i,j} \lambda_{ij} y_{ij}
\end{align*}
\]

**Max Directed Spanning Tree**

<table>
<thead>
<tr>
<th>Arcs</th>
<th>$\lambda_{ij}^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw$_2$ → that$_6$</td>
<td>2</td>
</tr>
<tr>
<td>liked$_8$ → he$_7$</td>
<td>1</td>
</tr>
<tr>
<td>movie$_4$ → that$_6$</td>
<td>-2</td>
</tr>
<tr>
<td>liked$_8$ → John$_1$</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
For \( t \) from 1 to \( T \),

1. Compute \( z^{(t)} \) and \( y^{(t)} \) for \( \lambda^{(t)} \).

2. If \( z^{(t)} \) and \( y^{(t)} \) are the same, return \( z^{(t)} \).

3. Update multipliers \( \lambda_{ij}^{(t+1)} \leftarrow \lambda_{ij}^{(t)} - \alpha_t (z_{ij}^{(t)} - y_{ij}^{(t)}) \).

### Optimal Parse

\[ \text{argmax}_{y \in \mathcal{Y}} f(y) \]

<table>
<thead>
<tr>
<th>Arcs</th>
<th>( \lambda_{ij}^{(3)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>saw_2 \rightarrow that_6</td>
<td>2</td>
</tr>
<tr>
<td>liked_8 \rightarrow he_7</td>
<td>1</td>
</tr>
<tr>
<td>movie_4 \rightarrow that_6</td>
<td>-2</td>
</tr>
<tr>
<td>liked_8 \rightarrow John_1</td>
<td>-1</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
Theorem (Subgradient Descent)

With an appropriate rate sequence \( \alpha_1, \alpha_2, \alpha_3, \ldots \),

\[
\lim_{t \to \infty} L(\lambda^{(t)}) = \min_{\lambda} L(\lambda)
\]
Formal Guarantees of Lagrangian Relaxation

**Theorem (Subgradient Descent)**

*With an appropriate rate sequence* \( \alpha_1, \alpha_2, \alpha_3, \ldots, \)*

\[
\lim_{t \to \infty} L(\lambda(t)) = \min_{\lambda} L(\lambda)
\]

**Theorem (Certificate of Optimality)**

*If the algorithm finds* \((y(t), z(t))\) *with the same tree then,*

\[
f(z(t)) = \max_{y \in Y} f(y)
\]

Otherwise return best dependency parse seen,

\[
\arg\max_{y \in y^{(1)}, y^{(2)}, \ldots, y^{(T)}} f(y)
\]
Dependency Parsing Results

Optimal Solutions (with Certificate)

<table>
<thead>
<tr>
<th>Language</th>
<th>Previous Best Model</th>
<th>Current Best Model</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cze</td>
<td>84.4</td>
<td>87.3</td>
<td>3.0</td>
</tr>
<tr>
<td>Eng</td>
<td>90.1</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Dan</td>
<td>90.7</td>
<td>92.1</td>
<td>1.4</td>
</tr>
<tr>
<td>Dut</td>
<td>82.3</td>
<td>85.6</td>
<td>3.3</td>
</tr>
<tr>
<td>Por</td>
<td>75.7</td>
<td>77.6</td>
<td>1.9</td>
</tr>
<tr>
<td>Slo</td>
<td>91.4</td>
<td>92.5</td>
<td>1.1</td>
</tr>
<tr>
<td>Swe</td>
<td>88.9</td>
<td>90.6</td>
<td>1.7</td>
</tr>
<tr>
<td>Tur</td>
<td>82.4</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

Prev Best includes:
- Local Search (McDonald and Pereira, 2006)
- Belief Propagation (Smith and Eisner, 2008)
- Linear Programming (Martins et al., 2009)

Lagrangian Relaxation for NLP
## Dependency Parsing Results

### Optimal Solutions (with Certificate)

![Bar chart showing dependency parsing accuracy for different languages.]

### Parsing Accuracy

Results in terms of UAS (% arcs correct)

<table>
<thead>
<tr>
<th>Language</th>
<th>First Order</th>
<th>Prev Best</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dan</td>
<td>89.7</td>
<td>91.5</td>
<td>91.8</td>
</tr>
<tr>
<td>Dut</td>
<td>82.3</td>
<td>85.6</td>
<td>85.8</td>
</tr>
<tr>
<td>Por</td>
<td>90.7</td>
<td>92.1</td>
<td>93.0</td>
</tr>
<tr>
<td>Slo</td>
<td>82.4</td>
<td>85.6</td>
<td>86.2</td>
</tr>
<tr>
<td>Swe</td>
<td>88.9</td>
<td>90.6</td>
<td>91.4</td>
</tr>
<tr>
<td>Tur</td>
<td>75.7</td>
<td>76.4</td>
<td>77.6</td>
</tr>
<tr>
<td>Cze</td>
<td>84.4</td>
<td>—</td>
<td>87.3</td>
</tr>
<tr>
<td>Eng</td>
<td>90.1</td>
<td>—</td>
<td>92.5</td>
</tr>
</tbody>
</table>

Prev Best includes:
- Local Search (McDonald and Pereira, 2006)
- Belief Propagation (Smith and Eisner, 2008)
- Linear Programming (Martins et al., 2009)
# Speed Results for English

<table>
<thead>
<tr>
<th></th>
<th>ILP</th>
<th>LR</th>
<th>LR-Early</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy</strong></td>
<td>92.7</td>
<td>92.7</td>
<td>92.7</td>
</tr>
<tr>
<td><strong>Speed</strong></td>
<td>0.69</td>
<td>12.5 (18x)</td>
<td>33.3 (48x)</td>
</tr>
</tbody>
</table>

![Graph showing speed per second for different methods](image)
Extension to Head Automata Models (Alshawi, 1996)

An important and widely-used generalization of dependency parsing:

- Latent-Variable Parsing Models (Balle et al., 2013)
- Parsing over Word Lattices (Alshawi, 1996)
- Tree-Adjoining Grammars (Carreras and Collins, 2009)
Overview

1. Lagrangian Relaxation for Dependency Parsing
2. Lagrangian Relaxation for Syntax-Based Translation
3. Future Work
Los detectives salvajes es la novela que lanzó a Roberto Bolaño a la fama literaria internacional antes de que 2666 estableciera su reputación para siempre.

The Savage Detectives is the novel that launched Roberto Bolaño to international literary fame before 2666 established his reputation forever.
... one naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: “This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.”

Letter from Warren Weaver to Norbert Weiner, 1947

\[
\arg\max_{y \in Y} f(y)
\]

Decoding
Translation systems are trained with a vast amount of data,
- Training uses 2.5 billion parallel documents.
- Language model trained with 500 billion English words.

Google Translate has used a statistical system since 2006
- 80 languages
- 200 million users a month
- Over 10 billion words translated a day
Australia is one of the few countries that have diplomatic relations with North Korea.
### Synchronous Rule

<table>
<thead>
<tr>
<th>#</th>
<th>Synchronous Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$S \rightarrow \text{le } X_1 X_2, \text{ the } X_1 X_2$</td>
</tr>
<tr>
<td>2</td>
<td>$S \rightarrow \text{le } X_1 X_2, \text{ the } X_2 X_1$</td>
</tr>
<tr>
<td>3</td>
<td>$X \rightarrow \text{chien, dog}$</td>
</tr>
<tr>
<td>4</td>
<td>$X \rightarrow \text{bleu, blue}$</td>
</tr>
<tr>
<td>5</td>
<td>$X \rightarrow \text{rouge, red}$</td>
</tr>
</tbody>
</table>

![Diagram of Synchronous Context-Free Grammar (SCFG)](image_url)
### Weighted Synchronous CFG

<table>
<thead>
<tr>
<th>#</th>
<th>Synchronous Rule</th>
<th>$w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$S \rightarrow \text{le } X_1 X_2 , \text{ the } X_1 X_2$</td>
<td>3.0</td>
</tr>
<tr>
<td>2</td>
<td>$S \rightarrow \text{le } X_1 X_2 , \text{ the } X_2 X_1$</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>$X \rightarrow \text{chien, dog}$</td>
<td>2.0</td>
</tr>
<tr>
<td>4</td>
<td>$X \rightarrow \text{bleu, blue}$</td>
<td>4.0</td>
</tr>
<tr>
<td>5</td>
<td>$X \rightarrow \text{rouge, red}$</td>
<td>3.0</td>
</tr>
</tbody>
</table>

$$w(S \rightarrow \text{le } X_1 X_2 , \text{ the } X_2 X_1) + \ w(X \rightarrow \text{chien, dog}) + \ w(X \rightarrow \text{bleu, blue})$$
<table>
<thead>
<tr>
<th>Synchronous Rules</th>
<th>$w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ of researchers $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ by a scientist</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ of scientists</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ scientists in $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ scientists</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ of scientists $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ scientists of $X_1$ generation</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ scientists $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ plant scientist $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ scientists $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 科学家 $X_2$</td>
<td>科学家 $X_2$ scientists at the $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 初期 $X_2$</td>
<td>初期 $X_2$ early period after the $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 初期 $X_2$</td>
<td>初期 $X_2$ during the early days of the $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 初期 $X_2$</td>
<td>初期 $X_2$ initial stage of $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 初期 $X_2$</td>
<td>初期 $X_2$ first few days after the $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 染色体 $X_2$</td>
<td>染色体 $X_2$ chromosomes</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 染色体 $X_2$</td>
<td>染色体 $X_2$ of chromosomes $X_1$</td>
</tr>
<tr>
<td>$X \rightarrow X_1$ 染色体 $X_2$</td>
<td>染色体 $X_2$ chromosome $X_1$</td>
</tr>
</tbody>
</table>
Consider all synchronous derivations that produce an input sentence.

Input: *le chien bleu*

```
S
 / 
le  X
   / 
   X
   |
   chien bleu

S
 / 
le  X
   / 
   X
   |
   chien bleu

S
 / 
the X
  / 
  X
  |
  dog blue

S
 / 
the X
  / 
  X
  |
  blue dog
```
Translation with a Synchronous Grammar

Consider all synchronous derivations that produce an input sentence.

Input: le chien bleu

<table>
<thead>
<tr>
<th>Forest Rule</th>
<th>$w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow \text{the } X_2 \ X_1$</td>
<td>3.0</td>
</tr>
<tr>
<td>$S \rightarrow \text{the } X_1 \ X_2$</td>
<td>1.0</td>
</tr>
<tr>
<td>$X_1 \rightarrow \text{blue}$</td>
<td>4.0</td>
</tr>
<tr>
<td>$X_2 \rightarrow \text{dog}$</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Incorporating a Language Model

<table>
<thead>
<tr>
<th>Bigram</th>
<th>$w_{lm}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>the blue</td>
<td>-1.0</td>
</tr>
<tr>
<td>the dog</td>
<td>-2.0</td>
</tr>
<tr>
<td>dog blue</td>
<td>-5.0</td>
</tr>
<tr>
<td>blue dog</td>
<td>-1.0</td>
</tr>
</tbody>
</table>

$w_{lm}(\text{the}, \text{dog}) = \log p(\text{dog}|\text{the})$

$$f(y) = w(S \rightarrow \text{the } X_2 X_1) + w(X_2 \rightarrow \text{dog}) + w(X_1 \rightarrow \text{blue}) + w_{lm}(\text{the}, \text{dog}) + w_{lm}(\text{dog}, \text{blue})$$
Define $\mathcal{Y}$ as all derivations in the translation forest.

\[
\arg\max_{y \in \mathcal{Y}} f(y)
\]
Define $\mathcal{Y}$ as all derivations in the translation forest.

$$\arg\max_{y \in \mathcal{Y}} f(y)$$
Relaxed Set: Augmented Derivations

Let $\mathcal{Z}$ be derivations augmented with any previous word, $\mathcal{Y} \subset \mathcal{Z}$

$$f(z) = w(S \rightarrow \text{the} \ X_2 \ X_1) + w(X_2 \rightarrow \text{dog}) + w(X_1 \rightarrow \text{blue}) + w_{lm}(\text{the}, \text{dog}) + w_{lm}(\text{dog}, \text{blue})$$

$$f(z) = w(S \rightarrow \text{the} \ X_2 \ X_1) + w(X_2 \rightarrow \text{dog}) + w(X_1 \rightarrow \text{blue}) + w_{lm}(\text{the}, \text{dog}) + w_{lm}(\text{the, blue})$$
Relaxed Set: Augmented Derivations

Let $\mathcal{Z}$ be derivations augmented with any previous word, $\mathcal{Y} \subset \mathcal{Z}$

\[
f(z) = w(S \rightarrow \text{the } X_2 X_1) + w(X_2 \rightarrow \text{dog}) + w(X_1 \rightarrow \text{blue}) + w_{lm}(\text{the}, \text{dog}) + w_{lm}(\text{dog}, \text{blue})
\]

\[
\arg\max_{y \in \mathcal{Y}} f(y)
\]

Challenging

\[
f(z) = w(S \rightarrow \text{the } X_2 X_1) + w(X_2 \rightarrow \text{dog}) + w(X_1 \rightarrow \text{blue}) + w_{lm}(\text{the}, \text{dog}) + w_{lm}(\text{the}, \text{blue})
\]

\[
\arg\max_{z \in \mathcal{Z}} f(z)
\]

Easy to compute

Alexander Rush (MIT CSAIL)  Lagrangian Relaxation for NLP
Greedy Augmenting Algorithm

1. For each terminal select the best previous word in the language model.

   - Dog
   - The
   - Blue
   - The

2. Find highest-weight derivation using dynamic programming (CKY).

   - S
   - X_1
   - X_2
   - The
   - The
   - Dog
   - Blue
Lagrangian Relaxation Algorithm

Constraint: Augmented words must be consistent with the translation.

For $t$ from 1 to $T$,

1. Find the current highest-scoring augmented derivation.
2. If it is a consistent, return as optimal translation.
3. Otherwise, update multipliers based on inconsistencies.
Extended Set and Inference Algorithm

Full Set $\mathcal{Z}$

- the
- blue

- dog
- S

- $X_1$
- $X_2$

Find highest-scoring derivation with dynamic programming.

Alexander Rush (MIT CSAIL)

Lagrangian Relaxation for NLP
Extended Set and Inference Algorithm

Full Set $\mathcal{Z}$

1. All-pairs shortest path.

2. Find highest-scoring derivation with dynamic programming.

Alexander Rush (MIT CSAIL)  
Lagrangian Relaxation for NLP
Syntax-Based Translation Results

<table>
<thead>
<tr>
<th>Method</th>
<th>ILP</th>
<th>DP</th>
<th>LR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal</td>
<td>100.0%</td>
<td>100.0%</td>
<td>97.8%</td>
</tr>
<tr>
<td>Speed</td>
<td>0.082</td>
<td>0.013</td>
<td>1.266 (15x, 98x)</td>
</tr>
</tbody>
</table>

Alexander Rush (MIT CSAIL)  
Lagrangian Relaxation for NLP
Syntax-Based Translation Results

Exact Solutions

Speed

% Exact

Cube Pruning [50]
Cube Pruning [500]
Lagrangian Relaxation

Sentences Per Second

100
90
80
70
60
50
40
30
20
10
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

Alexander Rush (MIT CSAIL)  Lagrangian Relaxation for NLP
Lagrangian Relaxation for Natural Language Inference

Simple
- Uses basic combinatorial algorithms.
- Can often utilize standard natural language solvers.

Efficient
- Faster than general-purpose solvers.
- Comparable to heuristic inference algorithms.

Strong Guarantees
- Gives a certificate of optimality when exact.
- Find a certificate on the vast majority of examples.
Impact

Many other published applications in NLP using Lagrangian Relaxation,

- Joint Parsing and Tagging (Rush et al., 2010)
- Bilingual Word Alignment (Denero and Macherey, 2010)
- Further Work on Dependency Parsing (Martins et al., 2011)
- Phrase-Based Machine Translation (Chang and Collins, 2011)
- CCG Supertagging (Auli and Lopez, 2011)
- Biomedical Event Extraction (Riedel and McCallum, 2011)
- Markov Logic Decomposition (Niu et al., 2011)
- Tagging with Global Constraints (Rush et al., 2012)
- Semantic Parsing (Das et al., 2012)
- Weighted Automata Problems (Paul and Eisner, 2012)
- Coordination Structure (Hanamoto et al., 2012)
- Latent-Variable Constituency Parsing (Le Roux et al., 2013)
- Faster Algorithms for Machine Translation (Rush et al., 2013)
Overview

1. Lagrangian Relaxation for Dependency Parsing

2. Lagrangian Relaxation for Syntax-Based Translation

3. Future Work
Abraham Lincoln was born February 12, 1809 in a one-room log cabin on the Sinking Spring Farm in Hardin County, Kentucky (now LaRue County). He is descended from Samuel Lincoln, who arrived in Hingham, Massachusetts, from Norfolk, England, in the 17th century...
Abraham Lincoln was born **February 12, 1809** in a one-room log cabin on the Sinking Spring Farm in **Hardin County, Kentucky** (now LaRue County). He is descended from Samuel Lincoln, who arrived in Hingham, Massachusetts, from Norfolk, England, in the 17th century...
Abraham Lincoln was born February 12, 1809 in a one-room log cabin on the Sinking Spring Farm in Hardin County, Kentucky (now LaRue County). He is descended from Samuel Lincoln, who arrived in Hingham, Massachusetts, from Norfolk, England, in the 17th century...

**Abraham Lincoln**

- Birth Date: 2/12/1809
- Birth Place: Hardin County, KY
- Spouse: ???
- Ancestors: Samuel Lincoln

**Samuel Lincoln**

- Birth Date: ???
- Birth Place: ???
That Lincoln, after winning the presidency, made the unprecedented decision to incorporate his eminent rivals into his political family, the cabinet, was evidence of a profound self-confidence and a first indication of what would prove to other a most unexpected greatness. Seward became secretary of state, Chase secretary of the treasury and Bates attorney general. ...
That **Lincoln**, after winning the presidency, made the unprecedented decision to incorporate his eminent rivals into his political family, the cabinet, was evidence of a profound self-confidence and a first indication of what would prove to other a most unexpected greatness. **Seward** became secretary of state, **Chase** secretary of the treasury and **Bates** attorney general. ...
Research Interest: Dialogue Systems

‘‘Is the G train running today?’’

\[ q = \lambda x. \text{alert}(x) \land \text{day}(x, \text{thur}) \land \text{train}(x, G) \]
Research Interest: Dialogue Systems

```

'Is the G train running today?'

\[ q = \lambda x. \ \text{alert}(x) \land \text{day}(x, \text{thur}) \land \text{train}(x, \text{G}) \]

⇓

\[ q \ y \Rightarrow \top, \ \text{warning}(y, \text{track maintenance}) \]

'No, it is closed for maintenance.'
```
Research Interest: Dialogue Systems

Is the G train running today?
No, it is closed for maintenance.

‘‘What line should I take instead?’’

\[ q = \lambda x. \ route(x) \land \text{day}(x, \text{thur}) \land \neg \text{train}(x, L) \]

\[ \Downarrow \]

\[ q \ y \Rightarrow \top \land \text{train}(y, \text{F\_Train}) \land \text{train}(y, \text{L\_Train}) \]

‘‘Take the F train and switch to the L.’’
Thank You
Future Work: Software Toolkits for Inference

www.pydecode.org

Optimized tools for inference and visualization.
Dependency Results

![Graph showing Dependency Results]

- % validation UAS
- % certificates
- % match K=5000

- % recomputed, g+s
- % recomputed, sib

Lagrangian Relaxation for NLP
Translation Intersection

FSA and Context-Free Grammar intersection (Bar-Hillel, 1961)

- **Bigram** -
  \[ O(|\mathcal{N}|^3|\mathcal{U}|^3) \]

- **Trigram** -
  \[ O(|\mathcal{N}|^3|\mathcal{U}|^6) \]
Idea: Predict within a fixed band of matrix.

Extension: Use vine parsing to prune possible arcs.
Vine Pruning: Dependency Parsing Speed

Third-Order Dependency Parsing

En
Bg
De
Pt
Sw
Zh
NoPrune
VineCascade

Relative Speed

Alexander Rush (MIT CSAIL)
Tagging with Inter-Sentence Constraints

- He saw an American man
- The smart girls stood outside
- Danny walks a long distance