Probabilistic Graphical Models: Lagrangian Relaxation Algorithms for Natural Language Processing

Alexander M. Rush

(based on joint work with Michael Collins, Tommi Jaakkola, Terry Koo, David Sontag)

Uncertainty in language

natural language is notoriusly ambiguous, even in toy sentences

`United flies some large jet`

problem becomes very difficult in real-world sentences

`Lin made the first start of his surprising N.B.A. career Monday just as the Knicks lost Amar'e Stoudemire and Carmelo Anthony, turning the roster upside down and setting the scene for another strange and wondrous evening at Madison Square Garden.`

NLP challenges

Lin made the first start of his surprising N.B.A. career Monday just as the Knicks lost Amar'e Stoudemire and Carmelo Anthony, turning the roster upside down and setting the scene for another strange and wondrous evening at Madison Square Garden.

- who lost Stoudemire?
- what is turning the roster upside down?
- are "strange and wonderous" paired?
- what about "down and setting"?

NLP applications

This lecture

1. introduce models for natural language processing in the context of decoding in graphical models
2. describe research in Lagrangian relaxation methods for inference in natural language problems.

Decoding in NLP

focus: decoding as a combinatorial optimization problem

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

where f is a scoring function and $\mathcal{Y}$ is a set of structures

also known as the MAP problem

$$f(y; x) = p(y|x)$$

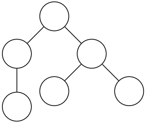
Algorithms for exact decoding

coming lectures will explore algorithms for decoding

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

for some problems, simple combinatorial algorithms are exact

- dynamic programming - e.g. tree-structured MRF's



- min cut - for certain types of weight structures
- minimum spanning tree - certain NLP problems

can depend on structure of the graphical model

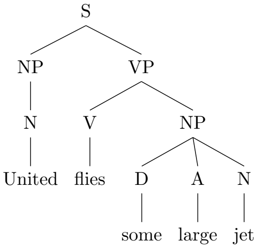
Tagging

United flies some large jet

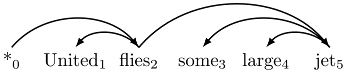
$$
\begin{tikzcd}
N \arrow[r] \arrow[d] & V \arrow[r] \arrow[d] & D \arrow[r] \arrow[d] & A \arrow[r] \arrow[d] & N \arrow[d] \\
\text{United}_1 & \text{flies}_2 & \text{some}_3 & \text{large}_4 & \text{jet}_5
\end{tikzcd}
$$

Parsing

United flies some large jet



United flies some large jet



Decoding complexity

issue: simple combinatorial algorithms do not scale to richer models

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

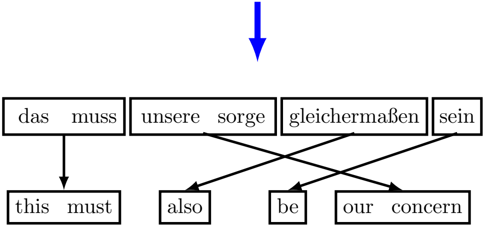
need decoding algorithms for complex natural language tasks

motivation:

- richer model structure often leads to improved accuracy
- exact decoding for complex models tends to be intractable

Phrase-based translation

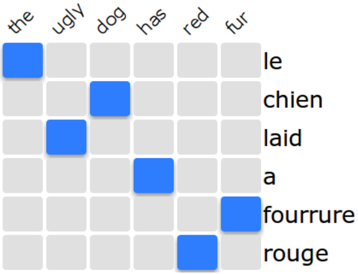
das muss unsere sorge gleichermaßen sein



Word alignment

the ugly dog has red fur

le chien laid a fourrure rouge



Decoding tasks

high complexity

- combined parsing and part-of-speech tagging (Rush et al., 2010)
- "loopy" HMM part-of-speech tagging
- syntactic machine translation (Rush and Collins, 2011)

NP-Hard

- symmetric HMM alignment (DeNero and Macherey, 2011)
- phrase-based translation (Chang and Collins, 2011)
- higher-order non-projective dependency parsing (Koo et al., 2010)

in practice:

- approximate decoding methods (coarse-to-fine, beam search, cube pruning, gibbs sampling, belief propagation)
- approximate models (mean field, variational models)

Motivation

cannot hope to find exact algorithms (particularly when NP-Hard)

aim: develop decoding algorithms with formal guarantees

method:

- derive fast algorithms that provide certificates of optimality
- show that for practical instances, these algorithms often yield exact solutions
- provide strategies for improving solutions or finding approximate solutions when no certificate is found

Lagrangian relaxation helps us develop algorithms of this form

Lagrangian relaxation

a general technique for constructing decoding algorithms

solve complicated models

$$y^* = \arg\max_y f(y)$$

by decomposing into smaller problems.

upshot: can utilize a toolbox of combinatorial algorithms.

- dynamic programming
- minimum spanning tree
- shortest path
- min cut
- ...

Lagrangian relaxation algorithms

Simple - uses basic combinatorial algorithms

Efficient - faster than solving exact decoding problems

Strong guarantees

- gives a certificate of optimality when exact
- direct connections to linear programming relaxations

Outline

1. background on exact NLP decoding
2. worked algorithm for combined parsing and tagging
3. important theorems and formal derivation
4. more examples from parsing and alignment

1. Background: Part-of-speech tagging

United flies some large jet

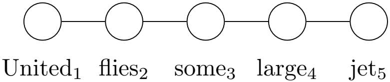
$$
\begin{tikzcd}
\mathrm{N} \arrow[r] \arrow[d] & \mathrm{V} \arrow[r] \arrow[d] & \mathrm{D} \arrow[r] \arrow[d] & \mathrm{A} \arrow[r] \arrow[d] & \mathrm{N} \arrow[d] \\
\text{United}_1 & \text{flies}_2 & \text{some}_3 & \text{large}_4 & \text{jet}_5
\end{tikzcd}
$$

Graphical model formulation

given:

- a sentence of length n and a tag set $\mathcal{T}$
- one variable for each word, takes values in $\mathcal{T}$
- edge potentials $\theta(i-1, i, t', t)$ for all $i \in n$, $t, t' \in \mathcal{T}$

example:



$$\mathcal{T} = \{A, D, N, V\}$$

note: for probabilistic HMM $\theta(i-1, i, t', t) = \log(p(w_i|t)p(t|t'))$

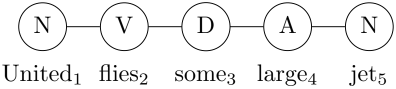
Decoding problem

let $\mathcal{Y} = \mathcal{T}^n$ be the set of assignments to the variables

decoding problem: $\arg\max_{y \in \mathcal{Y}} f(y)$

$$f(y) = \sum_{i=2}^{n} \theta(i-1, i, y(i-1), y(i))$$

example:



Combinatorial solution: Viterbi algorithm

Set $c(i,t) = -\infty$ for all $i \neq 1,\ t \in \mathcal{T}$

Set $c(1,t) = 0$ for all $t \in \mathcal{T}$

For $i = 2$ **to** n

  For $t = 1$ **to** $\mathcal{T}$

$$c(i,t) \leftarrow \max_{t' \in \mathcal{T}} c(i-1,t') + \theta(i-1,i,t',t)$$

Return $\max_{t' \in \mathcal{T}} c(n,t')$

Note: run time $O(n|\mathcal{T}|^2)$, can greatly reduce $|\mathcal{T}|$ in practice

special case of the belief propagation (next week)

Example run

POS Tag

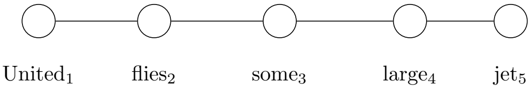
Best Path

A

D

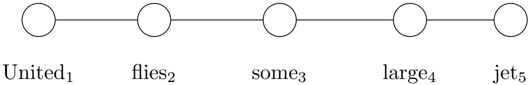
N

V



Example run

POS Tag	Best Path
A	0
D	0
N	0
V	0



Example run

POS Tag	Best Path				
A	0	1			
D	0	1			
N	0	2			
V	0	2			
	United_1	flies_2	some_3	large_4	jet_5

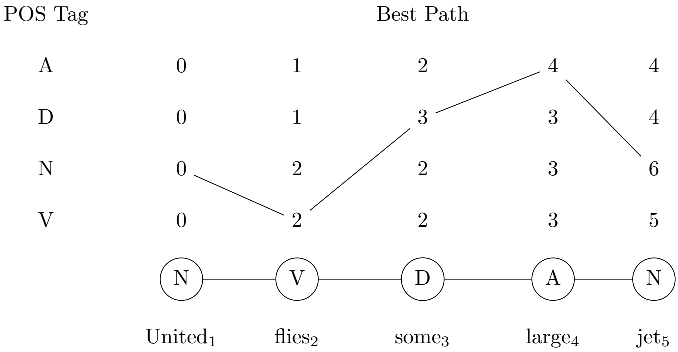
Example run

POS Tag			Best Path		
A	0	1	2		
D	0	1	3		
N	0	2	2		
V	0	2	2		
	United_1	flies_2	some_3	large_4	jet_5

Example run

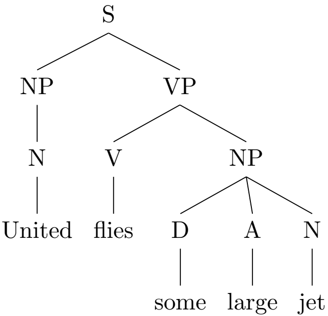
POS Tag	Best Path				
A	0	1	2	4	4
D	0	1	3	3	4
N	0	2	2	3	6
V	0	2	2	3	5
	United_1	flies_2	some_3	large_4	jet_5

Example run


POS Tag
Best Path
A 0 1 2 4 4
D 0 1 3 3 4
N 0 2 2 3 6
V 0 2 2 3 5
N V D A N
United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

Parsing

United flies some large jet



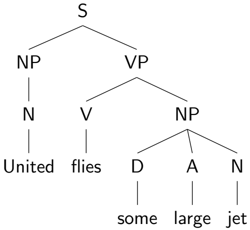
Parse decoding

parsing has potentials:

- $\theta(A \to B\ C)$ for each internal rules
- $\theta(A \to w)$ for pre-terminal (bottom) rules

not easily represented as a graphical model

- complicated independence structure



luckily, similar decoding algorithms to Viterbi

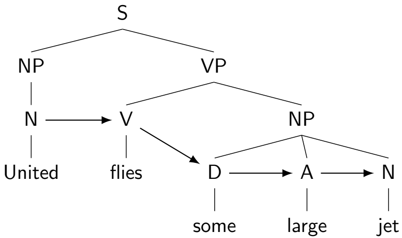
- CKY algorithm for context-free parsing - $O(G^3 n^3)$
- Eisner algorithm for dependency parsing - $O(n^3)$

2. Worked example

aim: walk through a Lagrangian relaxation algorithm for combined parsing and part-of-speech tagging

- introduce formal notation for parsing and tagging
- give assumptions necessary for decoding
- step through a run of the Lagrangian relaxation algorithm

Combined parsing and part-of-speech tagging



goal: find parse tree that optimizes

$$\mathit{score}(\mathrm{S} \to \mathrm{NP\ VP}) + \mathit{score}(\mathrm{VP} \to \mathrm{V\ NP}) +$$

$$\ldots + \mathit{score}(\mathrm{N} \to \mathrm{V}) + \mathit{score}(\mathrm{N} \to \mathrm{United}) + \ldots$$

Constituency parsing

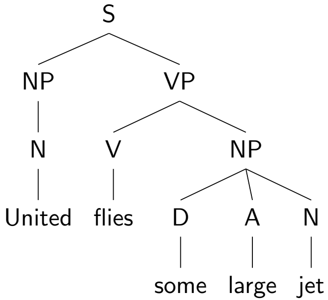
notation:

- $\mathcal{Y}$ is set of constituency parses for input
- $y \in \mathcal{Y}$ is a valid parse
- $f(y)$ scores a parse tree

goal:

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

example: a context-free grammar for constituency parsing



Part-of-speech tagging

notation:

- $\mathcal{Z}$ is set of tag sequences for input
- $z \in \mathcal{Z}$ is a valid tag sequence
- $g(z)$ scores of a tag sequence

goal:

$$\arg\max_{z \in \mathcal{Z}} g(z)$$

example: an HMM for part-of speech tagging

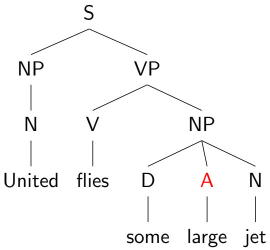
$$\begin{tikzcd}
\mathrm{N} \arrow[r] \arrow[d] & \mathrm{V} \arrow[r] \arrow[d] & \mathrm{D} \arrow[r] \arrow[d] & \mathrm{A} \arrow[r] \arrow[d] & \mathrm{N} \arrow[d] \\
\text{United}_1 & \text{flies}_2 & \text{some}_3 & \text{large}_4 & \text{jet}_5
\end{tikzcd}$$

Identifying tags

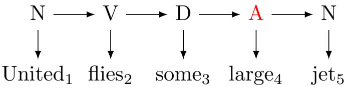
notation: identify the tag labels selected by each model

- $y(i, t) = 1$ when parse y selects tag t at position i
- $z(i, t) = 1$ when tag sequence z selects tag t at position i

example: a parse and tagging with $y(4, A) = 1$ and $z(4, A) = 1$



y



z

Combined optimization

goal:

$$\arg \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} f(y) + g(z)$$

such that for all $i = 1 \ldots n$, $t \in \mathcal{T}$,

$$y(i,t) = z(i,t)$$

i.e. find the best parse and tagging pair that agree on tag labels

equivalent formulation:

$$\arg \max_{y \in \mathcal{Y}} f(y) + g(l(y))$$

where $l : \mathcal{Y} \to \mathcal{Z}$ extracts the tag sequence from a parse tree

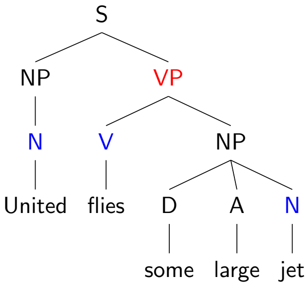
Exact method: Dynamic programming intersection

can solve by solving the product of the two models

example:

- parsing model is a context-free grammar
- tagging model is a first-order HMM
- can solve as CFG and finite-state automata intersection

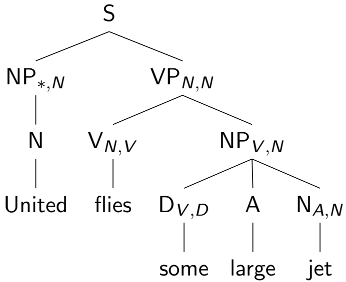
replace $\mathrm{VP} \rightarrow \mathrm{V}\ \mathrm{NP}$
with $\mathrm{VP}_{\mathrm{N,V}} \rightarrow \mathrm{V}_{\mathrm{N,V}}\ \mathrm{NP}_{\mathrm{V,N}}$



Intersected parsing and tagging complexity

let G be the number of grammar non-terminals

parsing CFG require $O(G^3 n^3)$ time with rules $\mathrm{VP} \rightarrow \mathrm{V}\ \mathrm{NP}$



with intersection $O(G^3 n^3 |\mathcal{T}|^3)$ with rules $\mathrm{VP_{N,V}} \rightarrow \mathrm{V_{N,V}}\ \mathrm{NP_{V,N}}$

becomes $O(G^3 n^3 |\mathcal{T}|^6)$ time for second-order HMM

Tagging assumption

assumption: optimization with u can be solved efficiently

$$\arg\max_{z \in \mathcal{Z}} g(z) - \sum_{i,t} u(i,t) z(i,t)$$

example: HMM with scores with edge potentials θ

$$g(z) = \sum_{i=2}^{n} \theta(i-1, i, z(i-1), z(i))$$

where

$$\begin{aligned} \arg\max_{z \in \mathcal{Z}} \quad & g(z) - \sum_{i,t} u(i,t) z(i,t) = \\ \arg\max_{z \in \mathcal{Z}} \quad & \sum_{i=2}^{n} (\theta(i-1, i, z(i-1), z(i)) - u(i, z(i))) - u(1, z(1)) \end{aligned}$$

(Modified) Viterbi algorithm

Set $c(i,t) = -\infty$ for all $i \neq 1,\ t \in \mathcal{T}$

Set $c(1,t) = u(1,t)$ for all $t \in \mathcal{T}$

For $i = 2$ **to** n

  For $t = 1$ **to** $\mathcal{T}$

$$c(i,t) \leftarrow \max_{t' \in \mathcal{T}} c(i-1,t') + \theta(i-1,i,t',t) - u(i,t)$$

Return $\max_{t' \in \mathcal{T}} c(n,t')$

Note: same algorithm as before with modified potentials

Parsing assumption

assumption: optimization with u can be solved efficiently

$$\arg\max_{y \in \mathcal{Y}} f(y) + \sum_{i,t} u(i,t) y(i,t)$$

example: CFG with rule scoring function θ

$$f(y) = \sum_{X \to Y\ Z \in y} \theta(X \to Y\ Z) + \sum_{(i,X) \in y} \theta(X \to w_i)$$

where

$$\arg\max_{y \in \mathcal{Y}} \quad f(y) + \sum_{i,t} u(i,t) y(i,t) =$$

$$\arg\max_{y \in \mathcal{Y}} \quad \sum_{X \to Y\ Z \in y} \theta(X \to Y\ Z) + \sum_{(i,X) \in y} (\theta(X \to w_i) + u(i,X))$$

Lagrangian relaxation algorithm

Set $u^{(1)}(i,t) = 0$ for all $i,\ t \in \mathcal{T}$

For $k = 1$ **to** K

$$y^{(k)} \leftarrow \arg\max_{y \in \mathcal{Y}} f(y) + \sum_{i,t} u^{(k)}(i,t)y(i,t) \text{ [Parsing]}$$

$$z^{(k)} \leftarrow \arg\max_{z \in \mathcal{Z}} g(z) - \sum_{i,t} u^{(k)}(i,t)z(i,t) \text{ [Tagging]}$$

If $y^{(k)}(i,t) = z^{(k)}(i,t)$ for all i,t **Return** $(y^{(k)}, z^{(k)})$

Else $u^{(k+1)}(i,t) \leftarrow u^{(k)}(i,t) - \alpha_k(y^{(k)}(i,t) - z^{(k)}(i,t))$

CKY Parsing

Penalties

$u(i,t) = 0$ for all i,t

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

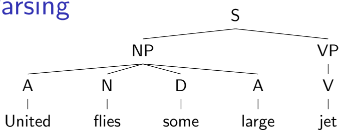
United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



Penalties

$u(i, t) = 0$ for all i,t

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

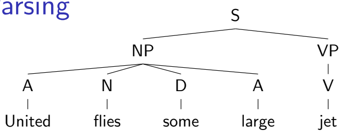
United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing

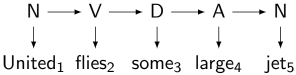


Penalties

$u(i,t) = 0$ for all i,t

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

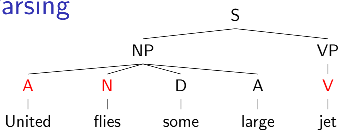


$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing

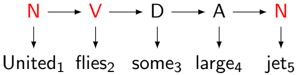


$$y^* = \arg\max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right)$$

Penalties

$u(i,t) = 0$ for all i,t

Viterbi Decoding

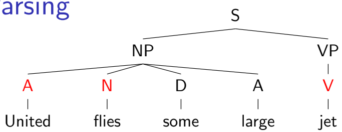


$$z^* = \arg\max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right)$$

Key

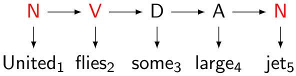
$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

Penalties

$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1, A)$	-1
$u(1, N)$	1
$u(2, N)$	-1
$u(2, V)$	1
$u(5, V)$	-1
$u(5, N)$	1

CKY Parsing

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Penalties

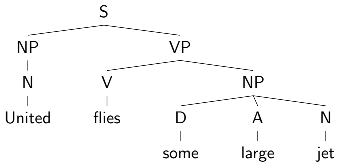
$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1,A)$	-1
$u(1,N)$	1
$u(2,N)$	-1
$u(2,V)$	1
$u(5,V)$	-1
$u(5,N)$	1

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

United_1 flies_2 some_3 large_4 jet_5

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Penalties

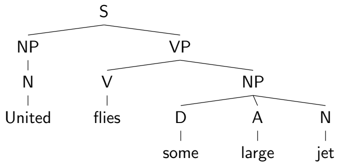
$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1,A)$	-1
$u(1,N)$	1
$u(2,N)$	-1
$u(2,V)$	1
$u(5,V)$	-1
$u(5,N)$	1

Key

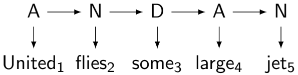
$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Penalties

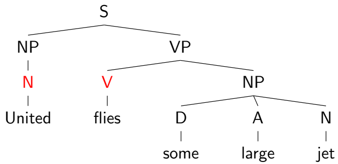
$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1, A)$	-1
$u(1, N)$	1
$u(2, N)$	-1
$u(2, V)$	1
$u(5, V)$	-1
$u(5, N)$	1

Key

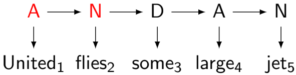
$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Penalties

$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1, A)$	-1
$u(1, N)$	1
$u(2, N)$	-1
$u(2, V)$	1
$u(5, V)$	-1
$u(5, N)$	1

Iteration 2	
$u(5, V)$	-1
$u(5, N)$	1

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Penalties

$u(i,t) = 0$ for all i,t

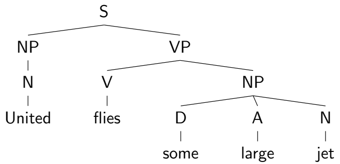
Iteration 1	
$u(1,A)$	-1
$u(1,N)$	1
$u(2,N)$	-1
$u(2,V)$	1
$u(5,V)$	-1
$u(5,N)$	1

Iteration 2	
$u(5,V)$	-1
$u(5,N)$	1

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding

United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Penalties

$u(i,t) = 0$ for all i,t

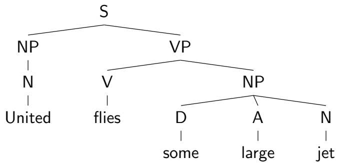
Iteration 1	
$u(1,A)$	-1
$u(1,N)$	1
$u(2,N)$	-1
$u(2,V)$	1
$u(5,V)$	-1
$u(5,N)$	1

Iteration 2	
$u(5,V)$	-1
$u(5,N)$	1

Key

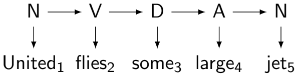
$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t) y(i,t))$$

Viterbi Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t) z(i,t))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

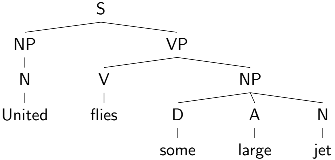
Penalties

$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1, A)$	-1
$u(1, N)$	1
$u(2, N)$	-1
$u(2, V)$	1
$u(5, V)$	-1
$u(5, N)$	1

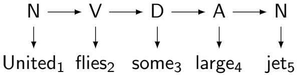
Iteration 2	
$u(5, V)$	-1
$u(5, N)$	1

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,t} u(i,t)y(i,t))$$

Viterbi Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,t} u(i,t)z(i,t))$$

Penalties

$u(i,t) = 0$ for all i,t

Iteration 1	
$u(1,A)$	-1
$u(1,N)$	1
$u(2,N)$	-1
$u(2,V)$	1
$u(5,V)$	-1
$u(5,N)$	1

Iteration 2	
$u(5,V)$	-1
$u(5,N)$	1

Converged

$$y^* = \arg\max_{y \in \mathcal{Y}} f(y) + g(y)$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	HMM
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Taggings
$y(i,t) = 1$	if	y contains tag t at position i			

Main theorem

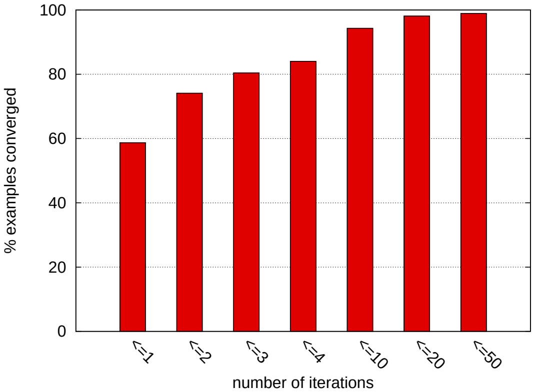
theorem: if at any iteration, for all i, $t \in \mathcal{T}$

$$y^{(k)}(i,t) = z^{(k)}(i,t)$$

then $(y^{(k)}, z^{(k)})$ is the global optimum

proof: focus of the next section

Convergence



2. Formal properties

aim: formal derivation of the algorithm given in the previous section

- derive Lagrangian dual
- prove three properties
  - upper bound
  - convergence
  - optimality
- describe subgradient method

Lagrangian

goal:

$$\arg \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} f(y) + g(z) \text{ such that } y(i,t) = z(i,t)$$

Lagrangian:

$$L(u,y,z) = f(y) + g(z) + \sum_{i,t} u(i,t) \left(y(i,t) - z(i,t)\right)$$

redistribute terms

$$L(u,y,z) \quad = \quad \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right) + \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right)$$

Lagrangian dual

Lagrangian:

$$L(u, y, z) = \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right) + \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right)$$

Lagrangian dual:

$$\begin{aligned} L(u) &= \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} L(u, y, z) \\ &= \max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right) + \\ &\quad \max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right) \end{aligned}$$

Theorem 1. Upper bound

define:

- y^*, z^* is the optimal combined parsing and tagging solution with $y^*(i,t) = z^*(i,t)$ for all i, t

theorem: for any value of u

$$L(u) \geq f(y^*) + g(z^*)$$

$L(u)$ provides an upper bound on the score of the optimal solution

note: upper bound may be useful as input to branch and bound or A* search

Theorem 1. Upper bound (proof)

theorem: for any value of u, $L(u) \geq f(y^*) + g(z^*)$

proof:

$$\begin{aligned}
L(u) &= \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} L(u, y, z) && (1)\\
&\geq \max_{y \in \mathcal{Y}, z \in \mathcal{Z} : y = z} L(u, y, z) && (2)\\
&= \max_{y \in \mathcal{Y}, z \in \mathcal{Z} : y = z} f(y) + g(z) && (3)\\
&= f(y^*) + g(z^*) && (4)
\end{aligned}$$

Formal algorithm (reminder)

Set $u^{(1)}(i,t) = 0$ for all $i,\ t \in \mathcal{T}$

For $k = 1$ **to** K

$$y^{(k)} \leftarrow \arg\max_{y \in \mathcal{Y}} f(y) + \sum_{i,t} u^{(k)}(i,t) y(i,t) \text{ [Parsing]}$$

$$z^{(k)} \leftarrow \arg\max_{z \in \mathcal{Z}} g(z) - \sum_{i,t} u^{(k)}(i,t) z(i,t) \text{ [Tagging]}$$

If $y^{(k)}(i,t) = z^{(k)}(i,t)$ for all i,t **Return** $(y^{(k)}, z^{(k)})$

Else $u^{(k+1)}(i,t) \leftarrow u^{(k)}(i,t) - \alpha_k (y^{(k)}(i,t) - z^{(k)}(i,t))$

Theorem 2. Convergence

notation:

- $u^{(k+1)}(i,t) \leftarrow u^{(k)}(i,t) + \alpha_k(y^{(k)}(i,t) - z^{(k)}(i,t))$ is update
- $u^{(k)}$ is the penalty vector at iteration k
- $\alpha_k > 0$ is the update rate at iteration k

theorem: for any sequence $\alpha^1, \alpha^2, \alpha^3, \ldots$ such that

$$\lim_{t\to\infty} \alpha^t = 0 \quad \text{and} \quad \sum_{t=1}^{\infty} \alpha^t = \infty,$$

we have

$$\lim_{t\to\infty} L(u^t) = \min_u L(u)$$

i.e. the algorithm converges to the tightest possible upper bound

proof: by subgradient convergence (next section)

Dual solutions

define:

- for any value of u

$$y_u = \arg\max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right)$$

and

$$z_u = \arg\max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right)$$

- y_u and z_u are the dual solutions for a given u

Theorem 3. Optimality

theorem: if there exists u such that

$$y_u(i, t) = z_u(i, t)$$

for all i, t then

$$f(y_u) + g(z_u) = f(y^*) + g(z^*)$$

i.e. if the dual solutions agree, we have an optimal solution

$$(y_u, z_u)$$

Theorem 3. Optimality (proof)

theorem: if u such that $y_u(i,t) = z_u(i,t)$ for all i, t then

$$f(y_u) + g(z_u) = f(y^*) + g(z^*)$$

proof: by the definitions of y_u and z_u

$$\begin{aligned} L(u) &= f(y_u) + g(z_u) + \sum_{i,t} u(i,t)(y_u(i,t) - z_u(i,t)) \\ &= f(y_u) + g(z_u) \end{aligned}$$

since $L(u) \geq f(y^*) + g(z^*)$ for all values of u

$$f(y_u) + g(z_u) \geq f(y^*) + g(z^*)$$

but y^* and z^* are optimal

$$f(y_u) + g(z_u) \leq f(y^*) + g(z^*)$$

Dual optimization

Lagrangian dual:

$$
\begin{aligned}
L(u) &= \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} L(u, y, z) \\
&= \max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right) + \\
&\quad \max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right)
\end{aligned}
$$

goal: dual problem is to find the tightest upper bound

$$
\min_{u} L(u)
$$

Dual subgradient

$$L(u) \quad = \quad \max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right) + \max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,t} u(i,t) z(i,t) \right)$$

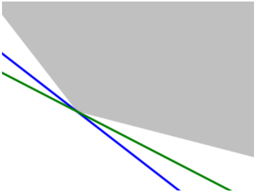
properties:

- L(u) is convex in u (no local minima)
- L(u) is not differentiable (because of max operator)

handle non-differentiability by using subgradient descent

define: a subgradient of $L(u)$ at u is a vector g_u such that for all v

$$L(v) \geq L(u) + g_u \cdot (v - u)$$



Subgradient algorithm

$$L(u) = \max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,t} u(i,t) y(i,t) \right) + \max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,j} u(i,t) z(i,t) \right)$$

recall, y_u and z_u are the argmax's of the two terms

subgradient:

$$g_u(i,t) = y_u(i,t) - z_u(i,t)$$

subgradient descent: move along the subgradient

$$u'(i,t) = u(i,t) - \alpha \left(y_u(i,t) - z_u(i,t) \right)$$

guaranteed to find a minimum with conditions given earlier for α

4. More examples

aim: demonstrate similar algorithms that can be applied to other decoding applications

- context-free parsing combined with dependency parsing
- combined translation alignment

Combined constituency and dependency parsing

(Rush et al., 2010)

setup: assume separate models trained for constituency and dependency parsing

problem: find constituency parse that maximizes the sum of the two models

example:

- combine lexicalized CFG with second-order dependency parser

Lexicalized constituency parsing

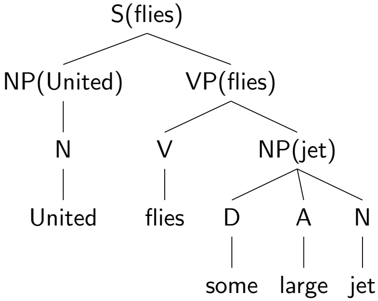
notation:

- $\mathcal{Y}$ is set of lexicalized constituency parses for input
- $y \in \mathcal{Y}$ is a valid parse
- $f(y)$ scores a parse tree

goal:

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

example: a lexicalized context-free grammar

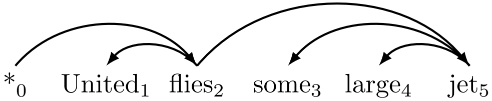


Dependency parsing

define:

- $\mathcal{Z}$ is set of dependency parses for input
- $z \in \mathcal{Z}$ is a valid dependency parse
- $g(z)$ scores a dependency parse

example:

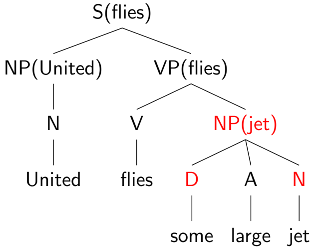


Identifying dependencies

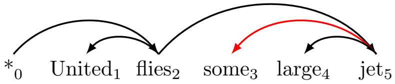
notation: identify the dependencies selected by each model

- $y(i,j) = 1$ when word i modifies of word j in constituency parse y
- $z(i,j) = 1$ when word i modifies of word j in dependency parse z

example: a constituency and dependency parse with $y(3,5) = 1$ and $z(3,5) = 1$



y



z

Combined optimization

goal:

$$\arg \max_{y \in \mathcal{Y}, z \in \mathcal{Z}} f(y) + g(z)$$

such that for all $i = 1 \ldots n$, $j = 0 \ldots n$,

$$y(i,j) = z(i,j)$$

CKY Parsing

Penalties

$u(i,j) = 0$ for all i,j

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Dependency Parsing

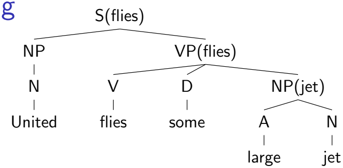
$*_0$ United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Dependency Parsing

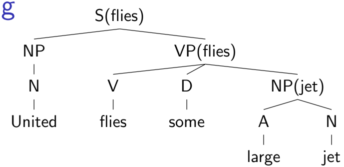
$*_0$ United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing

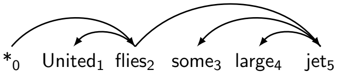


$$y^* = \arg\max_{y \in \mathcal{Y}} \Big(f(y) + \sum_{i,j} u(i,j) y(i,j) \Big)$$

Penalties

$u(i,j) = 0$ for all i,j

Dependency Parsing

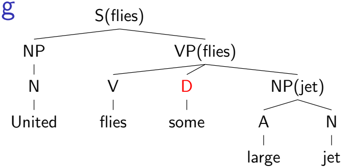


$$z^* = \arg\max_{z \in \mathcal{Z}} \Big(g(z) - \sum_{i,j} u(i,j) z(i,j) \Big)$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing

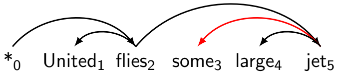


$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Dependency Parsing

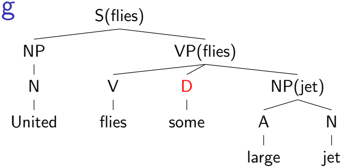


$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing



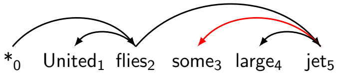
$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(2,3)$	-1
$u(5,3)$	1

Dependency Parsing



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(2,3)$	-1
$u(5,3)$	1

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Dependency Parsing

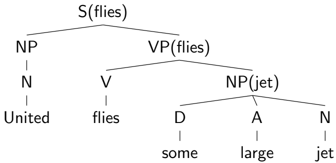
$*_0$ United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(2,3)$	-1
$u(5,3)$	1

Dependency Parsing

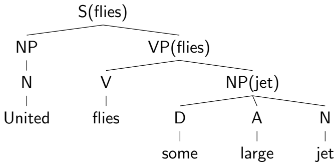
$*_0$ United$_1$ flies$_2$ some$_3$ large$_4$ jet$_5$

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing



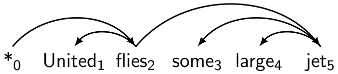
$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(2,3)$	-1
$u(5,3)$	1

Dependency Parsing

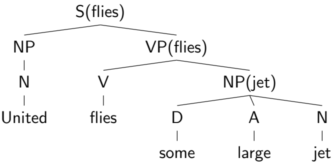


$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

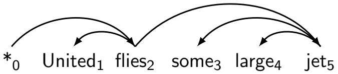
$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

CKY Parsing



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Dependency Parsing



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(2,3)$	-1
$u(5,3)$	1

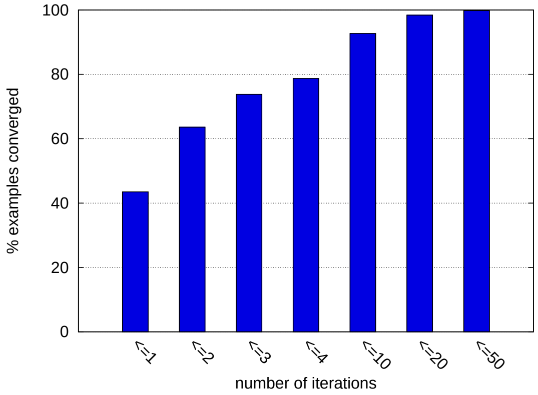
Converged

$$y^* = \arg\max_{y \in \mathcal{Y}} f(y) + g(y)$$

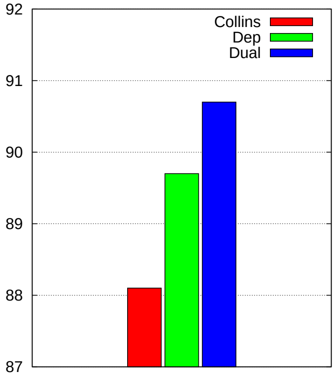
Key

$f(y)$	$\Leftarrow$	CFG	$g(z)$	$\Leftarrow$	Dependency Model
$\mathcal{Y}$	$\Leftarrow$	Parse Trees	$\mathcal{Z}$	$\Leftarrow$	Dependency Trees
$y(i,j) = 1$	if	y contains dependency i,j			

Convergence



Integrated Constituency and Dependency Parsing: Accuracy



F_1 Score

- Collins (1997) Model 1
- Fixed, First-best Dependencies from Koo (2008)
- Dual Decomposition

Corpus-level tagging

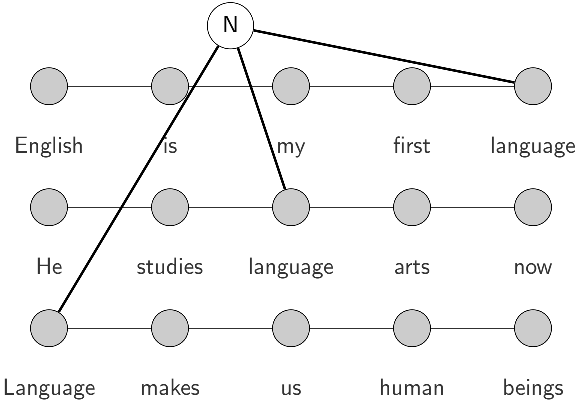
setup: given a corpus of sentences and a trained sentence-level tagging model

problem: find best tagging for each sentence, while at the same time enforcing inter-sentence soft constraints

example:

- test-time decoding with a trigram tagger
- constraint that each word type prefer a single POS tag

Corpus-level tagging



Sentence-level decoding

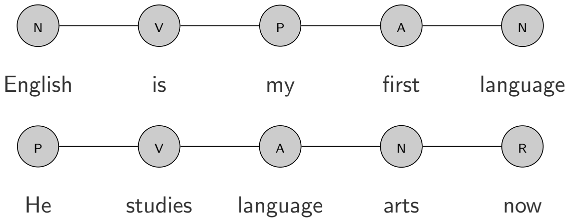
notation:

- $\mathcal{Y}_i$ is set of tag sequences for input sentence i
- $\mathcal{Y} = \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_m$ is set of tag sequences for the input corpus
- $Y \in \mathcal{Y}$ is a valid tag sequence for the corpus
- $F(Y) = \sum_i f(Y_i)$ is the score for tagging the whole corpus

goal:

$$\arg\max_{Y \in \mathcal{Y}} F(Y)$$

example: decode each sentence with a trigram tagger

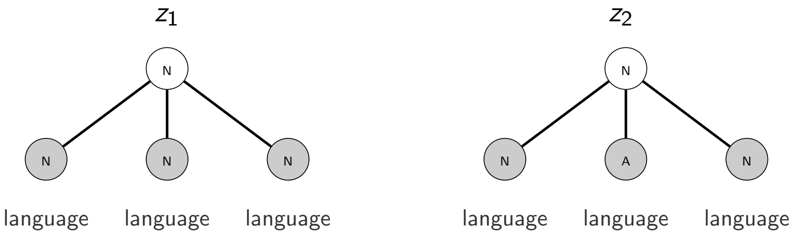


Inter-sentence constraints

notation:

- $\mathcal{Z}$ is set of possible assignments of tags to word types
- $z \in \mathcal{Z}$ is a valid tag assignment
- $g(z)$ is a scoring function for assignments to word types

example: an MRF model that encourages words of the same type to choose the same tag



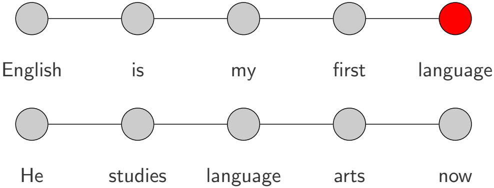
$$g(z_1) > g(z_2)$$

Identifying word tags

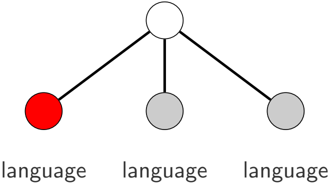
notation: identify the tag labels selected by each model

- $Y_s(i, t) = 1$ when the tagger for sentence s at position i selects tag t
- $z(s, i, t) = 1$ when the constraint assigns at sentence s position i the tag t

example: a parse and tagging with $Y_1(5, N) = 1$ and $z(1, 5, N) = 1$



Y



z

Combined optimization

goal:

$$\arg\max_{Y \in \mathcal{Y}, z \in \mathcal{Z}} F(Y) + g(z)$$

such that for all $s = 1 \ldots m$, $i = 1 \ldots n$, $t \in \mathcal{T}$,

$$Y_s(i,t) = z(s,i,t)$$

Penalties

$u(s, i, t) = 0$ for all s,i,t

Tagging

MRF

Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Penalties

$u(s,i,t) = 0$ for all s,i,t

Tagging

N	V	P	A	N
English	is	my	first	language
P	V	A	N	R
He	studies	language	arts	now
N	V	P	N	N
Language	makes	us	human	beings

MRF

Key

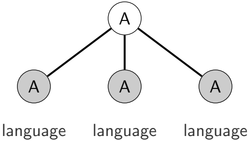
$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i,t) = 1$	if	sentence s has tag t at position i			

Penalties

$u(s, i, t) = 0$ for all s,i,t

Tagging

MRF



Key

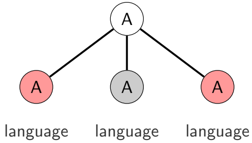
$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Penalties

$u(s, i, t) = 0$ for all s,i,t

Tagging

MRF



Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

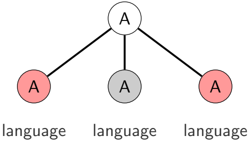
Tagging

Penalties

$u(s, i, t) = 0$ for all s,i,t

Iteration 1	
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1

MRF



Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Tagging

Penalties

$u(s,i,t) = 0$ for all s,i,t

Iteration 1	
$u(1,5,N)$	-1
$u(1,5,A)$	1
$u(3,1,N)$	-1
$u(3,1,A)$	1

MRF

Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i,t) = 1$	if	sentence s has tag t at position i			

Tagging

N	V	P	A	N
English	is	my	first	language
P	V	A	N	R
He	studies	language	arts	now
N	V	P	N	N
Language	makes	us	human	beings

Penalties

$u(s, i, t) = 0$ for all s,i,t

Iteration 1	
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1

MRF

Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

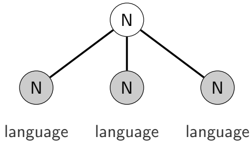
Tagging

Penalties

$u(s,i,t) = 0$ for all s,i,t

Iteration 1	
$u(1,5,N)$	-1
$u(1,5,A)$	1
$u(3,1,N)$	-1
$u(3,1,A)$	1

MRF



Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i,t) = 1$	if	sentence s has tag t at position i			

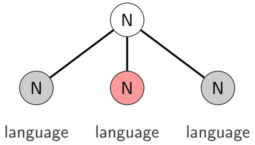
Penalties

$u(s, i, t) = 0$ for all s,i,t

	Iteration 1
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1

Tagging

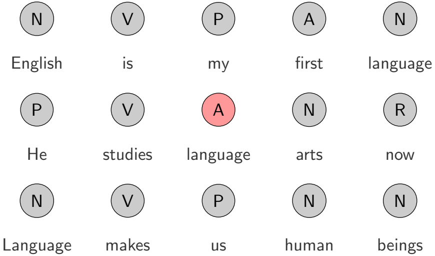
MRF



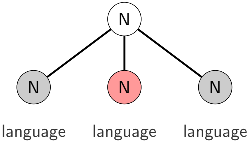
Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Tagging



MRF



Penalties

$u(s, i, t) = 0$ for all s,i,t

Iteration 1	
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1

Iteration 2	
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1
$u(2, 3, N)$	1
$u(2, 3, A)$	-1

Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Tagging

N	V	P	A	N
English	is	my	first	language
P	V	N	N	R
He	studies	language	arts	now
N	V	P	N	N
Language	makes	us	human	beings

MRF

Penalties

$u(s, i, t) = 0$ for all s,i,t

Iteration 1	
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1

Iteration 2	
$u(1, 5, N)$	-1
$u(1, 5, A)$	1
$u(3, 1, N)$	-1
$u(3, 1, A)$	1
$u(2, 3, N)$	1
$u(2, 3, A)$	-1

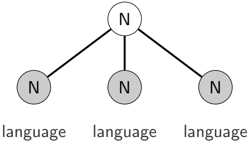
Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Tagging

N	V	P	A	N
English	is	my	first	language
P	V	N	N	R
He	studies	language	arts	now
N	V	P	N	N
Language	makes	us	human	beings

MRF



Penalties

$u(s, i, t) = 0$ for all s,i,t

Iteration 1	
$u(1,5,N)$	-1
$u(1,5,A)$	1
$u(3,1,N)$	-1
$u(3,1,A)$	1

Iteration 2	
$u(1,5,N)$	-1
$u(1,5,A)$	1
$u(3,1,N)$	-1
$u(3,1,A)$	1
$u(2,3,N)$	1
$u(2,3,A)$	-1

Key

$F(Y)$	$\Leftarrow$	Tagging model	$g(z)$	$\Leftarrow$	MRF
$\mathcal{Y}$	$\Leftarrow$	Sentence-level tagging	$\mathcal{Z}$	$\Leftarrow$	Inter-sentence constraints
$Y_s(i, t) = 1$	if	sentence s has tag t at position i			

Combined alignment (DeNero and Macherey, 2011)

setup: assume separate models trained for English-to-French and French-to-English alignment

problem: find an alignment that maximizes the score of both models

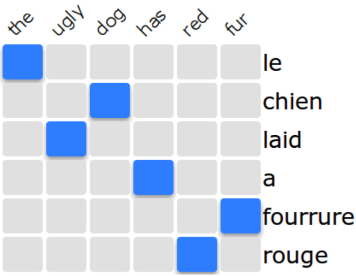
example:

- HMM models for both directional alignments (assume correct alignment is one-to-one for simplicity)

Alignment

the ugly dog has red fur

le chien laid a fourrure rouge

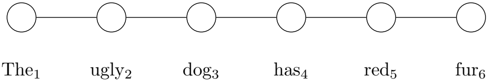


Graphical model formulation

given:

- French sentence of length n and English sentence of length m
- one variable for each English word, takes values in $\{1, \ldots, n\}$
- edge potentials $\theta(i-1, i, f', f)$ for all $i \in n$, $f, f' \in \{1, \ldots, n\}$

example:



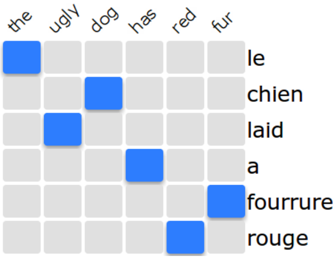
English-to-French alignment

define:

- $\mathcal{Y}$ is set of all possible English-to-French alignments
- $y \in \mathcal{Y}$ is a valid alignment
- $f(y)$ scores of the alignment

example: HMM alignment

$$\begin{tikzcd}
\text{Le}_1 \arrow[r] \arrow[d] & \text{laid}_3 \arrow[r] \arrow[d] & \text{chien}_2 \arrow[r] \arrow[d] & \text{a}_4 \arrow[r] \arrow[d] & \text{rouge}_6 \arrow[r] \arrow[d] & \text{fourrure}_5 \arrow[d] \\
\text{The}_1 & \text{ugly}_2 & \text{dog}_3 & \text{has}_4 & \text{red}_5 & \text{fur}_6
\end{tikzcd}$$



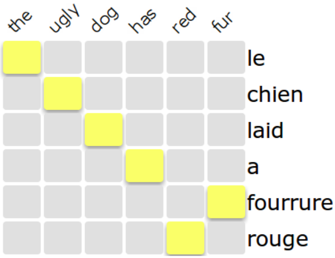
French-to-English alignment

define:

- $\mathcal{Z}$ is set of all possible French-to-English alignments
- $z \in \mathcal{Z}$ is a valid alignment
- $g(z)$ scores of an alignment

example: HMM alignment

$$
\begin{tikzcd}
\text{The}_1 \arrow[r] \arrow[d] & \text{ugly}_2 \arrow[r] \arrow[d] & \text{dog}_3 \arrow[r] \arrow[d] & \text{has}_4 \arrow[r] \arrow[d] & \text{fur}_6 \arrow[r] \arrow[d] & \text{red}_5 \arrow[d] \\
\text{Le}_1 & \text{chien}_2 & \text{laid}_3 & \text{a}_4 & \text{fourrure}_5 & \text{rouge}_6
\end{tikzcd}
$$

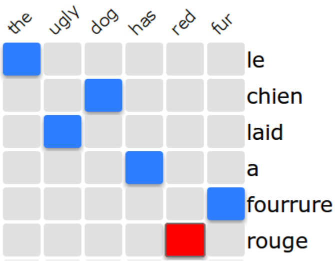


Identifying word alignments

notation: identify the tag labels selected by each model

- $y(i,j) = 1$ when e-to-f alignment y selects French word i to align with English word j
- $z(i,j) = 1$ when f-to-e alignment z selects French word i to align with English word j

example: two HMM alignment models with $y(6,5) = 1$ and $z(6,5) = 1$



Combined optimization

goal:

$$\arg\max_{y\in\mathcal{Y},z\in\mathcal{Z}} f(y) + g(z)$$

such that for all $i = 1 \ldots n$, $j = 1 \ldots n$,

$$y(i,j) = z(i,j)$$

English-to-French

Penalties

$u(i,j) = 0$ for all i,j

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

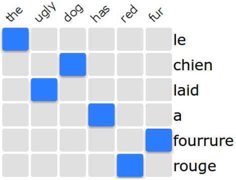
French-to-English

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} \left(f(y) + \sum_{i,j} u(i,j)y(i,j)\right)$$

French-to-English

$$z^* = \arg\max_{z \in \mathcal{Z}} \left(g(z) - \sum_{i,j} u(i,j)z(i,j)\right)$$

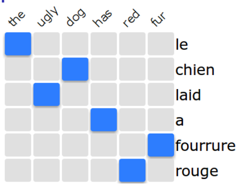
Penalties

$u(i,j) = 0$ for all i,j

Key

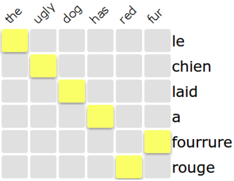
$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

French-to-English



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

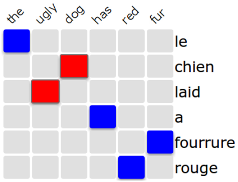
Penalties

$u(i,j) = 0$ for all i,j

Key

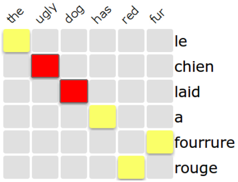
$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j)y(i,j))$$

French-to-English



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j)z(i,j))$$

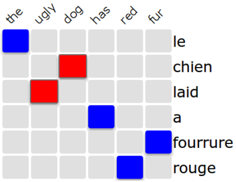
Penalties

$u(i,j) = 0$ for all i,j

Key

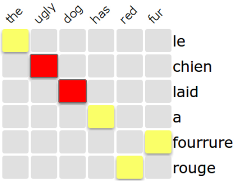
$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

French-to-English



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(3,2)$	-1
$u(2,2)$	1
$u(2,3)$	-1
$u(3,3)$	1

Key

$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French

$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

French-to-English

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Penalties

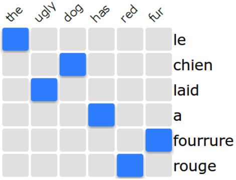
$u(i,j) = 0$ for all i,j

Iteration 1	
$u(3,2)$	-1
$u(2,2)$	1
$u(2,3)$	-1
$u(3,3)$	1

Key

$f(y)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model
$y(i,j) = 1$	if	French word i aligns to English word j
$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Z}$	$\Leftarrow$	French-to-English model

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(3,2)$	-1
$u(2,2)$	1
$u(2,3)$	-1
$u(3,3)$	1

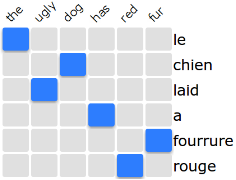
French-to-English

$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Key

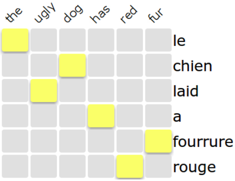
$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

French-to-English



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Penalties

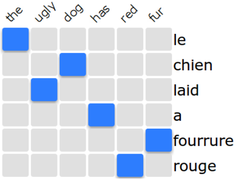
$u(i,j) = 0$ for all i,j

	Iteration 1
$u(3,2)$	-1
$u(2,2)$	1
$u(2,3)$	-1
$u(3,3)$	1

Key

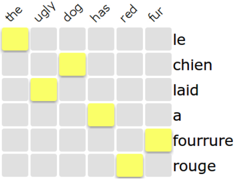
$f(y)$	$\Leftarrow$	HMM Alignment	$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model	$\mathcal{Z}$	$\Leftarrow$	French-to-English model
$y(i,j) = 1$	if	French word i aligns to English word j			

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j)y(i,j))$$

French-to-English



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j)z(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

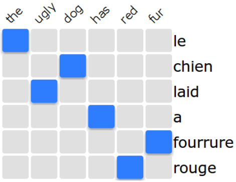
Iteration 1	
$u(3,2)$	-1
$u(2,2)$	1
$u(2,3)$	-1
$u(3,3)$	1

Key

$f(y)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model
$y(i,j) = 1$	if	French word i aligns to English word j

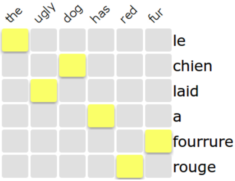
$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Z}$	$\Leftarrow$	French-to-English model

English-to-French



$$y^* = \arg\max_{y \in \mathcal{Y}} (f(y) + \sum_{i,j} u(i,j) y(i,j))$$

French-to-English



$$z^* = \arg\max_{z \in \mathcal{Z}} (g(z) - \sum_{i,j} u(i,j) z(i,j))$$

Penalties

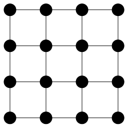
$u(i,j) = 0$ for all i,j

	Iteration 1
$u(3,2)$	-1
$u(2,2)$	1
$u(2,3)$	-1
$u(3,3)$	1

Key

$f(y)$	$\Leftarrow$	HMM Alignment
$\mathcal{Y}$	$\Leftarrow$	English-to-French model
$y(i,j) = 1$	if	French word i aligns to English word j
$g(z)$	$\Leftarrow$	HMM Alignment
$\mathcal{Z}$	$\Leftarrow$	French-to-English model

MAP problem in Markov random fields



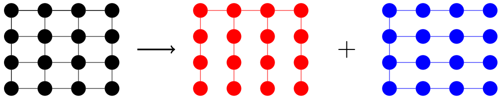
given: binary variables $x_1 \dots x_n$

goal: MAP problem

$$\arg\max_{x_1 \dots x_n} \sum_{(i,j) \in E} f_{i,j}(x_i, x_j)$$

where each $f_{i,j}(x_i, x_j)$ is a local potential for variables x_i, x_j

Dual decomposition for MRFs (Komodakis et al., 2010)



goal:

$$\arg\max_{x_1 \ldots x_n} \sum_{(i,j)\in E} f_{i,j}(x_i, x_j)$$

equivalent formulation:

$$\arg\max_{x_1 \ldots x_n, y_1 \ldots y_n} \sum_{(i,j)\in T_1} f'_{i,j}(x_i, x_j) + \sum_{(i,j)\in T_2} f'_{i,j}(y_i, y_j)$$

such that for $i = 1 \ldots n$,

$$x_i = y_i$$

Lagrangian:

$$L(u, x, y) = \sum_{(i,j)\in T_1} f'_{i,j}(x_i, x_j) + \sum_{(i,j)\in T_2} f'_{i,j}(y_i, y_j) + \sum_i u_i(x_i - y_i)$$

4. Practical issues

aim: overview of practical dual decomposition techniques

- tracking the progress of the algorithm
- choice of update rate α_k
- lazy update of dual solutions
- extracting solutions if algorithm does not converge

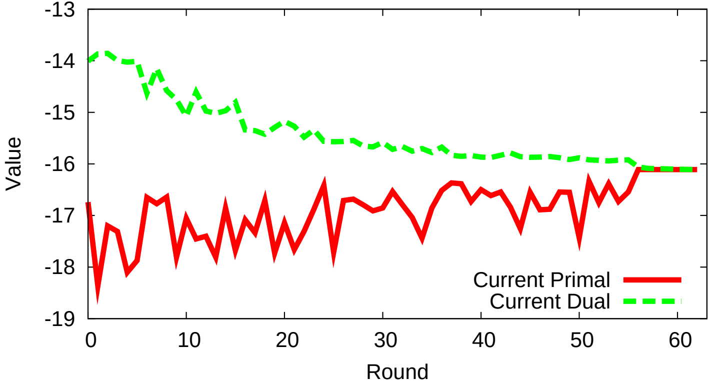
Optimization tracking

at each stage of the algorithm there are several useful values

track:

- $y^{(k)}$, $z^{(k)}$ are current dual solutions
- $L(u^{(k)})$ is the current dual value
- $y^{(k)}$, $l(y^{(k)})$ is a potential primal feasible solution
- $f(y^{(k)}) + g(l(y^{(k)}))$ is the potential primal value

Tracking example



example run from syntactic machine translation (later in talk)

- current primal

$$f(y^{(k)}) + g(l(y^{(k)}))$$

- current dual

$$L(u^{(k)})$$

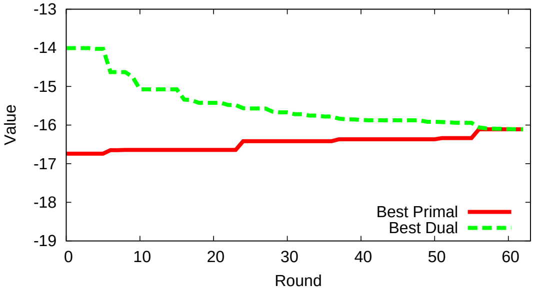
Optimization progress

useful signals:

- $L(u^{(k)}) - L(u^{(k-1)})$ is the dual change (may be positive)
- $\min_k L(u^{(k)})$ is the best dual value (tightest upper bound)
- $\max_k f(y^{(k)}) + g(l(y^{(k)}))$ is the best primal value

the optimal value must be between the best dual and primal values

Progress example

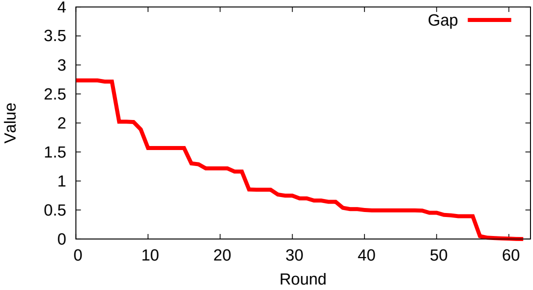


best primal

$$\max_k f(y^{(k)}) + g(l(y^{(k)}))$$

best dual

$$\min_k L(u^{(k)})$$



gap

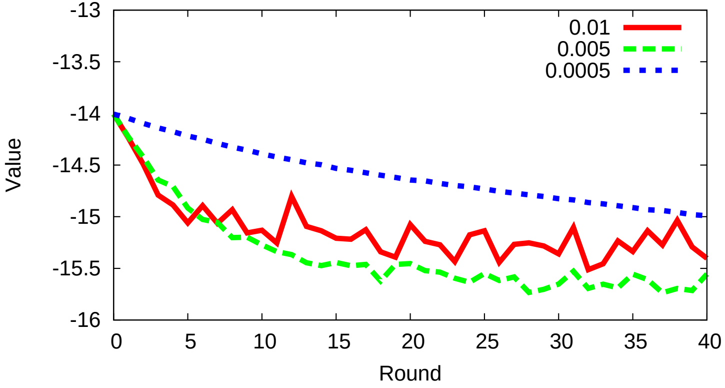
$$\min_k L(u^k) \quad -$$

$$\max_k f(y^{(k)}) \quad + \quad g(l(y^{(k)})$$

Update rate

choice of α_k has important practical consequences

- α_k too high causes dual value to fluctuate
- α_k too low means slow progress



Update rate

practical: find a rate that is robust to varying inputs

- $\alpha_k = c$ (constant rate) can be very fast, but hard to find constant that works for all problems
- $\alpha_k = \frac{c}{k}$ (decreasing rate) often cuts rate too aggressively, lowers value even when making progress
- rate based on dual progress
  - $\alpha_k = \frac{c}{t+1}$ where $t < k$ is number of iterations where dual value increased
  - robust in practice, reduces rate when dual value is fluctuating

Lazy decoding

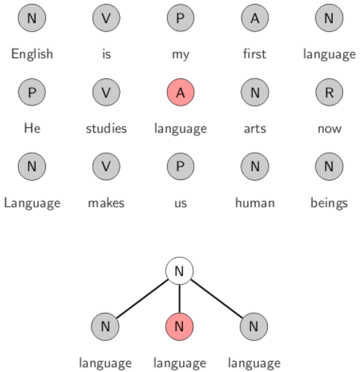
idea: don't recompute $y^{(k)}$ or $z^{(k)}$ from scratch each iteration

lazy decoding: if subgradient $u^{(k)}$ is sparse, then $y^{(k)}$ may be very easy to compute from $y^{(k-1)}$

use:

- helpful if y or z factor naturally into independent components
- can be important for fast decompositions

Lazy decoding example



recall corpus-level tagging example

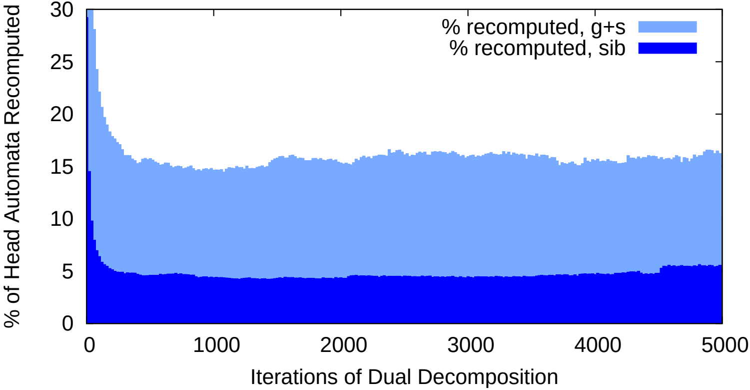
at this iteration, only sentence 2 receives a weight update

with lazy decoding

$$Y_1^{(k)} \leftarrow Y_1^{(k-1)}$$
$$Y_3^{(k)} \leftarrow Y_3^{(k-1)}$$

Lazy decoding results

lazy decoding is critical for the efficiency of some applications



recomputation statistics for non-projective dependency parsing

Approximate solution

upon agreement the solution is exact, but this may not occur

otherwise, there is an easy way to find an approximate solution

choose: the structure $y^{(k')}$ where

$$k' = \arg\max_k f(y^{(k)}) + g(l(y^{(k)}))$$

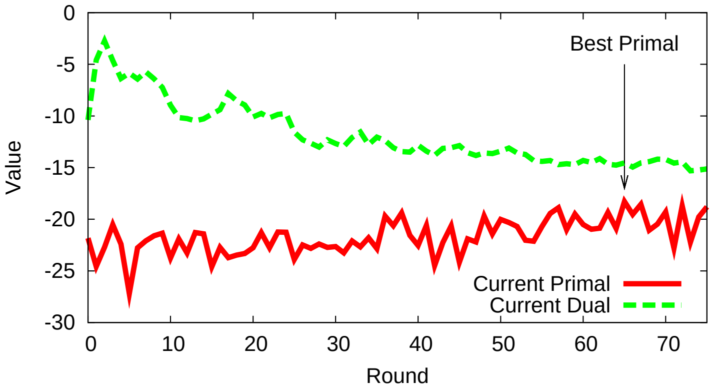
is the iteration with the best primal score

guarantee: the solution $y^{k'}$ is non-optimal by at most

$$(\min_k L(u^k)) - (f(y^{(k')}) + g(l(y^{(k')})))$$

there are other methods to estimate solutions, for instance by averaging solutions (see **?**)

Choosing best solution

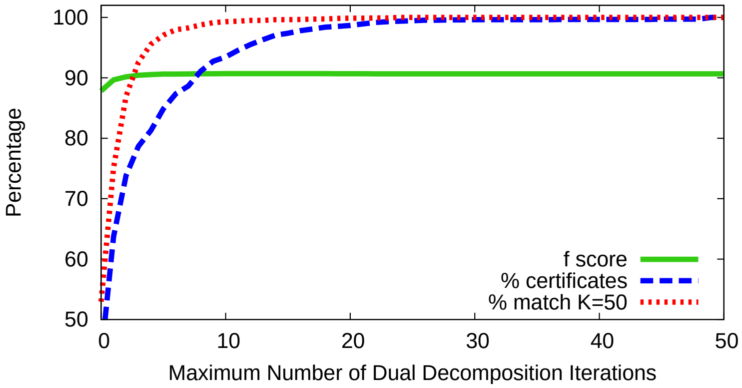


non-exact example from syntactic translation

best approximate primal solution occurs at iteration 63

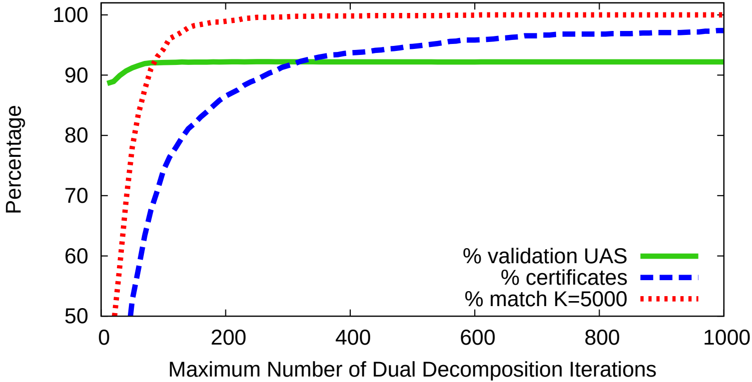
Early stopping results

early stopping results for constituency and dependency parsing



Early stopping results

early stopping results for non-projective dependency parsing



Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:

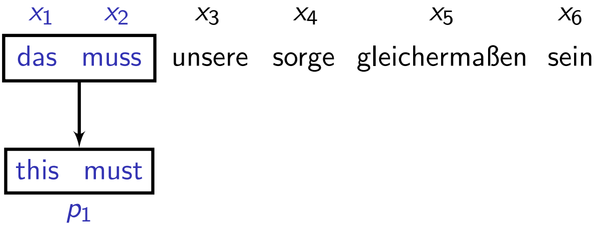
x_1	x_2	x_3	x_4	x_5	x_6
das	muss	unsere	sorge	gleichermaßen	sein

Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:



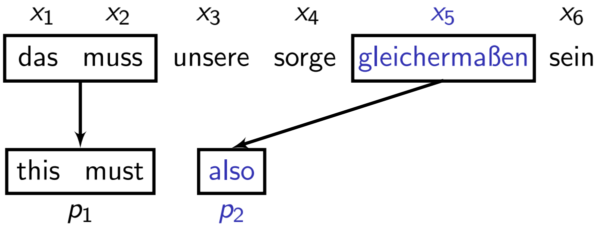
$y = \{(1, 2, \text{this must}),$

Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:



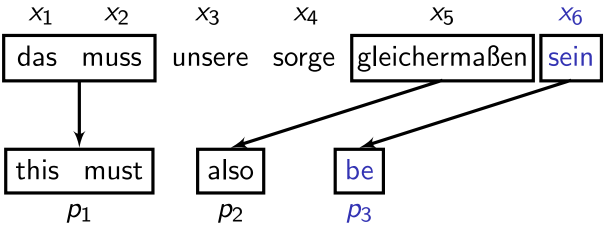
$y = \{(1, 2, \text{this must}),\ (5, 5, \text{also}),$

Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:



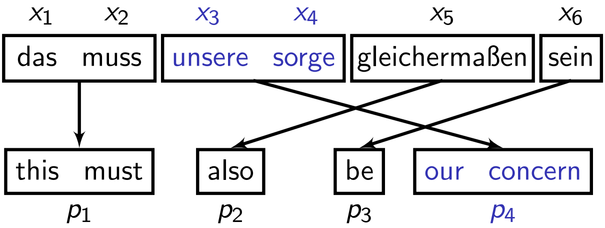
$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}),$

Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:



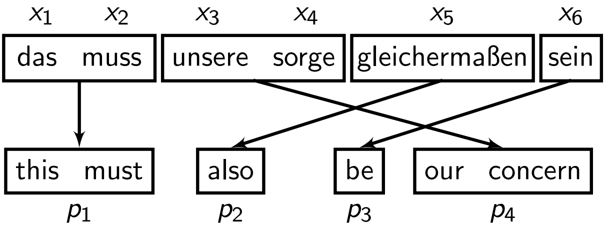
$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$

Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:



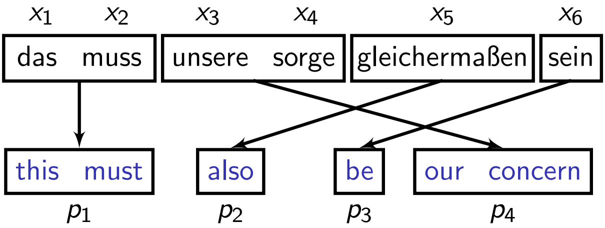
$$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$$

Phrase-Based Translation

define:

- source-language sentence words $x_1, \ldots, x_N$
- phrase translation $p = (s, e, t)$
- translation derivation $y = p_1, \ldots, p_L$

example:

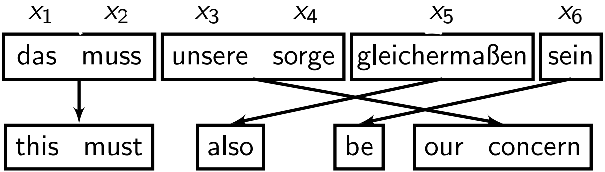


$$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$$

Scoring Derivations

derivation:

$$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$$



objective:

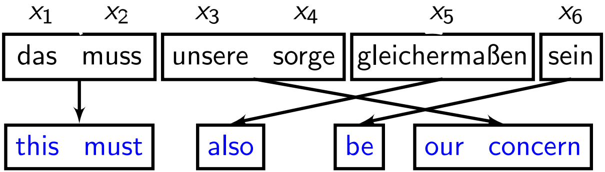
$$f(y) = h(e(y)) + \sum_{k=1}^{L} g(p_k) + \sum_{k=1}^{L-1} \eta |t(p_k) + 1 - s(p_{k+1})|$$

- language model score h
- phrase translation score g
- distortion penalty η

Scoring Derivations

derivation:

$$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$$



objective:

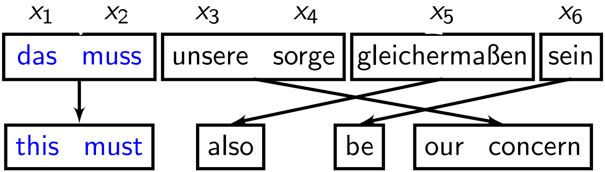
$$f(y) = h(e(y)) + \sum_{k=1}^{L} g(p_k) + \sum_{k=1}^{L-1} \eta |t(p_k) + 1 - s(p_{k+1})|$$

- language model score h
- phrase translation score g
- distortion penalty η

Scoring Derivations

derivation:

$$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$$



objective:

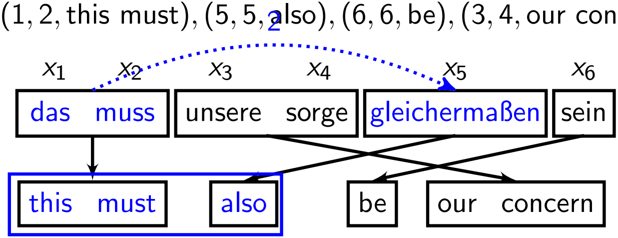
$$f(y) = h(e(y)) + \sum_{k=1}^{L} g(p_k) + \sum_{k=1}^{L-1} \eta |t(p_k) + 1 - s(p_{k+1})|$$

- language model score h
- phrase translation score g
- distortion penalty η

Scoring Derivations

derivation:

$$y = \{(1, 2, \text{this must}), (5, 5, \text{also}), (6, 6, \text{be}), (3, 4, \text{our concern})\}$$



objective:

$$f(y) = h(e(y)) + \sum_{k=1}^{L} g(p_k) + \sum_{k=1}^{L-1} \eta |t(p_k) + 1 - s(p_{k+1})|$$

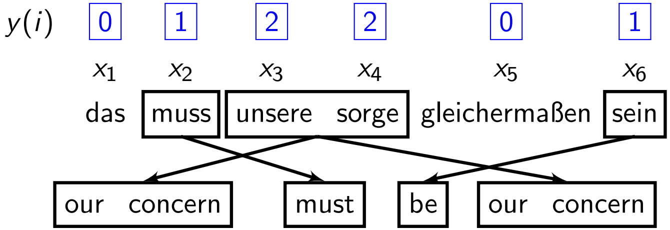
- language model score h
- phrase translation score g
- distortion penalty η

Relaxed Problem

$\mathcal{Y}'$: only requires the total number of words translated to be N

$$\mathcal{Y}' = \{y : \sum_{i=1}^{N} y(i) = N \text{ and the distortion limit } d \text{ is satisfied}\}$$

example:



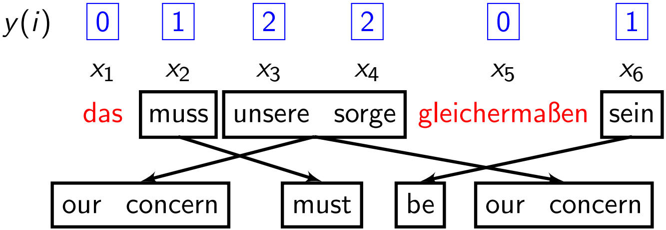
$(3, 4, \text{our concern})(2, 2, \text{must})(6, 6, \text{be})(3, 4, \text{our concern})$

Relaxed Problem

$\mathcal{Y}'$: only requires the total number of words translated to be N

$$\mathcal{Y}' = \{y : \sum_{i=1}^{N} y(i) = N \text{ and the distortion limit } d \text{ is satisfied}\}$$

example:



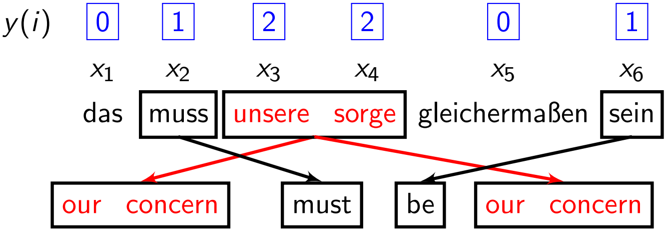
$(3, 4, \text{our concern})(2, 2, \text{must})(6, 6, \text{be})(3, 4, \text{our concern})$

Relaxed Problem

$\mathcal{Y}'$: only requires the total number of words translated to be N

$$\mathcal{Y}' = \{y : \sum_{i=1}^{N} y(i) = N \text{ and the distortion limit } d \text{ is satisfied}\}$$

example:



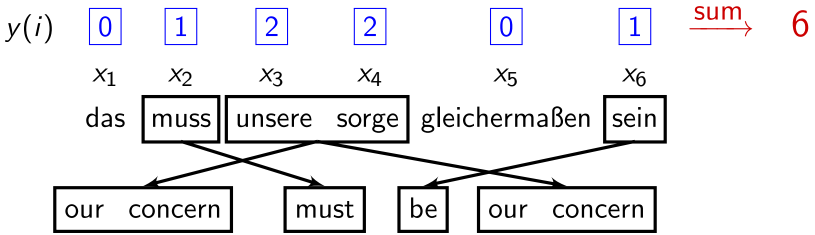
$(3, 4, \text{our concern})(2, 2, \text{must})(6, 6, \text{be})(3, 4, \text{our concern})$

Relaxed Problem

$\mathcal{Y}'$: only requires the total number of words translated to be N

$$\mathcal{Y}' = \{y : \sum_{i=1}^{N} y(i) = N \text{ and the distortion limit } d \text{ is satisfied}\}$$

example:



$(3, 4, \text{our concern})(2, 2, \text{must})(6, 6, \text{be})(3, 4, \text{our concern})$

Lagrangian Relaxation Method

original:

$$\underbrace{\arg\max_{y \in \mathcal{Y}} f(y)}$$

$$\mathcal{Y} = \{y : y(i) = 1 \ \forall i = 1 \ldots N\} \qquad \boxed{1}\,\boxed{1} \ldots \boxed{1}$$

rewrite:

$$\underbrace{\arg\max_{y \in \mathcal{Y}'} f(y)} \qquad \text{such that} \qquad \underbrace{y(i) = 1 \ \forall i = 1 \ldots N}$$

$$\mathcal{Y}' = \{y : \textstyle\sum_{i=1}^{N} y(i) = N\} \qquad \underbrace{\boxed{2}\,\boxed{0} \ldots \boxed{1}}_{\text{sum to } N}$$

Lagrangian Relaxation Method

original:

$$\underbrace{\arg\max_{y \in \mathcal{Y}} f(y)}_{\text{exact DP is NP-hard}}$$

$$\mathcal{Y} = \{y : y(i) = 1 \ \forall i = 1 \dots N\} \qquad \boxed{1}\,\boxed{1} \dots \boxed{1}$$

rewrite:

$$\underbrace{\arg\max_{y \in \mathcal{Y}'} f(y)}_{} \qquad \text{such that} \qquad \underbrace{y(i) = 1 \ \forall i = 1 \dots N}_{}$$

$$\mathcal{Y}' = \{y : \textstyle\sum_{i=1}^{N} y(i) = N\} \qquad \underbrace{\boxed{2}\,\boxed{0} \dots \boxed{1}}_{\text{sum to } N}$$

Lagrangian Relaxation Method

original:

$$\underbrace{\arg\max_{y \in \mathcal{Y}} f(y)}_{\text{exact DP is NP-hard}}$$

$$\mathcal{Y} = \{y : y(i) = 1 \ \forall i = 1 \ldots N\} \qquad \boxed{1}\,\boxed{1} \ldots \boxed{1}$$

rewrite:

$$\underbrace{\arg\max_{y \in \mathcal{Y}'} f(y)}\quad \text{such that} \quad \underbrace{y(i) = 1 \ \forall i = 1 \ldots N}$$

$$\mathcal{Y}' = \{y : \textstyle\sum_{i=1}^{N} y(i) = N\} \qquad \underbrace{\boxed{2}\,\boxed{0} \ldots \boxed{1}}_{\text{sum to } N}$$

Lagrangian Relaxation Method

original:

$$\underbrace{\arg\max_{y \in \mathcal{Y}} f(y)}_{\text{exact DP is NP-hard}}$$

$$\mathcal{Y} = \{y : y(i) = 1 \; \forall i = 1 \ldots N\} \qquad \boxed{1}\,\boxed{1}\ldots\boxed{1}$$

rewrite:

$$\underbrace{\arg\max_{y \in \mathcal{Y}'} f(y)}_{\text{can be solved efficiently by DP}} \qquad \text{such that} \qquad \underbrace{y(i) = 1 \; \forall i = 1 \ldots N}$$

$$\mathcal{Y}' = \{y : \sum_{i=1}^{N} y(i) = N\} \qquad \underbrace{\boxed{2}\,\boxed{0}\ldots\boxed{1}}_{\text{sum to } N}$$

Lagrangian Relaxation Method

original:

$$\underbrace{\arg\max_{y \in \mathcal{Y}} f(y)}_{\text{exact DP is NP-hard}}$$

$$\mathcal{Y} = \{y : y(i) = 1\ \forall i = 1 \ldots N\} \qquad \boxed{1}\,\boxed{1} \ldots \boxed{1}$$

rewrite:

$$\underbrace{\arg\max_{y \in \mathcal{Y}'} f(y)}_{\text{can be solved efficiently by DP}} \quad \text{such that} \quad \underbrace{y(i) = 1\ \forall i = 1 \ldots N}_{\text{using Lagrangian relaxation}}$$

$$\mathcal{Y}' = \{y : \sum_{i=1}^{N} y(i) = N\} \qquad \underbrace{\boxed{2}\,\boxed{0} \ldots \boxed{1}}_{\text{sum to } N}$$

Algorithm

Iteration 1:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

  $\alpha = 1$

$u(i)$	0	0	0	0	0	0
$y(i)$						
	x_1	x_2	x_3	x_4	x_5	x_6
	das	muss	unsere	sorge	gleichermaßen	sein

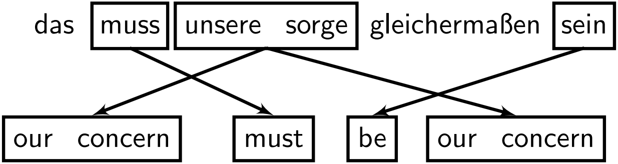
Algorithm

Iteration 1:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

  $\alpha = 1$

$u(i)$	0	0	0	0	0	0
$y(i)$	$\boxed{0}$	$\boxed{1}$	$\boxed{2}$	$\boxed{2}$	$\boxed{0}$	$\boxed{1}$
	x_1	x_2	x_3	x_4	x_5	x_6

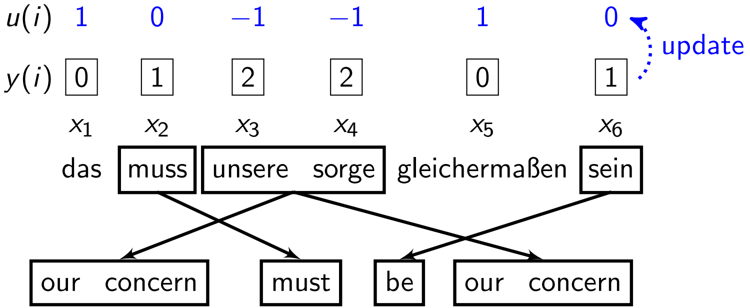


Algorithm

Iteration 1:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

$$\alpha = 1$$



Algorithm

Iteration 2:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

  $\alpha = 0.5$

$u(i)$	1	0	-1	-1	1	0
$y(i)$						
	x_1	x_2	x_3	x_4	x_5	x_6
	das	muss	unsere	sorge	gleichermaßen	sein

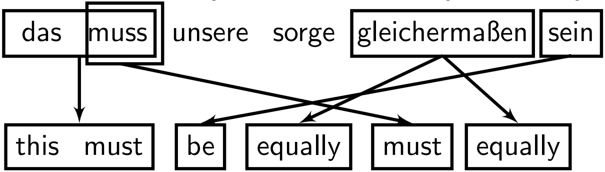
Algorithm

Iteration 2:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

  $\alpha = 0.5$

$u(i)$	1	0	-1	-1	1	0
$y(i)$	$\boxed{1}$	$\boxed{2}$	$\boxed{0}$	$\boxed{0}$	$\boxed{2}$	$\boxed{1}$
	x_1	x_2	x_3	x_4	x_5	x_6

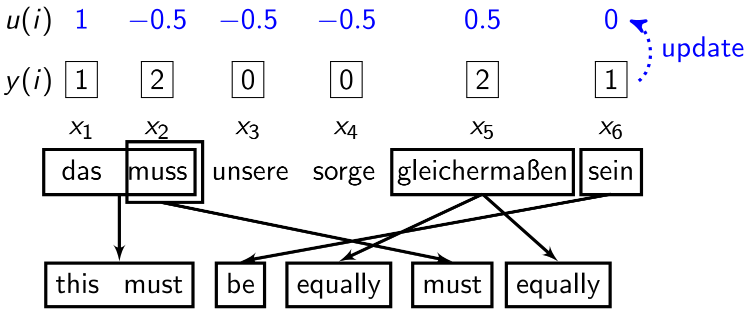


Algorithm

Iteration 2:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

  $\alpha = 0.5$



Algorithm

Iteration 3:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

  $\alpha = 0.5$

$u(i)$	1	-0.5	-0.5	-0.5	0.5	0
$y(i)$						
	x_1	x_2	x_3	x_4	x_5	x_6
	das	muss	unsere	sorge	gleichermaßen	sein

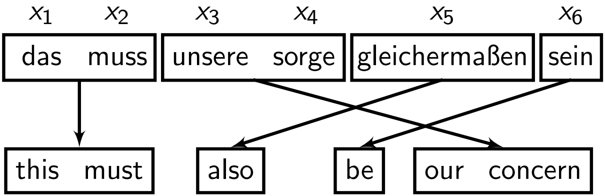
Algorithm

Iteration 3:

- update $u(i)$: $u(i) \leftarrow u(i) - \alpha(y(i) - 1)$

$$\alpha = 0.5$$

$u(i)$	1	-0.5	-0.5	-0.5	0.5	0
$y(i)$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$



Tightening the Relaxation

In some cases, we never reach $y(i) = 1$ for $i = 1 \ldots N$

If dual $L(u)$ is not decreasing fast enough

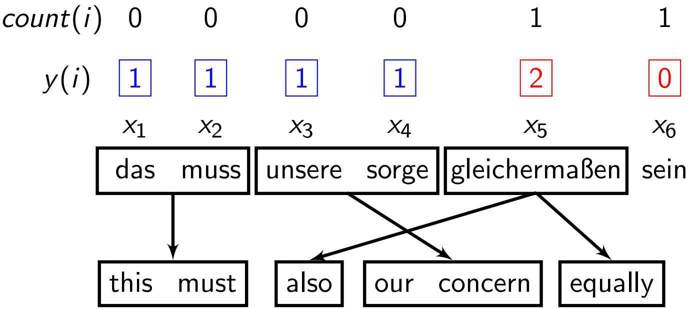
run for 10 more iterations

count number of times each constraint is violated

add 3 most often violated constraints

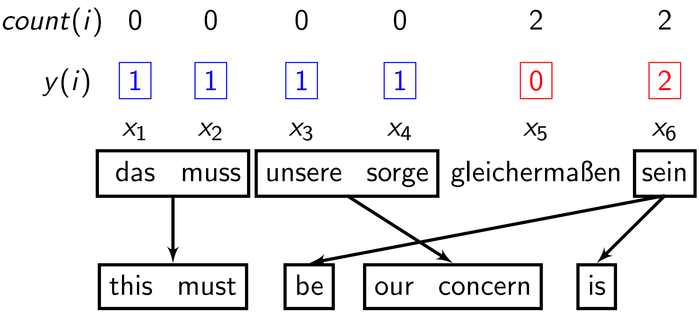
Tightening the Relaxation

Iteration 41:



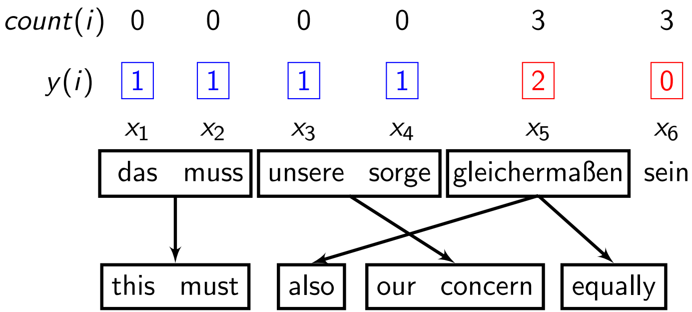
Tightening the Relaxation

Iteration 42:



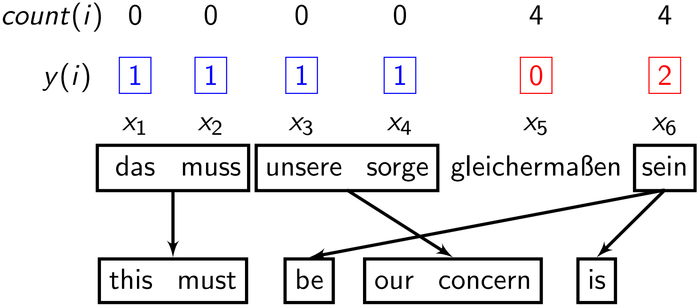
Tightening the Relaxation

Iteration 43:



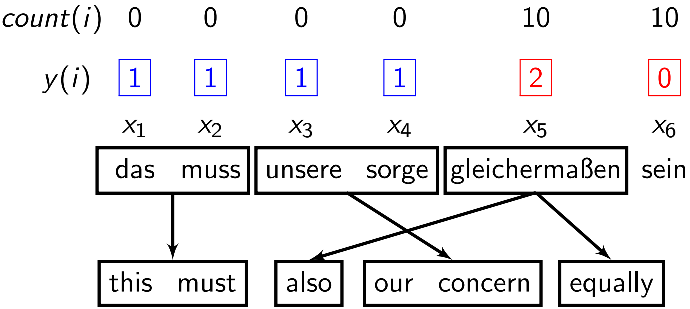
Tightening the Relaxation

Iteration 44:



Tightening the Relaxation

Iteration 50:



Tightening the Relaxation

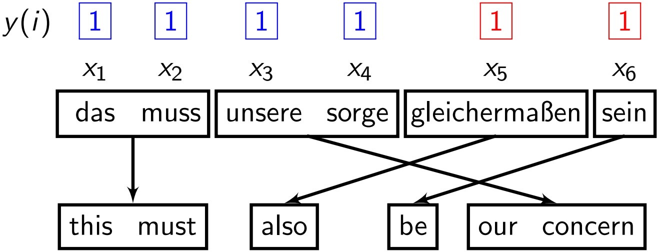
Iteration 51:

$y(i)$					$\boxed{1}$	$\boxed{1}$
	x_1	x_2	x_3	x_4	x_5	x_6
	das	muss	unsere	sorge	gleichermaßen	sein

Add 2 hard constraints (x_5, x_6) to the dynamic program

Tightening the Relaxation

Iteration 51:

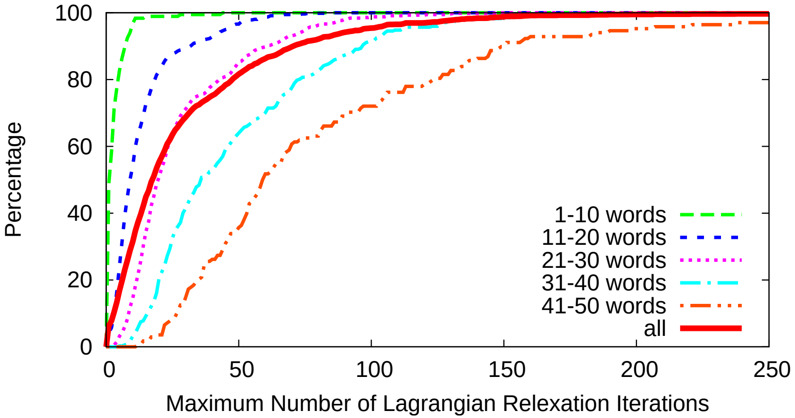


Add 2 hard constraints (x_5, x_6) to the dynamic program

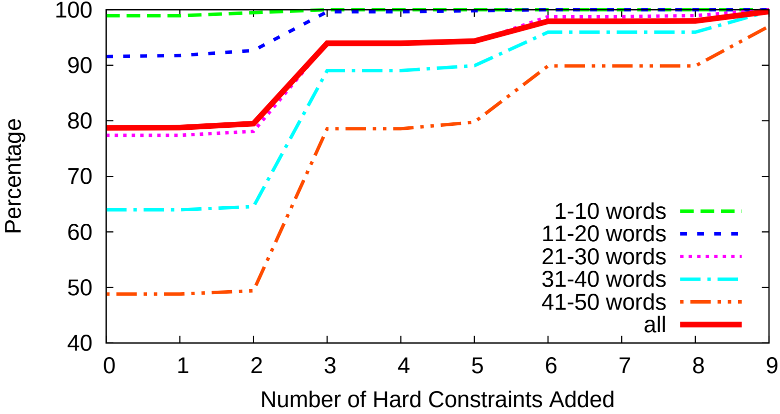
Experiments: German to English

- Europarl data: German to English
- Test on 1,824 sentences with length 1-50 words
- Converged: 1,818 sentences (99.67%)

Experiments: Number of Iterations



Experiments: Number of Hard Constraints Required



Experiments: Mean Time in Seconds

# words	1-10	11-20	21-30	31-40	41-50	All
mean	0.8	10.9	57.2	203.4	679.9	120.9
median	0.7	8.9	48.3	169.7	484.0	35.2

Comparison to ILP Decoding

	(sec.)	(sec.)
1-10	275.2	132.9
11-15	2,707.8	1,138.5
16-20	20,583.1	3,692.6

Higher-order non-projective dependency parsing

setup: given a model for higher-order non-projective dependency parsing (sibling features)

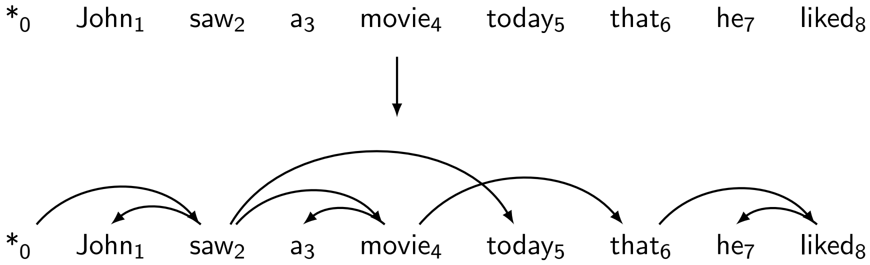
problem: find non-projective dependency parse that maximizes the score of this model

difficulty:

- model is NP-hard to decode
- complexity of the model comes from enforcing combinatorial constraints

strategy: design a decomposition that separates combinatorial constraints from direct implementation of the scoring function

Non-Projective Dependency Parsing



Important problem in many languages.

Problem is NP-Hard for all but the simplest models.

Dual Decomposition

A classical technique for constructing decoding algorithms.

Solve complicated models

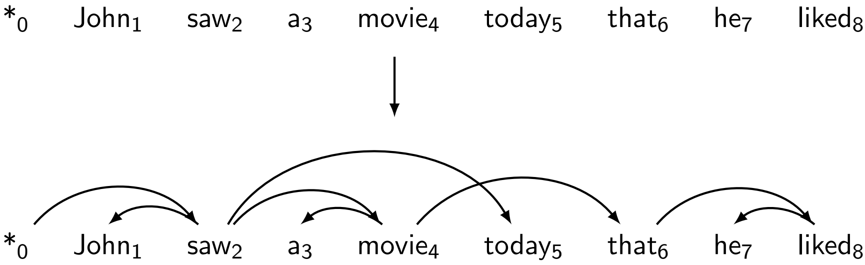
$$y^* = \arg\max_y f(y)$$

by decomposing into smaller problems.

Upshot: Can utilize a toolbox of combinatorial algorithms.

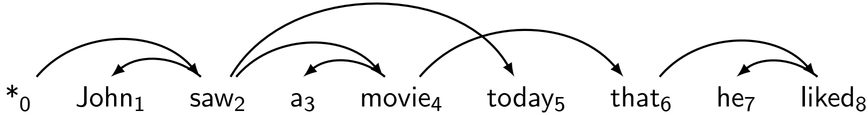
- Dynamic programming
- Minimum spanning tree
- Shortest path
- Min-Cut
- ...

Non-Projective Dependency Parsing



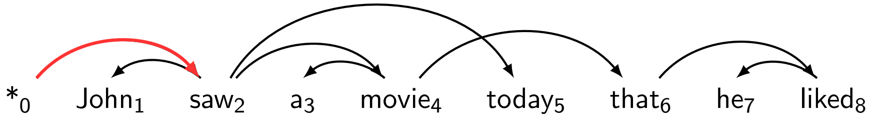
- Starts at the root symbol *
- Each word has a exactly one parent word
- Produces a tree structure (no cycles)
- Dependencies can cross

Arc-Factored



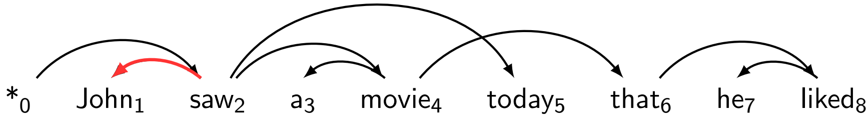
$f(y) =$

Arc-Factored



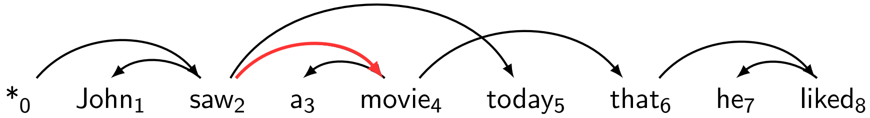
$f(y) = \textit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2)$

Arc-Factored



$$f(y) = \mathit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2) + \mathit{score}(\text{saw}_2, \text{John}_1)$$

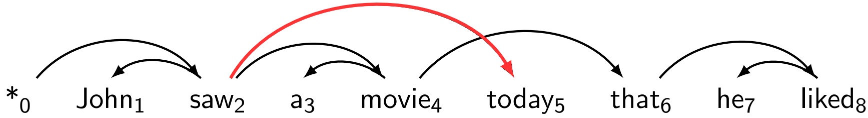
Arc-Factored



$$f(y) = \mathit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2) + \mathit{score}(\text{saw}_2, \text{John}_1)$$

$$+\mathit{score}(\text{saw}_2, \text{movie}_4)$$

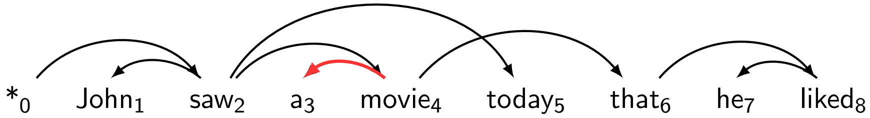
Arc-Factored



$$f(y) = \textit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2) + \textit{score}(\text{saw}_2, \text{John}_1)$$

$$+ \textit{score}(\text{saw}_2, \text{movie}_4) + \textit{score}(\text{saw}_2, \text{today}_5)$$

Arc-Factored

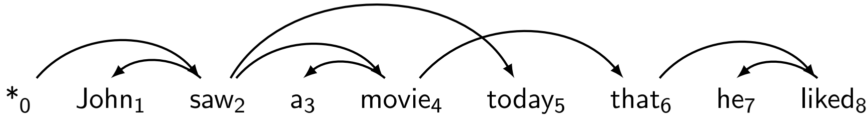


$$f(y) = \mathit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2) + \mathit{score}(\text{saw}_2, \text{John}_1)$$

$$+\mathit{score}(\text{saw}_2, \text{movie}_4) + \mathit{score}(\text{saw}_2, \text{today}_5)$$

$$+\mathit{score}(\text{movie}_4, \text{a}_3) + \ldots$$

Arc-Factored



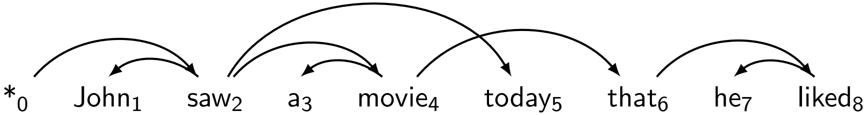
$$f(y) = \textit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2) + \textit{score}(\text{saw}_2, \text{John}_1)$$

$$+\textit{score}(\text{saw}_2, \text{movie}_4) + \textit{score}(\text{saw}_2, \text{today}_5)$$

$$+\textit{score}(\text{movie}_4, \text{a}_3) + \ldots$$

e.g. $\textit{score}(*_0, \text{saw}_2) = \log p(\text{saw}_2 | *_0)$ (generative model)

Arc-Factored



$$f(y) = \mathit{score}(\mathrm{head} = *_0, \mathrm{mod} = \mathrm{saw}_2) + \mathit{score}(\mathrm{saw}_2, \mathrm{John}_1)$$

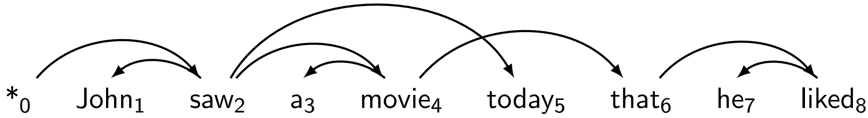
$$+ \mathit{score}(\mathrm{saw}_2, \mathrm{movie}_4) + \mathit{score}(\mathrm{saw}_2, \mathrm{today}_5)$$

$$+ \mathit{score}(\mathrm{movie}_4, \mathrm{a}_3) + \ldots$$

e.g. $\mathit{score}(*_0, \mathrm{saw}_2) = \log p(\mathrm{saw}_2 | *_0)$ (generative model)

or $\mathit{score}(*_0, \mathrm{saw}_2) = w \cdot \phi(\mathrm{saw}_2, *_0)$ (CRF/perceptron model)

Arc-Factored



$$f(y) = \mathit{score}(\text{head} = *_0, \text{mod} = \text{saw}_2) + \mathit{score}(\text{saw}_2, \text{John}_1)$$

$$+\mathit{score}(\text{saw}_2, \text{movie}_4) + \mathit{score}(\text{saw}_2, \text{today}_5)$$

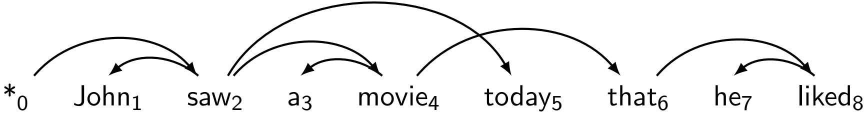
$$+\mathit{score}(\text{movie}_4, \text{a}_3) + \ldots$$

e.g. $\mathit{score}(*_0, \text{saw}_2) = \log p(\text{saw}_2 | *_0)$ (generative model)

or $\mathit{score}(*_0, \text{saw}_2) = w \cdot \phi(\text{saw}_2, *_0)$ (CRF/perceptron model)

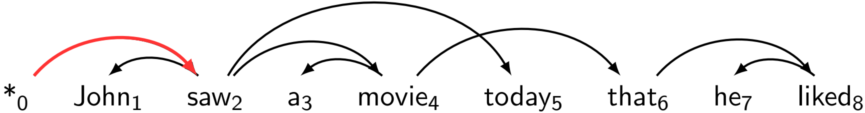
$$y^* = \arg\max_y f(y) \Leftarrow \text{Minimum Spanning Tree Algorithm}$$

Sibling Models



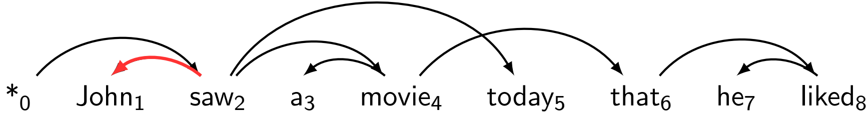
$f(y) =$

Sibling Models



$f(y) = \mathit{score}(\mathit{head} = *_0, \mathit{prev} = \mathrm{NULL}, \mathit{mod} = \mathrm{saw}_2)$

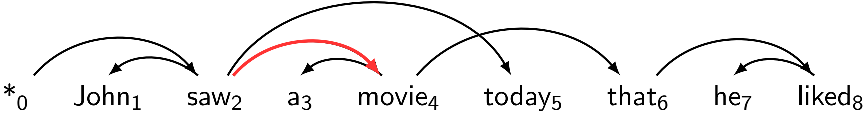
Sibling Models



$$f(y) = \mathit{score}(\mathit{head} = *_0, \mathit{prev} = \mathrm{NULL}, \mathit{mod} = \mathrm{saw}_2)$$

$$+\mathit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1)$$

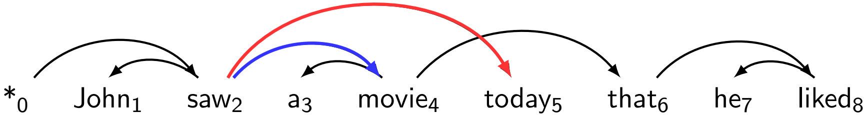
Sibling Models



$$f(y) = \mathit{score}(\mathit{head} = *_0, \mathit{prev} = \text{NULL}, \mathit{mod} = \text{saw}_2)$$

$$+\mathit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) \; +\mathit{score}(\text{saw}_2, \text{NULL}, \text{movie}_4)$$

Sibling Models

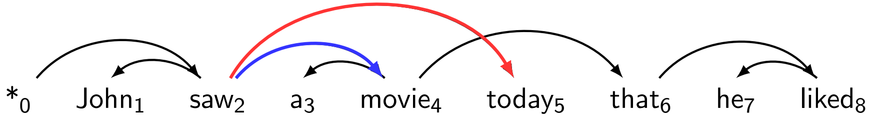


$$f(y) = \mathit{score}(\mathit{head} = *_0, \mathit{prev} = \mathrm{NULL}, \mathit{mod} = \mathrm{saw}_2)$$

$$+\mathit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1) + \mathit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{movie}_4)$$

$$+\mathit{score}(\mathrm{saw}_2, \mathrm{movie}_4, \mathrm{today}_5) + \ldots$$

Sibling Models



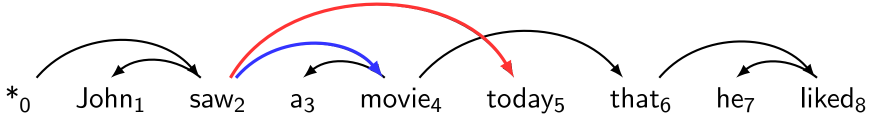
$$f(y) = \mathit{score}(\mathit{head} = *_0, \mathit{prev} = \mathrm{NULL}, \mathit{mod} = \mathrm{saw}_2)$$

$$+\mathit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1) + \mathit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{movie}_4)$$

$$+\mathit{score}(\mathrm{saw}_2, \mathrm{movie}_4, \mathrm{today}_5) + \ldots$$

e.g. $\mathit{score}(\mathrm{saw}_2, \mathrm{movie}_4, \mathrm{today}_5) = \log p(\mathrm{today}_5 | \mathrm{saw}_2, \mathrm{movie}_4)$

Sibling Models



$$f(y) = score(head = *_0, prev = \text{NULL}, mod = \text{saw}_2)$$

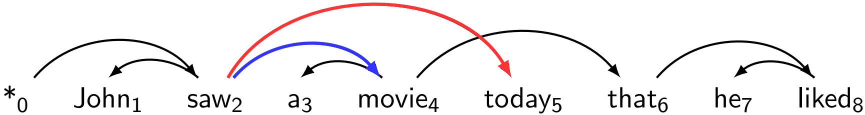
$$+score(\text{saw}_2, \text{NULL}, \text{John}_1) + score(\text{saw}_2, \text{NULL}, \text{movie}_4)$$

$$+score(\text{saw}_2, \text{movie}_4, \text{today}_5) + \ldots$$

e.g. $score(\text{saw}_2, \text{movie}_4, \text{today}_5) = \log p(\text{today}_5 | \text{saw}_2, \text{movie}_4)$

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

Sibling Models



$$f(y) = \mathit{score}(\mathit{head} = *_0, \mathit{prev} = \text{NULL}, \mathit{mod} = \text{saw}_2)$$

$$+\mathit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) + \mathit{score}(\text{saw}_2, \text{NULL}, \text{movie}_4)$$

$$+\mathit{score}(\text{saw}_2, \color{blue}{\text{movie}_4}, \color{red}{\text{today}_5}) + \ldots$$

e.g. $\mathit{score}(\text{saw}_2, \text{movie}_4, \text{today}_5) = \log p(\text{today}_5 | \text{saw}_2, \text{movie}_4)$

or $\mathit{score}(\text{saw}_2, \text{movie}_4, \text{today}_5) = w \cdot \phi(\text{saw}_2, \text{movie}_4, \text{today}_5)$

$$y^* = \arg\max_y f(y) \Leftarrow \text{NP-Hard}$$

Thought Experiment: Individual Decoding

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

Thoughtexperiment: Individual Decoding

$*_0$ John_1 saw_2 a_3 movie_4 today_5 that_6 he_7 liked_8

$$\textit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) + \textit{score}(\text{saw}_2, \text{NULL}, \text{movie}_4)$$
$$+\textit{score}(\text{saw}_2, \text{movie}_4, \text{today}_5)$$

Thought Experiment: Individual Decoding

$*_0$ $\quad$ John_1 $\quad$ saw_2 $\quad$ a_3 $\quad$ movie_4 $\quad$ today_5 $\quad$ that_6 $\quad$ he_7 $\quad$ liked_8

$$\textit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) + \textit{score}(\text{saw}_2, \text{NULL}, \text{movie}_4)$$
$$+\textit{score}(\text{saw}_2, \text{movie}_4, \text{today}_5)$$

$$\textit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) + \textit{score}(\text{saw}_2, \text{NULL}, \text{that}_6)$$

Thought Experiment: Individual Decoding

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$\textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1) + \textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{movie}_4)$
$+\textit{score}(\mathrm{saw}_2, \mathrm{movie}_4, \mathrm{today}_5)$

$\textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1) + \textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{that}_6)$

$\textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{a}_3) + \textit{score}(\mathrm{saw}_2, \mathrm{a}_3, \mathrm{he}_7)$

Thought Experiment: Individual Decoding

$*_0$ $\quad$ John_1 $\quad$ saw_2 $\quad$ a_3 $\quad$ movie_4 $\quad$ today_5 $\quad$ that_6 $\quad$ he_7 $\quad$ liked_8

2^{n-1} **possibilities**

$$\textit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) + \textit{score}(\text{saw}_2, \text{NULL}, \text{movie}_4) + \textit{score}(\text{saw}_2, \text{movie}_4, \text{today}_5)$$

$$\textit{score}(\text{saw}_2, \text{NULL}, \text{John}_1) + \textit{score}(\text{saw}_2, \text{NULL}, \text{that}_6)$$

$$\textit{score}(\text{saw}_2, \text{NULL}, \text{a}_3) + \textit{score}(\text{saw}_2, \text{a}_3, \text{he}_7)$$

Thought Experiment: Individual Decoding

$*_0$ $\quad$ John$_1$ $\quad$ saw$_2$ $\quad$ a$_3$ $\quad$ movie$_4$ $\quad$ today$_5$ $\quad$ that$_6$ $\quad$ he$_7$ $\quad$ liked$_8$

2^{n-1} **possibilities**

$$\textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1) + \textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{movie}_4) + \textit{score}(\mathrm{saw}_2, \mathrm{movie}_4, \mathrm{today}_5)$$

$$\textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{John}_1) + \textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{that}_6)$$

$$\textit{score}(\mathrm{saw}_2, \mathrm{NULL}, \mathrm{a}_3) + \textit{score}(\mathrm{saw}_2, \mathrm{a}_3, \mathrm{he}_7)$$

Under Sibling Model, can solve for each word with Viterbi decoding.

Thought Experiment Continued

$*_0$ John_1 saw_2 a_3 movie_4 today_5 that_6 he_7 liked_8

Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

Thought Experiment Continued

$*_0$ John_1 saw_2 a_3 movie_4 today_5 that_6 he_7 liked_8

Idea: Do individual decoding for each head word using dynamic programming.

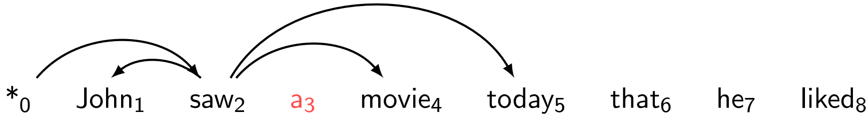
If we're lucky, we'll end up with a valid final tree.

Thought Experiment Continued

Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

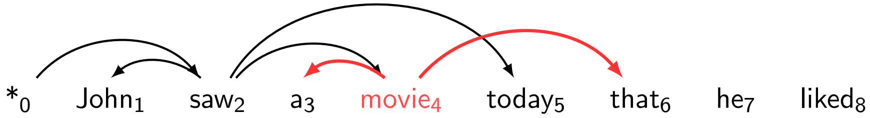
Thought Experiment Continued



Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

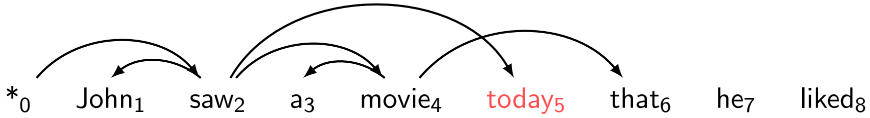
Thought Experiment Continued



Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

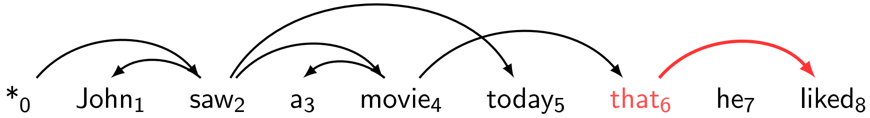
Thought Experiment Continued



Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

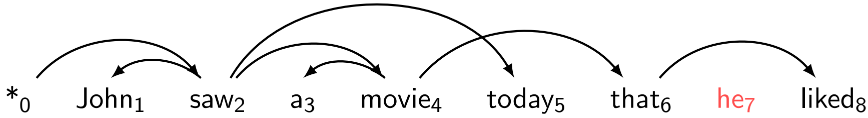
Thought Experiment Continued



Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

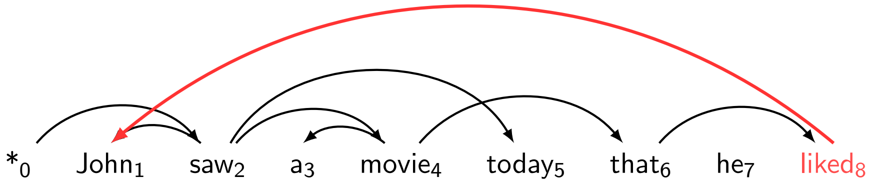
Thought Experiment Continued



Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

Thought Experiment Continued



Idea: Do individual decoding for each head word using dynamic programming.

If we're lucky, we'll end up with a valid final tree.

But we might violate some constraints.

Dual Decomposition Structure

$$\text{Goal } y^* = \arg\max_{y \in \mathcal{Y}} f(y)$$

Dual Decomposition Structure

Goal $y^* = \arg\max_{y \in \mathcal{Y}} f(y)$

Rewrite as $\underset{z \in \mathcal{Z},\ y \in \mathcal{Y}}{\text{argmax}} \quad f(z) + g(y)$

such that $z = y$

Dual Decomposition Structure

$$\text{Goal } y^* = \arg\max_{y \in \mathcal{Y}} f(y)$$

$$\text{Rewrite as} \quad \underset{\substack{z \in \mathcal{Z},\ y \in \mathcal{Y}}}{\text{argmax}} \quad f(z) \ + \ g(y)$$

All Possible

$$\text{such that } z = y$$

Dual Decomposition Structure

$$\text{Goal } y^* = \arg\max_{y \in \mathcal{Y}} f(y)$$

$$\text{Rewrite as } \underset{\underset{\text{All Possible}}{z \in \mathcal{Z}},\ \underset{\text{Valid Trees}}{y \in \mathcal{Y}}}{\operatorname{argmax}} \quad f(z) \ + \ g(y)$$

such that $z = y$

Dual Decomposition Structure

$$\text{Goal } y^* = \arg\max_{y \in \mathcal{Y}} f(y)$$

$$\text{Rewrite as} \quad \operatorname*{argmax}_{\underbrace{z \in \mathcal{Z}}_{\textbf{All Possible}},\ \underbrace{y \in \mathcal{Y}}_{\textbf{Valid Trees}}} \quad \overbrace{f(z)}^{\textbf{Sibling}} + g(y)$$

$$\text{such that } z = y$$

Dual Decomposition Structure

$$\text{Goal } y^* = \arg\max_{y \in \mathcal{Y}} f(y)$$

$$\text{Rewrite as} \quad \underset{z \in \mathcal{Z},\ y \in \mathcal{Y}}{\text{argmax}} \quad \overbrace{f(z)}^{\textbf{Sibling}} + \overbrace{g(y)}^{\textbf{Arc-Factored}}$$

All Possible **Valid Trees**

$$\text{such that } z = y$$

Dual Decomposition Structure

$$\text{Goal } y^* = \arg\max_{y \in \mathcal{Y}} f(y)$$

$$\text{Rewrite as} \quad \underset{\underbrace{z \in \mathcal{Z}}_{\textbf{All Possible}},\ \underbrace{y \in \mathcal{Y}}_{\textbf{Valid Trees}}}{\operatorname{argmax}} \quad \overbrace{f(z)}^{\textbf{Sibling}} + \overbrace{g(y)}^{\textbf{Arc-Factored}}$$

$$\text{such that } \underbrace{z = y}_{\textbf{Constraint}}$$

Algorithm Sketch

Set penalty weights equal to 0 for all edges.

For $k = 1$ **to** K

Algorithm Sketch

Set penalty weights equal to 0 for all edges.

For $k = 1$ **to** K

$z^{(k)} \leftarrow$ Decode $(f(z) + \mathrm{penalty})$ by Individual Decoding

Algorithm Sketch

Set penalty weights equal to 0 for all edges.

For $k = 1$ **to** K

$z^{(k)} \leftarrow$ Decode $(f(z) + \mathrm{penalty})$ by Individual Decoding

$y^{(k)} \leftarrow$ Decode $(g(y) - \mathrm{penalty})$ by Minimum Spanning Tree

Algorithm Sketch

Set penalty weights equal to 0 for all edges.

For $k = 1$ **to** K

$z^{(k)} \leftarrow$ Decode $(f(z) + \mathrm{penalty})$ by Individual Decoding

$y^{(k)} \leftarrow$ Decode $(g(y) - \mathrm{penalty})$ by Minimum Spanning Tree

If $y^{(k)}(i,j) = z^{(k)}(i,j)$ for all i,j **Return** $(y^{(k)}, z^{(k)})$

Algorithm Sketch

Set penalty weights equal to 0 for all edges.

For $k = 1$ **to** K

$z^{(k)} \leftarrow$ Decode $(f(z) + \mathrm{penalty})$ by Individual Decoding

$y^{(k)} \leftarrow$ Decode $(g(y) - \mathrm{penalty})$ by Minimum Spanning Tree

If $y^{(k)}(i,j) = z^{(k)}(i,j)$ for all i,j **Return** $(y^{(k)}, z^{(k)})$

Else Update penalty weights based on $y^{(k)}(i,j) - z^{(k)}(i,j)$

Individual Decoding

Penalties

$u(i,j) = 0$ for all i,j

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding

Penalties

$u(i,j) = 0$ for all i,j

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

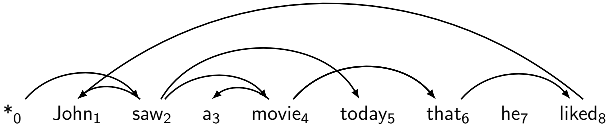
Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding

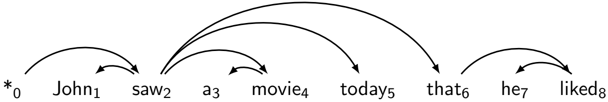
Penalties

$u(i,j) = 0$ for all i,j



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree



$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

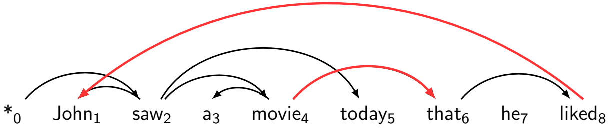
Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding

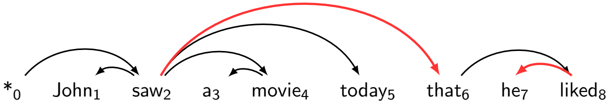
Penalties

$u(i,j) = 0$ for all i,j



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree

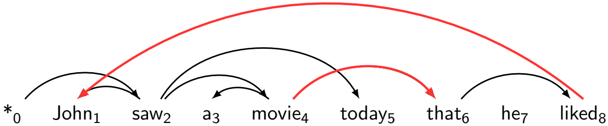


$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Key

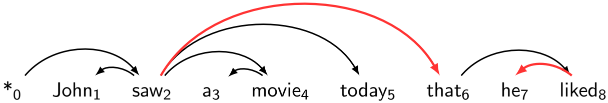
$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree



$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Penalties

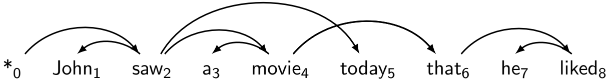
$u(i,j) = 0$ for all i,j

	Iteration 1
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Minimum Spanning Tree

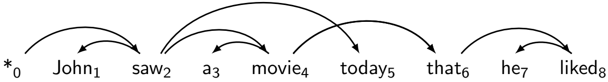
$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding



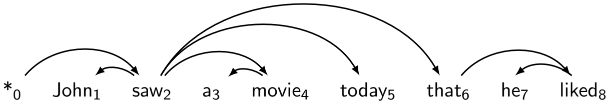
$$z^* = \arg\max_{z \in \mathcal{Z}} \left(f(z) + \sum_{i,j} u(i,j) z(i,j) \right)$$

Penalties

$u(i,j) = 0$ for all i,j

	Iteration 1
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Minimum Spanning Tree

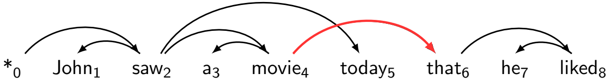


$$y^* = \arg\max_{y \in \mathcal{Y}} \left(g(y) - \sum_{i,j} u(i,j) y(i,j) \right)$$

Key

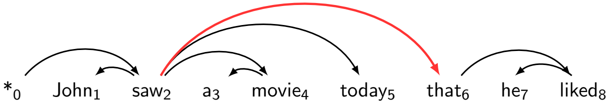
$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree



$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Iteration 2	
$u(8,1)$	-1
$u(4,6)$	-2
$u(2,6)$	2
$u(8,7)$	1

Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding

*$_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree

*$_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Iteration 2	
$u(8,1)$	-1
$u(4,6)$	-2
$u(2,6)$	2
$u(8,7)$	1

Key

$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree

$*_0$ John$_1$ saw$_2$ a$_3$ movie$_4$ today$_5$ that$_6$ he$_7$ liked$_8$

$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

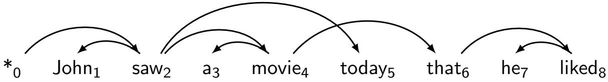
Iteration 1	
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Iteration 2	
$u(8,1)$	-1
$u(4,6)$	-2
$u(2,6)$	2
$u(8,7)$	1

Key

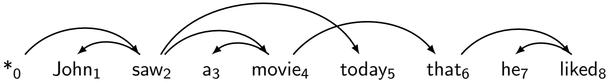
$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree



$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Penalties

$u(i,j) = 0$ for all i,j

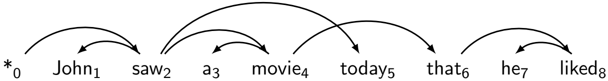
Iteration 1	
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Iteration 2	
$u(8,1)$	-1
$u(4,6)$	-2
$u(2,6)$	2
$u(8,7)$	1

Key

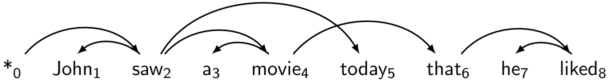
$f(z)$	$\Leftarrow$	Sibling Model	$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints	$\mathcal{Y}$	$\Leftarrow$	Tree Constraints
$y(i,j) = 1$	if	y contains dependency i,j			

Individual Decoding



$$z^* = \arg\max_{z \in \mathcal{Z}} (f(z) + \sum_{i,j} u(i,j) z(i,j))$$

Minimum Spanning Tree



$$y^* = \arg\max_{y \in \mathcal{Y}} (g(y) - \sum_{i,j} u(i,j) y(i,j))$$

Key

$f(z)$	$\Leftarrow$	Sibling Model
$\mathcal{Z}$	$\Leftarrow$	No Constraints
$y(i,j) = 1$	if	y contains dependency i,j

$g(y)$	$\Leftarrow$	Arc-Factored Model
$\mathcal{Y}$	$\Leftarrow$	Tree Constraints

Penalties

$u(i,j) = 0$ for all i,j

Iteration 1	
$u(8,1)$	-1
$u(4,6)$	-1
$u(2,6)$	1
$u(8,7)$	1

Iteration 2	
$u(8,1)$	-1
$u(4,6)$	-2
$u(2,6)$	2
$u(8,7)$	1

Converged

$$y^* = \arg\max_{y \in \mathcal{Y}} f(y) + g(y)$$

Guarantees

Theorem

If at any iteration $y^{(k)} = z^{(k)}$, then $(y^{(k)}, z^{(k)})$ is the global optimum.

In experiments, we find the global optimum on 98% of examples.

Guarantees

Theorem

If at any iteration $y^{(k)} = z^{(k)}$, then $(y^{(k)}, z^{(k)})$ is the global optimum.

In experiments, we find the global optimum on 98% of examples.

If we do not converge to a match, we can still return an approximate solution (more in the paper).

Extensions

- Grandparent Models

$*_0$ John_1 saw_2 a_3 movie_4 today_5 that_6 he_7 liked_8

$$f(y) = \ldots + \mathit{score}(\mathit{gp} = *_0, \mathit{head} = \text{saw}_2, \mathit{prev} = \text{movie}_4, \mathit{mod} = \text{today}_5)$$

- Head Automata (Eisner, 2000)

  Generalization of Sibling models

  Allow arbitrary automata as local scoring function.

Experiments

Properties:

- Exactness
- Parsing Speed
- Parsing Accuracy
- Comparison to Individual Decoding
- Comparison to LP/ILP

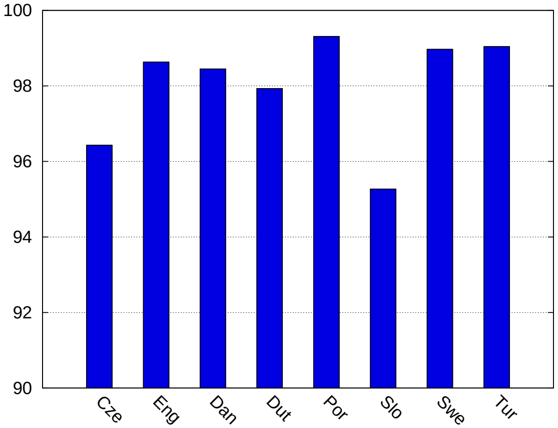
Training:

- Averaged Perceptron (more details in paper)

Experiments on:

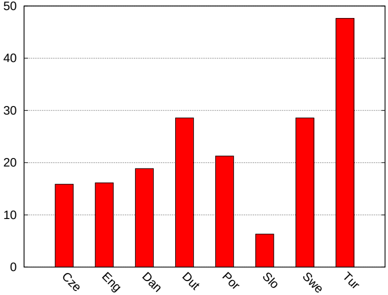
- CoNLL Datasets
- English Penn Treebank
- Czech Dependency Treebank

How often do we exactly solve the problem?

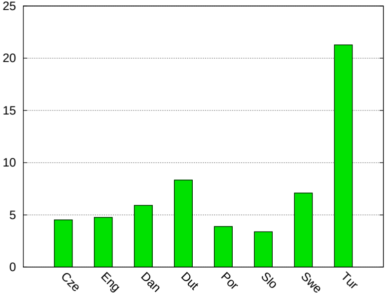


- Percentage of examples where the dual decomposition finds an exact solution.

Parsing Speed



Sibling model



Grandparent model

- Number of sentences parsed per second
- Comparable to dynamic programming for projective parsing

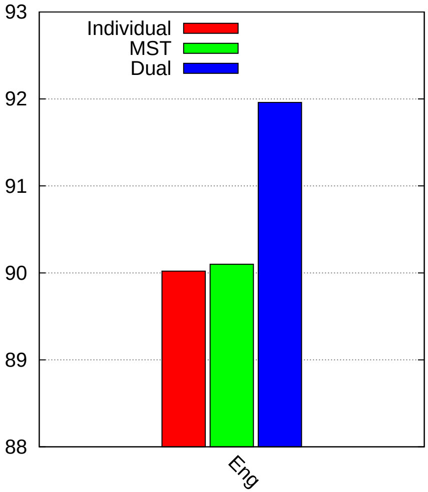
Accuracy

	Arc-Factored	Prev Best	Grandparent
Dan	89.7	91.5	**91.8**
Dut	82.3	85.6	**85.8**
Por	90.7	92.1	**93.0**
Slo	82.4	85.6	**86.2**
Swe	88.9	90.6	**91.4**
Tur	75.7	76.4	**77.6**
Eng	90.1	—	**92.5**
Cze	84.4	—	**87.3**

Prev Best - Best reported results for CoNLL-X data set, includes

- Approximate search (McDonald and Pereira, 2006)
- Loop belief propagation (Smith and Eisner, 2008)
- (Integer) Linear Programming (Martins et al., 2009)

Comparison to Subproblems



F_1 for dependency accuracy

Comparison to LP/ILP

Martins et al.(2009): Proposes two representations of non-projective dependency parsing as a linear programming relaxation as well as an exact ILP.

- LP (1)
- LP (2)
- ILP

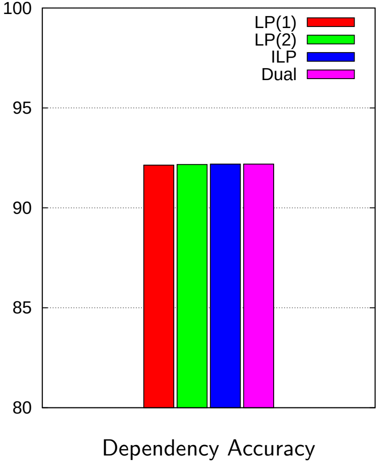
Use an LP/ILP Solver for decoding

We compare:

- Accuracy
- Exactness
- Speed

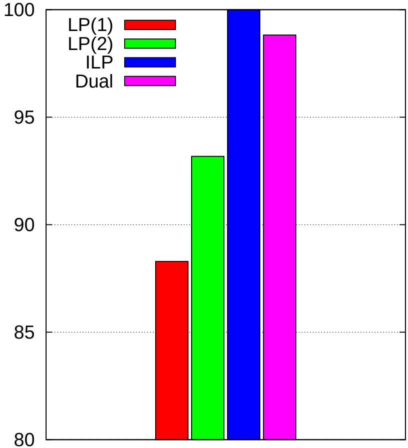
Both LP and dual decomposition methods use the same model, features, and weights w.

Comparison to LP/ILP: Accuracy

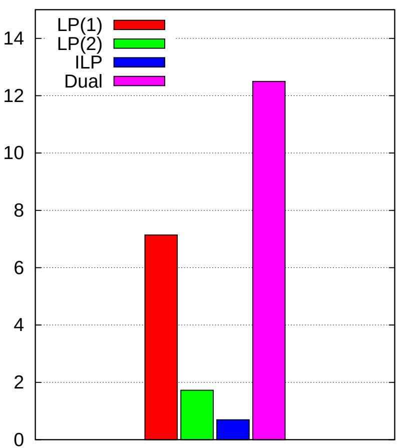


- All decoding methods have comparable accuracy

Comparison to LP/ILP: Exactness and Speed



Percentage with exact solution



Sentences per second

Summary

presented Lagrangian relaxation as a method for decoding in NLP

formal guarantees

- gives certificate or approximate solution
- can improve approximate solutions by tightening relaxation

efficient algorithms

- uses fast combinatorial algorithms
- can improve speed with lazy decoding

widely applicable

- demonstrated algorithms for a wide range of NLP tasks (parsing, tagging, alignment, mt decoding)

References I

Y. Chang and M. Collins. Exact Decoding of Phrase-based Translation Models through Lagrangian Relaxation. In *To appear proc. of EMNLP*, 2011.

J. DeNero and K. Macherey. Model-Based Aligner Combination Using Dual Decomposition. In *Proc. ACL*, 2011.

J. Duchi, D. Tarlow, G. Elidan, and D. Koller. Using Combinatorial Optimization within Max-Product Belief Propagation. In *NIPS*, pages 369–376, 2007.

D. Klein and C.D. Manning. Factored A* Search for Models over Sequences and Trees. In *Proc IJCAI*, volume 18, pages 1246–1251. Citeseer, 2003.

N. Komodakis, N. Paragios, and G. Tziritas. Mrf energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010. ISSN 0162-8828.

References II

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. Dual decomposition for parsing with non-projective head automata. In *EMNLP*, 2010. URL `http://www.aclweb.org/anthology/D10-1125`.

B.H. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Verlag, 2008.

A.M. Rush and M. Collins. Exact Decoding of Syntactic Translation Models through Lagrangian Relaxation. In *Proc. ACL*, 2011.

A.M. Rush, D. Sontag, M. Collins, and T. Jaakkola. On Dual Decomposition and Linear Programming Relaxations for Natural Language Processing. In *Proc. EMNLP*, 2010.

D.A. Smith and J. Eisner. Dependency Parsing by Belief Propagation. In *Proc. EMNLP*, pages 145–156, 2008. URL `http://www.aclweb.org/anthology/D08-1016`.