Decoding for SMT

Wilker Aziz

April 19, 2016

Table of Contents

Task

Translate a source text (e.g. sentence)
Examples:

um conto de duas cidades	$\rightarrow$	a tale of two cities
nosso amigo comum	$\rightarrow$	our mutual friend
a loja de antiguidades	$\rightarrow$	the old curiosity shop
o grill da lareira	$\rightarrow$	the cricket on the hearth

Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations [Lopez, 2008]

Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations [Lopez, 2008]

Example:

- a word replacement model

Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations [Lopez, 2008]

Example:

- a word replacement model
- operates in monotone left-to-right order

Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations [Lopez, 2008]

Example:

- a word replacement model
- operates in monotone left-to-right order
- with no insertions or deletions

Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations [Lopez, 2008]

Example:

- a word replacement model
- operates in monotone left-to-right order
- with no insertions or deletions
- constrained to known word-to-word bilingual mappings (rule set)

Monotone word-by-word translation: solutions

Source: *um conto de duas cidades*

Translation rules[1]

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

[1] Unrealistically simple

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades
a tale of two cities

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades

a tale of two cities

a tale of two **towns**

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades

a tale of two cities

a tale of two **towns**

a tale of two **villages**

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades

a tale of two cities

a tale of two **towns**

a tale of two **villages**

a tale of **couple cities**

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades

a tale of two cities

a tale of two **towns**

a tale of two **villages**

a tale of **couple cities**

a tale of couple **towns**

Monotone word-by-word translation: solutions

um	{a, some, one}
conto	{tale, story, narrative, novella}
de	{of, from, 's}
duas	{two, couple}
cidades	{cities, towns, villages}

um conto de duas cidades

a tale of two cities

a tale of two **towns**

a tale of two **villages**

a tale of **couple cities**

a tale of couple **towns**

...

This can go very far :(

Monotone word-by-word translation: complexity

Say

- the input has I words
- we know at most t translation options per source word

Monotone word-by-word translation: complexity

Say

- the input has I words
- we know at most t translation options per source word

This makes $O(t^I)$ solutions

Monotone word-by-word translation: complexity

Say

- the input has I words
- we know at most t translation options per source word

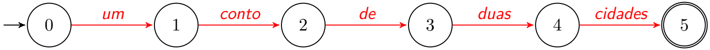
This makes $O(t^I)$ solutions

Note

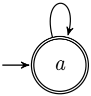
- WMT14's shared task: $I = 40$ on average
- last I checked Moses default was $t = 100$

  (for a more complex model)
- silly monotone word replacement model: 10^{80} solutions

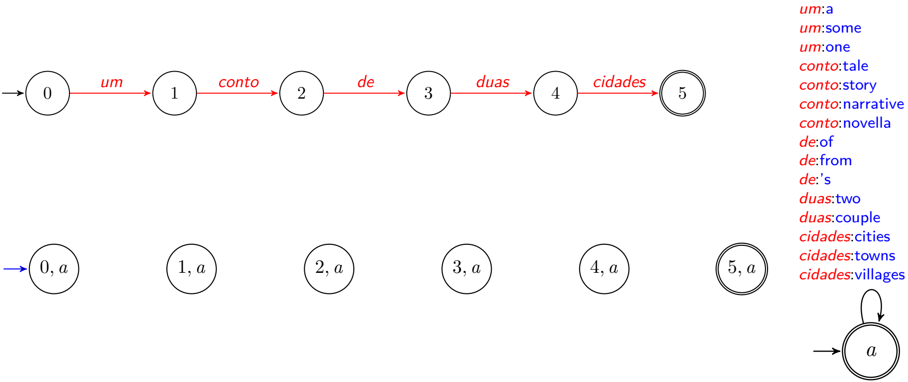
Space of solutions as intersection/composition



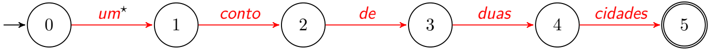
um:a
um:some
um:one
conto:tale
conto:story
conto:narrative
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages



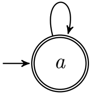
Space of solutions as intersection/composition



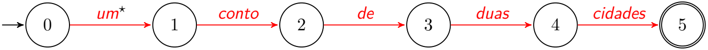
Space of solutions as intersection/composition



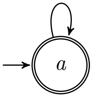
um:a ←
um:some
um:one
conto:tale
conto:story
conto:narrative
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages



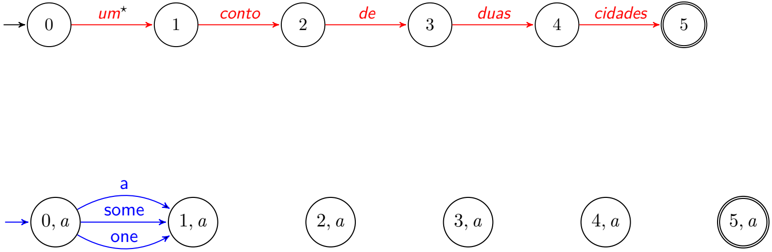
Space of solutions as intersection/composition



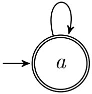
um:a ✓
um:some ←
um:one
conto:tale
conto:story
conto:narrative
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages



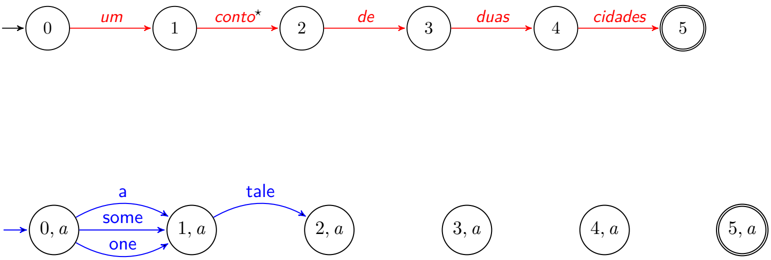
Space of solutions as intersection/composition



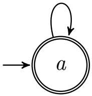
um:a ✓
um:some ✓
um:one ←
conto:tale
conto:story
conto:narrative
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages



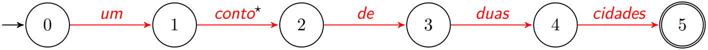
Space of solutions as intersection/composition

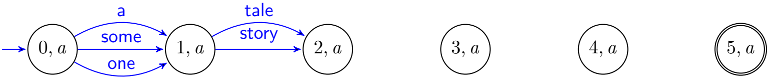


um:a ✓
um:some ✓
um:one ✓
conto:tale ←
conto:story
conto:narrative
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages

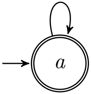


Space of solutions as intersection/composition

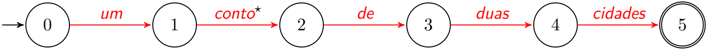




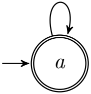
um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ←
conto:narrative
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages



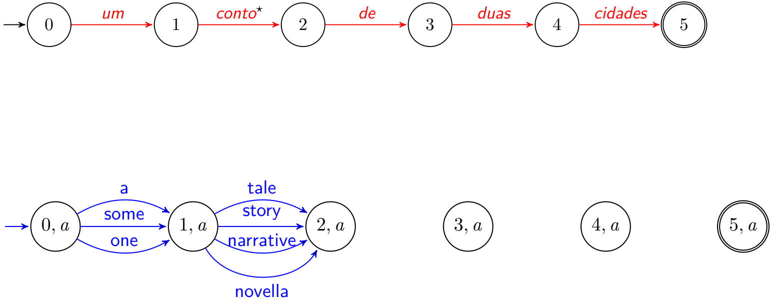
Space of solutions as intersection/composition



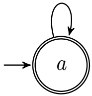
um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ←
conto:novella
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages



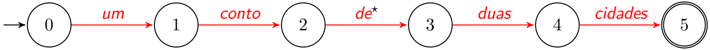
Space of solutions as intersection/composition

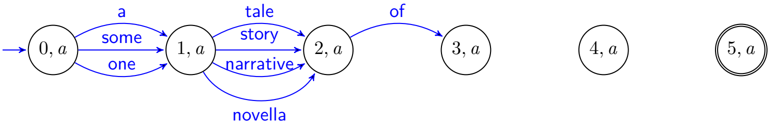


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella←
de:of
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages

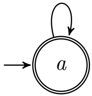


Space of solutions as intersection/composition

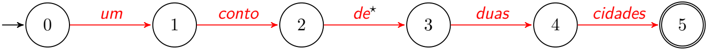


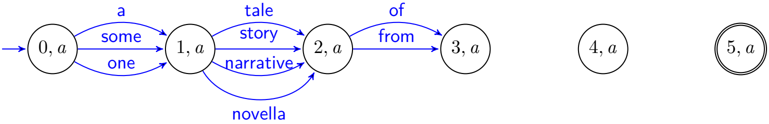


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ←
de:from
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages

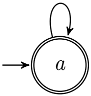


Space of solutions as intersection/composition

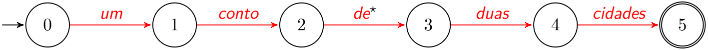


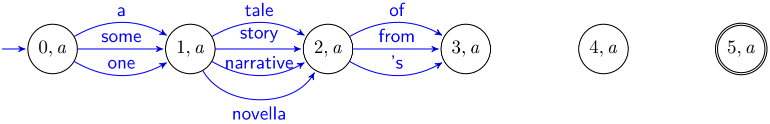


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ←
de:'s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages

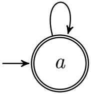


Space of solutions as intersection/composition

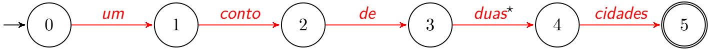


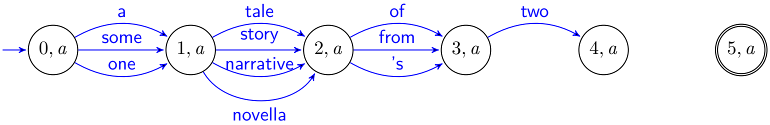


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ✓
de:'s ←
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages

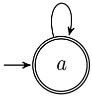


Space of solutions as intersection/composition

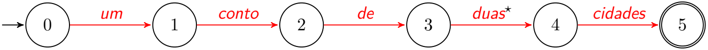


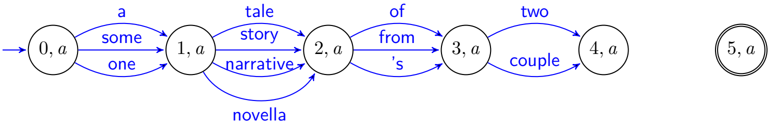


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ←
duas:couple
cidades:cities
cidades:towns
cidades:villages

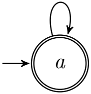


Space of solutions as intersection/composition

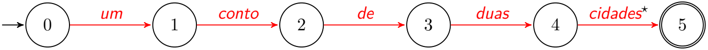


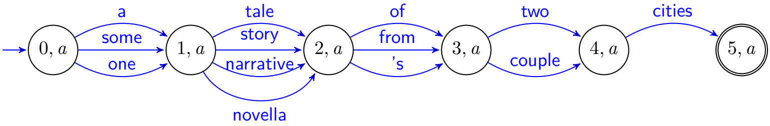


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ✓
duas:couple ←
cidades:cities
cidades:towns
cidades:villages

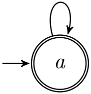


Space of solutions as intersection/composition

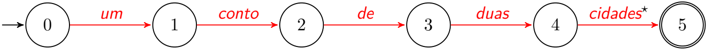


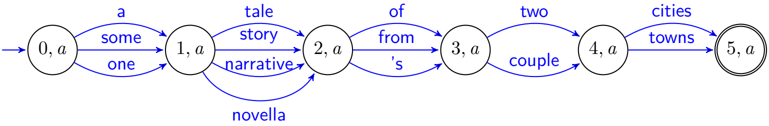


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ✓
duas:couple ✓
cidades:cities ←
cidades:towns
cidades:villages

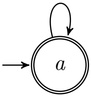


Space of solutions as intersection/composition

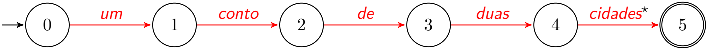


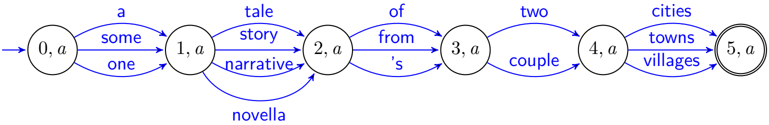


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ✓
duas:couple ✓
cidades:cities ✓
cidades:towns ←
cidades:villages

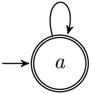


Space of solutions as intersection/composition

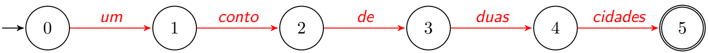


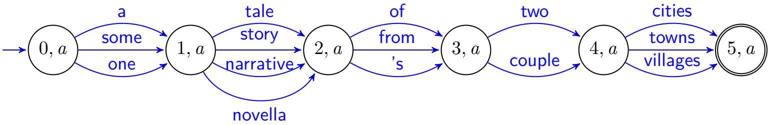


um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella ✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ✓
duas:couple ✓
cidades:cities ✓
cidades:towns ✓
cidades:villages ←

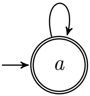


Space of solutions as intersection/composition





um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ✓
duas:couple ✓
cidades:cities ✓
cidades:towns ✓
cidades:villages ✓



$3 \times 4 \times 3 \times 2 \times 3 = 216$ solutions

- 6 states
- $3 + 4 + 3 + 2 + 3 = 15$ transitions

Packing solutions with finite-state automata

Same $O(t^I)$ solutions using

- $O(I)$ states
- $O(tI)$ transitions

Recap 1

Recap 1

Model of translational equivalences

- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Recap 1

Model of translational equivalences

- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model

- easy to represent using finite-state transducers

Recap 1

Model of translational equivalences

- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model

- easy to represent using finite-state transducers
- set of translations given by composition

Recap 1

Model of translational equivalences

- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model

- easy to represent using finite-state transducers
- set of translations given by composition
- exponential number of solutions in linear space

Recap 1

Model of translational equivalences

- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model

- easy to represent using finite-state transducers
- set of translations given by composition
- exponential number of solutions in linear space
- translates infinitely many sentences

Recap 1

Model of translational equivalences

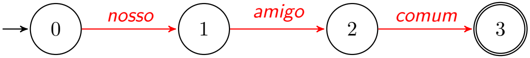
- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model

- easy to represent using finite-state transducers
- set of translations given by composition
- exponential number of solutions in linear space
- translates infinitely many sentences
  but not nearly enough interesting cases!

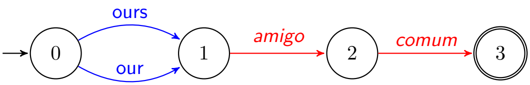
Monotone word-by-word translation: fail!

nosso {our, ours}
amigo {friend, mate}
comum {ordinary, common, usual, mutual}



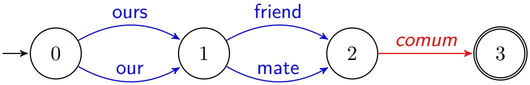
Monotone word-by-word translation: fail!

nosso	{our, ours}
amigo	{friend, mate}
comum	{ordinary, common, usual, mutual}



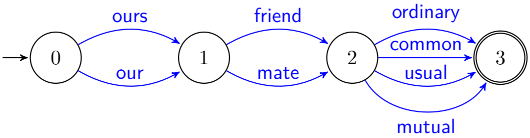
Monotone word-by-word translation: fail!

nosso	{our, ours}
amigo	{friend, mate}
comum	{ordinary, common, usual, mutual}



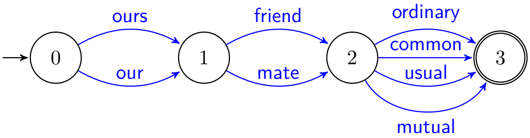
Monotone word-by-word translation: fail!

nosso	{our, ours}
amigo	{friend, mate}
comum	{ordinary, common, usual, mutual}



Monotone word-by-word translation: fail!

nosso	{our, ours}
amigo	{friend, mate}
comum	{ordinary, common, usual, mutual}



We simply cannot obtain a correct translation

our mutual friend

Reordering

Our model of translational equivalences assumes monotonicity

- a word replacement model
- operates in **monotone** left-to-right order
- with no insertions or deletions
- constrained to known word-to-word bilingual mappings (rule set)

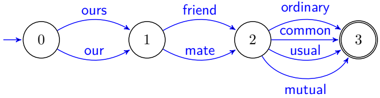
Reordering

Not anymore!

- a word replacement model
- operates in **arbitrary** order
- with no insertions or deletions
- constrained to known word-to-word bilingual mappings (rule set)

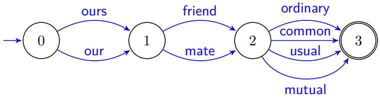
Translating arbitrary permutations

nosso amigo comum

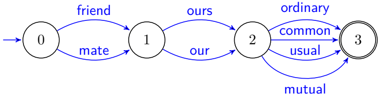


Translating arbitrary permutations

nosso amigo comum

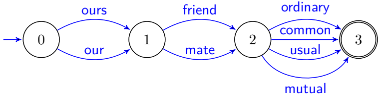


amigo nosso comum

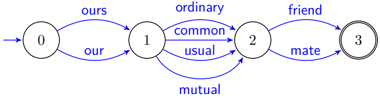


Translating arbitrary permutations

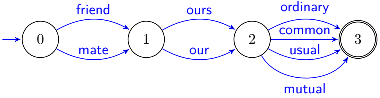
nosso amigo comum



nosso comum amigo

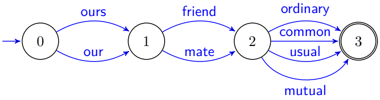


amigo nosso comum

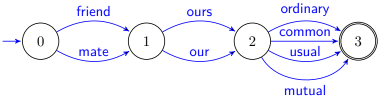


Translating arbitrary permutations

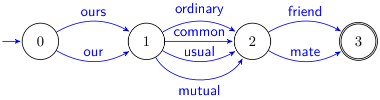
nosso amigo comum



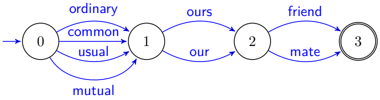
amigo nosso comum



nosso comum amigo

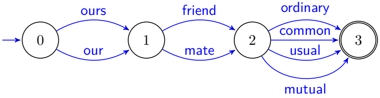


comum nosso amigo

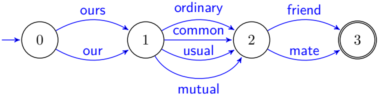


Translating arbitrary permutations

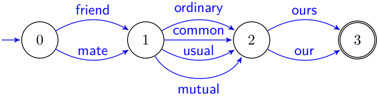
nosso amigo comum



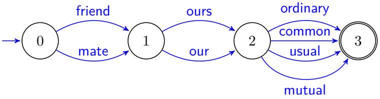
nosso comum amigo



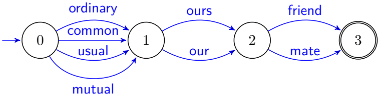
amigo comum nosso



amigo nosso comum

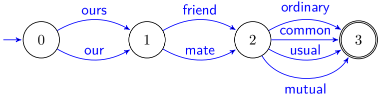


comum nosso amigo

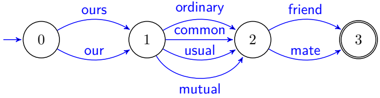


Translating arbitrary permutations

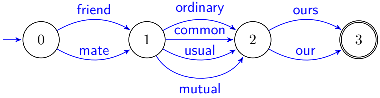
nosso amigo comum



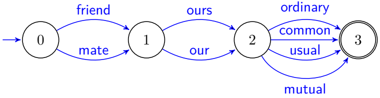
nosso comum amigo



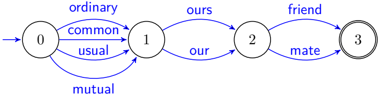
amigo comum nosso



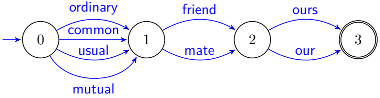
amigo nosso comum



comum nosso amigo

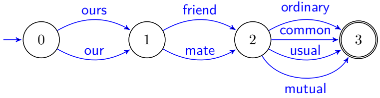


comum amigo nosso

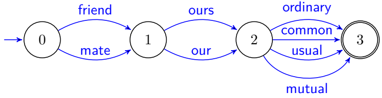


Translating arbitrary permutations

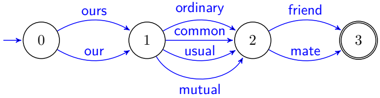
nosso amigo comum



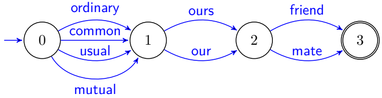
amigo nosso comum



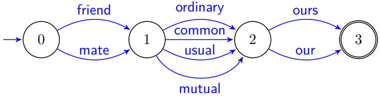
nosso comum amigo



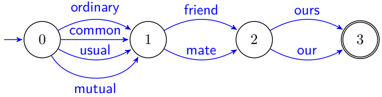
comum nosso amigo



amigo comum nosso



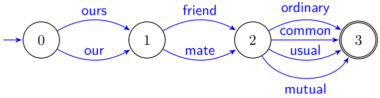
comum amigo nosso



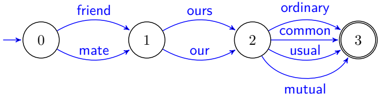
$3! = 3 \times 2 \times 1 = 6$ permutations

Translating arbitrary permutations

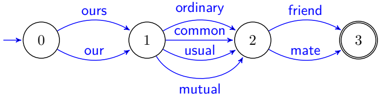
nosso amigo comum



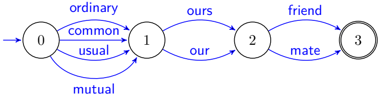
amigo nosso comum



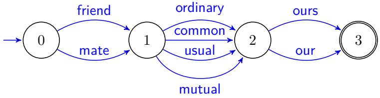
nosso comum amigo



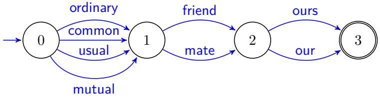
comum nosso amigo



amigo comum nosso



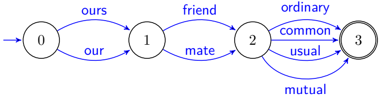
comum amigo nosso



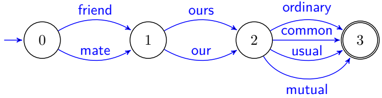
each has $2 \times 2 \times 4 = 16$ translations

Translating arbitrary permutations

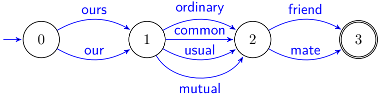
nosso amigo comum



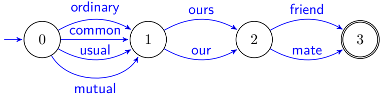
amigo nosso comum



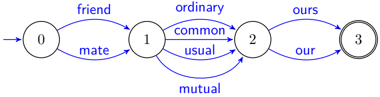
nosso comum amigo



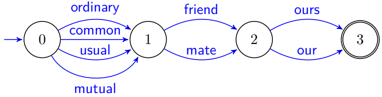
comum nosso amigo



amigo comum nosso



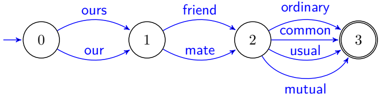
comum amigo nosso



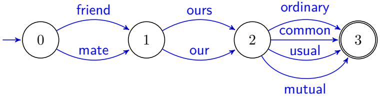
amounting to $6 \times 16 = 96$ solutions

Translating arbitrary permutations

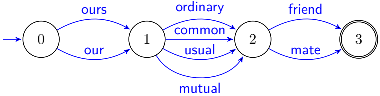
nosso amigo comum



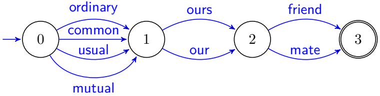
amigo nosso comum



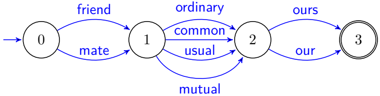
nosso comum amigo



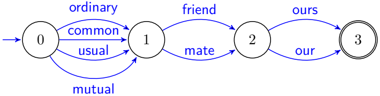
comum nosso amigo



amigo comum nosso



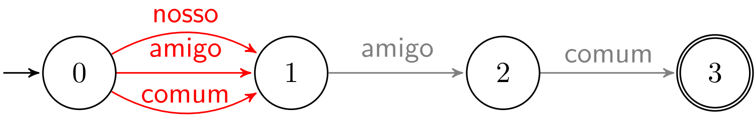
comum amigo nosso



$I!$ permutations $\times$ t^I translations

Packing permutations

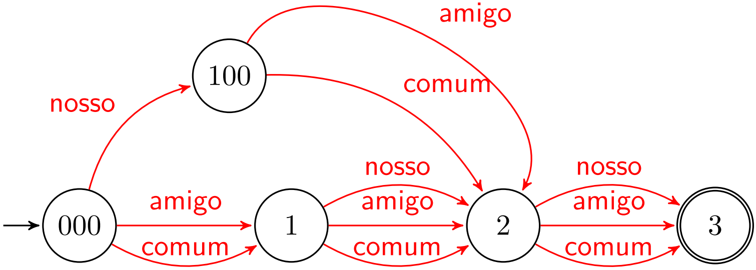
Packing permutations



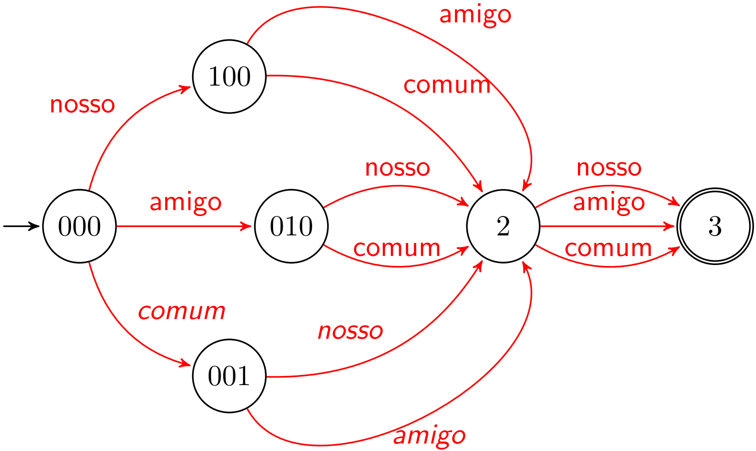
Packing permutations

Packing permutations

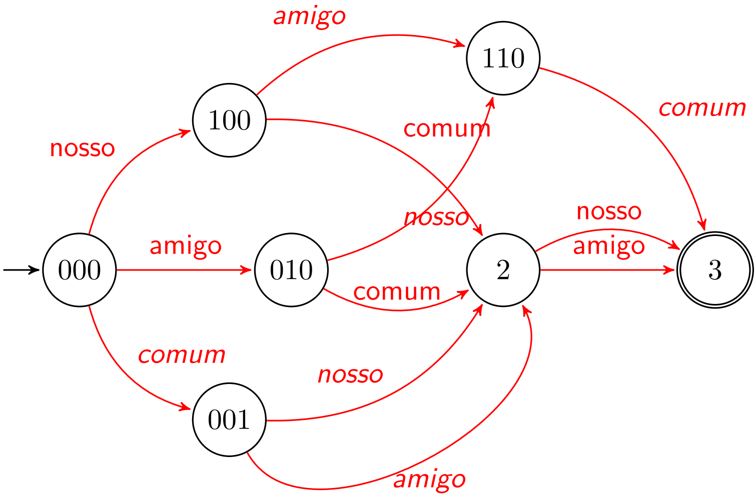
Packing permutations



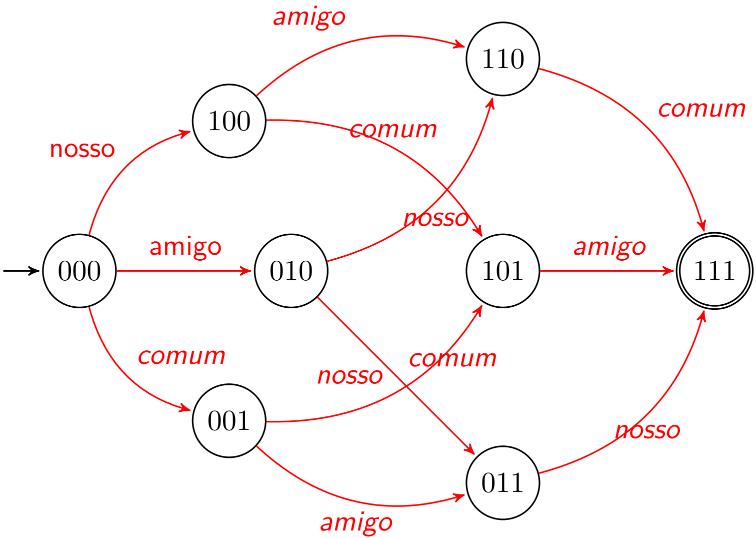
Packing permutations



Packing permutations



Packing permutations



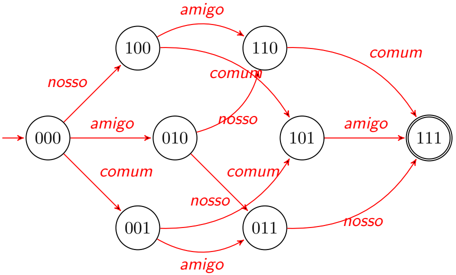
Packing permutations

Powerset (*all subsets*) of $\{1, 2, \ldots, I\}$

- 2^I subsets
  think of a vector of I bits ;)

Lattice

- $O(2^I)$ states
- $O(I \times 2^I)$ transitions



Deductive logic

$$\textsc{Item} \quad \left[\{0,1\}^I\right]$$

$$\textsc{Goal} \quad \left[1^I\right]$$

$$\textsc{Axiom}$$

$$\frac{}{[0^I]}$$

$$\textsc{Expand}$$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$

where $\alpha_i(C)$ is a copy of C with $c_i = \bar{1}$

Template

- items $\rightarrow$ states
- deduction rules $\rightarrow$ transitions

Deductive logic

$\textsc{Item}$ $\left[\{0,1\}^I\right]$

$\textsc{Goal}$ $\left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{matrix} 1 \le i \le I \\ c_i = \bar{0} \end{matrix}$$

where $\alpha_i(C)$ is a copy of C with $c_i = \bar{1}$

- a subset of $\{1, \ldots, I\}$ encoded as a bit vector of length I

Deductive logic

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\overline{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$

where $\alpha_i(C)$ is a copy of C with $c_i = \bar{1}$

- we start with an empty sentence e.g. $I = 3 \rightarrow 0^3 = 000$

Deductive logic

ITEM $\left[\{0,1\}^I\right]$

GOAL $\left[1^I\right]$

AXIOM

$$\overline{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$

where $\alpha_i(C)$ is a copy of C with $c_i = \bar{1}$

- and continue one word at a time
  e.g. $[000](i = 1) \rightarrow [100]$

Deductive logic

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{matrix} 1 \leq i \leq I \\ c_i = \bar{0} \end{matrix}$$

where $\alpha_i(C)$ is a copy of C with $c_i = \bar{1}$

- until we have a complete sentence e.g. $[111]$

Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

ITEM $\left[\{0,1\}^I\right]$

GOAL $\left[1^I\right]$

AXIOM

$$\frac{}{[0^I]}$$

EXPAND

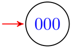
$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$

Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$



$$\textsc{Item} \quad \left[\{0,1\}^I\right]$$

$$\textsc{Goal} \quad \left[1^I\right]$$

$$\textsc{Axiom}$$

$$\overline{[0^I]}$$

$$\textsc{Expand}$$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$

Instantiated deductive logic program

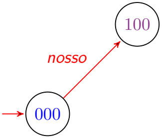
Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i = 1) \rightarrow [100]$



$\textsc{Item}\quad \left[\{0,1\}^I\right]$

$\textsc{Goal}\quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$

Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

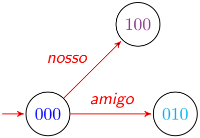
`Axiom`

$[000]$

`Expand`

$[000](i = 1) \rightarrow [100]$

$[000](i = 2) \rightarrow [010]$



ITEM $\left[\{0,1\}^I\right]$

GOAL $\left[1^I\right]$

AXIOM

$$\frac{}{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$

Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

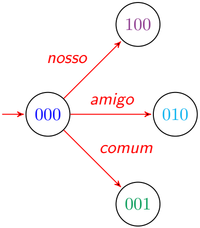
$[000]$

`Expand`

$[000](i = 1) \rightarrow [100]$

$[000](i = 2) \rightarrow [010]$

$[000](i = 3) \rightarrow [001]$



$\textsc{Item} \quad \left[\{0,1\}^I\right]$

$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$

Instantiated deductive logic program

ITEM $\left[\{0,1\}^I\right]$

GOAL $\left[1^I\right]$

AXIOM

$$\frac{}{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

Axiom

$[000]$

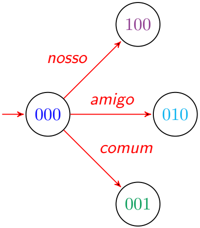
Expand

$[000](i = 1) \rightarrow [100]$

$[000](i = 2) \rightarrow [010]$

$[000](i = 3) \rightarrow [001]$

$[100](i = 1)$ ✗



Instantiated deductive logic program

$$\textsc{Item}\quad \left[\{0,1\}^I\right]$$
$$\textsc{Goal}\quad \left[1^I\right]$$
$$\textsc{Axiom}\quad \frac{}{[0^I]}$$
$$\textsc{Expand}\quad \frac{[C]}{[\alpha_i(C)]}\quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

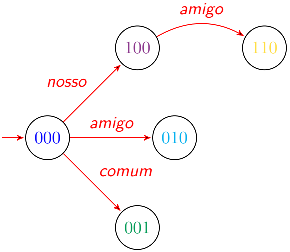
$[000](i=1) \rightarrow [100]$

$[000](i=2) \rightarrow [010]$

$[000](i=3) \rightarrow [001]$

$[100](i=1)$ ✗

$[100](i=2) \rightarrow [110]$



Instantiated deductive logic program

ITEM $\left[\{0,1\}^I\right]$

GOAL $\left[1^I\right]$

AXIOM

$$\overline{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \to [100]$

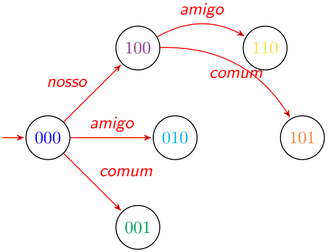
$[000](i=2) \to [010]$

$[000](i=3) \to [001]$

$[100](i=1)$ ✗

$[100](i=2) \to [110]$

$[100](i=3) \to [101]$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$

$[000](i=2) \rightarrow [010]$

$[000](i=3) \rightarrow [001]$

$[100](i=1)$ ✗

$[100](i=2) \rightarrow [110]$

$[100](i=3) \rightarrow [101]$

$[010](i=1) \rightarrow [110]$

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

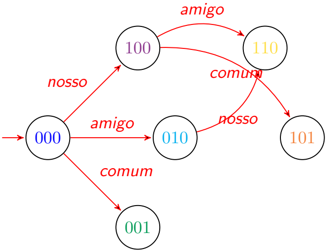
$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i = 1) \rightarrow [100]$
$[000](i = 2) \rightarrow [010]$
$[000](i = 3) \rightarrow [001]$
$[100](i = 1)$ ✗
$[100](i = 2) \rightarrow [110]$
$[100](i = 3) \rightarrow [101]$
$[010](i = 1) \rightarrow [110]$
$[010](i = 2)$ ✗

ITEM $\left[\{0,1\}^I\right]$

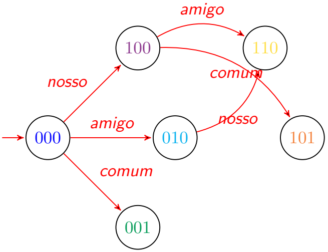
GOAL $\left[1^I\right]$

AXIOM

$$\overline{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$

$[000](i=2) \rightarrow [010]$

$[000](i=3) \rightarrow [001]$

$[100](i=1)$ ✗

$[100](i=2) \rightarrow [110]$

$[100](i=3) \rightarrow [101]$

$[010](i=1) \rightarrow [110]$

$[010](i=2)$ ✗

$[010](i=3) \rightarrow [011]$

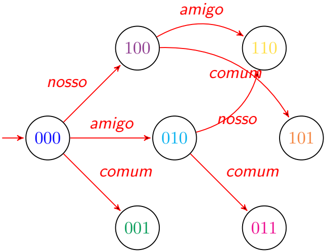
ITEM	$\left[\{0,1\}^I\right]$
GOAL	$\left[1^I\right]$

AXIOM

$$\frac{}{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$

$[000](i=2) \rightarrow [010]$

$[000](i=3) \rightarrow [001]$

$[100](i=1)$ ✗

$[100](i=2) \rightarrow [110]$

$[100](i=3) \rightarrow [101]$

$[010](i=1) \rightarrow [110]$

$[010](i=2)$ ✗

$[010](i=3) \rightarrow [011]$

$[001](i=1) \rightarrow [101]$

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

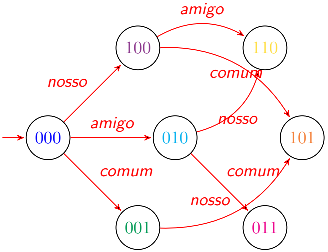
$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$

$[000](i=2) \rightarrow [010]$

$[000](i=3) \rightarrow [001]$

$[100](i=1)$ ✗

$[100](i=2) \rightarrow [110]$

$[100](i=3) \rightarrow [101]$

$[010](i=1) \rightarrow [110]$

$[010](i=2)$ ✗

$[010](i=3) \rightarrow [011]$

$[001](i=1) \rightarrow [101]$

$[001](i=2) \rightarrow [011]$

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

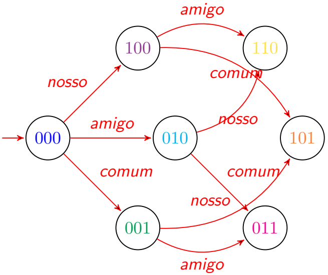
$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$

$[000](i=2) \rightarrow [010]$

$[000](i=3) \rightarrow [001]$

$[100](i=1)$ ✗

$[100](i=2) \rightarrow [110]$

$[100](i=3) \rightarrow [101]$

$[010](i=1) \rightarrow [110]$

$[010](i=2)$ ✗

$[010](i=3) \rightarrow [011]$

$[001](i=1) \rightarrow [101]$

$[001](i=2) \rightarrow [011]$

$[001](i=3)$ ✗

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

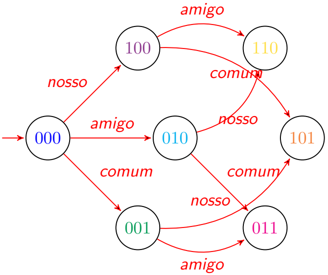
$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \to [100]$
$[000](i=2) \to [010]$
$[000](i=3) \to [001]$
$[100](i=2) \to [110]$
$[100](i=3) \to [101]$
$[010](i=1) \to [110]$
$[010](i=3) \to [011]$
$[001](i=1) \to [101]$
$[001](i=2) \to [011]$

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

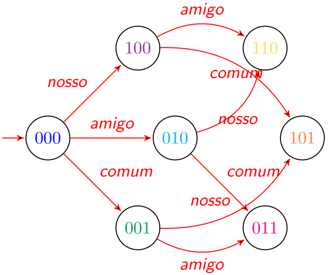
$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$
$[000](i=2) \rightarrow [010]$
$[000](i=3) \rightarrow [001]$
$[100](i=2) \rightarrow [110]$
$[100](i=3) \rightarrow [101]$
$[010](i=1) \rightarrow [110]$
$[010](i=3) \rightarrow [011]$
$[001](i=1) \rightarrow [101]$
$[001](i=2) \rightarrow [011]$
$[110](i=3) \rightarrow [111]$

$\textsc{Item} \quad \left[\{0,1\}^I\right]$

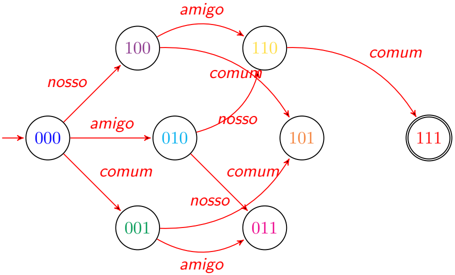
$\textsc{Goal} \quad \left[1^I\right]$

$\textsc{Axiom}$

$$\frac{}{[0^I]}$$

$\textsc{Expand}$

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$
$[000](i=2) \rightarrow [010]$
$[000](i=3) \rightarrow [001]$
$[100](i=2) \rightarrow [110]$
$[100](i=3) \rightarrow [101]$
$[010](i=1) \rightarrow [110]$
$[010](i=3) \rightarrow [011]$
$[001](i=1) \rightarrow [101]$
$[001](i=2) \rightarrow [011]$
$[110](i=3) \rightarrow [111]$
$[101](i=2) \rightarrow [111]$

ITEM $\left[\{0,1\}^I\right]$

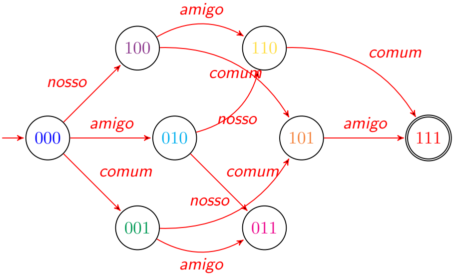
GOAL $\left[1^I\right]$

AXIOM

$$\overline{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \to [100]$
$[000](i=2) \to [010]$
$[000](i=3) \to [001]$
$[100](i=2) \to [110]$
$[100](i=3) \to [101]$
$[010](i=1) \to [110]$
$[010](i=3) \to [011]$
$[001](i=1) \to [101]$
$[001](i=2) \to [011]$
$[110](i=3) \to [111]$
$[101](i=2) \to [111]$
$[011](i=1) \to [111]$

ITEM $\left[\{0,1\}^I\right]$

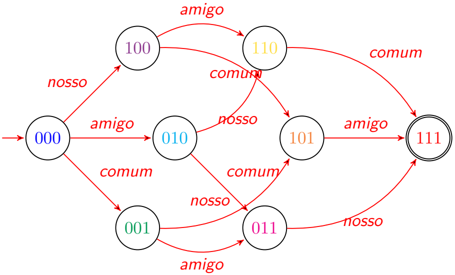
GOAL $\left[1^I\right]$

AXIOM

$$\overline{[0^I]}$$

EXPAND

$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \le i \le I \\ c_i = \bar{0} \end{array}$$



Instantiated deductive logic program

Source: *nosso*$_1$ *amigo*$_2$ *comum*$_3$

`Axiom`

$[000]$

`Expand`

$[000](i=1) \rightarrow [100]$
$[000](i=2) \rightarrow [010]$
$[000](i=3) \rightarrow [001]$
$[100](i=2) \rightarrow [110]$
$[100](i=3) \rightarrow [101]$
$[010](i=1) \rightarrow [110]$
$[010](i=3) \rightarrow [011]$
$[001](i=1) \rightarrow [101]$
$[001](i=2) \rightarrow [011]$
$[110](i=3) \rightarrow [111]$
$[101](i=2) \rightarrow [111]$
$[011](i=1) \rightarrow [111]$

`Goal`

$[111]$

ITEM $\left[\{0,1\}^I\right]$

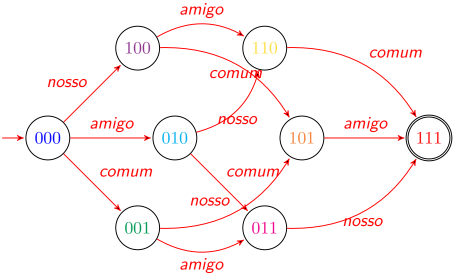
GOAL $\left[1^I\right]$

AXIOM

$$\frac{}{[0^I]}$$

EXPAND

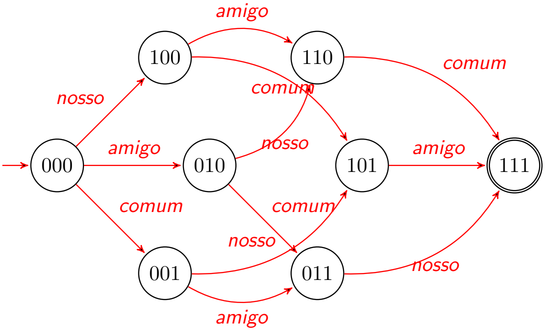
$$\frac{[C]}{[\alpha_i(C)]} \quad \begin{array}{l} 1 \leq i \leq I \\ c_i = \bar{0} \end{array}$$



Word replacement with unconstrained reordering

Source: *nosso amigo comum*

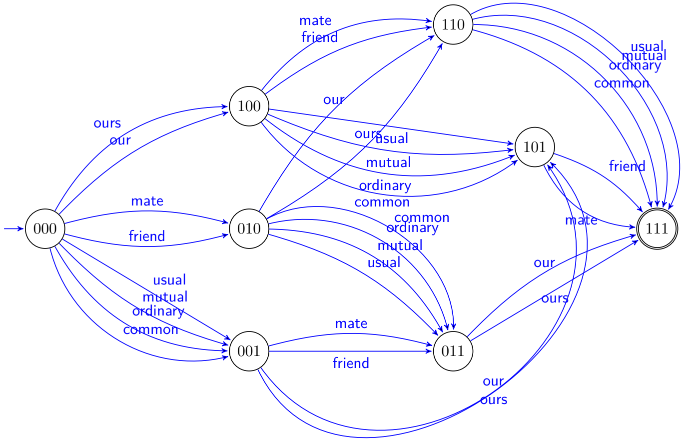
Word replacement with unconstrained reordering



Source: *nosso amigo comum*

1. arbitrary permutations: $O(2^I)$ states

Word replacement with unconstrained reordering



Source: *nosso amigo comum*

1. arbitrary permutations: $O(2^I)$ states
2. intersection with the rule set: $O(tI2^I)$ transitions

Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of **all permutations**?

Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of **all permutations**?

Solution

Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of **all permutations**?

Solution

- constrain reordering **:D**

Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of **all permutations**?

Solution

- constrain reordering **:D**
- **0.o** but how?

Ad-hoc distortion limit

Several flavours of distortion limit [Lopez, 2009]

Ad-hoc distortion limit

Several flavours of distortion limit [Lopez, 2009]

- limit reordering as a function of the number of skipped words

Ad-hoc distortion limit

Several flavours of distortion limit [Lopez, 2009]

- limit reordering as a function of the number of skipped words

Moses allows reordering within a window of length d

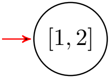
- starting from the leftmost uncovered word

WLd (example)

Suppose $d = 2$ and $I = 3$

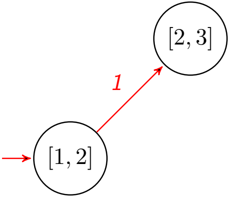
WLd (example)

Suppose $d = 2$ and $I = 3$



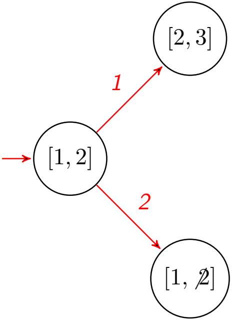
WLd (example)

Suppose $d = 2$ and $I = 3$



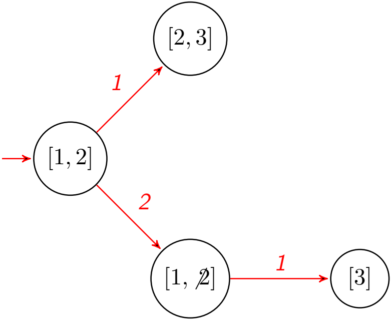
WLd (example)

Suppose $d = 2$ and $I = 3$



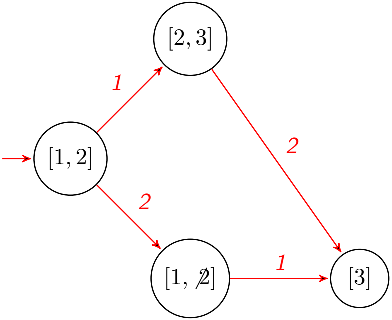
WLd (example)

Suppose $d = 2$ and $I = 3$



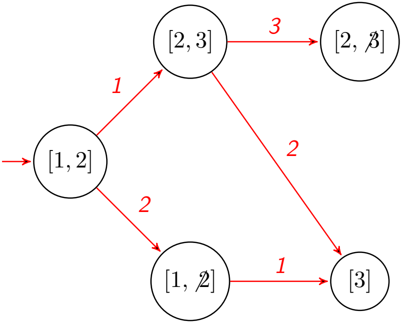
WLd (example)

Suppose $d = 2$ and $I = 3$



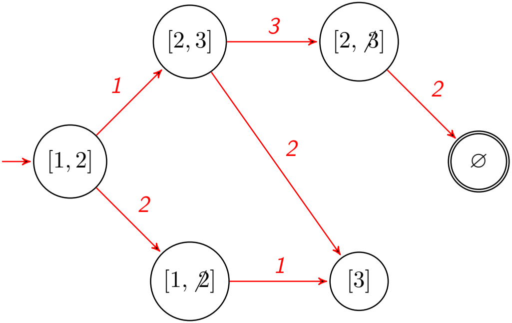
WLd (example)

Suppose $d = 2$ and $I = 3$



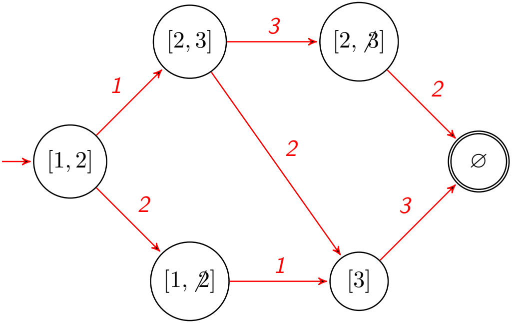
WLd (example)

Suppose $d = 2$ and $I = 3$



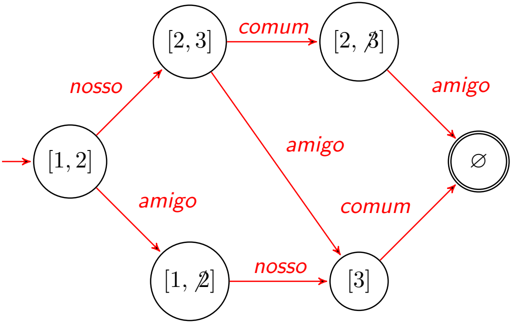
WLd (example)

Suppose $d = 2$ and $I = 3$



WLd (example)

Suppose $d = 2$ and $I = 3$ (e.g. *nosso amigo comum*)



WLd (logic)

$$\textsc{Item} \quad \left[[1..I+1], \{0,1\}^{d-1}\right]$$

$$\textsc{Goal} \quad [I+1, C]$$

$$\textsc{Axiom}$$

$$\frac{}{[1, 0^{d-1}]}$$

$$\textsc{Adjacent}$$

$$\frac{[l, C]}{[l+n, C \ll n]} \quad i = l$$

where $n = \#_1(C) + 1$

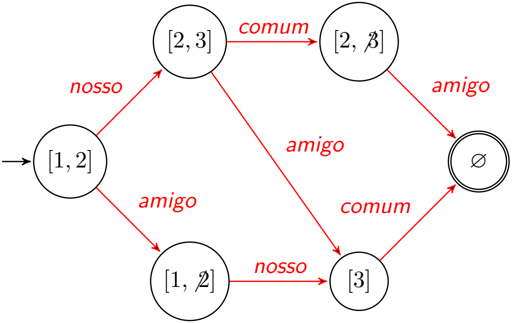
$$\textsc{Non-Adjacent}$$

$$\frac{[l, C]}{[l, \alpha_l^i(C)]} \quad \begin{array}{l} l < i \leq I \\ \delta(i, l) \leq d \\ c_{i-l} = \bar{0} \end{array}$$

- $O(Id2^{d-1})$ states
- $O(Id2^{d-1})$ transitions

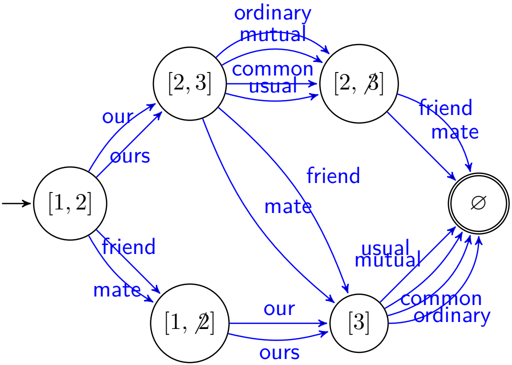
Word replacement with reordering constrained by WL2

Complexity: $O(I2^{d-1})$ states



Word replacement with reordering constrained by WL2

Complexity: $O(tI2^{d-1})$ transitions



Ad-hoc distortion limit: expressiveness

Limit reordering to a fixed-length window

Ad-hoc distortion limit: expressiveness

Limit reordering to a fixed-length window

- convenient (linear complexity), but

Ad-hoc distortion limit: expressiveness

Limit reordering to a fixed-length window

- convenient (linear complexity), but
- what about languages with very different syntax?
  e.g. OV vs VO, head-initial vs head-final

Ad-hoc distortion limit: expressiveness

Limit reordering to a fixed-length window

- convenient (linear complexity), but
- what about languages with very different syntax?
  e.g. OV vs VO, head-initial vs head-final
- can we do better?

ITGs

Inversion Transduction Grammars (ITGs) [Wu, 1997]

ITGs

Inversion Transduction Grammars (ITGs) [Wu, 1997]

- $X \rightarrow XX$
  direct order

ITGs

Inversion Transduction Grammars (ITGs) [Wu, 1997]

- $X \to XX$
  direct order
- $X \to \langle XX \rangle$
  inverted order

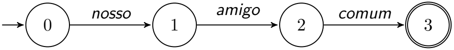
ITGs

Inversion Transduction Grammars (ITGs) [Wu, 1997]

- $X \rightarrow XX$
  direct order
- $X \rightarrow \langle XX \rangle$
  inverted order
- $X \rightarrow f/e$, where $(f, e) \in R$
  bilingual mappings

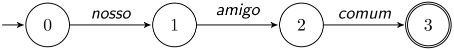
Parsing, intersection and hypergraphs

Source



Parsing, intersection and hypergraphs

Source



Grammar

$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$

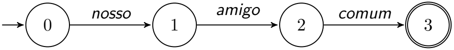
$X \rightarrow$ *nosso*

$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*

Parsing, intersection and hypergraphs

Source



Grammar

$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$

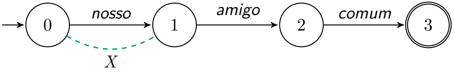
$X \rightarrow$ *nosso*

$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*

Parsing, intersection and hypergraphs

Source



Grammar

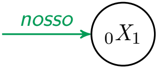
$X \to XX$

$X \to \langle XX \rangle$

$X \to$ *nosso*

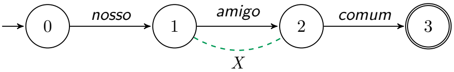
$X \to$ *amigo*

$X \to$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

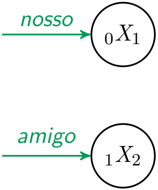
$X \to XX$

$X \to \langle XX \rangle$

$X \to$ *nosso*

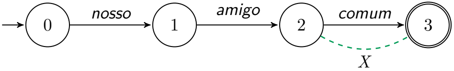
$X \to$ *amigo* $\Longleftarrow$

$X \to$ *comum*



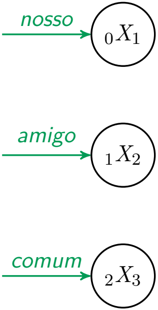
Parsing, intersection and hypergraphs

Source



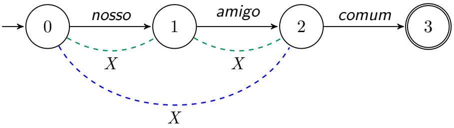
Grammar

$X \rightarrow XX$
$X \rightarrow \langle XX \rangle$
$X \rightarrow$ *nosso*
$X \rightarrow$ *amigo*
$X \rightarrow$ *comum* $\Longleftarrow$



Parsing, intersection and hypergraphs

Source



Grammar

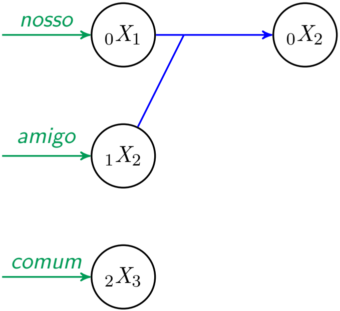
$X \rightarrow XX$ $\Longleftarrow$

$X \rightarrow \langle XX \rangle$

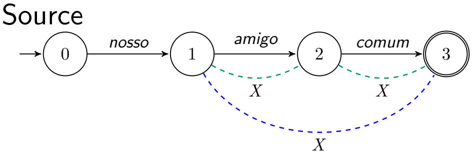
$X \rightarrow$ *nosso*

$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs



Grammar

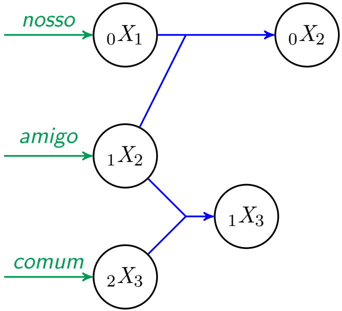
$X \to XX$ $\Longleftarrow$

$X \to \langle XX \rangle$

$X \to$ *nosso*

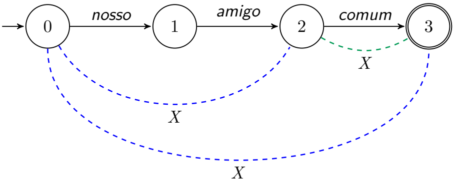
$X \to$ *amigo*

$X \to$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

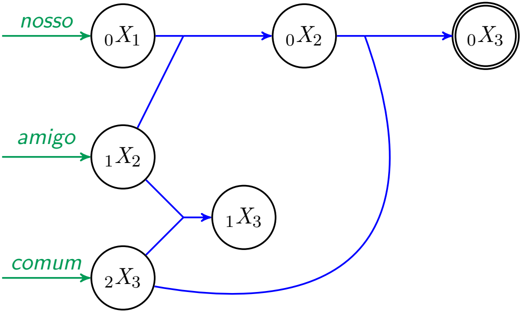
$X \to XX$

$X \to \langle XX \rangle$

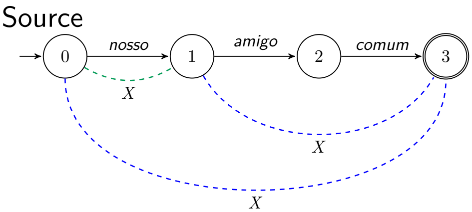
$X \to$ *nosso*

$X \to$ *amigo*

$X \to$ *comum*



Parsing, intersection and hypergraphs



Grammar

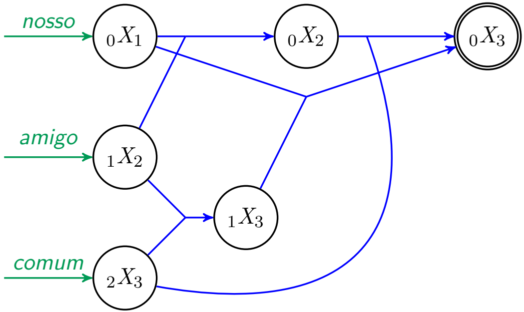
$X \rightarrow XX$ $\Longleftarrow$

$X \rightarrow \langle XX \rangle$

$X \rightarrow$ *nosso*

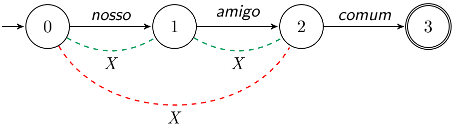
$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

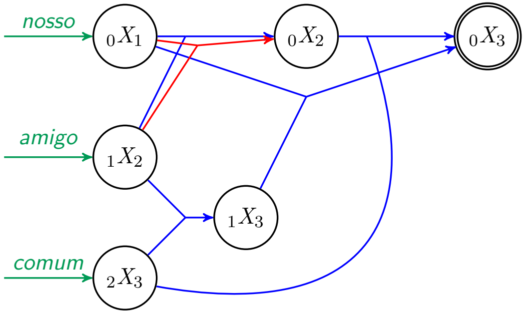
$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$ $\Longleftarrow$

$X \rightarrow$ *nosso*

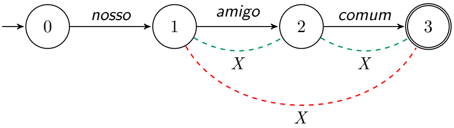
$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

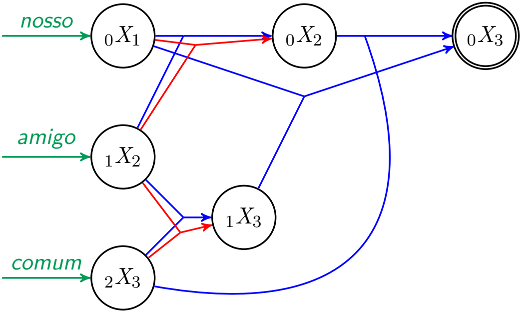
$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$ $\Leftarrow$

$X \rightarrow$ *nosso*

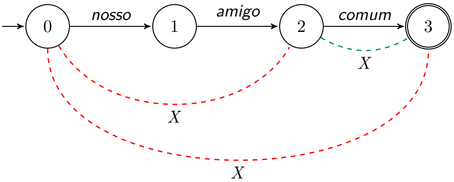
$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

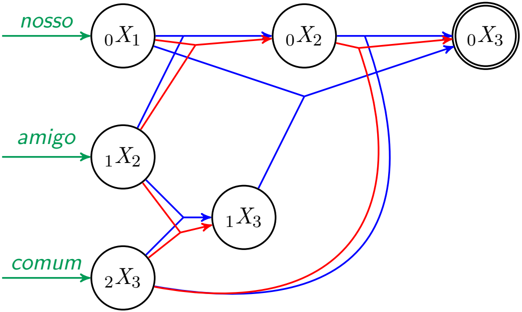
$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$ $\Longleftarrow$

$X \rightarrow$ *nosso*

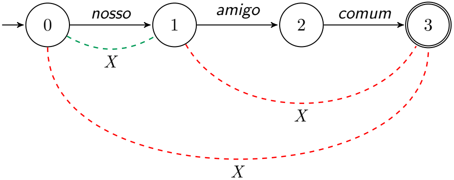
$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

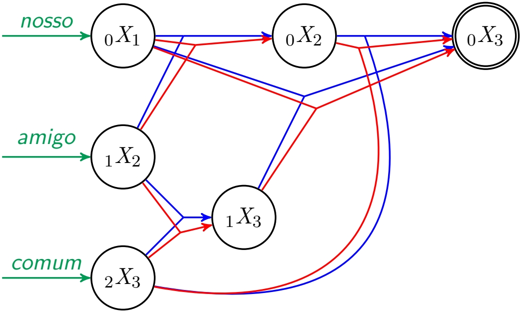
$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$ $\Longleftarrow$

$X \rightarrow$ *nosso*

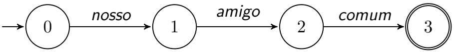
$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

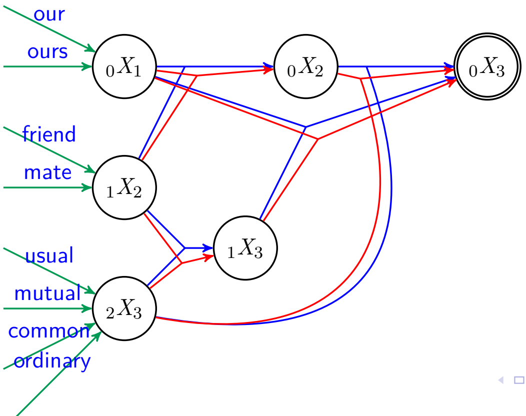
$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$

$X \rightarrow$ *nosso*

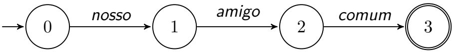
$X \rightarrow$ *amigo*

$X \rightarrow$ *comum*



Parsing, intersection and hypergraphs

Source



Grammar

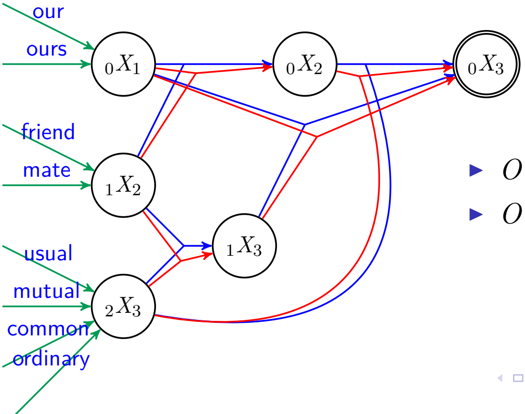
$X \to XX$

$X \to \langle XX \rangle$

$X \to$ *nosso*

$X \to$ *amigo*

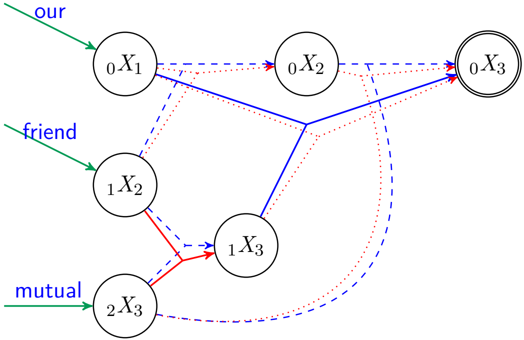
$X \to$ *comum*



- $O(I^3)$ nodes
- $O(tI^3)$ edges

Example

(nosso ⟨amigo comum⟩) $\rightarrow$ **our mutual friend**



Recap 2

1. our first model of translational equivalences assumed **monotonicity**

Recap 2

1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input

Recap 2

1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input
3. to avoid NP-completeness, we constrained permutations using a **distortion limit**

Recap 2

1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input
3. to avoid NP-completeness, we constrained permutations using a **distortion limit**
4. we can instead constrain permutations using an **ITG**

Recap 2

1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input
3. to avoid NP-completeness, we constrained permutations using a **distortion limit**
4. we can instead constrain permutations using an **ITG**

But we still perform translation word-by-word with no insertion or deletion!

1-1 mappings: fail!

Source: o$_1$ grilo$_2$ *da*$_3$ lareira$_4$
Target: the$_1$ cricket$_2$ [on the]$_3$ hearth$_4$

Insertion and deletion

Implicitly modelled by moving from words to phrases

Insertion and deletion

Implicitly modelled by moving from words to phrases

- a phrase replacement model

Insertion and deletion

Implicitly modelled by moving from words to phrases

- a phrase replacement model
- operating with an ITG (or with a distortion limit)

Insertion and deletion

Implicitly modelled by moving from words to phrases

- a phrase replacement model
- operating with an ITG (or with a distortion limit)
- with no phrase-insertion or phrase-deletion

Insertion and deletion

Implicitly modelled by moving from words to phrases

- a phrase replacement model
- operating with an ITG (or with a distortion limit)
- with no phrase-insertion or phrase-deletion
- constrained to known phrase-to-phrase bilingual mappings (rule set)

Phrase mappings

Mappings of contiguous sequences of words

Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)

Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data

Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)

Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)

Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)
- their words need not align monotonically
  which gives us a bit of reordering power as well ;)

Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)
- their words need not align monotonically
  which gives us a bit of reordering power as well ;)
  e.g. *a loja de antiguidades*/old curiosity shop

Generalising the rule set (FST)

Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Generalising the rule set (FST)

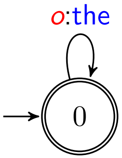
Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



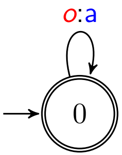
Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

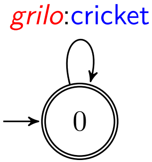
Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)

Rules

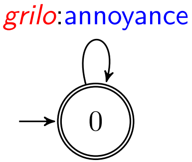
o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}



Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



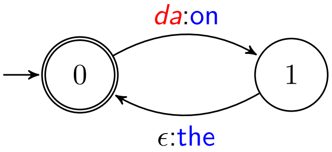
Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



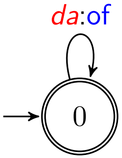
Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



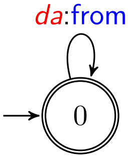
Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



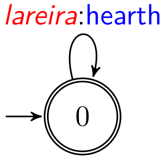
Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)



Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer

Generalising the rule set (FST)

Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer
- the union represents the rule set

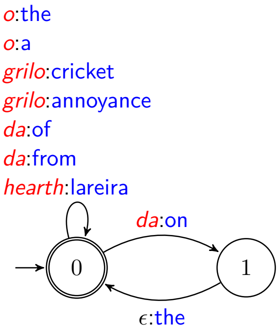
Generalising the rule set (FST)

Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer
- the union represents the rule set



Generalising the rule set (FST)

Rules

o	{the, a}
grilo	{cricket, annoyance}
da	{on the, of, from}
hearth	{lareira}

Using FST

- each rule can be seen as a transducer
- the union represents the rule set
- standard intersection mechanisms do the rest

Phrase permutations' translation with WLd

We can translate a lattice encoding the WLd permutations

Phrase permutations' translation with WLd

We can translate a lattice encoding the WLd permutations

- a truncated window controls reordering

Phrase permutations' translation with WLd

We can translate a lattice encoding the WLd permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input

Phrase permutations' translation with WLd

We can translate a lattice encoding the WLd permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input
  - $O(I^2)$ segments

Phrase permutations' translation with WLd

We can translate a lattice encoding the WLd permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input
  - $O(I^2)$ segments
  - it is sensible to limit phrases to a maximum length

Phrase permutations' translation with WLd

We can translate a lattice encoding the WLd permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input
  - $O(I^2)$ segments
  - it is sensible to limit phrases to a maximum length
- complexity remains
  - linear with sentence length
  - exponential with distortion limit

Generalising the rule set (ITG)

Simply extend the terminal rules

Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \to XX$
  direct order

Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \to XX$
  direct order
- $X \to \langle XX \rangle$
  inverted order

Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  direct order
- $X \rightarrow \langle XX \rangle$
  inverted order
- $X \rightarrow r_i$, where $r_i \in R$
  bilingual mappings

Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  direct order
- $X \rightarrow \langle XX \rangle$
  inverted order
- $X \rightarrow r_i$, where $r_i \in R$
  bilingual mappings

Examples

$X \rightarrow$ o/the
$X \rightarrow$ grilo/cricket
$X \rightarrow$ da/on the

Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  direct order
- $X \rightarrow \langle XX \rangle$
  inverted order
- $X \rightarrow r_i$, where $r_i \in R$
  bilingual mappings

Examples
$X \rightarrow$ o/the
$X \rightarrow$ grilo/cricket
$X \rightarrow$ da/on the
The intersection mechanisms do the rest

- $O(I^3)$ nodes (phrases are limited in length)
- $O(tI^3)$ edges

Recap 3

We have

Recap 3

We have

1. defined different models of translational equivalence

Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases

Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order

Recap 3

We have

1. defined different models of translational equivalence
    - by translating words or phrases
    - in arbitrary order
    - or according to an ITG

Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order
   - or according to an ITG
2. efficiently represented the set of translations supported by these models for a given input sentence

Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order
   - or according to an ITG
2. efficiently represented the set of translations supported by these models for a given input sentence
   - trivially expressed in terms of intersection/composition

Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order
   - or according to an ITG
2. efficiently represented the set of translations supported by these models for a given input sentence
   - trivially expressed in terms of intersection/composition
   - a logic program can do the same
     (sometimes more convenient, e.g. WLd constraints)

Remarks

Phrase-based SMT [Koehn et al., 2003]

Remarks

Phrase-based SMT [Koehn et al., 2003]

- the space of solutions grows linearly with input length and exponentially with the distortion limit

Remarks

Phrase-based SMT [Koehn et al., 2003]

- the space of solutions grows linearly with input length and exponentially with the distortion limit

ITG [Wu, 1997]

Remarks

Phrase-based SMT [Koehn et al., 2003]

- the space of solutions grows linearly with input length and exponentially with the distortion limit

ITG [Wu, 1997]

- the space of solutions is cubic in length

Remarks

Phrase-based SMT [Koehn et al., 2003]

- the space of solutions grows linearly with input length and exponentially with the distortion limit

ITG [Wu, 1997]

- the space of solutions is cubic in length
- better motivated constraints on reordering

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

[1] Other than monotone translation with *glue rules*

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)

[1] Other than monotone translation with *glue rules*

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)

[1] Other than monotone translation with *glue rules*

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. $X \rightarrow$ loja de antiguidades/old curiosity shop

[1] Other than monotone translation with *glue rules*

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. $X \rightarrow$ loja de antiguidades/old curiosity shop
- as well as lexicalised recursive rules
  e.g. $X \rightarrow X_1$ de X_2 / X_2 's X_1

[1] Other than monotone translation with *glue rules*

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. $X \rightarrow$ loja de antiguidades/old curiosity shop
- as well as lexicalised recursive rules
  e.g. $X \rightarrow X_1$ de X_2 / X_2 's X_1
- no purely unlexicalised rules[1]

[1] Other than monotone translation with *glue rules*

Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. $X \rightarrow$ loja de antiguidades/old curiosity shop
- as well as lexicalised recursive rules
  e.g. $X \rightarrow X_1$ de X_2 / X_2 's X_1
- no purely unlexicalised rules[1]
- same cubic dependency on input length (as ITGs)

[1] Other than monotone translation with *glue rules*

What are we missing?

We have characterised the set of solutions "backed" by our transfer model

What are we missing?

We have characterised the set of solutions "backed" by our transfer model

- these solutions are unweighted

What are we missing?

We have characterised the set of solutions “backed” by our transfer model

- these solutions are unweighted
- there is no obvious way to discriminate them

What are we missing?

We have characterised the set of solutions “backed” by our transfer model

- these solutions are unweighted
- there is no obvious way to discriminate them
- we cannot make decisions like that

What are we missing?

We have characterised the set of solutions “backed” by our transfer model

- these solutions are unweighted
- there is no obvious way to discriminate them
- we cannot make decisions like that

We are missing a parameterisation of the model

- the scoring function which will guide the decision making process

Linear models

Let's call **derivation**

Linear models

Let's call **derivation**

- a translation string

Linear models

Let's call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model e.g. phrase segmentation, alignment

Linear models

Let's call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

$$f(\mathbf{d}) = \sum_k \lambda_k H_k(\mathbf{d})$$

where $\mathbf{d}$ is the derivation, and H_k is one of m feature functions

Linear models

Let's call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

$$f(\mathbf{d}) = \sum_k \lambda_k H_k(\mathbf{d})$$

where $\mathbf{d}$ is the derivation, and H_k is one of m feature functions

It assigns a real-valued score to each and every derivation

Linear models

Let's call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

$$f(\mathbf{d}) = \sum_k \lambda_k H_k(\mathbf{d})$$

where $\mathbf{d}$ is the derivation, and H_k is one of m feature functions

It assigns a real-valued score to each and every derivation

Think of it as a surrogate for translation quality at decoding time

[Berger et al., 1996]
[Och and Ney, 2002]

Feature functions

Independently capture different aspects of the translation, such as

- adequacy
  - translation probabilities
  - confidence on lexical choices
- fluency
  - LM probabilities
  - confidence on reodering

Independence assumptions

Our transfer model makes independence assumptions

- "translation happens by concatenating isolated rules" e.g. flat mappings, hierarchical mappings

Independence assumptions

Our transfer model makes independence assumptions

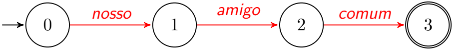
- "translation happens by concatenating isolated rules" e.g. flat mappings, hierarchical mappings

Certain aspects of translation quality comply with such assumptions

- how likely a certain translation rule is
  e.g. relative frequency in a bilingual corpus

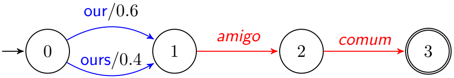
Structural independence: scoring rules in isolation

Scoring rules independently



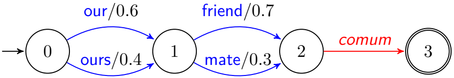
Structural independence: scoring rules in isolation

Scoring rules independently



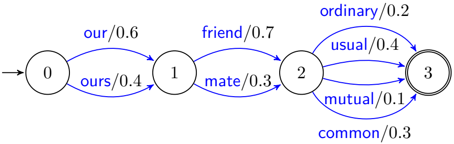
Structural independence: scoring rules in isolation

Scoring rules independently



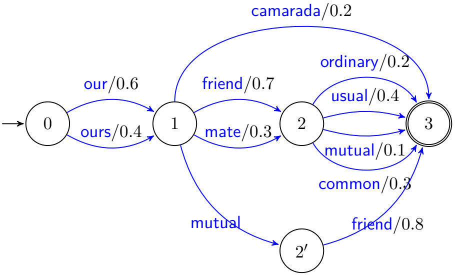
Structural independence: scoring rules in isolation

Scoring rules independently



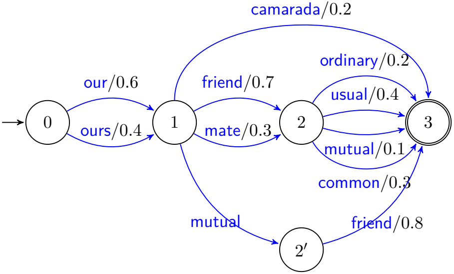
Structural independence: scoring rules in isolation

Scoring rules independently



Structural independence: scoring rules in isolation

Scoring rules independently



inference runs in time linear with the size of the automaton

Independence assumptions

Our transfer model makes independence assumptions

- "translation happens by concatenating isolated rules" e.g. flat mappings, hierarchical mappings

Certain aspects of translation quality comply with such assumptions

- how likely a certain translation rule is
  e.g. relative frequency in a bilingual corpus

Independence assumptions

Our transfer model makes independence assumptions

- "translation happens by concatenating isolated rules" e.g. flat mappings, hierarchical mappings

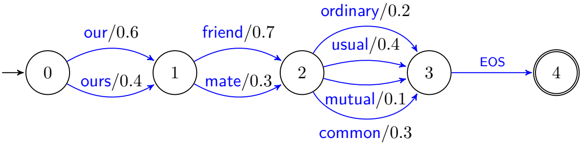
Certain aspects of translation quality comply with such assumptions

- how likely a certain translation rule is
  e.g. relative frequency in a bilingual corpus

Certain aspects do not comply with such assumptions

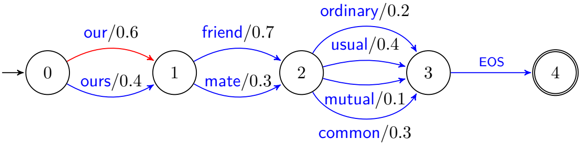
- fluency as captured by an n-gram LM component

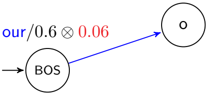
Scoring strings with a 2-gram LM



requires unpacking the representation

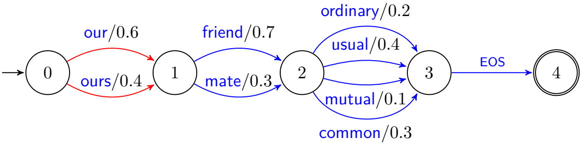
Scoring strings with a 2-gram LM

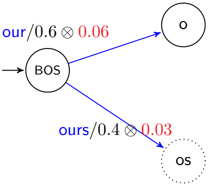




requires unpacking the representation

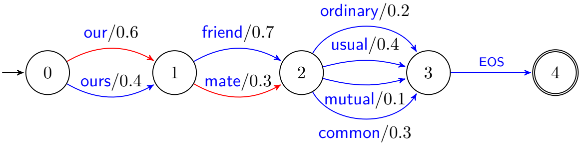
Scoring strings with a 2-gram LM

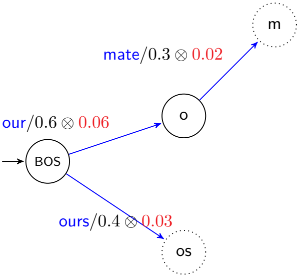




requires unpacking the representation

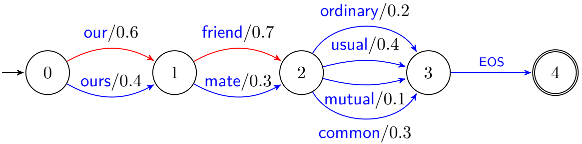
Scoring strings with a 2-gram LM

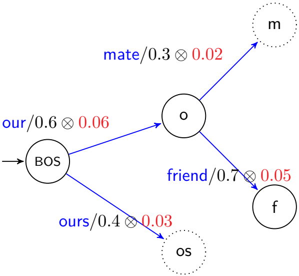




requires unpacking the representation

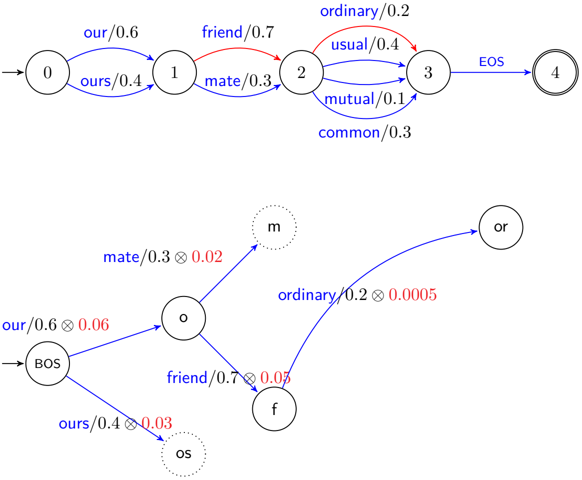
Scoring strings with a 2-gram LM





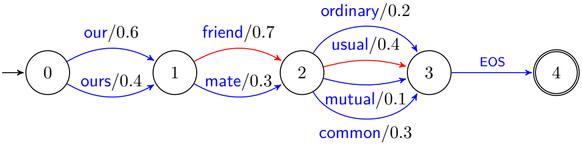
requires unpacking the representation

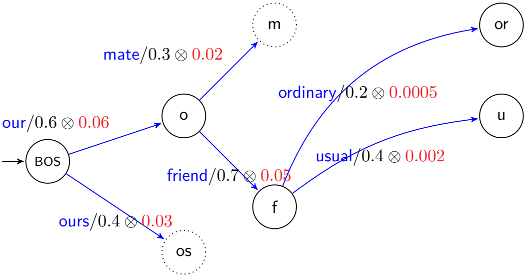
Scoring strings with a 2-gram LM



requires unpacking the representation

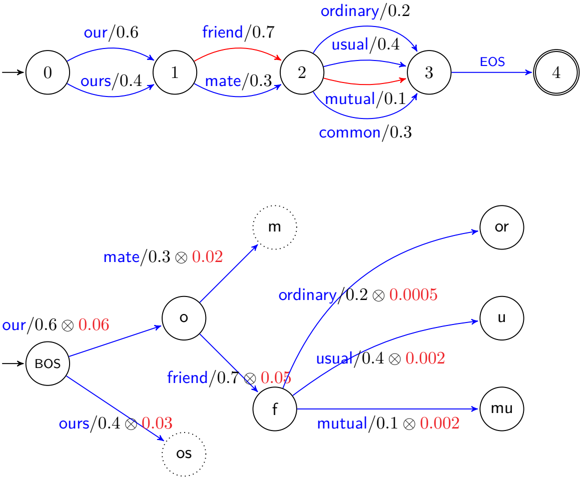
Scoring strings with a 2-gram LM





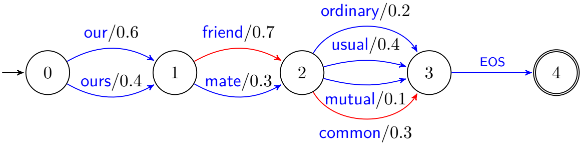
requires unpacking the representation

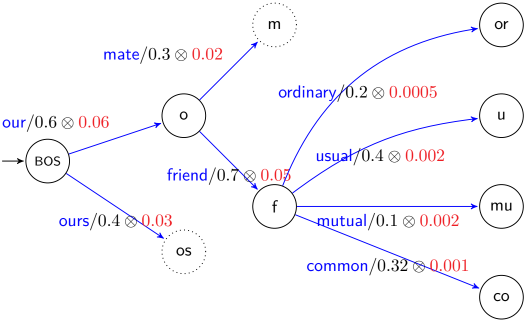
Scoring strings with a 2-gram LM



requires unpacking the representation

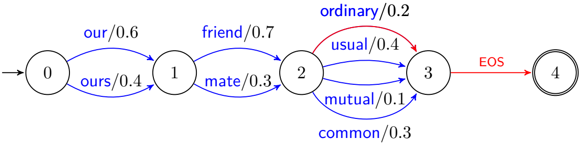
Scoring strings with a 2-gram LM

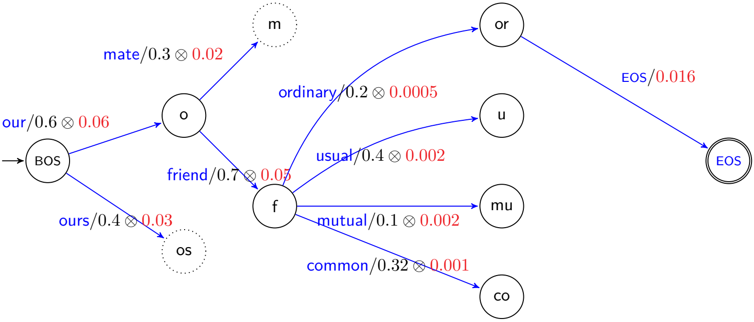




requires unpacking the representation

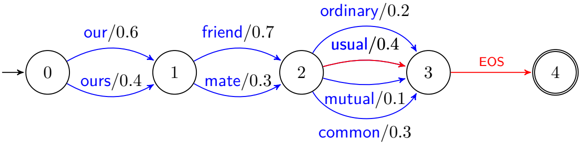
Scoring strings with a 2-gram LM

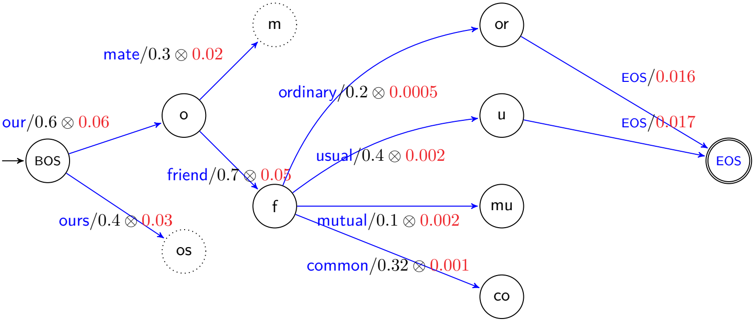




requires unpacking the representation

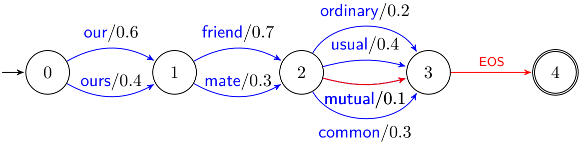
Scoring strings with a 2-gram LM

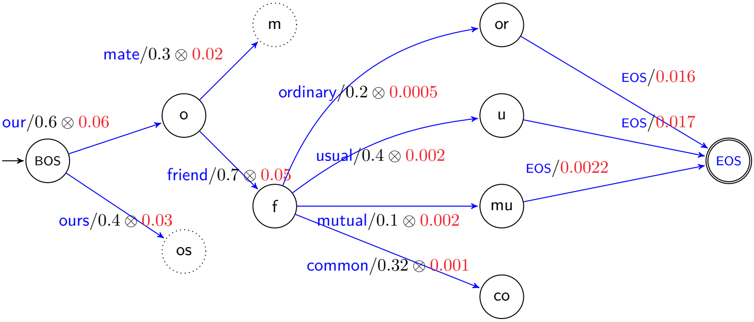




requires unpacking the representation

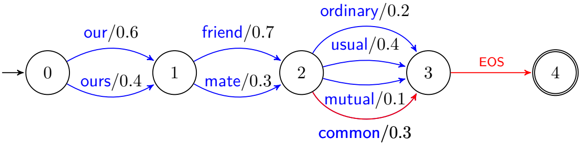
Scoring strings with a 2-gram LM

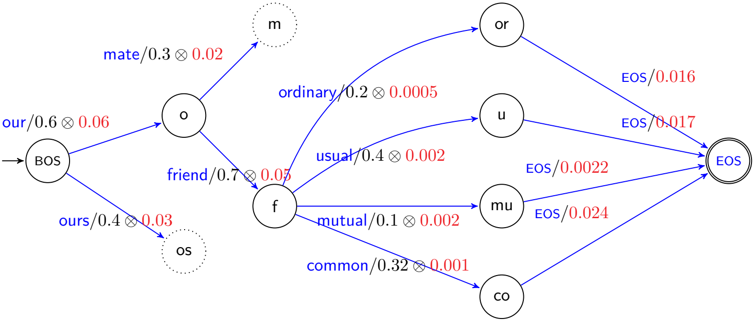




requires unpacking the representation

Scoring strings with a 2-gram LM





requires unpacking the representation

Scoring whole sentences

Imagine a feature function that requires a complete translation

Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  e.g. via suffix arrays [Zhang and Vogel, 2006]
  e.g. via RNN language model
- estimated overall translation quality

Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  e.g. via suffix arrays [Zhang and Vogel, 2006]
  e.g. via RNN language model
- estimated overall translation quality

No factorisation at the phrase (nor n-gram) level

Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  e.g. via suffix arrays [Zhang and Vogel, 2006]
  e.g. via RNN language model
- estimated overall translation quality

No factorisation at the phrase (nor n-gram) level

- requires fully unpacking the representation

Scoring whole sentences

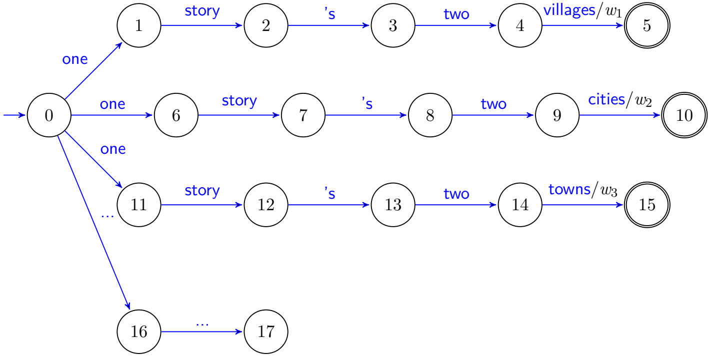
Imagine a feature function that requires a complete translation

- unbounded LM
  e.g. via suffix arrays [Zhang and Vogel, 2006]
  e.g. via RNN language model
- estimated overall translation quality

No factorisation at the phrase (nor n-gram) level

- requires fully unpacking the representation
- making dependencies explicit through the graphical structure

Scoring whole sentences: example



Exhaustive enumeration

Not all is lost

Most features we can reliably estimate

Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context

Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context
- are quite incremental

Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context
- are quite incremental

n-gram LMs are good examples

- there are up to $|\Delta|^{n-1}$ contexts that must be made explicit

Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context
- are quite incremental

n-gram LMs are good examples

- there are up to $|\Delta|^{n-1}$ contexts that must be made explicit
- nodes must group derivations sharing the same context

Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context
- are quite incremental

n-gram LMs are good examples

- there are up to $|\Delta|^{n-1}$ contexts that must be made explicit
- nodes must group derivations sharing the same context
- polynomial, though often prohibitive (impracticable)

Recap 4

1. a characterisation the space of solutions

Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model

Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

What's left?

Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

What's left?

- more examples of models and impact on representation
  - distance-based reordering
  - lexicalised models
  - a global feature function
- inference algorithms

Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

What's left?

- more examples of models and impact on representation
  - distance-based reordering
  - lexicalised models
  - a global feature function
- inference algorithms
- techniques to make inference feasible for interesting models

Picking one solution

What do we pick out of the (whole) weighted space of solutions?

- best translation
- "minimum-loss" translation

Best translation

MAP

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathrm{y}[\mathbf{d}]=\mathbf{y}} f(\mathbf{d}|\mathbf{x})$$

Best translation

MAP

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathrm{y}[\mathbf{d}]=\mathbf{y}} f(\mathbf{d}|\mathbf{x})$$

- summing alternative derivations of the same string
  NP-complete: related to determinisation [Sima'an, 1996]

Best translation

MAP

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathrm{y}[\mathbf{d}]=\mathbf{y}} f(\mathbf{d}|\mathbf{x})$$

- summing alternative derivations of the same string
  NP-complete: related to determinisation [Sima'an, 1996]

Viterbi (approximation to MAP)

$$\mathbf{d}^* = \operatorname*{argmax}_{\mathbf{d}} f(\mathbf{d}|\mathbf{x})$$

- assumes the most likely derivation is enough

Minimum Bayes Risk translation

MBR

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \operatorname*{argmin}_{\mathbf{y}} \langle \text{loss}(\mathbf{y}, \mathbf{y}') \rangle_{p(\mathbf{y}'|\mathbf{x})}$$

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y}} \left\langle \mathrm{gain}(\mathbf{y}, \mathbf{y}') \right\rangle_{p(\mathbf{y}'|\mathbf{x})}$$

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \underset{\mathbf{y}}{\operatorname{argmax}} \left\langle \mathrm{BLEU}(\mathbf{y}, \mathbf{y}') \right\rangle_{p(\mathbf{y}'|\mathbf{x})}$$

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \underset{\mathbf{y}}{\operatorname{argmax}} \left\langle \mathrm{BLEU}(\mathbf{y}, \mathbf{y}') \right\rangle_{p(\mathbf{y}'|\mathbf{x})}$$

- assesses the risk associated with choosing any one translation

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathbf{y}'} \mathrm{BLEU}(\mathbf{y}, \mathbf{y}') p(\mathbf{y}'|\mathbf{x})$$

- assesses the risk associated with choosing any one translation
- requires the computation of expectations

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathbf{y}'} \mathrm{BLEU}(\mathbf{y}, \mathbf{y}') p(\mathbf{y}' | \mathbf{x})$$

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

$$p(\mathbf{d}|\mathbf{x}) = \frac{f(\mathbf{d}|\mathbf{x})}{\sum_{\mathbf{d}'} f(\mathbf{d}'|\mathbf{x})}$$

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathbf{y}' \sim p(\mathbf{y}'|\mathbf{x})} \mathrm{BLEU}(\mathbf{y}, \mathbf{y}')$$

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

$$p(\mathbf{d}|\mathbf{x}) = \frac{f(\mathbf{d}|\mathbf{x})}{\sum_{\mathbf{d}'} f(\mathbf{d}'|\mathbf{x})}$$

- can be estimated by sampling translations

Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y}} \sum_{\mathbf{y}'} \sum_{\mathbf{d}' \sim p(\mathbf{d}'|\mathbf{x})} \mathrm{BLEU}(\mathbf{y}, \mathrm{y}[\mathbf{d}'])$$

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

$$p(\mathbf{d}|\mathbf{x}) = \frac{f(\mathbf{d}|\mathbf{x})}{\sum_{\mathbf{d}'} f(\mathbf{d}'|\mathbf{x})}$$

- can be estimated by sampling translations
- can be estimated from samples of derivations

DP-based Viterbi

Explore a truncated version of the full space

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best
  - cube pruning: enumerates k edges in near best-first order

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best
  - cube pruning: enumerates k edges in near best-first order

In order to compare hypotheses more fairly

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best
  - cube pruning: enumerates k edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
    - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best
    - cube pruning: enumerates k edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates
- heuristic view of outside weights

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best
  - cube pruning: enumerates k edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates
- heuristic view of outside weights
- cheap dynamic program that estimates the best possible way to complete any translation prefix

DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but k-best
  - cube pruning: enumerates k edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates
- heuristic view of outside weights
- cheap dynamic program that estimates the best possible way to complete any translation prefix

[Koehn et al., 2003]
[Chiang, 2007]

DP-based MBR

Uses derivations in an n-best list as samples

DP-based MBR

Uses derivations in an n-best list as samples

- arguably poor proxy to samples

DP-based MBR

Uses derivations in an n-best list as samples

- arguably poor proxy to samples
- arbitrarily biased (due to pruning)

DP-based MBR

Uses derivations in an n-best list as samples

- arguably poor proxy to samples
- arbitrarily biased (due to pruning)
- centred around the Viterbi solution by design (due to beam search)

DP-based MBR

Uses derivations in an n-best list as samples

- arguably poor proxy to samples
- arbitrarily biased (due to pruning)
- centred around the Viterbi solution by design (due to beam search)

[Kumar and Byrne, 2004]
[Tromble et al., 2008]

Sampling

Gibbs sampling

Sampling

Gibbs sampling

1. start with a draft translation

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)

Importance sampling

Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)

Importance sampling

- you will hear from us (project 14) ;)

Sampling

Disadvantages

Sampling

Disadvantages

- hard to do it without introducing bias

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

1. broad view of distribution

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)
3. sometimes unbiased

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)
3. sometimes unbiased
4. ideal for MBR and tuning

Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)
3. sometimes unbiased
4. ideal for MBR and tuning
5. typically stupid simple to parallelise

Thanks!

Questions?

References I

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, March 1996. ISSN 0891-2017. URL http://dl.acm.org/citation.cfm?id=234285.234289.

David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 263–270, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219873. URL http://dx.doi.org/10.3115/1219840.1219873.

David Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33:201–228, 2007. URL http://www.mitpressjournals.org/doi/abs/10.1162/coli.2007.33.2.201.

References II

Kevin Knight. Decoding complexity in word-replacement translation models. *Comput. Linguist.*, 25(4):607–615, December 1999. ISSN 0891-2017. URL http://dl.acm.org/citation.cfm?id=973226.973232.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1073445.1073462. URL http://dx.doi.org/10.3115/1073445.1073462.

References III

Shankar Kumar and William Byrne. Minimum Bayes-risk decoding for statistical machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 169–176, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

Adam Lopez. Statistical machine translation. *ACM Computing Surveys*, 40(3):8:1–8:49, August 2008. ISSN 0360-0300. doi: 10.1145/1380584.1380586. URL http://doi.acm.org/10.1145/1380584.1380586.

Adam Lopez. Translation as weighted deduction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 532–540, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1609067.1609126.

References IV

Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 295–302, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073133. URL http://dx.doi.org/10.3115/1073083.1073133.

Khalil Sima'an. Computational complexity of probabilistic disambiguation by means of tree-grammars. In *Proceedings of the 16th conference on Computational linguistics - Volume 2*, COLING '96, pages 1175–1180, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics. doi: 10.3115/993268.993392. URL http://dx.doi.org/10.3115/993268.993392.

References V

Roy W. Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. Lattice minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 620–629, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1613715.1613792`.

Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, September 1997. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=972705.972707`.

Ying Zhang and Stephan Vogel. Suffix array and its applications in empirical natural language processing. Technical report, CMU, Pittsburgh, PA, USA, December 2006.