

# Machine Translation

(Following slides are modified from Prof. Raymond Mooney's slides.)

# Machine Translation

- Automatically translate one natural language into another.

**Mary didn't slap the green witch.**



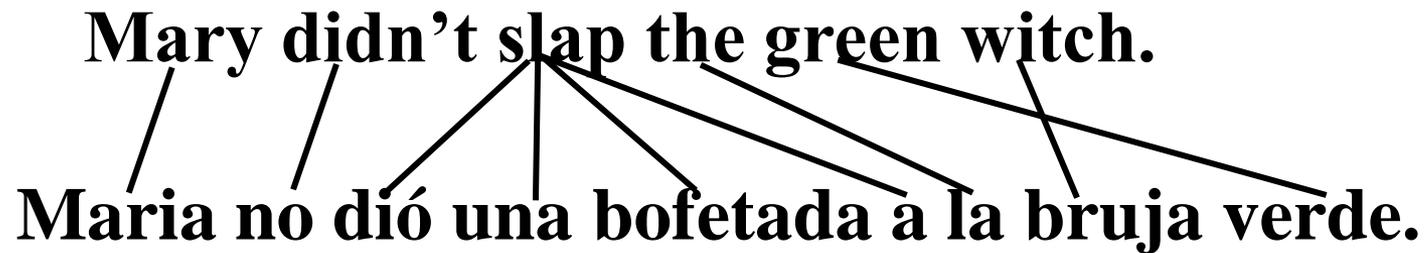
**Maria no dió una bofetada a la bruja verde.** (Spanish)

# Ambiguity Resolution is Required for Translation

- Syntactic and semantic ambiguities must be properly resolved for correct translation:
  - “John plays the guitar.” → “John toca la guitarra.”
  - “John plays soccer.” → “John juega el fútbol.”
- An apocryphal story is that an early MT system gave the following results when translating from English to Russian and then back to English:
  - “The spirit is willing but the flesh is weak.” ⇒ “The liquor is good but the meat is spoiled.”
  - “Out of sight, out of mind.” ⇒ “Invisible idiot.”

# Word Alignment

- Shows mapping between words in one language and the other.



# Translation Quality

- Achieving literary quality translation is very difficult.
- Existing MT systems can generate rough translations that convey at least the gist of a document.
- High quality translations possible when specialized to narrow domains, e.g. weather forecasts.
- Some MT systems used in ***computer-aided translation*** in which a bilingual human post-edits the output to produce more readable accurate translations.
- Frequently used to aid ***localization*** of software interfaces and documentation to adapt them to other languages.

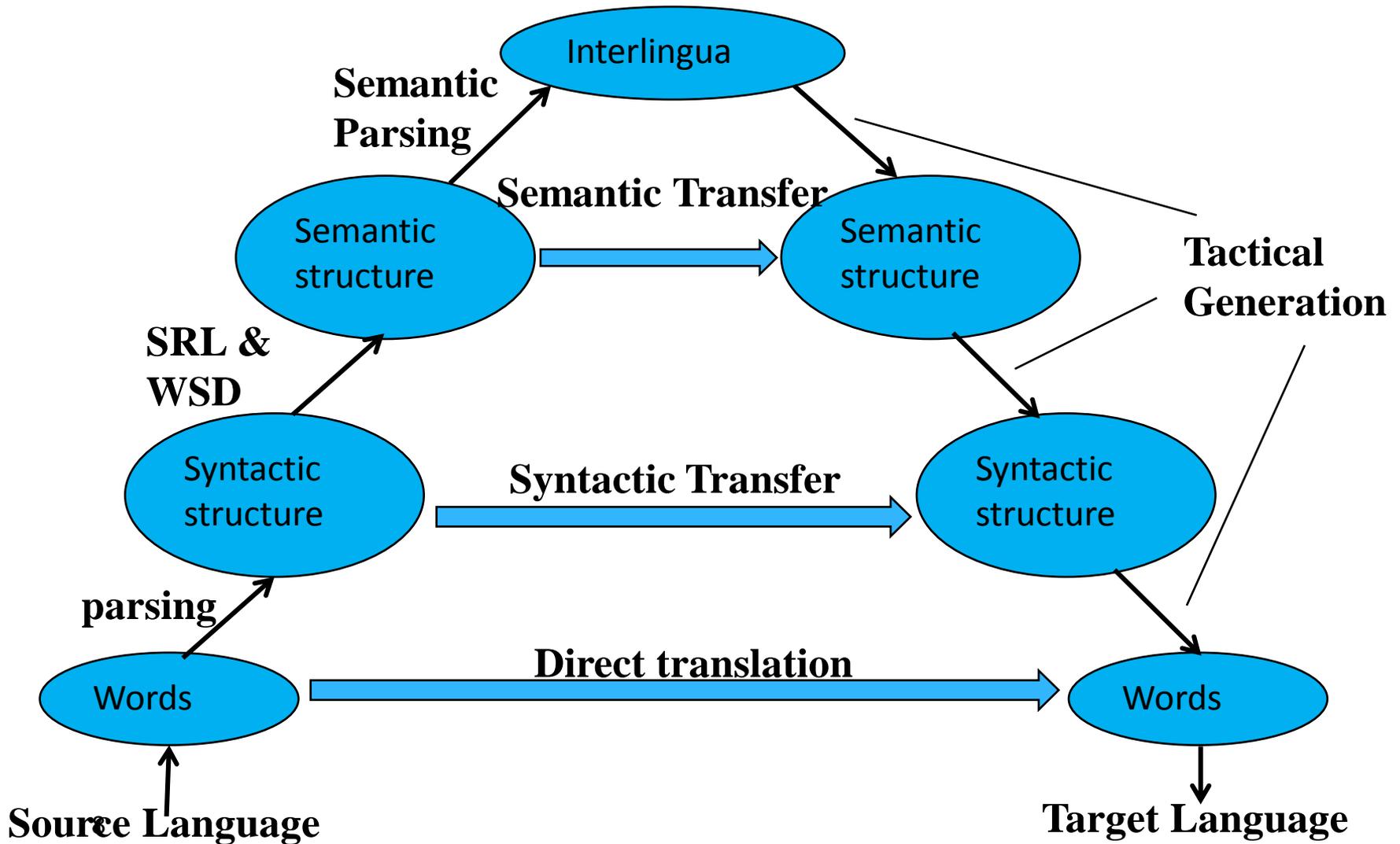
# Linguistic Issues Making MT Difficult

- Morphological issues with ***agglutinative***, ***fusional*** and ***polysynthetic*** languages with complex word structure.
- Syntactic variation between ***SVO*** (e.g. English), ***SOV*** (e.g. Hindi), and ***VSO*** (e.g. Arabic) languages.
  - SVO languages use prepositions
  - SOV languages use postpositions
- ***Pro-drop*** languages regularly omit subjects that must be inferred.

# Lexical Gaps

- Some words in one language do not have a corresponding term in the other.
  - Rivière (river that flows into ocean) and fleuve (river that does not flow into ocean) in French
  - Schedenfraude (feeling good about another's pain) in German.
  - Oyakoko (filial piety) in Japanese

# “Vauquois Triangle”



# Direct Transfer

- Morphological Analysis

- Mary didn't slap the green witch. →  
Mary DO:PAST not slap the green witch.

- Lexical Transfer

- Mary DO:PAST not slap the green witch.

- Maria no dar:PAST una bofetada a la verde bruja.

- Lexical Reordering

- Maria no dar:PAST una bofetada a la bruja verde.

- Morphological generation

- Maria no dió una bofetada a la bruja verde.

# Syntactic Transfer

- Simple lexical reordering does not adequately handle more dramatic reordering such as that required to translate from an SVO to an SOV language.
- Need syntactic transfer rules that map parse tree for one language into one for another.
  - English to Spanish:
    - $NP \rightarrow \text{Adj Nom} \Rightarrow NP \rightarrow \text{Nom ADJ}$
  - English to Japanese:
    - $VP \rightarrow V NP \Rightarrow VP \rightarrow NP V$
    - $PP \rightarrow P NP \Rightarrow PP \rightarrow NP P$

# Semantic Transfer

- Some transfer requires semantic information.
- Semantic roles can determine how to properly express information in another language.
- In Chinese, PPs that express a goal, destination, or benefactor occur **before** the verb but those expressing a recipient occur **after** the verb.
- Transfer Rule
  - English to Chinese
    - $VP \rightarrow V PP[+\text{benefactor}] \Rightarrow VP \rightarrow PP[+\text{benefactor}] V$

# Statistical MT

- Manually encoding comprehensive bilingual lexicons and transfer rules is difficult.
- SMT acquires knowledge needed for translation from a *parallel corpus* or *bitext* that contains the same set of documents in two languages.
- The Canadian Hansards (parliamentary proceedings in French and English) is a well-known parallel corpus.
- First align the sentences in the corpus based on simple methods that use coarse cues like sentence length to give bilingual sentence pairs.

# Picking a Good Translation

- A good translation should be ***faithful*** and correctly convey the information and tone of the original source sentence.
- A good translation should also be ***fluent***, grammatically well structured and readable in the target language.
- Final objective:

$$T_{best} = \operatorname{argmax}_{T \in \text{Target}} \text{faithfulness}(T, S) \text{ fluency}(T)$$

# “Noisy Channel Model”

- Based on analogy to information-theoretic model used to decode messages transmitted via a communication channel that adds errors.
- Assume that source sentence was generated by a “noisy” transformation of some target language sentence and then use Bayesian analysis to recover the most likely target sentence that generated it.

Translate foreign language sentence  $F = f_1, f_2, \dots, f_m$  to an English sentence  $\hat{E} = e_1, e_2, \dots, e_l$  that maximizes  $P(E | F)$

# Bayesian Analysis of **Noisy Channel**

$$\begin{aligned}\hat{E} &= \operatorname{argmax}_{E \in \text{English}} P(E | F) \\ &= \operatorname{argmax}_{E \in \text{English}} \frac{P(F | E)P(E)}{P(F)} \\ &= \operatorname{argmax}_{E \in \text{English}} \underbrace{P(F | E)}_{\text{Translation Model}} \underbrace{P(E)}_{\text{Language Model}}\end{aligned}$$

A **decoder** determines the most probable translation  $\hat{E}$  given  $F$

# Language Model

- Use a standard  $n$ -gram language model for  $P(E)$ .
- Can be trained on a large, unsupervised mono-lingual corpus for the target language  $E$ .
- Could use a more sophisticated PCFG language model to capture long-distance dependencies.
- Terabytes of web data have been used to build a large 5-gram model of English.

# Phrase-Based Translation Model

- Base  $P(F | E)$  on translating phrases in  $E$  to phrases in  $F$ .
- First segment  $E$  into a sequence of phrases  $\bar{e}_1, \bar{e}_1, \dots, \bar{e}_I$
- Then translate each phrase  $\bar{e}_i$  into  $f_i$ , based on **translation probability**  $\phi(f_i | \bar{e}_i)$
- Then reorder translated phrases based on **distortion probability**  $d(i)$  for the  $i$ th phrase.

$$P(F | E) = \prod_{i=1}^I \phi(\bar{f}_i, \bar{e}_i) d(i)$$

# Translation Probabilities

- Assuming a **phrase aligned** parallel corpus is available or constructed that shows matching between phrases in  $E$  and  $F$ .
- Then compute (MLE) estimate of  $\phi$  based on simple frequency counts.

$$\phi(\bar{f}, \bar{e}) = \frac{\text{count}(\bar{f}, \bar{e})}{\sum_{\bar{f}} \text{count}(\bar{f}, \bar{e})}$$

# Distortion Probability

- Measure distortion of phrase  $i$  as the distance between the start of the  $f$  phrase generated by  $\bar{e}_i$ , ( $a_i$ ) and the end of the end of the  $f$  phrase generated by the previous phrase  $\bar{e}_{i-1}$ , ( $b_{i-1}$ ).
- Typically assume the probability of a distortion decreases exponentially with the distance of the movement.

$$d(i) = c\alpha^{|a_i - b_{i-1}|}$$

**Set  $0 < \alpha < 1$  based on fit to phrase-aligned training data**  
**Then set  $c$  to normalize  $d(i)$  so it sums to 1.**

# Sample Translation Model

Position	1	2	3	4	5	6
English	Mary	did not	slap	the	green	witch
Spanish	Maria	no	dió una bofetada a	la	bruja	verde
$a_i - b_{i-1}$	1	1	1	1	2	1

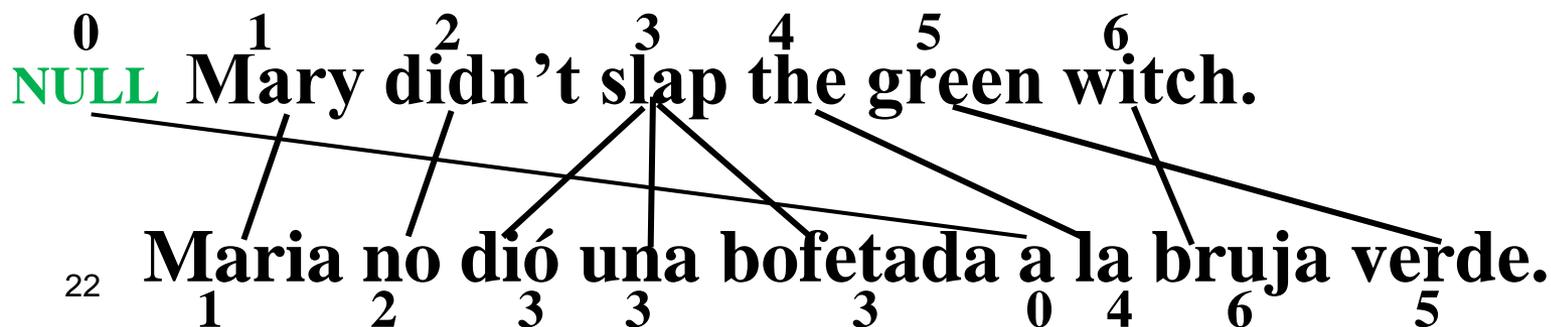
$$p(F | E) = \phi(\text{Maria, Mary})c\alpha^1\phi(\text{no, did not})c\alpha^1\phi(\text{slap, dio una bofetada a})c\alpha^1\phi(\text{la, the})c\alpha^1\phi(\text{verde, green})c\alpha^2\phi(\text{bruja, witch})c\alpha^1$$

# Word Alignment

- Directly constructing phrase alignments is difficult, so rely on first constructing word alignments.
- Can learn to align from supervised word alignments, but human-aligned bitexts are rare and expensive to construct.
- Typically use an unsupervised EM-based approach to compute a word alignment from unannotated parallel corpus.

# One to Many Alignment

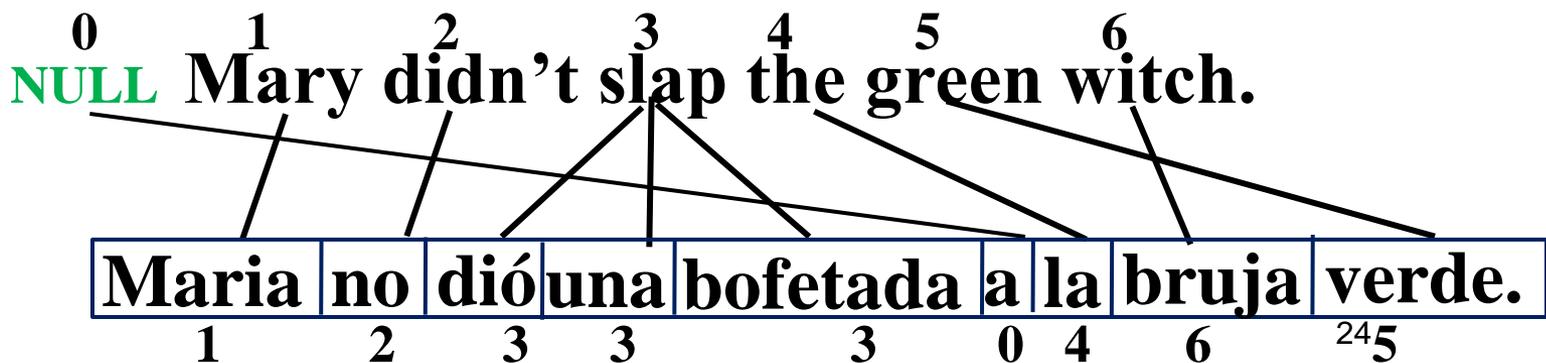
- To simplify the problem, typically assume **each word in  $F$  aligns to 1 word in  $E$**  (but assume each word in  $E$  may generate more than one word in  $F$ ).
- Some words in  $F$  may be generated by the NULL element of  $E$ .
- Therefore, alignment can be specified by a vector  $A$  giving, for each word in  $F$ , the index of the word in  $E$  which generated it.



# IBM Model 1

- First model proposed in seminal paper by Brown *et al.* in 1993 as part of CANDIDE, the first complete SMT system.
- Assumes following simple generative model of producing  $F$  from  $E=e_1, e_2, \dots, e_l$ 
  1. Choose  $J$  as the sentence length for  $F$
  2. Choose a 1 to many alignment  $A=a_1, a_2, \dots, a_j$
  3. For each position in  $F$ , generate a word  $f_j$  from the aligned word in  $E$ :  $e_{a_j}$

# Sample IBM Model 1 Generation



- Assumes following simple generative model of producing  $F$  from  $E=e_1, e_2, \dots, e_l$ 
  1. Choose  $J$  as the sentence length for  $F$
  2. Choose a 1 to many alignment  $A=a_1, a_2, \dots, a_j$
  3. For each position in  $F$ , generate a word  $f_j$  from the aligned word in  $E$ :  $e_{a_j}$

# Computing $P(F | E)$ in IBM Model 1

- Assume some length distribution  $P(J | E)$
- Assume all alignments are equally likely. Since there are  $(I + 1)^J$  possible alignments:

$$P(A | E) = P(A | E, J)P(J | E) = \frac{P(J | E)}{(I + 1)^J}$$

- Assume  $t(f_x, e_y)$  is the prob of translating  $e_y$  as  $f_x$

$$P(F | E, A) = \prod_{j=1}^J t(f_j, e_{a_j})$$

- Determine  $P(F | E)$  by summing over all alignments:

$$P(F | E) = \sum_A P(F | E, A)P(A | E) = \sum_A \frac{P(J | E)}{(I + 1)^J} \prod_{j=1}^J t(f_j, e_{a_j})$$

# Decoding Alignment for IBM Model 1

- Goal is to find the most probable alignment given a parameterized model.

$$\begin{aligned}\hat{A} &= \operatorname{argmax}_A P(F, A | E) \\ &= \operatorname{argmax}_A \frac{P(J | E)}{(I + 1)^J} \prod_{j=1}^J t(f_j, e_{a_j}) \\ &= \operatorname{argmax}_A \prod_{j=1}^J t(f_j, e_{a_j})\end{aligned}$$

Since translation choice for each position  $j$  is independent, the product is maximized by maximizing each term:

$$a_j = \operatorname{argmax}_{0 \leq i \leq I} t(f_j, e_i) \quad 1 \leq j \leq J$$

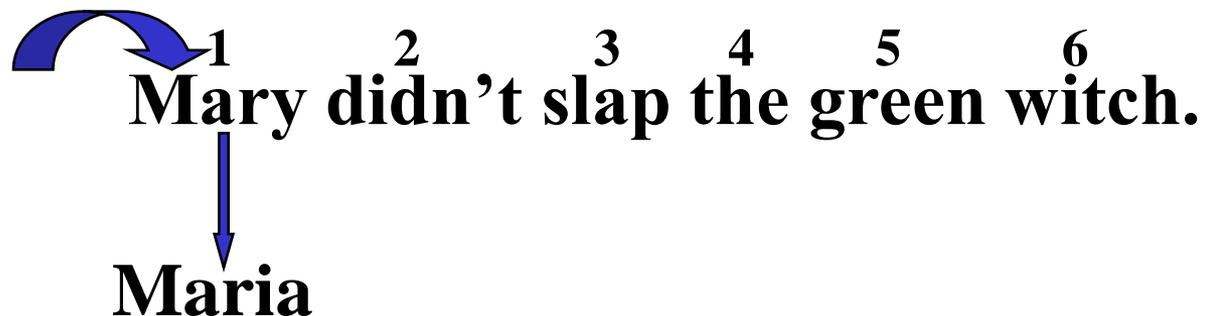
# HMM-Based Word Alignment

- IBM Model 1 assumes all alignments are equally likely and does not take into account **locality**:
  - If two words appear together in one language, then their translations are likely to appear together in the result in the other language.
- An alternative model of word alignment based on an HMM model **does** account for locality by making longer jumps in switching from translating one word to another less likely.

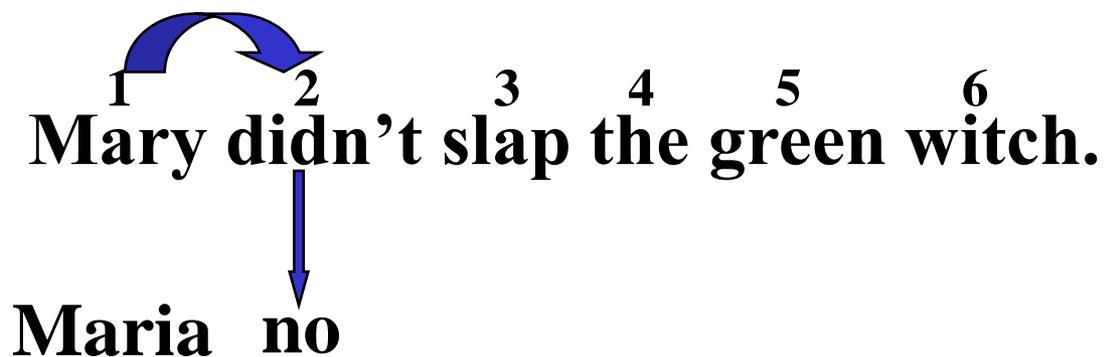
# HMM Model

- Assumes the hidden state is the specific word occurrence  $e_i$  in  $E$  currently being translated (i.e. there are  $I$  states, one for each word in  $E$ ).
- Assumes the observations from these hidden states are the possible translations  $f_j$  of  $e_i$ .
- Generation of  $F$  from  $E$  then consists of moving to the initial  $E$  word to be translated, generating a translation, moving to the next word to be translated, and so on.

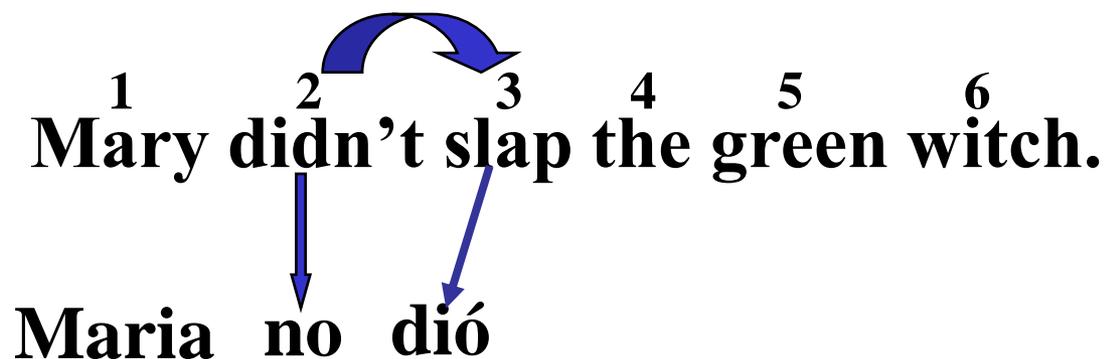
# Sample HMM Generation



# Sample HMM Generation



# Sample HMM Generation



# Sample HMM Generation

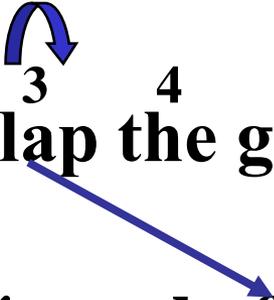
<sup>1</sup> Mary <sup>2</sup> didn't <sup>3</sup> slap <sup>4</sup> the <sup>5</sup> green <sup>6</sup> witch.

Maria no dió una



# Sample HMM Generation

<sup>1</sup> Mary <sup>2</sup> didn't <sup>3</sup> slap <sup>4</sup> the <sup>5</sup> green <sup>6</sup> witch.

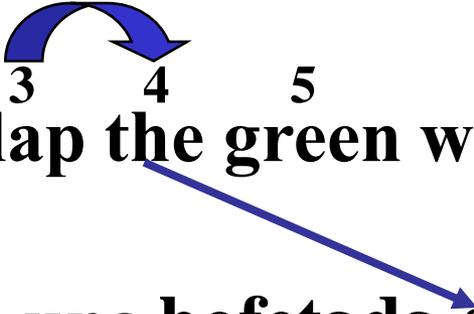


Maria no dió una bofetada

# Sample HMM Generation

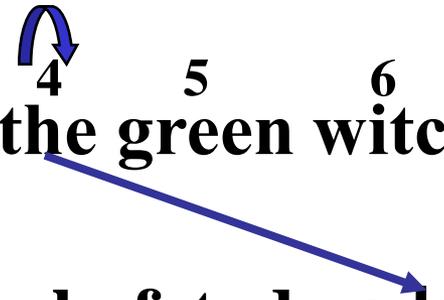
<sup>1</sup> Mary <sup>2</sup> didn't <sup>3</sup> slap <sup>4</sup> the <sup>5</sup> green <sup>6</sup> witch.

Maria no dió una bofetada a



# Sample HMM Generation

<sup>1</sup> Mary <sup>2</sup> didn't <sup>3</sup> slap <sup>4</sup> the <sup>5</sup> green <sup>6</sup> witch.



Maria no dió una bofetada a la

# Sample HMM Generation

1 2 3 4 5 6  
**Mary didn't slap the green witch.**

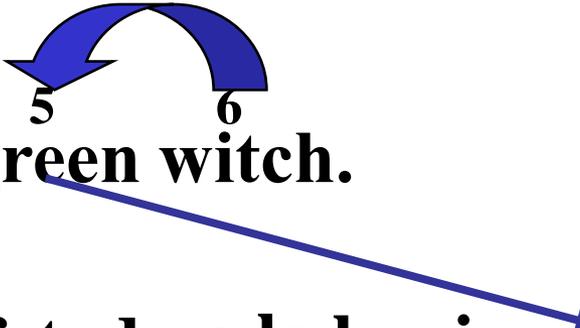


**Maria no dió una bofetada a la bruja**



# Sample HMM Generation

<sup>1</sup> Mary <sup>2</sup> didn't <sup>3</sup> slap <sup>4</sup> the <sup>5</sup> green <sup>6</sup> witch.



**Maria no dió una bofetada a la bruja verde.**

# Sample HMM Generation

<sup>1</sup> Mary <sup>2</sup> didn't <sup>3</sup> slap <sup>4</sup> the <sup>5</sup> green <sup>6</sup> witch.

**Maria no dió una bofetada a la bruja verde.**

# HMM Parameters

- Transition and observation parameters of states for HMMs for all possible source sentences are “tied” to reduce the number of free parameters that have to be estimated.
- **Observation probabilities:**  $b_j(f_i) = P(f_i | e_j)$  the same for all states representing an occurrence of the same English word.
- **State transition probabilities:**  $a_{ij} = s(j-i)$  the same for all transitions that involve the same **jump width** (and direction).

# Computing $P(F | E)$ in the HMM Model

- Given the observation and state-transition probabilities,  $P(F | E)$  (observation likelihood) can be computed using the standard *forward algorithm* for HMMs.

# Decoding for the HMM Model

- Use the standard ***Viterbi algorithm*** to efficiently compute the most likely alignment (i.e. most likely state sequence).

# Training Word Alignment Models

- Both the IBM model 1 and HMM model can be trained on a parallel corpus to set the required parameters.
- For supervised (hand-aligned) training data, parameters can be estimated directly using frequency counts.
- For unsupervised training data, EM can be used to estimate parameters, e.g. Baum-Welch for the HMM model.

# Sketch of EM Algorithm for Word Alignment

Randomly set model parameters.

(making sure they represent legal distributions)

Until converge (i.e. parameters no longer change) do:

E Step: Compute the probability of all possible alignments of the training data using the current model.

M Step: Use these alignment probability estimates to re-estimate values for all of the parameters.

**Note: Use dynamic programming (as in Baum-Welch) to avoid explicitly enumerating all possible alignments**

# Sample EM Trace for Alignment

(IBM Model 1 with no NULL Generation)

**Training Corpus**      **green house**                      **the house**  
                                  **casa verde**                                      **la casa**

	<b>verde</b>	<b>casa</b>	<b>la</b>
<b>green</b>	1/3	1/3	1/3
<b>house</b>	1/3	1/3	1/3
<b>the</b>	1/3	1/3	1/3

**Assume uniform initial probabilities**

**Translation Probabilities**

**Compute Alignment Probabilities**

<b>green house</b>	<b>green house</b>	<b>the house</b>	<b>the house</b>
<b>casa verde</b>	<del><b>casa verde</b></del>	<b>la casa</b>	<del><b>la casa</b></del>
$1/3 \times 1/3 = 1/9$	$1/3 \times 1/3 = 1/9$	$1/3 \times 1/3 = 1/9$	$1/3 \times 1/3 = 1/9$

**Normalize to get P(A | F, E)**

$\frac{1/9}{2/9} = \frac{1}{2}$	$\frac{1/9}{2/9} = \frac{1}{2}$	$\frac{1/9}{2/9} = \frac{1}{2}$	$\frac{1/9}{2/9} = \frac{1}{2}$
---------------------------------	---------------------------------	---------------------------------	---------------------------------

# Example cont.

green house  
 casa verde  
 $1/2$

~~green house~~  
~~casa verde~~  
 $1/2$

the house  
 la casa  
 $1/2$

~~the house~~  
~~la casa~~  
 $1/2$

Compute  
 weighted  
 translation  
 counts

	verde	casa	la
green	$1/2$	$1/2$	0
house	$1/2$	$1/2 + 1/2$	$1/2$
the	0	$1/2$	$1/2$

Normalize  
 rows to sum  
 to one to  
 estimate  $P(f | e)$

	verde	casa	la
green	$1/2$	$1/2$	0
house	$1/4$	$1/2$	$1/4$
the	0	$1/2$	$1/2$

# Example cont.

		verde	casa	la
Translation Probabilities	green	1/2	1/2	0
	house	1/4	1/2	1/4
	the	0	1/2	1/2

Recompute  
Alignment  
Probabilities  
 $P(A, F | E)$

green house  
casa verde  
 $1/2 \times 1/4 = 1/8$

~~green house~~  
~~casa verde~~  
 $1/2 \times 1/2 = 1/4$

~~the house~~  
~~la casa~~  
 $1/2 \times 1/2 = 1/4$

~~the house~~  
~~la casa~~  
 $1/2 \times 1/4 = 1/8$

Normalize  
to get  
 $P(A | F, E)$

$$\frac{1/8}{3/8} = \frac{1}{3}$$

$$\frac{1/4}{3/8} = \frac{2}{3}$$

$$\frac{1/4}{3/8} = \frac{2}{3}$$

$$\frac{1/8}{3/8} = \frac{1}{3}$$

Continue EM iterations until translation  
parameters converge

# Phrase Alignments from Word Alignments

- Phrase-based approaches to MT have been shown to be better than word-based models.
- However, alignment algorithms (IBM Model 1 or HMM Aligner) produce one to many word translations rather than many to many phrase translations.
- Combine  $E \rightarrow F$  and  $F \rightarrow E$  word alignments to produce a phrase alignment.
  - ➔ "Symmetrization technique"

# Phrase Alignment via “Symmetrization”

## Spanish to English (using HMM Alignment Model)

	Maria	no	dio	una	bofetada	a	la	bruja	verde
Mary	XXXX								
did		XX							
not		XX							
slap					XXXXXX				
the							XX		
green									XXXX
witch								XXXXX	

# Phrase Alignment via “Symmetrization”

## English to Spanish (using HMM Alignment Model)

	Maria	no	dio	una	bofetada	a	la	bruja	verde
Mary	XXXX								
did						XX			
not		XX							
slap			XXX	XXX	XXXXXXX				
the							XX		
green									XXXX
witch								XXXXX	

# Phrase Alignment via “Symmetrization”

**Intersection of previous two alignments  
(high precision word-to-word alignment)**

	Maria	no	dio	una	bofetada	a	la	bruja	verde
Mary	XXXX								
did									
not		XX							
slap					XXXXXXX				
the							XX		
green									XXXX
witch								XXXXXX	

# Phrase Alignment via “Symmetrization”

**Phrase alignments are obtained by expanding intersection to union** (with certain rules or classifiers)

	Maria	no	dio	una	bofetada	a	la	bruja	verde
Mary	XXXX								
did		XX							
not		XX							
slap			XXX	XXX	XXXXXXX				
the						XX	XX		
green									XXXX
witch								XXXXXX	

# Decoding

- Goal is to find a translation that maximizes the product of the translation and language models.

$$\operatorname{argmax}_{E \in \text{English}} P(F | E)P(E)$$

- Cannot explicitly enumerate and test the combinatorial space of all possible translations.
- Must efficiently (heuristically) search the space of translations that approximates the solution to this difficult optimization problem.
- The optimal decoding problem for all reasonable model's (e.g. IBM model 1) is NP-complete.

# Space of Translations

- The phrase translation table from phrase alignments defines a space of all possible translations.
- Why is this NP-hard?

<b>Maria</b>	<b>no</b>	<b>dio</b>	<b>una</b>	<b>bofetada</b>	<b>a</b>	<b>la</b>	<b>bruja</b>	<b>verde</b>
<u>Mary</u>	<u>not</u>	<u>give</u>	<u>a</u>	<u>slap</u>	<u>to</u>	<u>the</u>	<u>witch</u>	<u>green</u>
	<u>did not</u>		<u>a slap</u>		<u>to the</u>		<u>green witch</u>	
	<u>no</u>		<u>slap</u>		<u>to</u>			
	<u>did not give</u>				<u>the</u>			
			<u>slap</u>			<u>the witch</u>		

# Software

- [Giza++](#) a training tool for IBM Model 1-5 ([version for gcc-4](#))
- [Moses](#), a complete SMT system
- [Pharaoh](#) a decoder for phrase-based SMT
- [Rewrite](#) a decoder for IBM Model 4
- [BLEU scoring tool](#) for machine translation evaluation

# Stack Decoding

- Use a version of heuristic A\* search to explore the space of phrase translations to find the best scoring subset that covers the source sentence.

**Initialize priority queue  $Q$  (stack) to empty translation.**

**Loop:**

**$s = \text{pop}(Q)$**

**If  $h$  is a complete translation, exit loop and return it.**

**For each refinement  $s'$  of  $s$  created by adding a phrase translation**

**Compute score  $f(s')$**

**Add  $s'$  to  $Q$**

**Sort  $Q$  by score  $f$**

# Search Heuristic

- A\* is best-first search using the function  $f$  to sort the search queue:
  - $f(s) = g(s) + h(s)$
  - $g(s)$ : Cost of existing partial solution
  - $h(s)$ : Estimated cost of completion of solution
- If  $h(s)$  is an underestimate of the true remaining cost (*admissible heuristic*) then A\* is guaranteed to return an optimal solution.

## Current Cost: $g(s)$

- Known quality of partial translation,  $E$ , composed of a set of chosen phrase translations  $S$  based on phrase translation and language models.

$$g(s) = \log \frac{1}{\left( \prod_{i \in S} \phi(\bar{f}_i, \bar{e}_i) d(i) \right) P(E)}$$

## Estimated Future Cost: $h(s)$

- True future cost requires knowing the way of translating the remainder of the sentence in a way that maximizes the probability of the final translation.
- However, this is not computationally tractable.
- Therefore under-estimate the cost of remaining translation by ignoring the distortion component and computing the most probable remaining translation ignoring distortion (which is efficiently computable using the Viterbi algorithm)

# Beam Search

- However,  $Q$  grows too large to be efficient and guarantee an optimal result with full  $A^*$  search.
- Therefore, always cut  $Q$  back to only the best (lowest cost)  $K$  items to approximate the best translation

**Initialize priority queue  $Q$  (stack) to empty translation.**

**Loop:**

**If top item on  $Q$  is a complete translation, exit loop and return it.**

**For each element  $s$  of  $Q$  do**

**For each refinement  $s'$  of  $s$  created by adding a phrase translation**

**Compute score  $f(s')$**

**Add  $s'$  to  $Q$**

**Sort  $Q$  by score  $f$**

**Prune  $Q$  back to only the first (lowest cost)  $K$  items**

# Multistack Decoding

- It is difficult to compare translations that cover different fractions of the foreign sentence, so maintain multiple priority queues (stacks), one for each number of foreign words currently translated.
- Finally, return best scoring translation in the queue of translations that cover all of the words in  $F$ .

# Evaluating MT

- Human subjective evaluation is the best but is time-consuming and expensive.
- Automated evaluation comparing the output to multiple human reference translations is cheaper and correlates with human judgements.

# Human Evaluation of MT

- Ask humans to estimate MT output on several dimensions.
  - **Fluency**: Is the result grammatical, understandable, and readable in the target language.
  - **Fidelity**: Does the result correctly convey the information in the original source language.
    - **Adequacy**: Human judgment on a fixed scale.
      - Bilingual judges given source and target language.
      - Monolingual judges given reference translation and MT result.
    - **Informativeness**: Monolingual judges must answer questions about the source sentence given only the MT translation (task-based evaluation).

# Computer-Aided Translation Evaluation

- **Edit cost:** Measure the number of changes that a human translator must make to correct the MT output.
  - Number of words changed
  - Amount of time taken to edit
  - Number of keystrokes needed to edit

# Automatic Evaluation of MT

- Collect one or more human *reference translations* of the source.
- Compare MT output to these reference translations.
- Score result based on similarity to the reference translations.
  - BLEU
  - NIST
  - TER
  - METEOR

# BLEU

- Determine number of  $n$ -grams of various sizes that the MT output shares with the reference translations.
- Compute a modified precision measure of the  $n$ -grams in MT result.

# BLEU Example

Cand 1: **Mary** no **slap** **the** **witch** **green**

Cand 2: Mary did not give a smack to a green witch.

Ref 1: **Mary** did not **slap** **the** **green** **witch**.

Ref 2: **Mary** did not smack **the** **green** **witch**.

Ref 3: **Mary** did not hit a **green** sorceress.

**Cand 1 Unigram Precision: 5/6**

# BLEU Example

Cand 1: Mary no slap the witch green.

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

**Cand 1 Bigram Precision: 1/5**

# BLEU Example

How about: **Mary** **Mary** **Mary** **Mary** **Mary** **Mary.**

Ref 1: **Mary** did not slap the green witch.

Ref 2: **Mary** did not smack the green witch.

Ref 3: **Mary** did not hit a green sorceress.

**Unigram Precision: 6/6 ???**

# BLEU Example

How about: **Mary** Mary Mary Mary Mary Mary.

Ref 1: **Mary** did not slap the green witch.

Ref 2: **Mary** did not smack the green witch.

Ref 3: **Mary** did not hit a green sorceress.

**Clip match count of each  $n$ -gram to maximum count of the  $n$ -gram in any single reference translation**

**Unigram Precision: 1/6**

# BLEU Example

Cand 1: Mary no slap the witch green.

Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.

Ref 2: Mary did not smack the green witch.

Ref 3: Mary did not hit a green sorceress.

Clip match count of each  $n$ -gram to maximum count of the  $n$ -gram in any single reference translation

**Cand 2 Unigram Precision: 7/10**

# BLEU Example

**Cand 1:** Mary no slap the witch green.

**Cand 2:** Mary did not give a smack to a green witch.

**Ref 1:** Mary did not slap the green witch.

**Ref 2:** Mary did not smack the green witch.

**Ref 3:** Mary did not hit a green sorceress.

**Cand 2 Bigram Precision: 4/9**

# Modified $N$ -Gram Precision

- Average  $n$ -gram precision over all  $n$ -grams up to size  $N$  (typically 4) using geometric mean.

$$p_n = \frac{\sum_{C \in \text{corpus}} \sum_{n\text{-gram} \in C} \text{count}_{\text{clip}}(n\text{-gram})}{\sum_{C \in \text{corpus}} \sum_{n\text{-gram} \in C} \text{count}(n\text{-gram})}$$

$$p = \sqrt[N]{\prod_{n=1}^N p_n}$$

**Cand 1:**  $p = \sqrt[2]{\frac{5}{6} \frac{1}{5}} = 0.408$

**Cand 2:**  $p = \sqrt[2]{\frac{7}{10} \frac{4}{9}} = 0.558$

# BLEU is *roughly* Precision

- Why not n-gram Recall?
- What is the problem with computing Recall?
- What is the problem of ***not*** computing Recall?

# Brevity Penalty

- Not easy to compute recall to complement precision since there are multiple alternative gold-standard references and don't need to match all of them.
- Instead, use a penalty for translations that are shorter than the reference translations.
- Define effective reference length,  $r$ , for each sentence as the length of the reference sentence with the largest number of  $n$ -gram matches. Let  $c$  be the candidate sentence length.

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

# BLEU Score

- Final BLEU Score:  $BLEU = BP \times \text{avg-ngram-prec}$

**Cand 1:** Mary no slap the witch green.

**Best Ref:** Mary did not slap the green witch.

$$c = 6, \quad r = 7, \quad BP = e^{(1-7/6)} = 0.846$$

$$BLEU = 0.846 \times 0.408 = 0.345$$

**Cand 2:** Mary did not give a smack to a green witch.

**Best Ref:** Mary did not smack the green witch.

$$c = 10, \quad r = 7, \quad BP = 1$$

$$BLEU = 1 \times 0.558 = 0.558$$

# BLEU Score Issues

- BLEU has been shown to correlate with human evaluation when comparing outputs from different SMT systems.
- However, it does not correlate with human judgments when comparing SMT systems with manually developed MT (Systran) or MT with human translations.
- Other MT evaluation metrics have been proposed that claim to overcome some of the limitations of BLEU.

# Syntax-Based Statistical Machine Translation

- Recent SMT methods have adopted a syntactic transfer approach.
- Improved results demonstrated for translating between more distant language pairs, e.g. Chinese/English.

# Synchronous Grammar

- **Multiple parse trees** in a single derivation.
- Used by (Chiang, 2005; Galley et al., 2006).
- Describes the **hierarchical structures** of a sentence and its translation, and also the **correspondence** between their sub-parts.

# Synchronous Productions

- Has two RHSs, one for each language.

*Chinese:*                      *English:*  
**X** → **X** 是甚麼 / **What is X**

# Syntax-Based MT Example

**Input:** 俄亥俄州的首府是甚麼？

# Syntax-Based MT Example



**Input:** 俄亥俄州的首府是甚麼？

# Syntax-Based MT Example



**Input:** 俄亥俄州的首府是甚麼？

**X** → **X** 是甚麼 / What is **X**

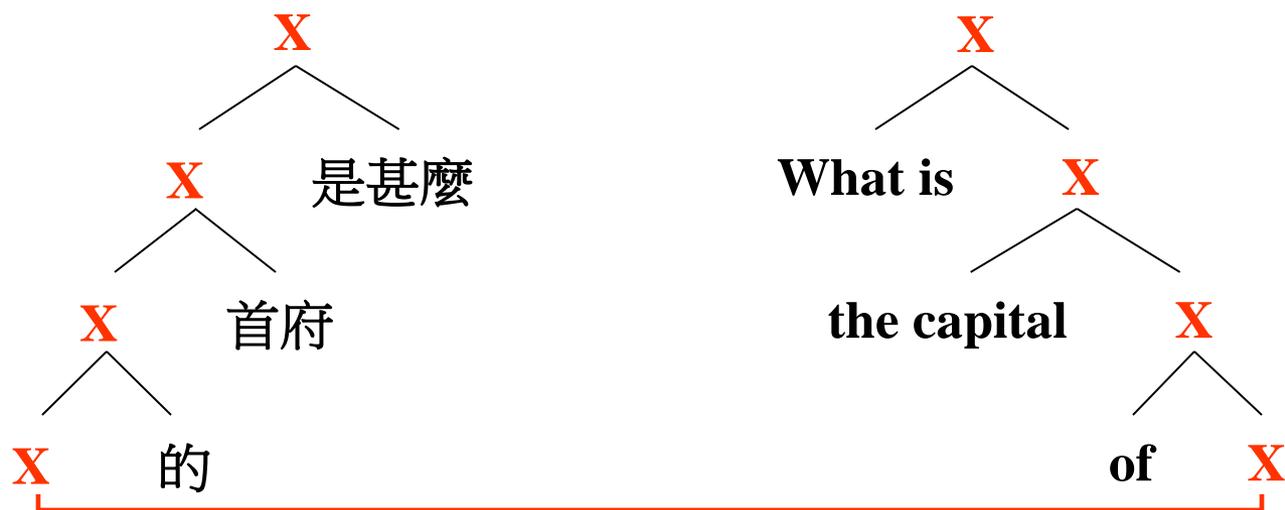
# Syntax-Based MT Example



**Input:** 俄亥俄州的首府是甚麼？

**X** → **X** 首府 / the capital **X**

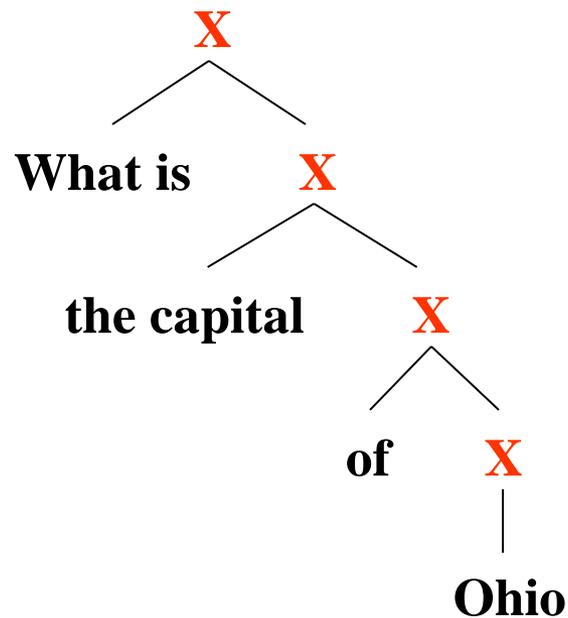
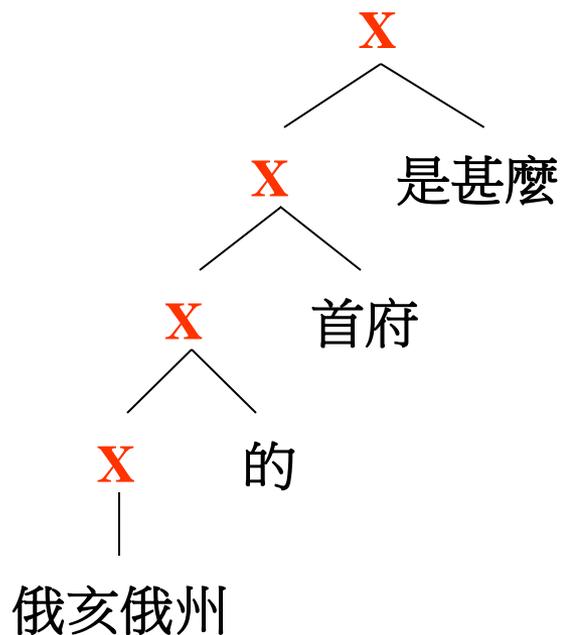
# Syntax-Based MT Example



Input: 俄亥俄州的首府是甚麼？

$X \rightarrow X \text{ 的 } / \text{ of } X$

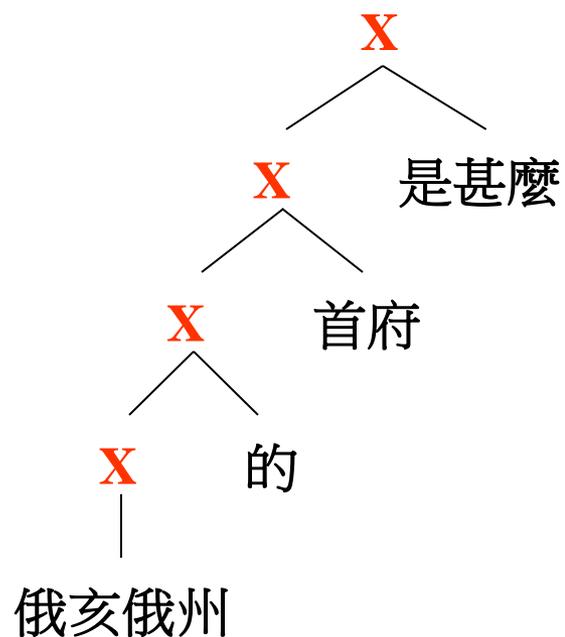
# Syntax-Based MT Example



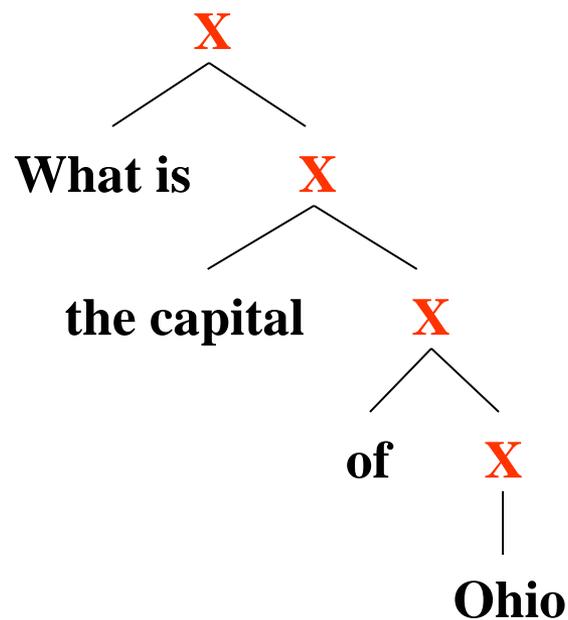
**Input:** 俄亥俄州的首府是甚麼？

**X** → 俄亥俄州 / Ohio

# Syntax-Based MT Example



**Input:** 俄亥俄州的首府是甚麼？



**Output:** What is the capital of Ohio?

# Synchronous Derivations and Translation Model

- Need to make a probabilistic version of synchronous grammars to create a translation model for  $P(F | E)$ .
- Each synchronous production rule is given a weight  $\lambda_j$ , that is used in a maximum-entropy (log linear) model.
- Parameters are learned to maximize the conditional log-likelihood of the training data.

$$\lambda^* = \arg \max_{\lambda} \sum_j \log \Pr_{\lambda}(\mathbf{f}_j | \mathbf{e}_j)$$

# Log-Linear Models for MT

- Noisy channel model takes into account just two factors:
  - translation model  $P(F | E)$
  - language model  $P(E)$
- A max-ent (log-linear) model can incorporate arbitrary other factors/features:
  - Language model:  $P(E)$
  - Translation mode:  $P(F | E)$
  - Reverse translation model:  $P(E | F)$
  - unknown word penalty, phrase penalty, etc

# Minimum Error Rate Training (**MERT**)

- Noisy channel model is not trained to directly minimize the final MT evaluation metric, e.g. BLEU.
- A max-ent (log-linear) model can be trained by
  - standard maximum entropy training, or these days,
  - minimum error rate training (MERT)
    - “Minimum Error Rate Training in Statistical Machine Translation”, [Franz Josef Och](#), *ACL*, 2003

# Conclusions

- MT methods can usefully exploit various amounts of syntactic and semantic processing along the Vauquois triangle.
- Statistical MT methods can automatically learn a translation system from a parallel corpus.
- Typically use a noisy-channel model to exploit both a bilingual translation model and a monolingual language model.
- Automatic word alignment methods can learn a translation lexicon from a parallel corpus.
- Phrase-based and syntax based SMT methods are currently the state-of-the-art.