CS447: Natural Language Processing

*http://courses.engr.illinois.edu/cs447*

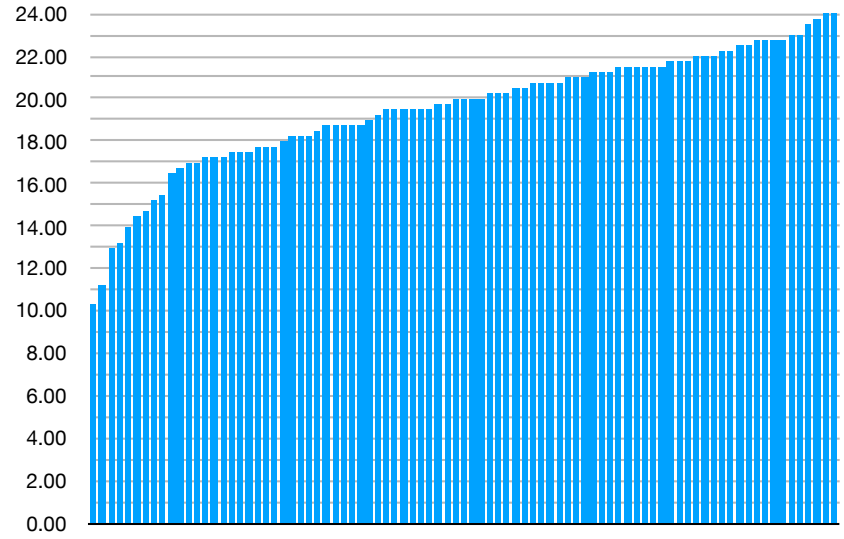
# Lecture 16: Statistical Machine Translation

Julia Hockenmaier

*juliahmr@illinois.edu*

3324 Siebel Center

# Midterm results out on Gradescope




Please check your exam!

— We will accept regrade requests until next Friday.

— Most people did really well (overall Max: 24, Median 20) but many of the top students are grad students

We will set **23** points=100%

(Gradescope's percentages assume 25 points = 100%)

It's difficult to translate exam percentages to letter grades because:

… letter grades will depend on overall performance (incl. MPs)

… we use the undergrads as yardstick, but we don't have that information in Gradescope.

Come and talk to us if you're worried about your results

# MPs and Autograder

Apologies for the confusion and frustration.
Many thanks for your feedback (and patience)!

This is a learning experience for us as well
— we're redesiging several MPs
AND putting them on Gradescope for the first time.

The hope was that the feedback that the autograder provides would be helpful to you…

# Great talk at 2pm today — No office hours today

**Distinguished Lecture** In Computer Science

## Explainable AI: Making Visual Question Answering Systems more Transparent

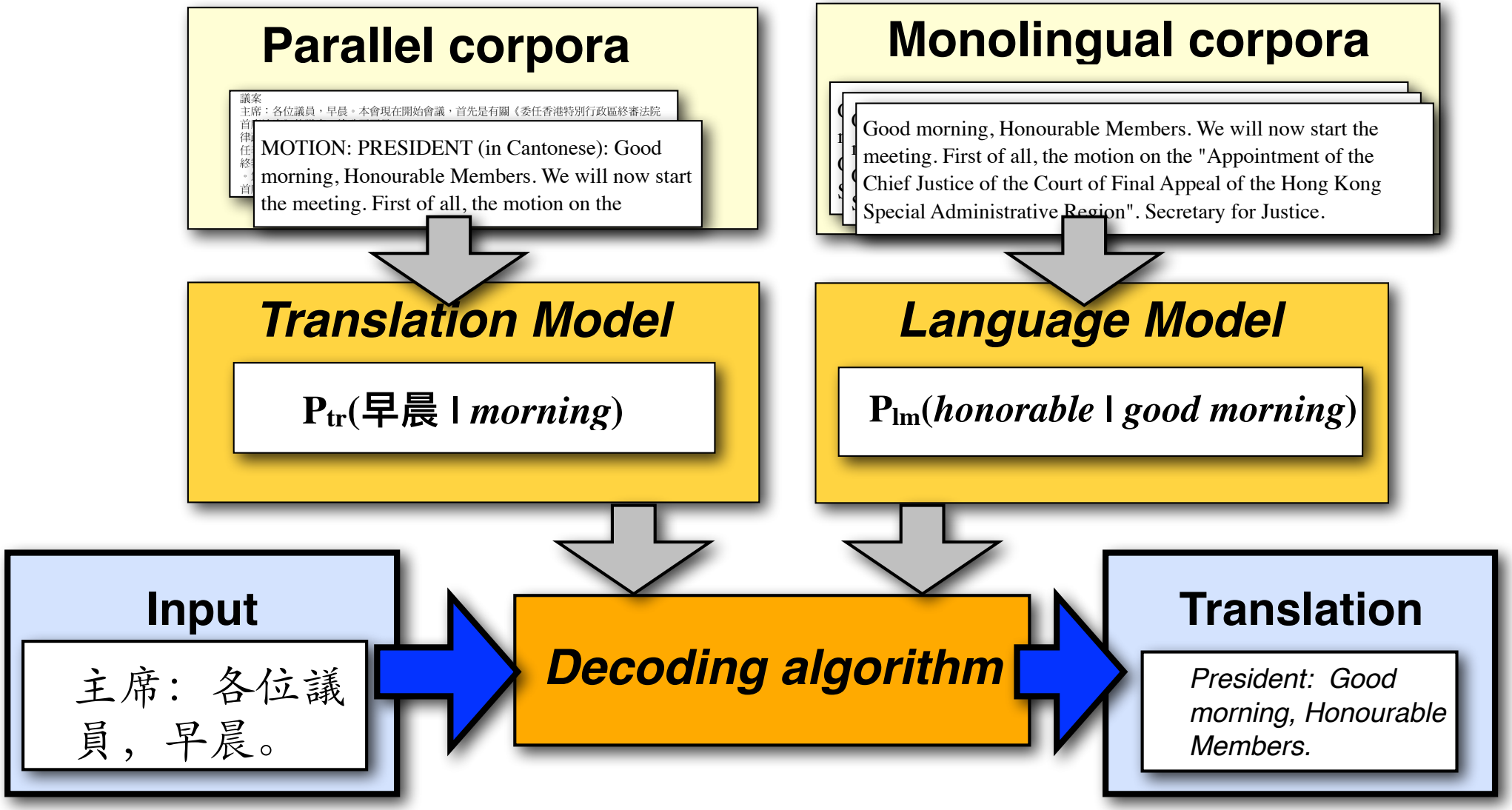
***A Distinguished Lecture Sponsored by the Department of Computer Science***
**Guest Speaker:** Raymond Mooney, Professor, University of Texas at Austin
**Date/Time:** Friday, Oct. 18, 2019, 2:00 pm
**Location:** 2405 Thomas M. Siebel Center for Computer Science

**Abstract:** Artificial Intelligence systems' ability to explain their conclusions is crucial to their utility and trustworthiness. Deep neural networks have enabled significant progress on many challenging problems such as visual question answering (VQA), the task of answering natural language questions about images. However, most of them are opaque black boxes with limited explanatory capability. The goal of Explainable AI is to increase the transparency of complex AI systems such as deep networks. We have developed a novel approach to XAI and used it to build a high-performing VQA system that can elucidate its answers with multi-modal natural-language and visual explanations that faithfully reflect important aspects of its underlying reasoning while capturing the style of comprehensible human explanations. Crowd-sourced human evaluation of these explanations demonstrate the advantages of our approach.

**Bio:** Raymond J. Mooney is a Professor in the Department of Computer Science at the University of Texas at Austin. He received his Ph.D. in 1988 from the University of Illinois at Urbana/Champaign. He is an author of over 170 published research papers, primarily in the areas of machine learning and natural language processing. He was the President of the International Machine Learning Society from 2008-2011, program co-chair for AAAI 2006, general chair for HLT-EMNLP 2005, and co-chair for ICML 1990. He is a Fellow of AAAI, ACM, and ACL and the recipient of the Classic Paper award from AAAI-19 and best paper awards from AAAI-96, KDD-04, ICML-05 and ACL-07.

# Back to the material…

# Statistical MT with the noisy channel model



# IBM models

First statistical MT models, based on noisy channel:

Translate from source $\mathbf{f}$ to target $\mathbf{e}$

via a **translation model** $P(\mathbf{f} \mid \mathbf{e})$ and a **language model** $P(\mathbf{e})$

The translation model goes **from target $\mathbf{e}$ to source $\mathbf{f}$**

via **word alignments $\mathbf{a}$**: $P(\mathbf{f} \mid \mathbf{e}) = \sum_{\mathbf{a}} P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$

Original purpose: Word-based translation models
Today: Can be used to obtain word alignments, which are then used to obtain phrase alignments for phrase-based translation models

Sequence of 5 translation models

Model 1 is too simple to be used by itself,
but can be trained very easily on parallel data.

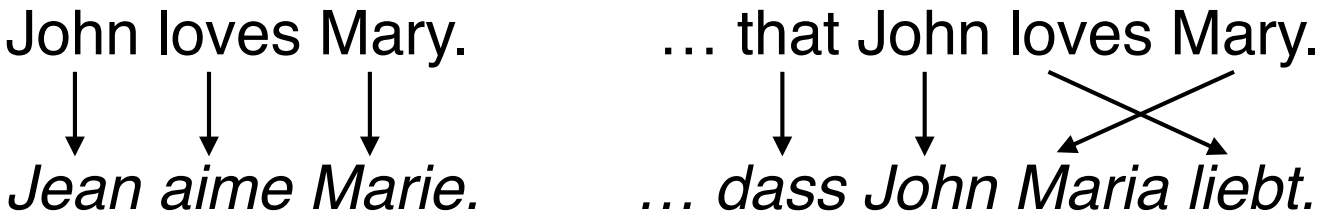
# IBM translation models: assumptions

The model "generates" the 'foreign' source sentence $\mathbf{f}$
conditioned on the 'English' target sentence $\mathbf{e}$
by the following stochastic process:

1. Generate the **length** of the source $\mathbf{f}$
   with probability $p$ = ...
2. Generate the **alignment** of the source $\mathbf{f}$
   to the target $\mathbf{e}$ with probability $p$ = ...
3. Generate the **words** of the source $\mathbf{f}$
   with probability $p$ = ...

# Word alignments in the IBM models

# Word alignment




| | Jean | aime | Marie |
|---|---|---|---|
| John | ■ | | |
| loves | | ■ | |
| Mary | | | ■ |

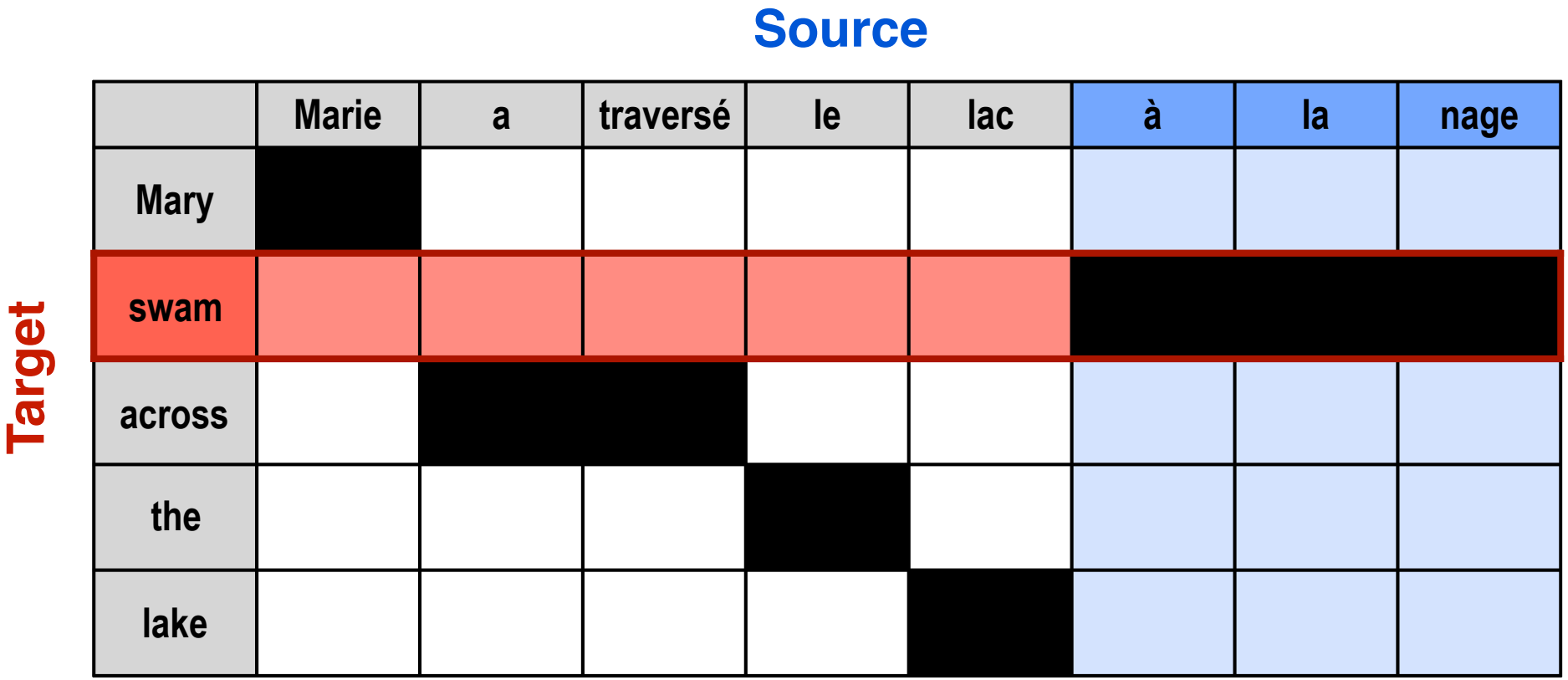
| | dass | John | Maria | liebt |
|---|---|---|---|---|
| that | ■ | | | |
| John | | ■ | | |
| loves | | | | ■ |
| Mary | | | ■ | |

# Word alignment

# Word alignment

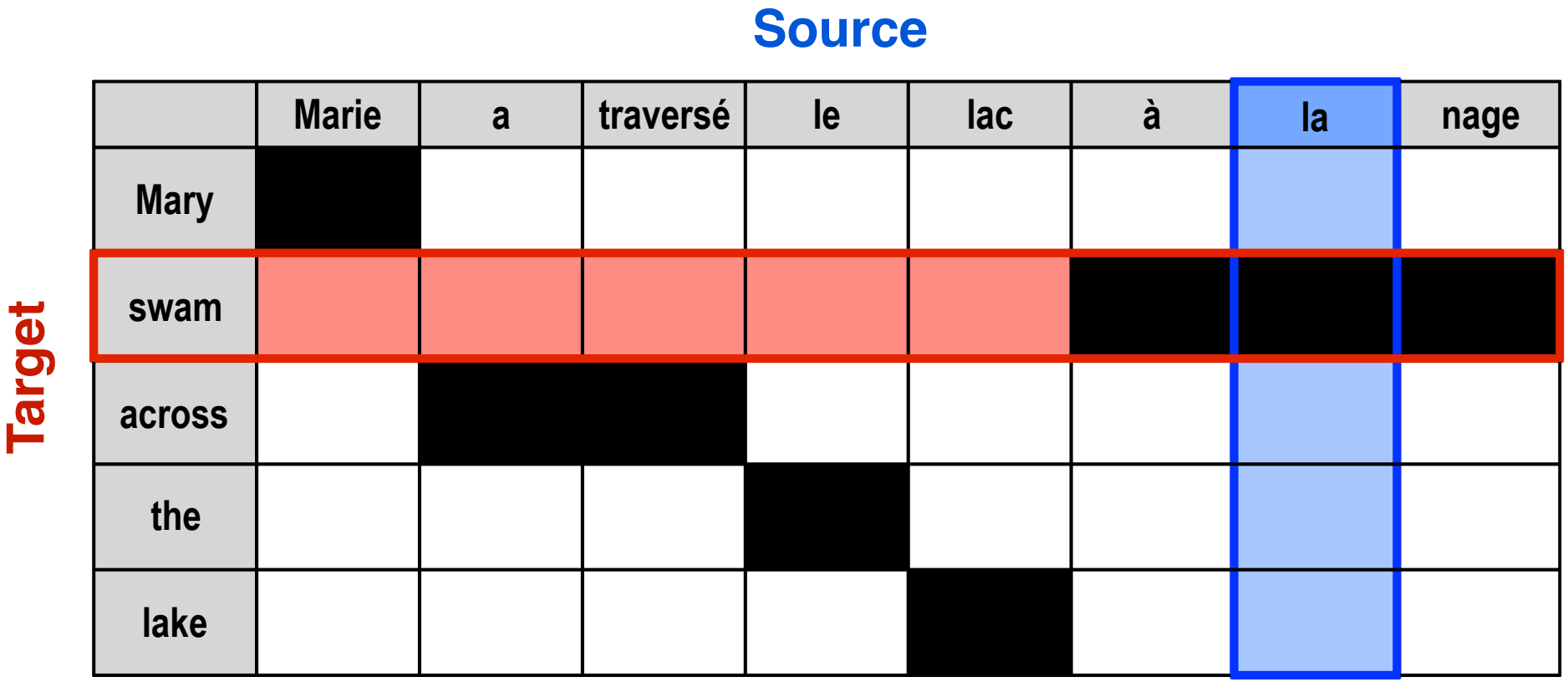
| | Marie | a | traversé | le | lac | à | la | nage |
|---|---|---|---|---|---|---|---|---|
| Mary | ■ | | | | | | | |
| swam | | | | | | ■ | ■ | ■ |
| across | | ■ | ■ | | | | | |
| the | | | | ■ | | | | |
| lake | | | | | ■ | | | |

# Word alignment



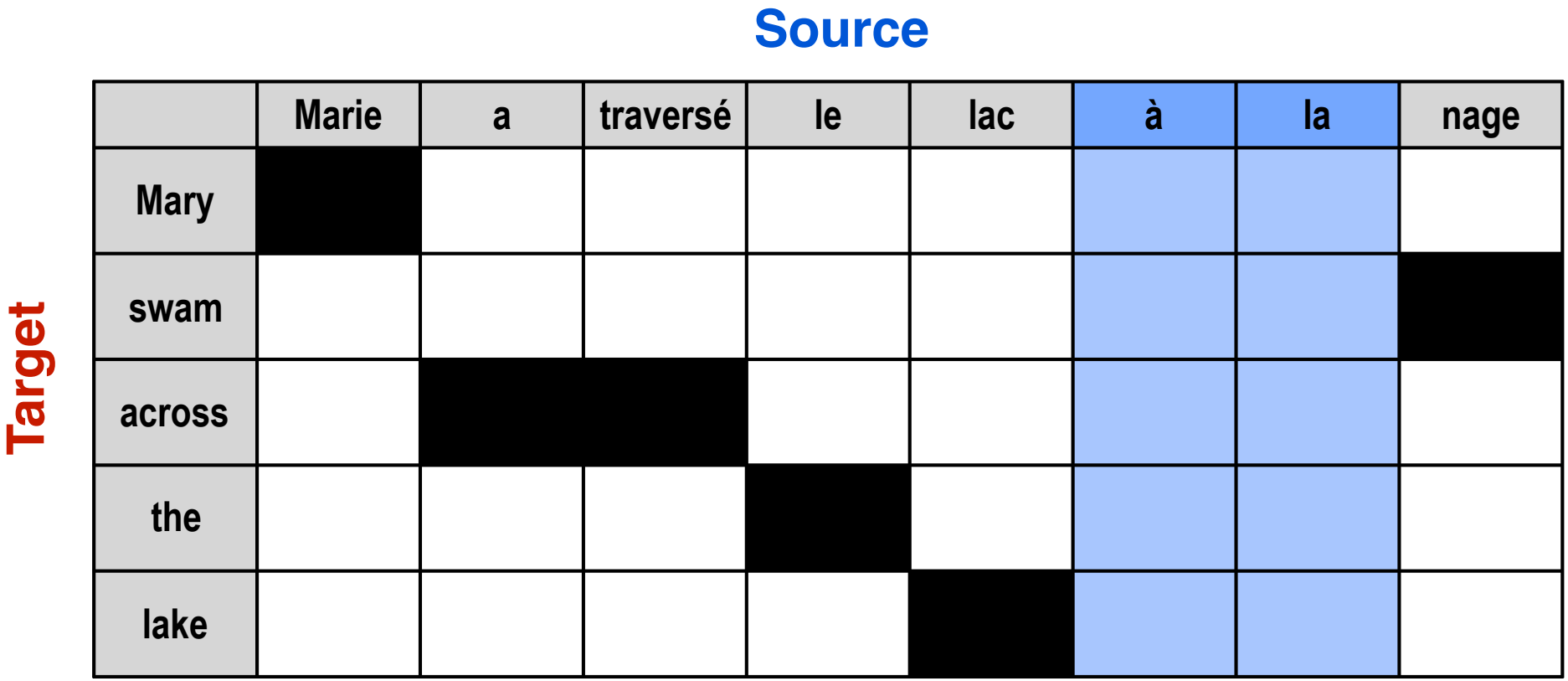

**One target word** can be aligned to **many source words**.

# Word alignment



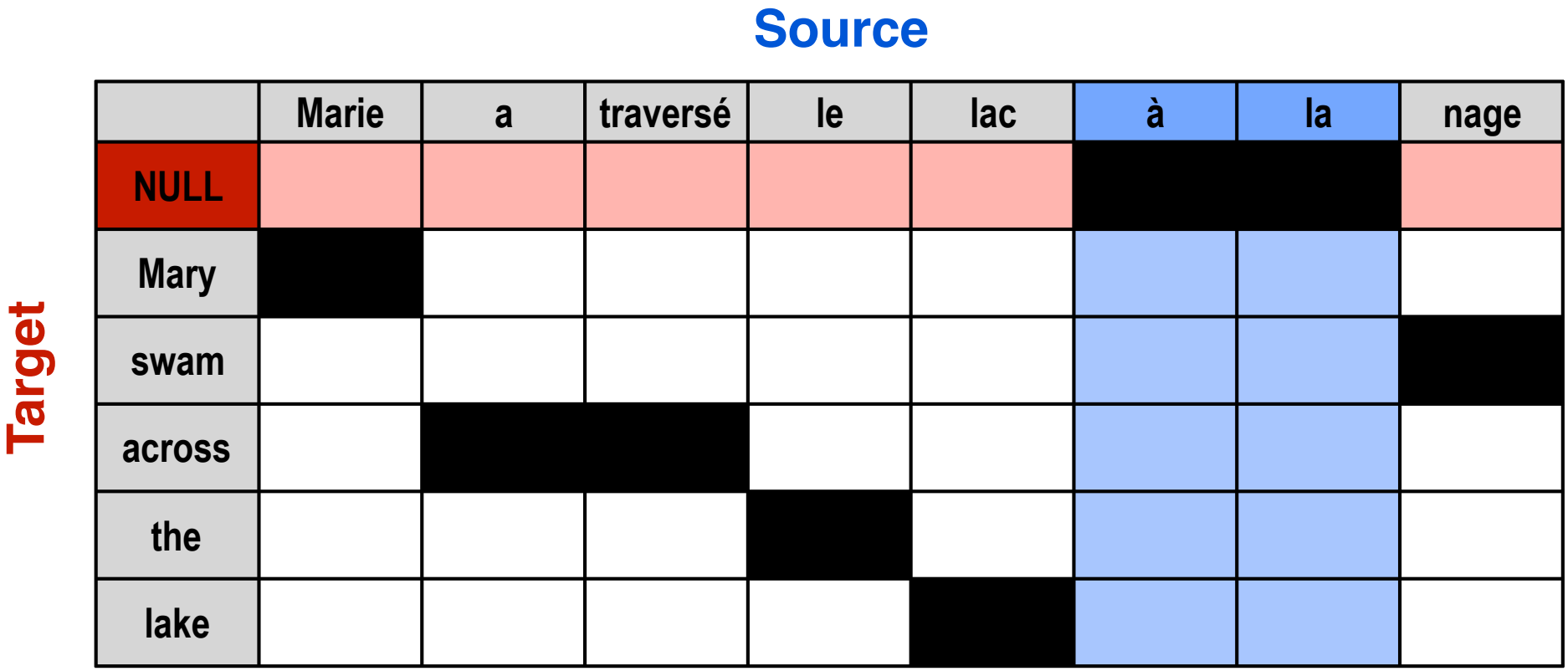

**One target word** can be aligned to **many source words**.
But **each source word** can only be aligned to **one target word.**
This allows us to model ***P*(source | target)**

# Word alignment




**Some source words** may **not align** to ***any*** **target words**.

# Word alignment




**Some source words** may **not align** to ***any* target words**.

To handle this we assume a NULL word in the target sentence.

# Representing word alignments

| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | | Marie | a | traversé | le | lac | à | la | nage |
| 0 | NULL | | | | | | ■ | ■ | |
| 1 | Mary | ■ | | | | | | | |
| 2 | swam | | | | | | | | ■ |
| 3 | across | | ■ | ■ | | | | | |
| 4 | the | | | | ■ | | | | |
| 5 | lake | | | | | ■ | | | |

| Position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Foreign | Marie | a | traversé | le | lac | à | la | nage |
| Alignment | 1 | 3 | 3 | 4 | 5 | 0 | 0 | 2 |

Every source word **f[i]** is aligned to **one** target word **e[j]** (incl. NULL). We represent alignments as a vector **a** (of the same length as the source) with **a[i] = j**

# The IBM alignment models

# The IBM models

Use the noisy channel (Bayes rule) to get the best (most likely) target translation $\mathbf{e}$ for source sentence $\mathbf{f}$:

$$\arg\max_{\mathbf{e}} P(\mathbf{e}|\mathbf{f}) = \arg\max_{\mathbf{e}} P(\mathbf{f}|\mathbf{e})P(\mathbf{e})$$

noisy channel

The translation model $P(\mathbf{f} \mid \mathbf{e})$ requires **alignments** $\mathbf{a}$

$$P(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}\in\mathcal{A}(\mathbf{e},\mathbf{f})} P(\mathbf{f},\mathbf{a}|\mathbf{e})$$

marginalize (=sum) over all alignments a

Generate $\mathbf{f}$ and the alignment $\mathbf{a}$ with $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$:

$$P(\mathbf{f},\mathbf{a}|\mathbf{e}) = \underbrace{P(m|\mathbf{e})}_{\text{Length: } |\mathbf{f}|=\mathrm{m}} \prod_{j=1}^{m} \underbrace{P(a_j|a_{1..j-1}, f_{1..j-1}, m, \mathbf{e})}_{\text{Word alignment } a_j} \underbrace{P(f_j|a_{1..j} f_{1..j-1}, \mathbf{e}, m)}_{\text{Translation } f_j}$$

m = #words in fj

probability of alignment aj

probability of word fj

# IBM model 1: Generative process

For each target sentence $\mathbf{e} = e_1..e_n$ of length $n$:

| 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| NULL | **Mary** | **swam** | **across** | **the** | **lake** |

1. **Choose a length** $m$ for the source sentence (e.g $m = 8$)

| Position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|

2. **Choose an alignment** $\mathbf{a} = a_1...a_m$ for the source sentence

Each $a_j$ corresponds to a word $e_i$ in $\mathbf{e}$: $0 \leq a_j \leq n$

| Position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Alignment | 1 | 3 | 3 | 4 | 5 | 0 | 0 | 2 |

3. **Translate each target word** $e_{aj}$ into the source language

| Position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Alignment | 1 | 3 | 3 | 4 | 5 | 0 | 0 | 2 |
| Translation | **Marie** | **a** | **traversé** | **le** | **lac** | **à** | **la** | **nage** |

# Model parameters

Length probability $P(m \mid n)$:

What's the probability of generating a source sentence of length $m$ given a target sentence of length $n$?
Count in training data, or use a constant

Alignment probability: $\mathrm{P}(\mathbf{a} \mid m, n)$:

Model 1 assumes all alignments have the same probability:
For each position $\mathrm{a}_1 \ldots \mathrm{a}_m$, pick one of the $n+1$ target positions uniformly at random

Translation probability: $\mathrm{P}(\mathrm{f_j} = lac \mid \mathrm{a_j} = \mathrm{i}, \mathrm{e_i} = lake)$:

In Model 1, these are the only parameters we have to learn.

# IBM model 1: details

The **length probability** is constant: $P(m \mid \mathrm{e}) = \varepsilon$

The **alignment probability** is uniform ($n$ = length of target string): $P(a_i \mid \mathrm{e}) = 1/(n+1)$

The **translation probability** depends only on $e_{ai}$ (the corresponding target word): $P(f_i \mid \mathrm{e}_{\mathrm{ai}})$

$$P(\mathbf{f}, \mathbf{a}|\mathbf{e}) = \underbrace{P(m|\mathbf{e})}_{\text{Length: } |\mathbf{f}|=\mathrm{m}} \prod_{j=1}^{m} \underbrace{P(a_j|a_{1..j-1}, f_{1..j-1}, m, \mathbf{e})}_{\text{Word alignment } a_j} \underbrace{P(f_j|a_{1..j}f_{1..j-1}, \mathbf{e}, m)}_{\text{Translation } f_j}$$

$$= \epsilon \prod_{j=1}^{m} \frac{1}{n+1} P(f_j|e_{a_j})$$

All alignments have the same probability

Translation depends only on the aligned English word

$$\frac{\epsilon}{(n+1)^m} \prod_{j=1}^{m} P(f_j|e_{a_j})$$

# Finding the best alignment

How do we find the **best alignment** between $\mathbf{e}$ and $\mathbf{f}$?

$$\begin{aligned}
\hat{\mathbf{a}} &= \arg\max_{\mathbf{a}} P(\mathbf{f}, \mathbf{a}|\mathbf{e}) \\
&= \arg\max_{\mathbf{a}} \frac{\epsilon}{(n+1)^m} \prod_{j=1}^{m} P(f_j|e_{a_j}) \\
&= \arg\max_{\mathbf{a}} \prod_{j=1}^{m} P(f_j|e_{a_j})
\end{aligned}$$

$$\hat{a}_j = \arg\max_{a_j} P(f_j|e_{a_j})$$

# Learning translation probabilities

The only parameters that need to be learned are the **translation probabilities** $P(\mathrm{f} \mid \mathrm{e})$

$$\mathrm{P}(\mathrm{f_j} = lac \mid \mathrm{e_i} = lake)$$

If the training corpus had word alignments, we could simply count how often 'lake' is aligned to 'lac':

$$\mathrm{P}(lac \mid lake) = \mathrm{count}(lac, lake) / \sum_w \mathrm{count}(w, lake)$$

But we don't have gold word alignments.
So, instead of relative frequencies, we have to use *expected* relative frequencies:

$$\mathrm{P}(lac \mid lake) = \langle \mathrm{count}(lac, lake) \rangle / \langle \sum_w \mathrm{count}(w, lake) \rangle$$

# Training Model 1 with EM

The only parameters that need to be learned are the **translation probabilities** $P(\mathrm{f} \mid \mathrm{e})$

We use the **EM algorithm** to estimate these parameters from a corpus with $S$ sentence pairs $s = \langle \boldsymbol{f}^{(s)}, \boldsymbol{e}^{(s)} \rangle$ with alignments $\mathcal{A}(\mathbf{f}^{(s)}, \mathbf{e}^{(s)})$

- **Initialization:** guess $P(\mathrm{f} \mid \mathrm{e})$
- **Expectation step:** compute expected counts

$$\langle c(f, e) \rangle = \sum_{s \in S} \langle c(f, e | \mathbf{e}^{(s)}, \mathbf{f}^{(s)}) \rangle$$

- **Maximization step:** recompute probabilities $P(\mathrm{f} \mid \mathrm{e})$

$$\hat{P}(f|e) = \frac{\langle c(f, e) \rangle}{\sum_{f'} \langle c(f', e) \rangle}$$

# Expectation-Maximization (EM)

1. Initialize a first model, $M_0$

2. Expectation (E) step:
   Go through training data to gather expected counts
   $\langle \text{count}(lac, lake) \rangle$
3. Maximization (M) step:
   Use expected counts to compute a new model $M_{i+1}$
   $P_{i+1}(\ lac \mid lake) = \langle \text{count}(lac, lake) \rangle / \langle \sum w \ \text{count}(w, lake) \rangle$
4. Check for convergence:
   Compute log-likelihood of training data with $M_{i+1}$
   If the difference between new and old log-likelihood smaller than a threshold, stop. Else go to 2.

# The E-step

Compute the expected count $\langle c(f, e|\mathbf{f}, \mathbf{e})\rangle$:

$$\langle c(f, e|\mathbf{f}, \mathbf{e})\rangle = \sum_{\mathbf{a}\in\mathcal{A}(\mathbf{f},\mathbf{e})} P(\mathbf{a}|\mathbf{f}, \mathbf{e}) \cdot \underbrace{c(f, e|\mathbf{a}, \mathbf{e}, \mathbf{f})}_{\text{How often are } f,e \text{ aligned in } \mathbf{a}?}$$

$$P(\mathbf{a}|\mathbf{f}, \mathbf{e}) = \frac{P(\mathbf{a}, \mathbf{f}|\mathbf{e})}{P(\mathbf{f}|\mathbf{e})} = \frac{P(\mathbf{a}, \mathbf{f}|\mathbf{e})}{\sum_{\mathbf{a}'} P(\mathbf{a}', \mathbf{f}|\mathbf{e})}$$

$$P(\mathbf{a}, \mathbf{f}|\mathbf{e}) = \prod_j P(f_j|e_{a_j})$$

$$\langle c(f, e|\mathbf{f}, \mathbf{e})\rangle = \sum_{\mathbf{a}\in\mathcal{A}(\mathbf{f},\mathbf{e})} \frac{\prod_j P(f_j|e_{a_j})}{\sum_{a'} \prod_j P(f_j|e_{a'_j})} \cdot c(f, e|\mathbf{a}, \mathbf{e}, \mathbf{f})$$

We need to know $P(f_j|e_{a_j})$, the probability that word $f_j$ is aligned to word $e_{aj}$ under the alignment $a$

# Other translation models

Model 1 is a very simple (and not very good) translation model.

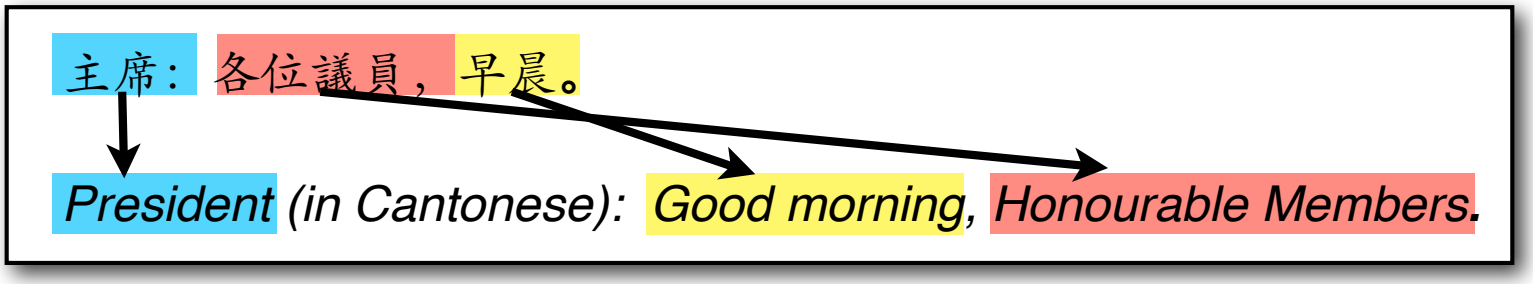
IBM models 2-5 are more complex. They take into account:

- **"fertility":** the number of foreign words generated by each target word
- the **word order** and **string position** of the aligned words

# Phrase-based translation models

# Phrase-based translation models

Assumption: fundamental units of translation are **phrases**:




**Phrase-based model of** $P(F \mid E)$:

1. Split target sentence deterministically into phrases $ep_1 \ldots ep_n$
2. Translate each target phrase $ep_i$ into source phrase $fp_i$ with translation probability $\varphi(fp_i \mid ep_i)$
3. Reorder foreign phrases with distortion probability $d(a_i - b_{i-1}) = c^{|a_i - b_{i-1} - 1|}$

$a_i$ = start position of source phrase generated by $e_i$

$b_{i-1}$ = end position of source phrase generated by $e_{i-1}$

# Phrase-based models of $P(f \mid e)$

**Split target sentence** $\boldsymbol{e}=e_{1..n}$ into phrases $\mathbf{ep}_1..\mathbf{ep}_N$:
[*The green witch*] [*is*] [*at home*] [*this week*]

**Translate each target phrase** $\mathbf{ep}_i$ into source phrase $\mathbf{fp}_i$ with **translation probability** $P(fp_i \mid ep_i)$:
[*The green witch*] = [*die grüne Hexe*], ...

**Arrange the set of source phrases** $\{fp_i\}$ to get $\mathbf{s}$ with **distortion probability** $P(\mathbf{fp} \mid \{fp_i\})$:
[*Diese Woche*] [*ist*] [*die grüne Hexe*] [*zuhause*]

$$P(\mathbf{f}|\mathbf{e} = \langle ep_1, ..., ep_l) \quad = \quad \prod_i P(fp_i|ep_i) P(\mathbf{fp}|\{fp_i\})$$

# Translation probability $P(fp_i \mid ep_i)$

Phrase translation probabilities can be obtained from a **phrase table:**

| EP | FP | count |
|---|---|---|
| green witch | grüne Hexe | … |
| at home | zuhause | 10534 |
| at home | daheim | 9890 |
| is | ist | 598012 |
| this week | diese Woche | …. |

This requires **phrase alignment**

# Word alignment

# Phrase alignment

# Obtaining phrase alignments

We'll skip over details, but here's the basic idea:

For a given parallel corpus (F-E)

1. Train **two word aligners**, (F→E and E→F)
2. Take the **intersection** of these alignments to get a **high-precision** word alignment
3. **Grow** these high-precision alignments until all words in both sentences are included in the alignment.

   Consider any pair of words in the **union** of the alignments, and incrementally add them to the existing alignments
4. Consider all phrases that are **consistent** with this improved word alignment

# Decoding
# (for phrase-based MT)

# Phrase-based models of $P(f \mid e)$

**Split target sentence** $\boldsymbol{e}=e_{1..n}$ into phrases $\mathbf{ep}_1..\mathbf{ep}_N$:
[*The green witch*] [*is*] [*at home*] [*this week*]

**Translate each target phrase** $\mathbf{ep}_i$ into source phrase $\mathbf{fp}_i$ with **translation probability** $P(fp_i \mid ep_i)$:
[*The green witch*] = [*die grüne Hexe*], ...

**Arrange the set of source phrases** $\{fp_i\}$ to get $\mathbf{s}$ with **distortion probability** $P(fp \mid \{fp_i\})$:
[*Diese Woche*] [*ist*] [*die grüne Hexe*] [*zuhause*]

$$P(\mathbf{f}|\mathbf{e} = \langle ep_1, ..., ep_l) \quad = \quad \prod_i P(fp_i|ep_i)P(\mathbf{fp}|\{fp_i\})$$

# Translating

How do we translate a foreign sentence (e.g. *"Diese Woche ist die grüne Hexe zuhause"*) into English?

- We need to find $\hat{e} = argmax_e P(f \mid e)P(e)$
- There is an exponential number of candidate translations $e$
- But we can look up phrase translations $ep$ and $P(fp \mid ep)$ in the phrase table:

| **diese** | **Woche** | **ist** | **die** | **grüne** | **Hexe** | **zuhause** |
|---|---|---|---|---|---|---|
| this 0.2 | week 0.7 | is 0.8 | the 0.3 | green 0.3 | witch 0.5 | home 1.00 |
| these 0.5 | | | the green 0.4 | | sorceress 0.6 | |
| this week 0.6 | | | | green witch 0.7 | | |
| is this week 0.4 | | | the green witch 0.7 | | | |

# Generating a (random) translation

1. Pick the first Target phrase $ep_1$ from the candidate list.

$$P := P_{LM}(<s> ep_1)P_{Trans}(fp_1 \mid ep_1)$$

*E = the, F= <….die…>*

2. Pick the next target phrase $ep_2$ from the candidate list

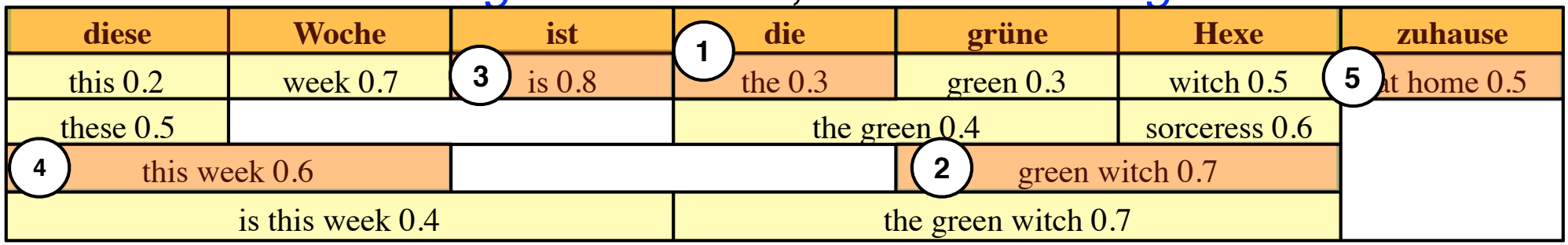
$$P := P \times P_{LM}(ep_2 \mid ep_1)P_{Trans}(fp_2 \mid ep_2)$$

*E = the green witch, F = <….die grüne Hexe...>*

3. Keep going: pick target phrases $ep_i$ until the entire source sentence is translated

$$P := P \times P_{LM}(ep_i \mid ep_{1\ldots i-1})P_{Trans}(fp_i \mid ep_i)$$

*E = the green witch is, F = <….ist die grüne Hexe...>*



# Finding the best translation

How can we find the *best* translation efficiently?

There is an exponential number of possible translations.

We will use a *heuristic* search algorithm

We cannot guarantee to find the best (= highest-scoring) translation, but we're likely to get close.

We will use a *"stack-based"* decoder

(If you've taken Intro to AI: this is A* ("A-star") search)

We will score partial translations based on how good we expect the corresponding completed translation to be.

Or, rather: we will score partial translations on how **bad** we expect the corresponding complete translation to be.

That is, our scores will be **costs (high=bad, low=good)**

# Scoring partial translations

Assign expected costs to *partial* translations $(E, F)$:

$$expected\_cost(E,F) = current\_cost(E,F) + future\_cost(E,F)$$

The current cost is based on the score of the partial translation $(E, F)$

e.g. $current\_cost(E,F) = \log P(E)P(F \mid E)$

The (estimated) future cost is a **lower bound** on the actual cost of completing the partial translation $(E, F)$:

$true\_cost(E,F)$ $(= current\_cost(E,F) + actual\_future\_cost(E,F))$

$\geq expected\_cost(E,F)$ $(= current\_cost(E,F) + est\_future\_cost(E,F))$

because $actual\_future\_cost(E,F) \geq est\_future\_cost(E,F)$

(The estimated future cost ignores the distortion cost)

# Stack-based decoding

Maintain a **priority queue** (='stack') of **partial translations** (hypotheses) with their **expected costs**.

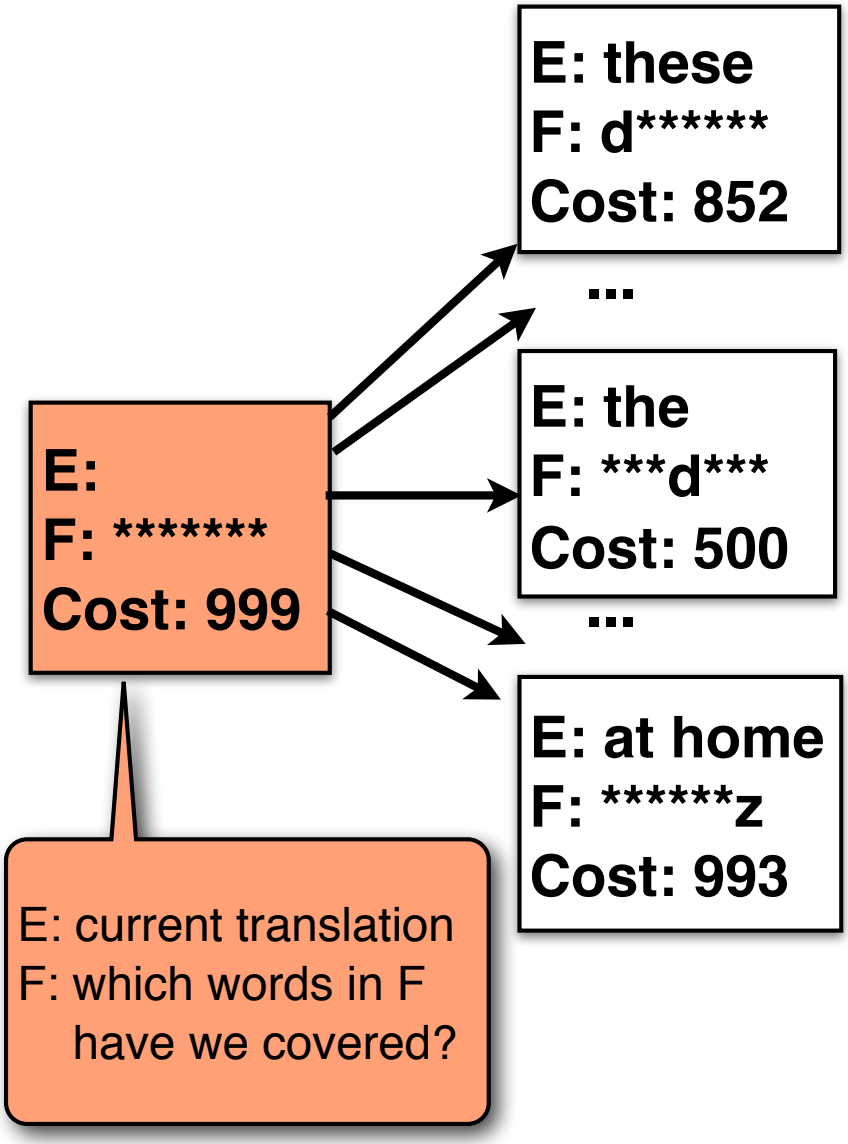
Each element on the stack is **open** (we haven't yet pursued this hypothesis) or **closed** (we have already pursued this hypothesis)

At each step:

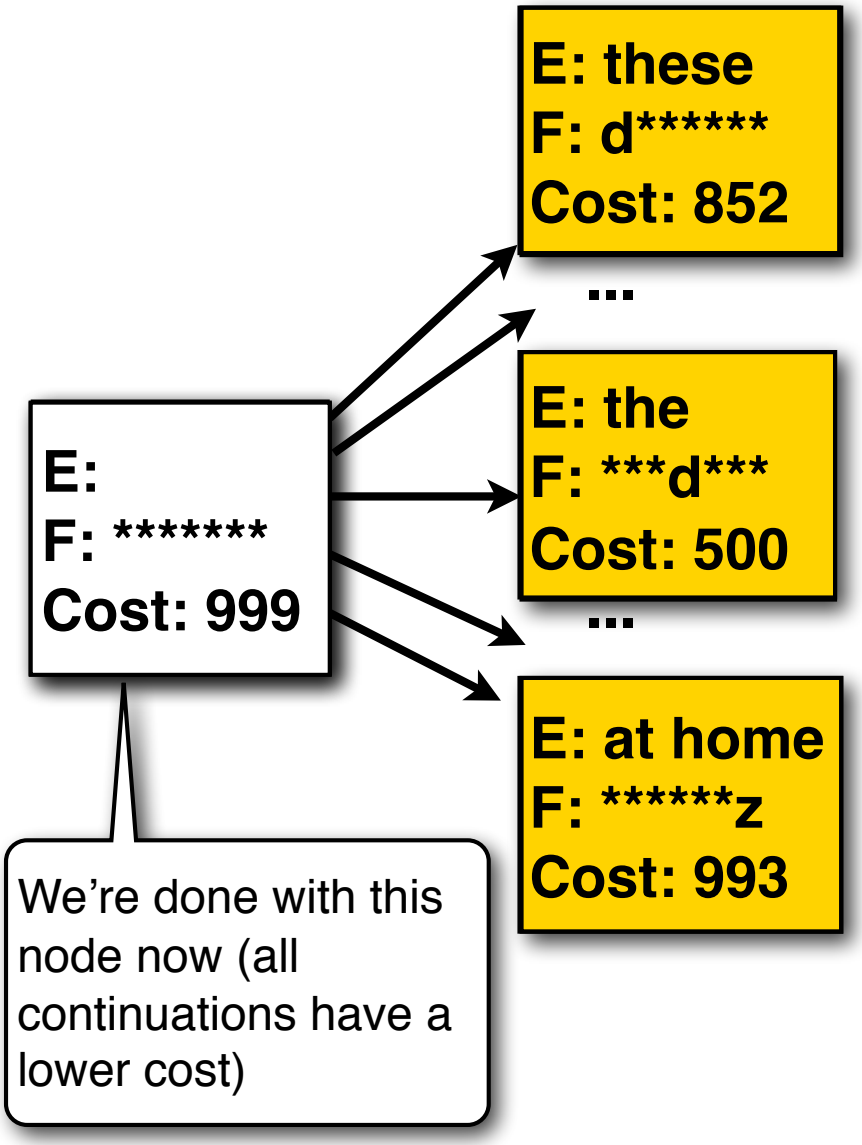
- **Expand** the best open hypothesis (the open translation with the lowest expected cost) in all possible ways.
- These new translations become new **open elements** on the stack.
- **Close** the best open hypothesis.

**Additional Pruning** (*n*-best / beam search):
Only keep the *n* best open hypotheses around

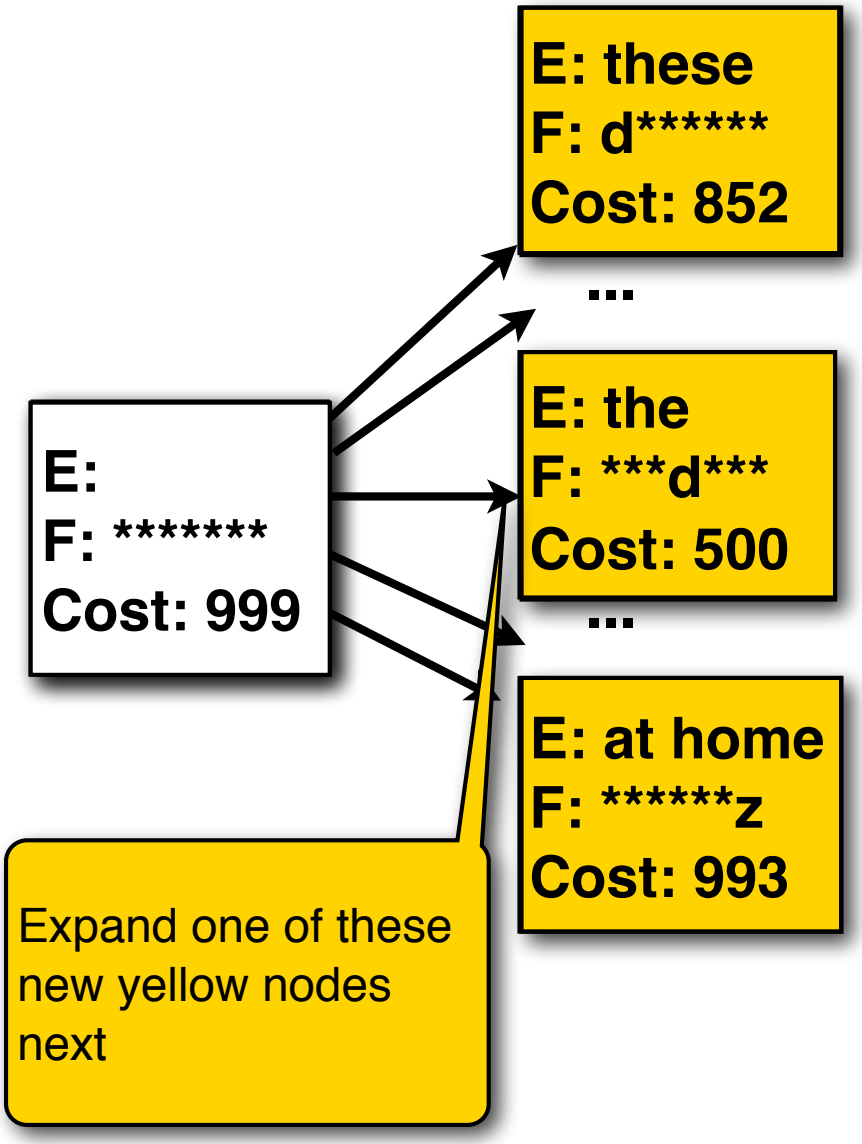
# Stack-based decoding



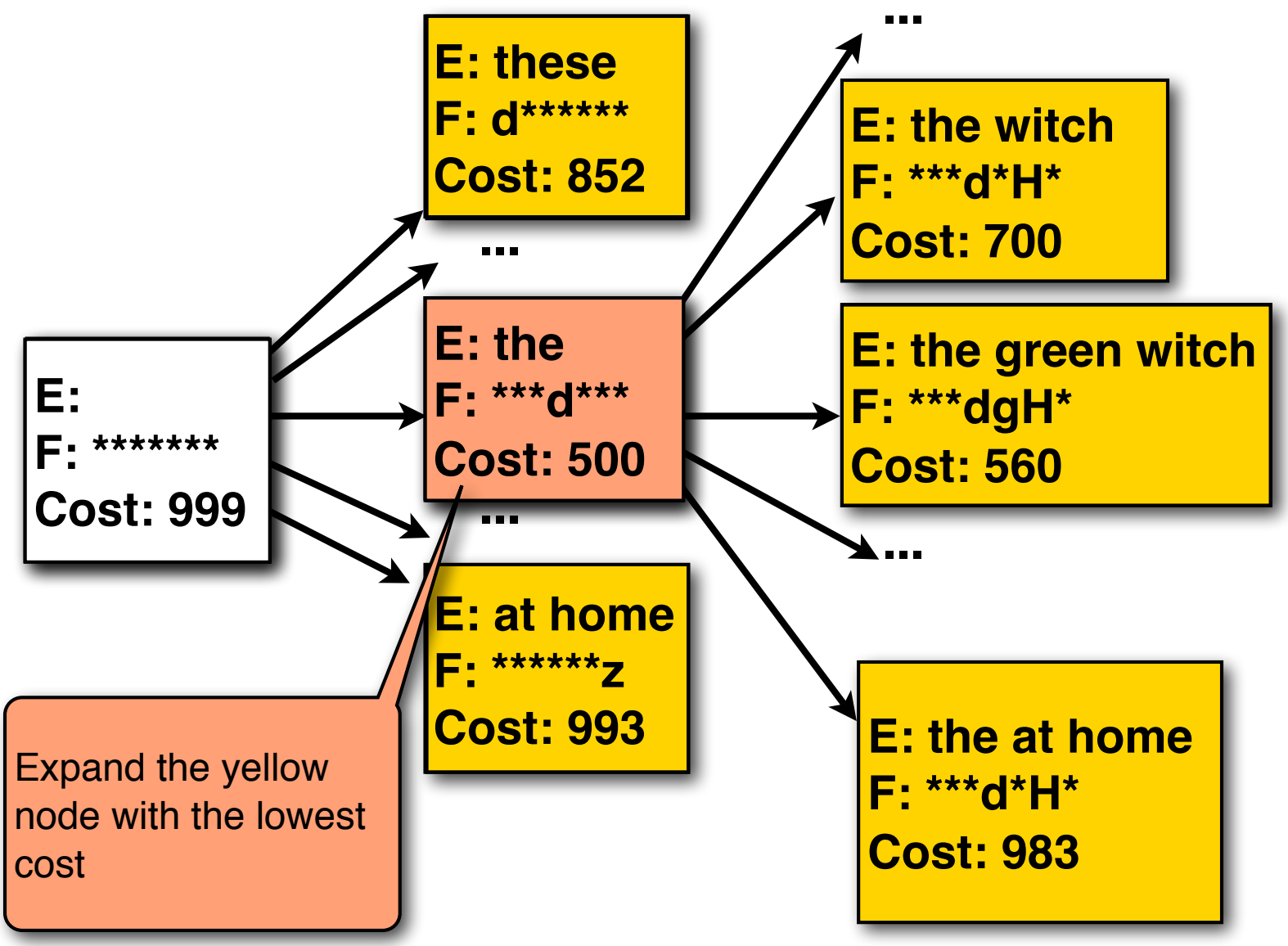
# Stack-based decoding



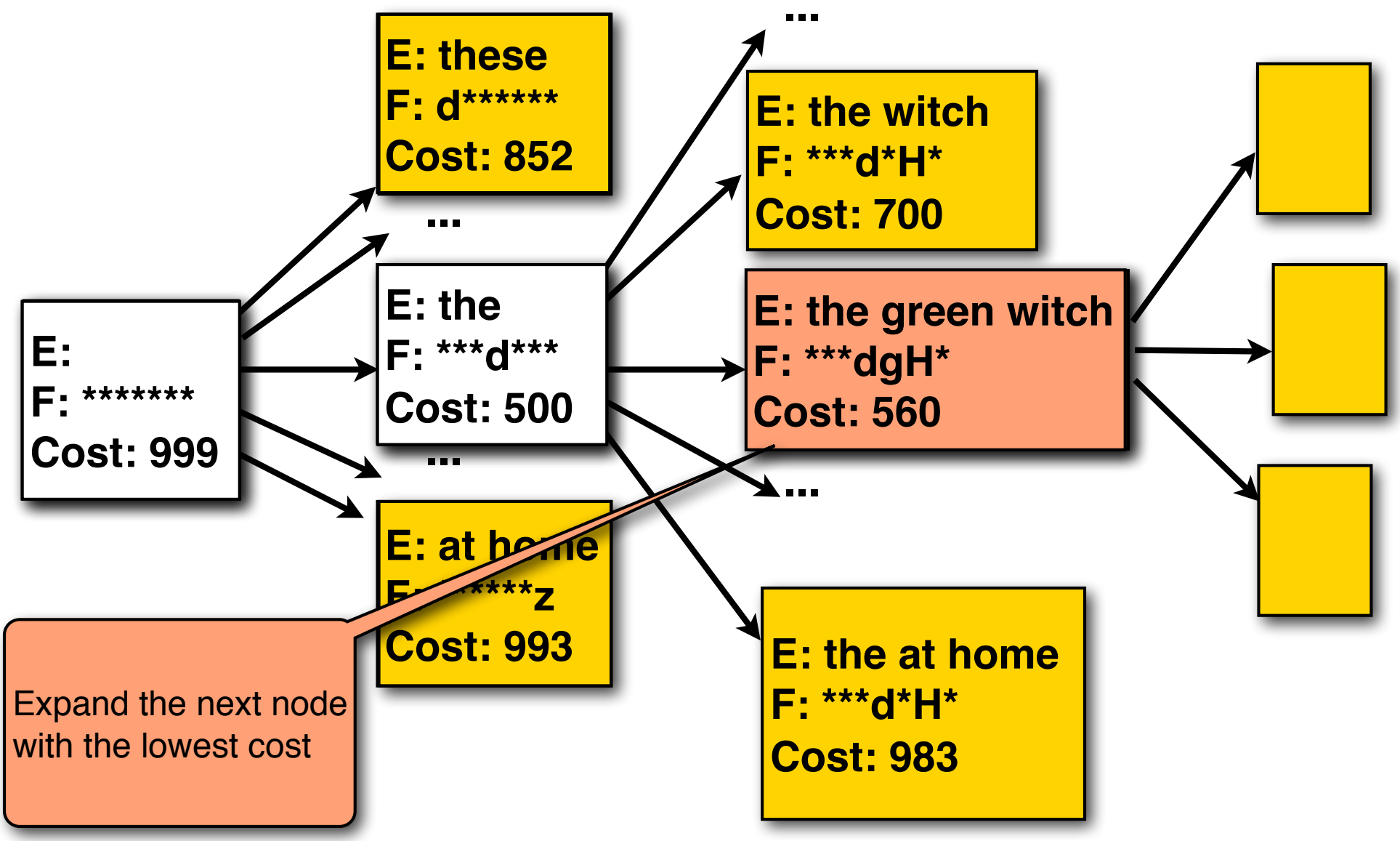
# Stack-based decoding



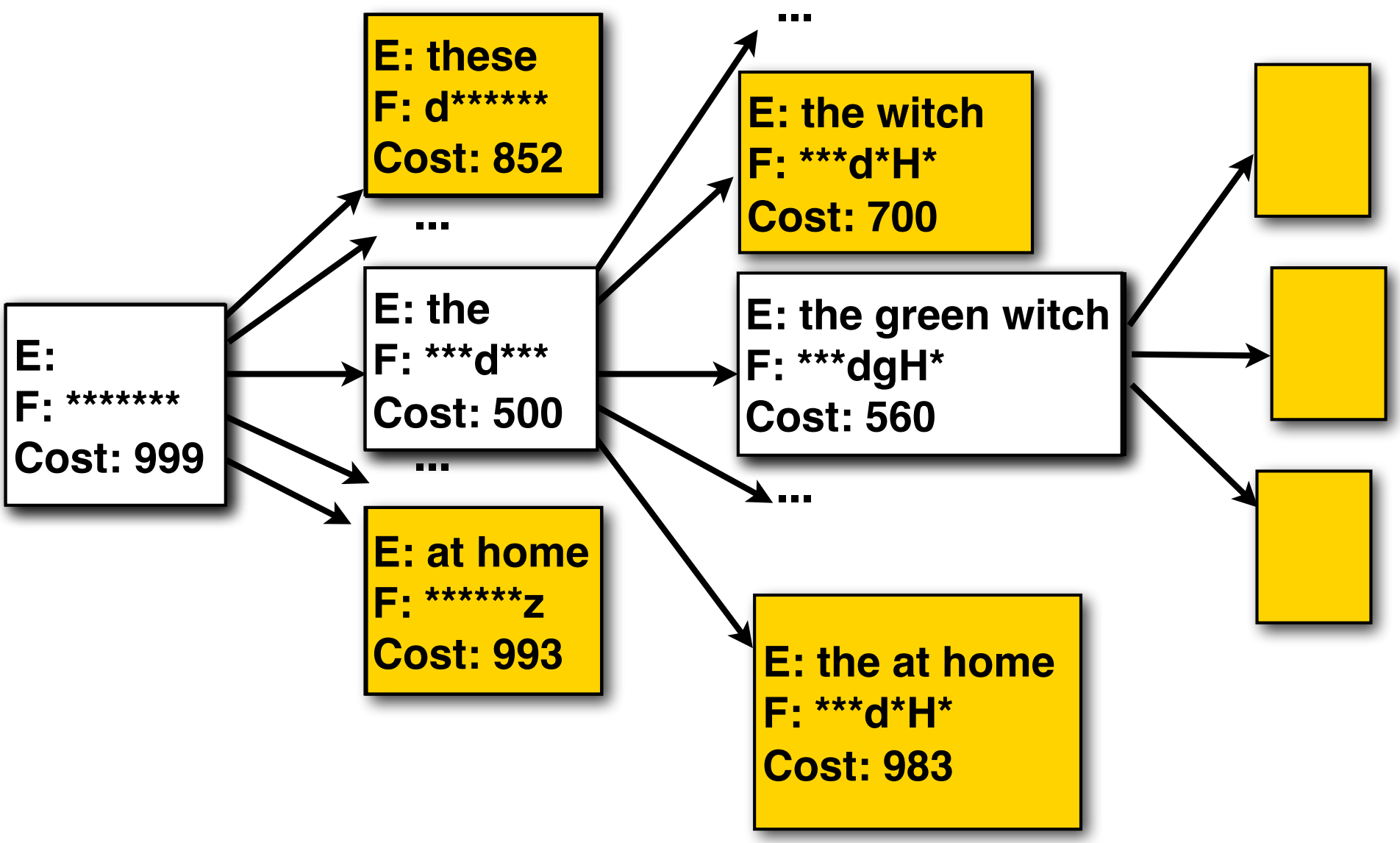
# Stack-based decoding



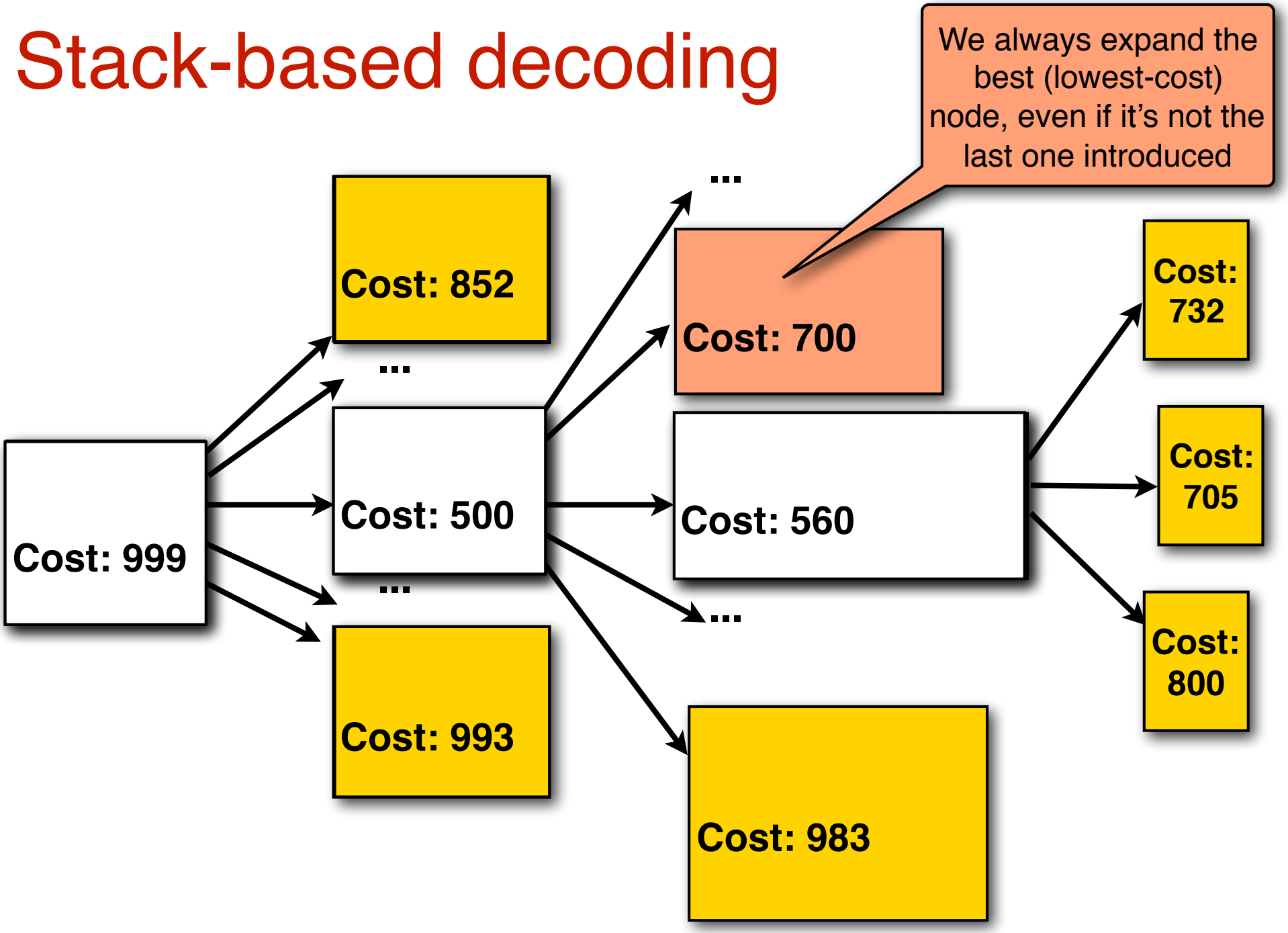
# Stack-based decoding



# Stack-based decoding



# Stack-based decoding



# Summary: Machine Translation

# Machine translation models

Current MT models all rely on statistics.

Many current models do estimate $P(\mathrm{E} \mid \mathrm{F})$ directly, but may use features based on language models (capturing $P(\mathrm{E})$) and IBM-style translation models ($\mathrm{P}(\mathrm{F} \mid \mathrm{E})$) internally.

There are a number of syntax-based models, e.g. using synchronous context-free grammars, which consist of pairs of rules for the two languages in which each RHS NT in language A corresponds to a RHS NT in language B:

Language A: XP → YP ZP  Language B: XP → ZP YP

# Outlook: Neural MT

Neural network-based approaches:

- Recurrent neural networks (RNN) can model sequences (e.g. strings, sentences, etc.)
- Use one RNN (the encoder) to process the input in the source language
- Pass its output to another RNN (the decoder) to generate the output in the target language

See e.g. http://www.tensorflow.org/tutorials/seq2seq/index.md#sequence-to-sequence_basics

# Today's key concepts

Why is machine translation hard?

- Linguistic divergences: morphology, syntax, semantics

Different approaches to machine translation:

- Vauquois triangle
- Statistical MT: Noisy Channel, IBM Model 1 (more on this next time)

# Great talk at 2pm today — No office hours today

**Distinguished Lecture** In Computer Science

## Explainable AI: Making Visual Question Answering Systems more Transparent

***A Distinguished Lecture Sponsored by the Department of Computer Science***
**Guest Speaker:** Raymond Mooney, Professor, University of Texas at Austin
**Date/Time:** Friday, Oct. 18, 2019, 2:00 pm
**Location:** 2405 Thomas M. Siebel Center for Computer Science

**Abstract:** Artificial Intelligence systems' ability to explain their conclusions is crucial to their utility and trustworthiness. Deep neural networks have enabled significant progress on many challenging problems such as visual question answering (VQA), the task of answering natural language questions about images. However, most of them are opaque black boxes with limited explanatory capability. The goal of Explainable AI is to increase the transparency of complex AI systems such as deep networks. We have developed a novel approach to XAI and used it to build a high-performing VQA system that can elucidate its answers with multi-modal natural-language and visual explanations that faithfully reflect important aspects of its underlying reasoning while capturing the style of comprehensible human explanations. Crowd-sourced human evaluation of these explanations demonstrate the advantages of our approach.

**Bio:** Raymond J. Mooney is a Professor in the Department of Computer Science at the University of Texas at Austin. He received his Ph.D. in 1988 from the University of Illinois at Urbana/Champaign. He is an author of over 170 published research papers, primarily in the areas of machine learning and natural language processing. He was the President of the International Machine Learning Society from 2008-2011, program co-chair for AAAI 2006, general chair for HLT-EMNLP 2005, and co-chair for ICML 1990. He is a Fellow of AAAI, ACM, and ACL and the recipient of the Classic Paper award from AAAI-19 and best paper awards from AAAI-96, KDD-04, ICML-05 and ACL-07.