Statistical Machine Translation: Word Based Translation Models

Michael Wohlmayr

Statistical Machine Translation

- There is not THE ONE english translation **e** of a foreign sentence **f**.
- Some translations **e** are more likely than others:

Das ist ein schnelles Auto.

probable:	*It is a rapid vehicle.*
probable:	*This is a fast car.*
less probable:	*It‘s a sunny day.*

Statistical Machine Translation

- Assign each sentence pair (**e**,**f**) a probability p(**e**|**f**)
- Assume we have a model to calculate p(**e**|**f**)
- To translate a foreign sentence into english, find:

$$\hat{\mathbf{e}} = \arg\max_{\mathrm{e}} p(\mathbf{e} \mid \mathbf{f})$$

Well formed strings

- Problem: p(**e**|**f**) must concentrate its probability on *well-formed* english sentences

well-formed	***ill-formed***
I live in a house.	I house a in live.
Have you seen my keys?	Seen keys have my you?

Bayes Rule

- Relax contraint by using alternative statement:

$$\hat{\mathbf{e}} = \arg\max_{\mathrm{e}} p(\mathbf{e})\, p(\mathbf{f} \mid \mathbf{e})$$

- Bayesian Reasoning: „What is the chance that the producer of **f** had **e** in mind and then translated it to **f**"

Source Channel Model

Language and Translation Model

$$\hat{\mathbf{e}} = \arg\max_{\mathrm{e}} p(\mathbf{e})\, p(\mathbf{f} \mid \mathbf{e})$$

- p(**e**): language model. Gives a low probability to ill-formed strings.
- p(**f**|**e**): translation model. Must not concentrate on well-formed strings anymore. Gives the probability that a english *bag of words* will translate into a french *bag of words*.
- *Probabilistic model split into 2 simpler models*.

Decoding

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} p(\mathbf{e})\, p(\mathbf{f} \mid \mathbf{e})$$

- Process of finding **e** that maximizes the above product.

Translation Model

- „Explains“ how english sentence becomes a french sentence.
- Based on a set of parameters.
- Given the parameters and a sentence pair (**e**,**f**), computation of p(**f**|**e**) is possible.
- IBM Model 1-5, developped in *Candide* Project
- Lets start with model 3…

Model 3

- Very simplistic way of explaining how sentence **e** is translated to foreign sentence **f**
- Based on probabilities, simply reproduce each word a number of times, translate each word then reposition them
- Don‘t worry: This model is not used for translation, just to judge p(**f**|**e**)

Fertility

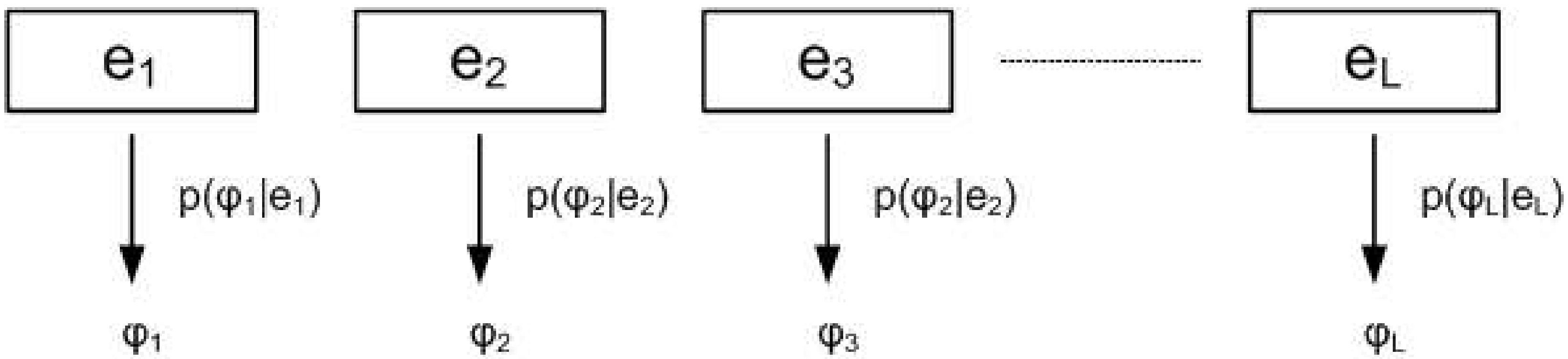
- Each word in the english sentence may produce a certain amount of foreign words.
- e.g.

This is not true.

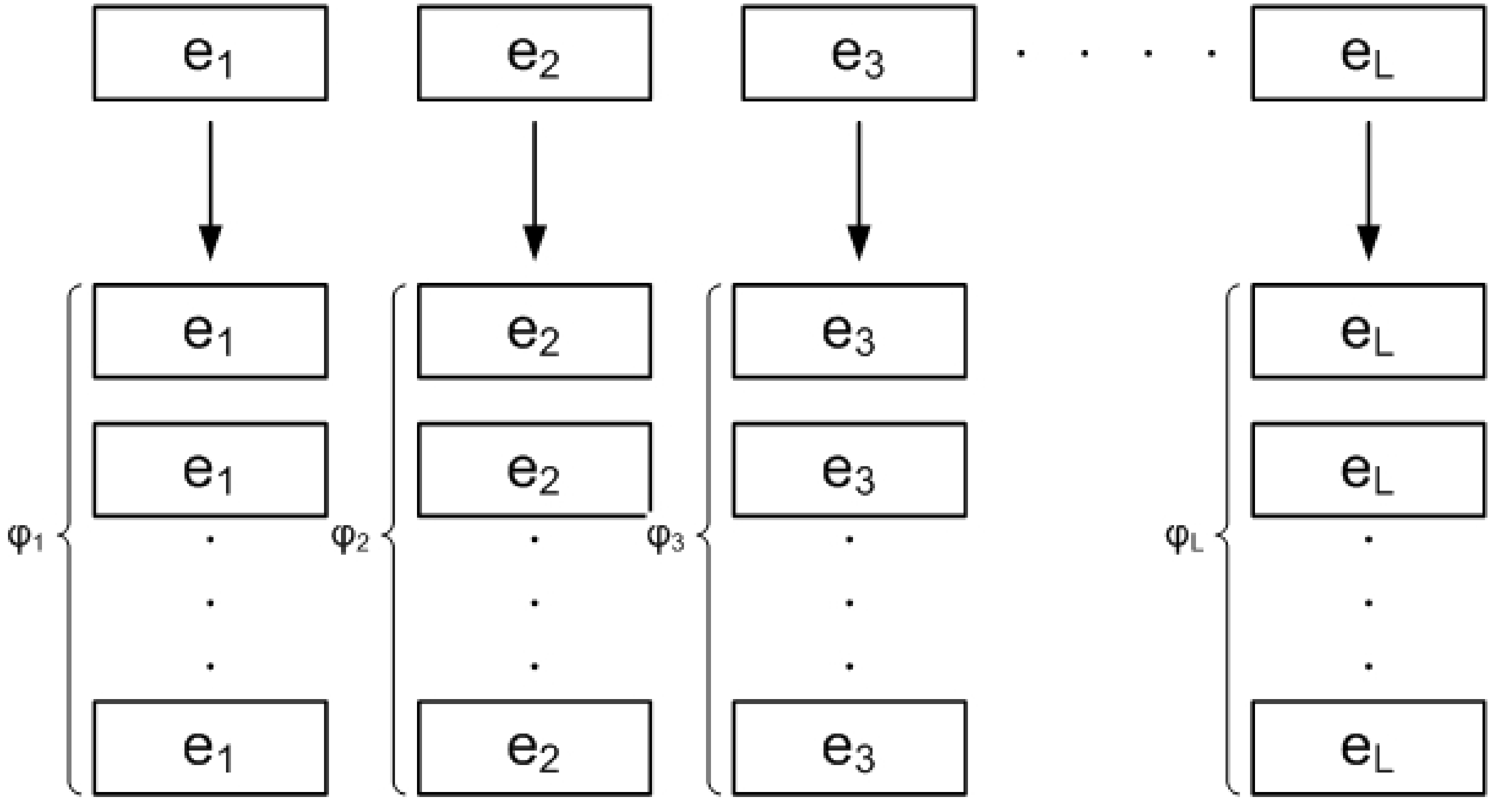
Ce n'est pas vrais.

Fertility

- Fertility probability $p_n(\varphi|e)$

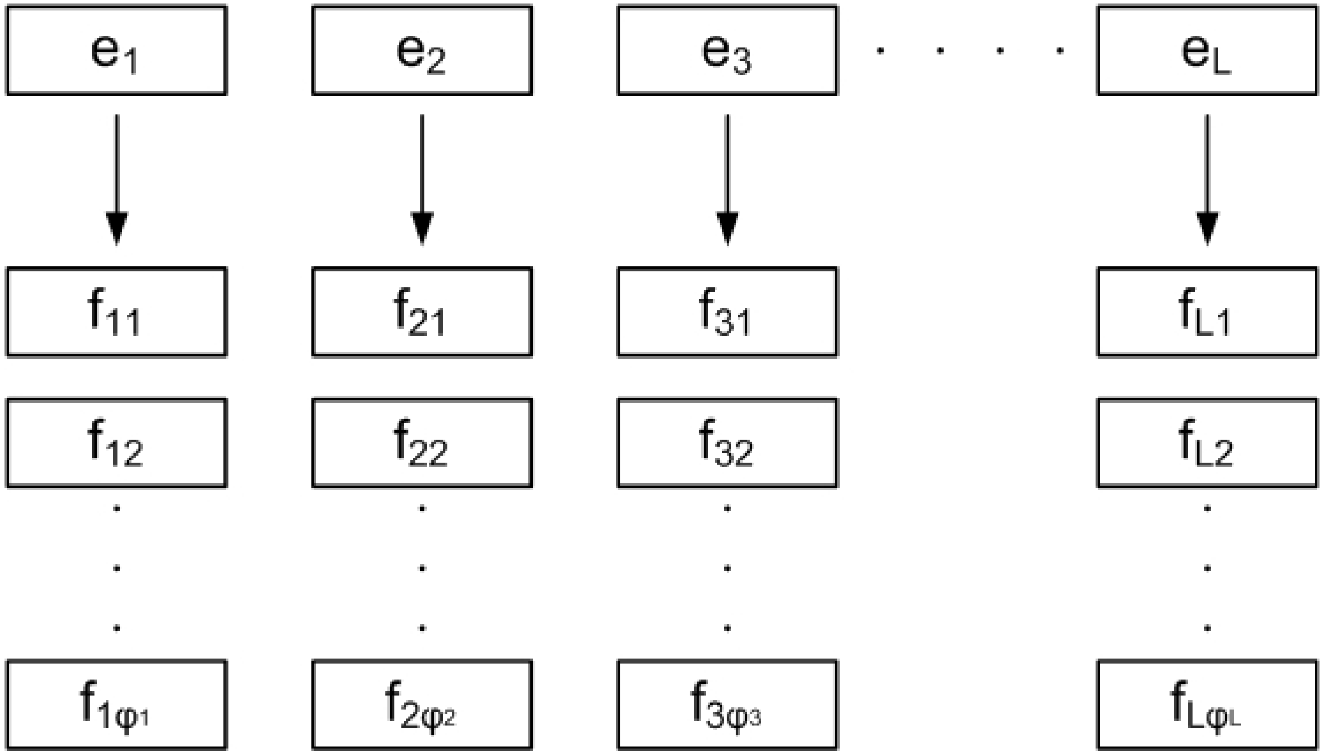


Fertility



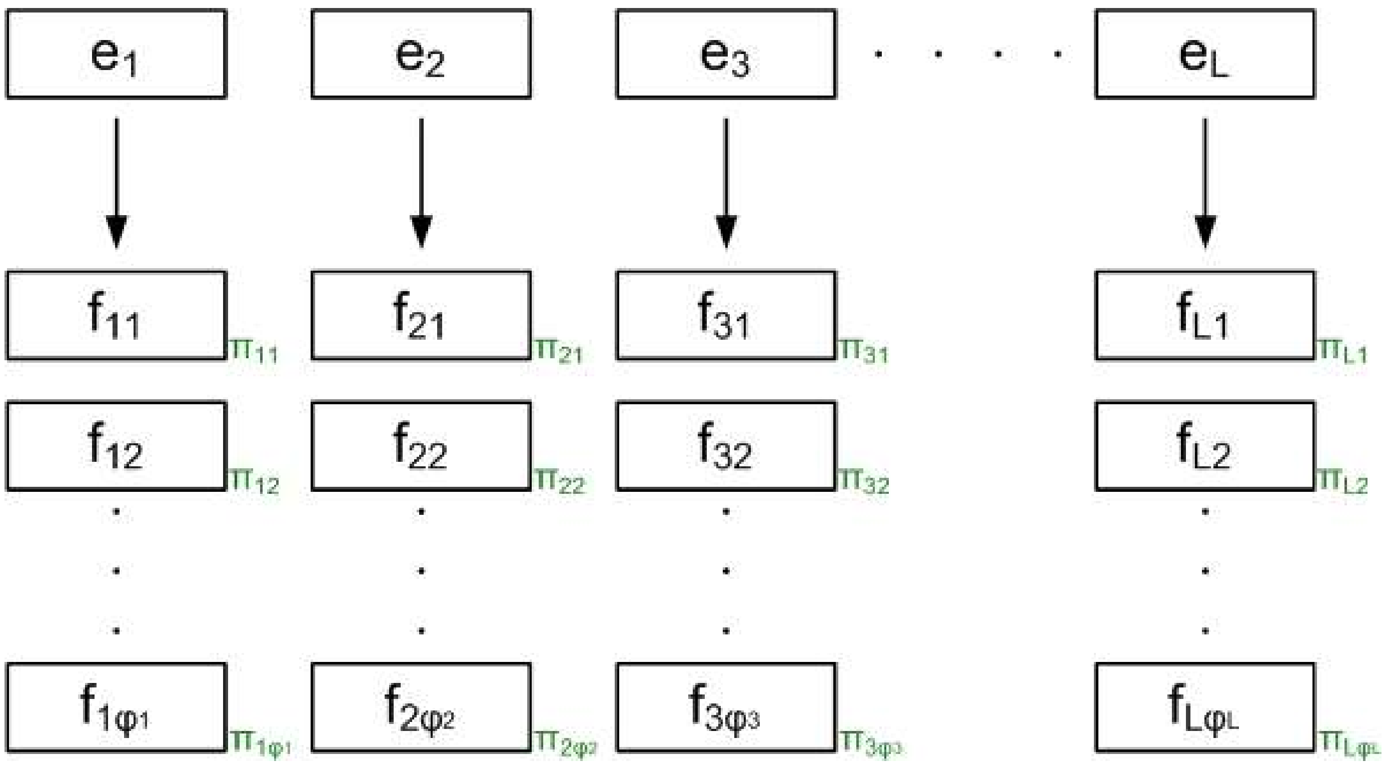
Word Translation

- Word translation probability $p_t(f|e)$

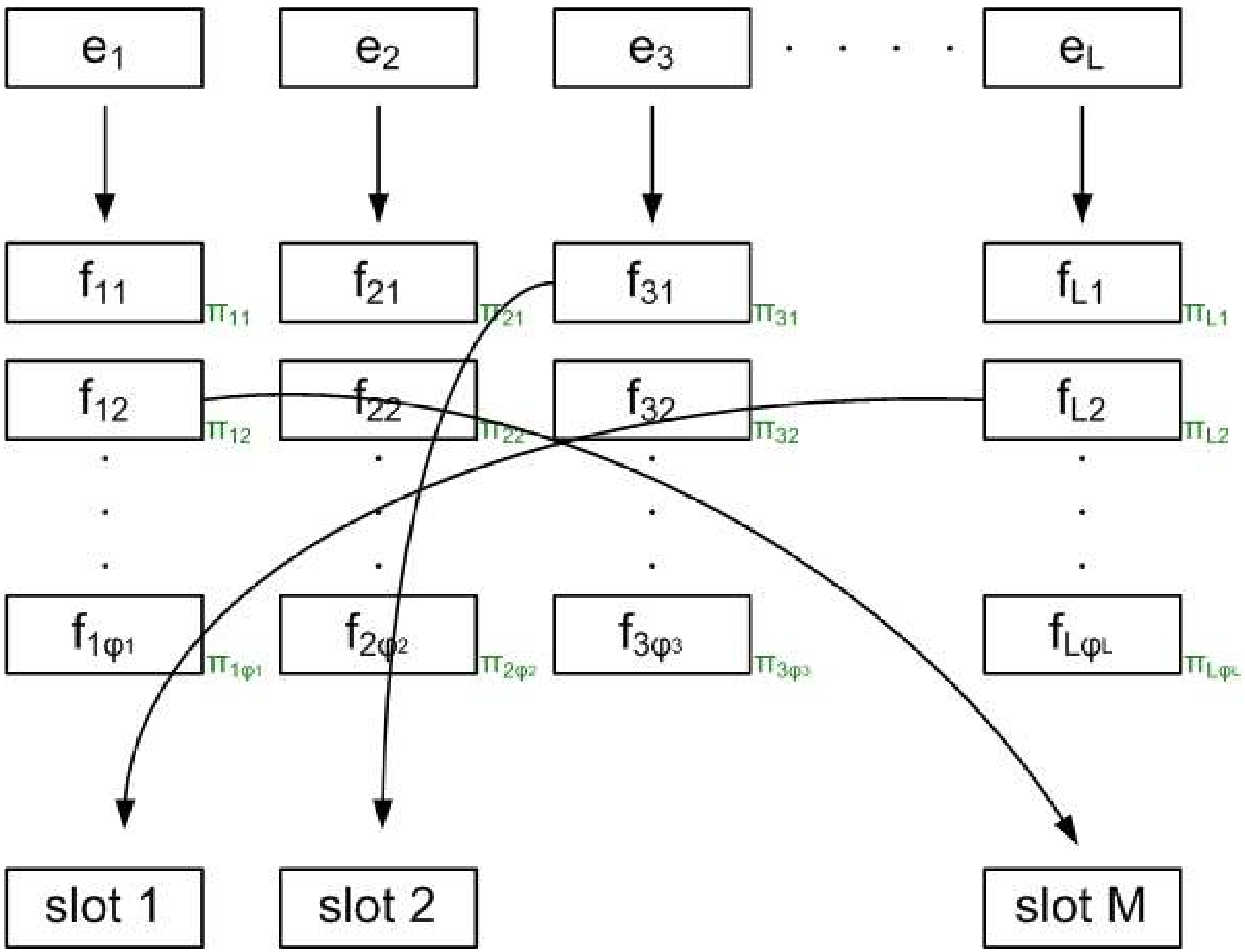


Positioning

- distortion probability $p_p(\pi|i,L,M)$



Positioning



Spurious Words

- Words that appear in the translation „although no word in the english sentence can be held responsible for them"

I go home.

Ich gehe nach Hause.

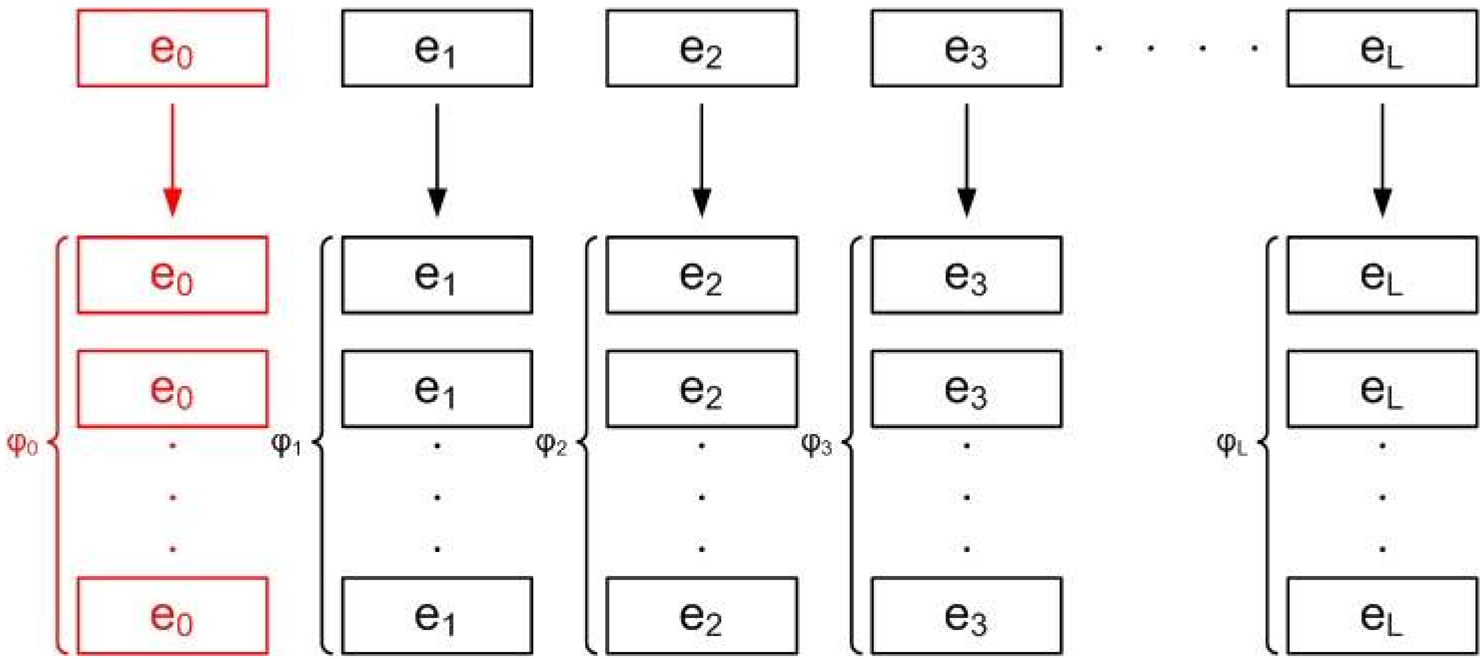
Spurious Words

- Spurious words are added after the „normal“ words have been generated.
- After each foreign word, add a spurious word with probability p_1

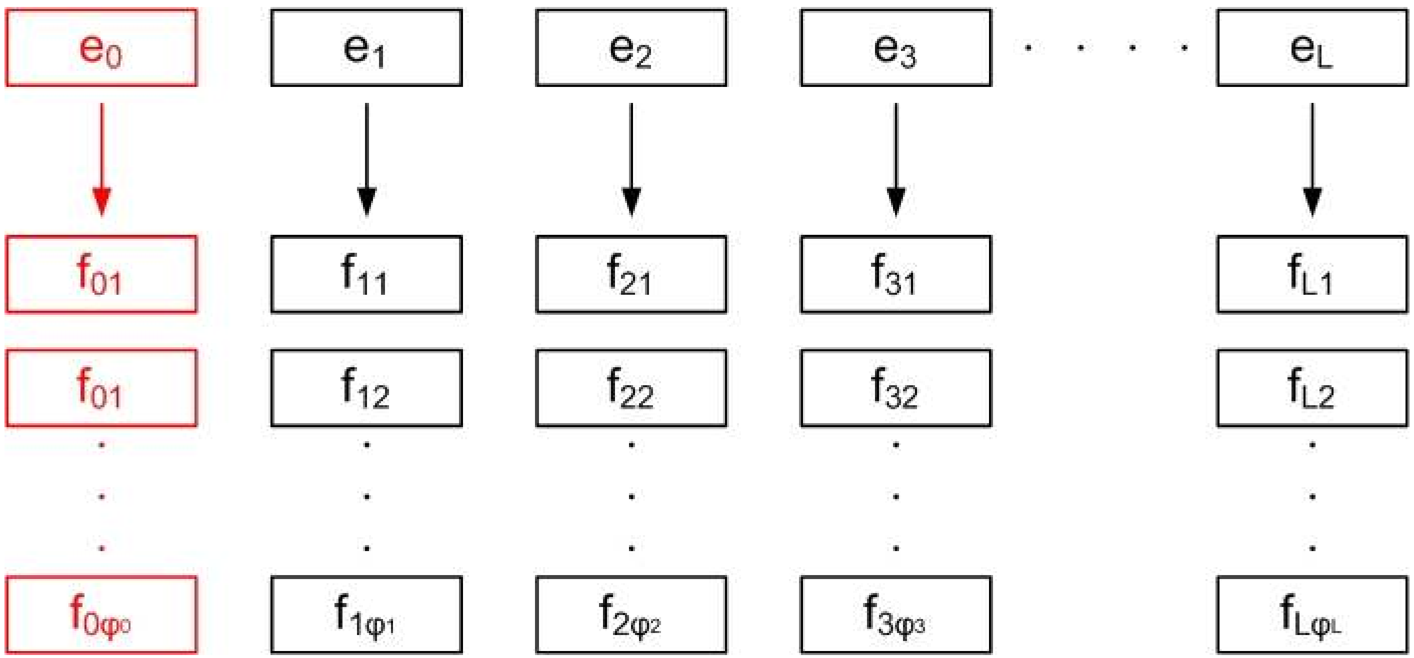
$$M_n = \varphi_1 + \varphi_2 + \ldots + \varphi_L$$

$$p(\varphi_0) = \binom{M_n}{\varphi_0} (1 - p_1)^{M_n - \varphi_0} p_1^{\varphi_0}$$

Spurious Words



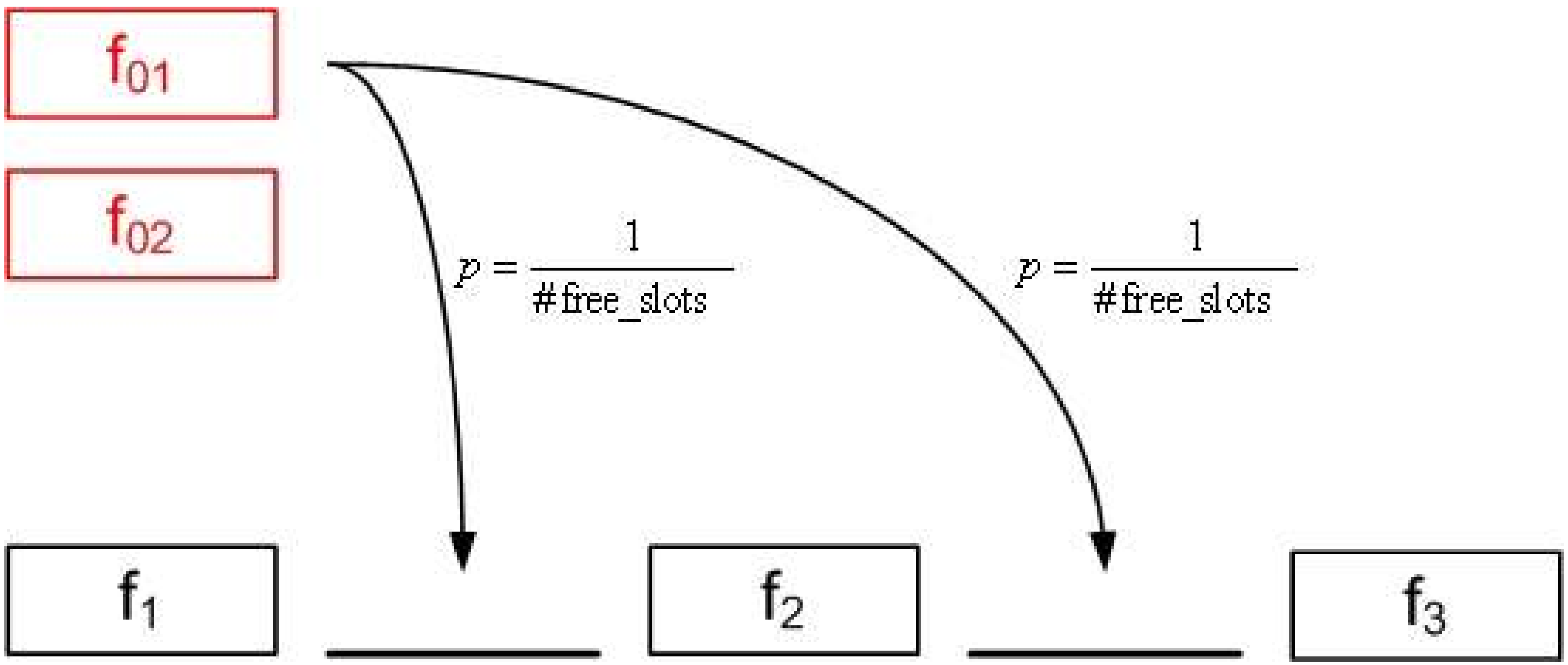
Word Translation



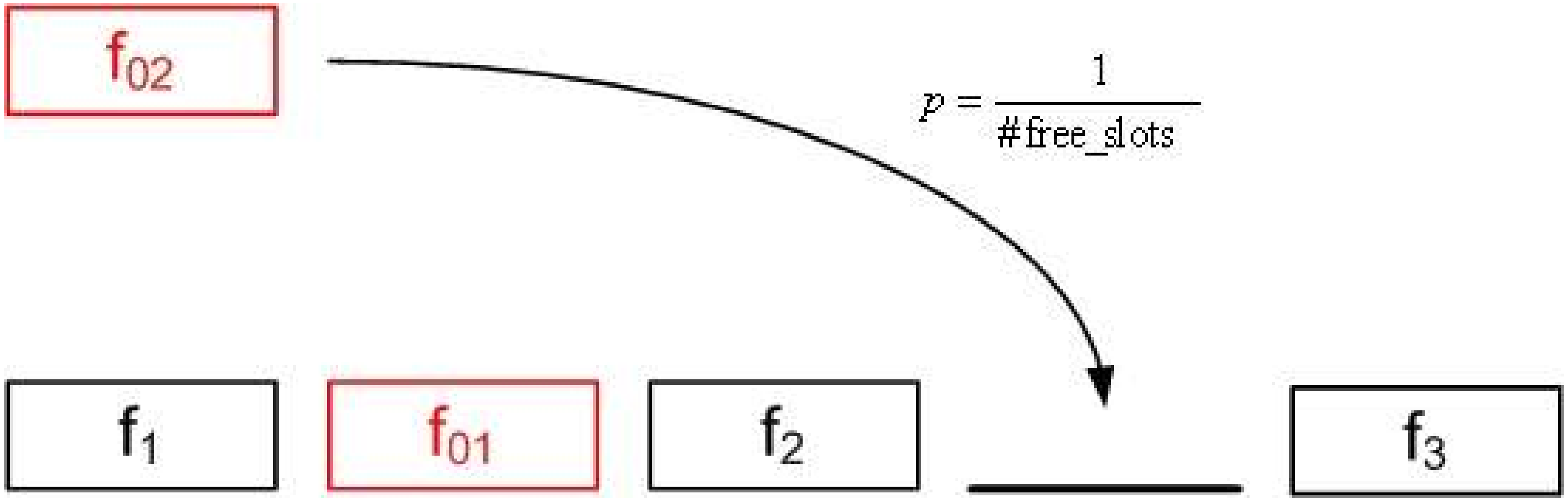
Positioning of spurious words

- Final foreign sentence has $M = M_n + \varphi_0$ words.
- After positioning the normal words, assign each spurious word a random position in one of the remaining free positions.

Positioning of spurious words



Positioning of spurious words



Model 3 Parameters

- fertility $p_n(\varphi|e)$
- translation $p_t(f|e)$
- positioning $p_p(\pi|i,L,M)$
- spurious fertility p_1

HOW TO ESTIMATE?

Training sentences

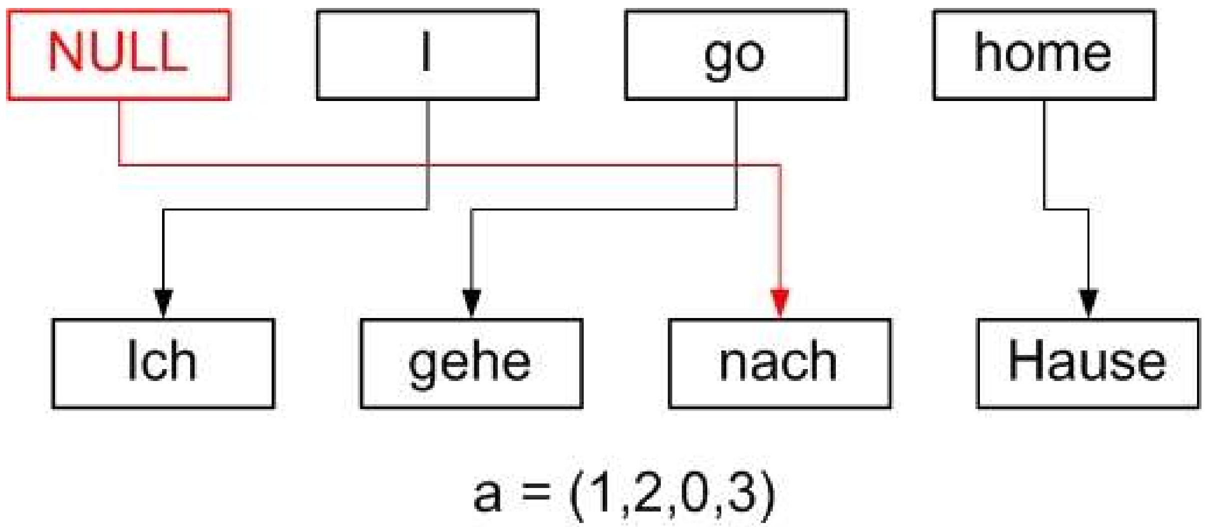
- Based on large corpora, consisting of many translation pairs (**e**,**f**)
- e.g.: European Parliament Proceedings Parallel Corpus 1996-2003

http://people.csail.mit.edu/koehn/publications/europarl/

Mr President , I respond to an invitation yesterday afternoon by the President of the House to speak on behalf of my group on a matter referred to in the Minutes .

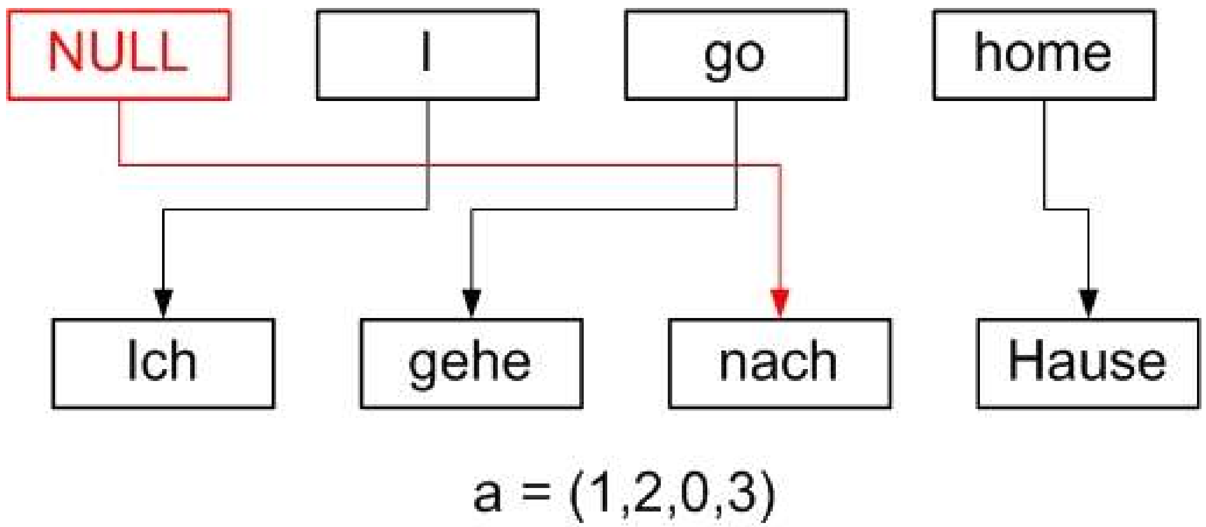
Herr Präsident , ich entspreche hiermit einer vom Präsidenten des Hauses gestern nachmittag geäußerten Aufforderung , im Namen meiner Fraktion zu der im Protokoll genannten Angelegenheit Stellung zu nehmen .

Word-for-Word Alignments



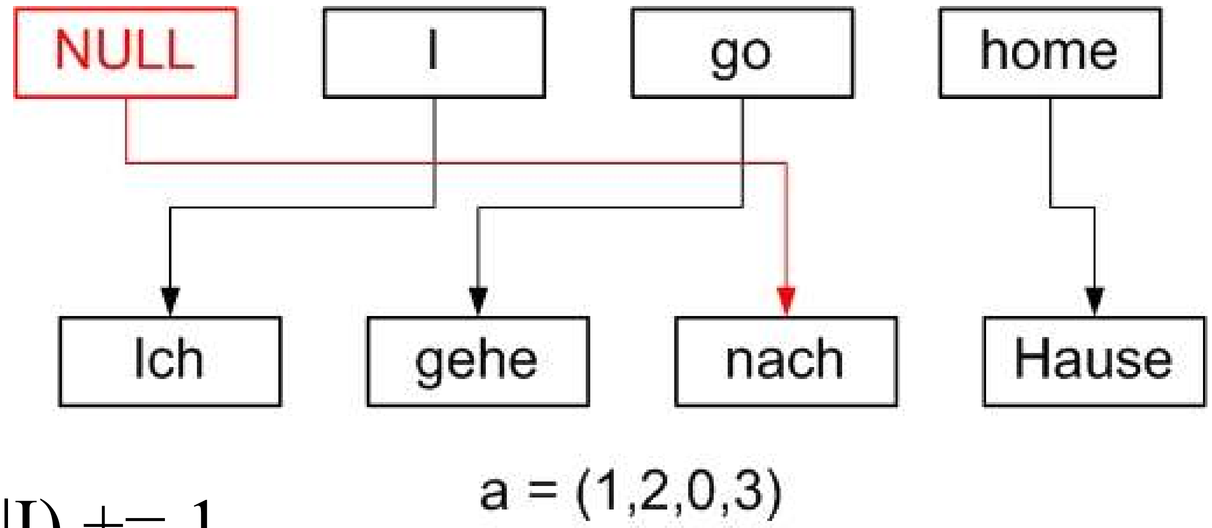
- Each word e might be connected to one or more words in **f**
- Each word f is connected to one and only one word in **e**.

Word-for-Word Alignments



- Given a large corpus together with alignment information, parameters can be estimated
- Count events in each sentence

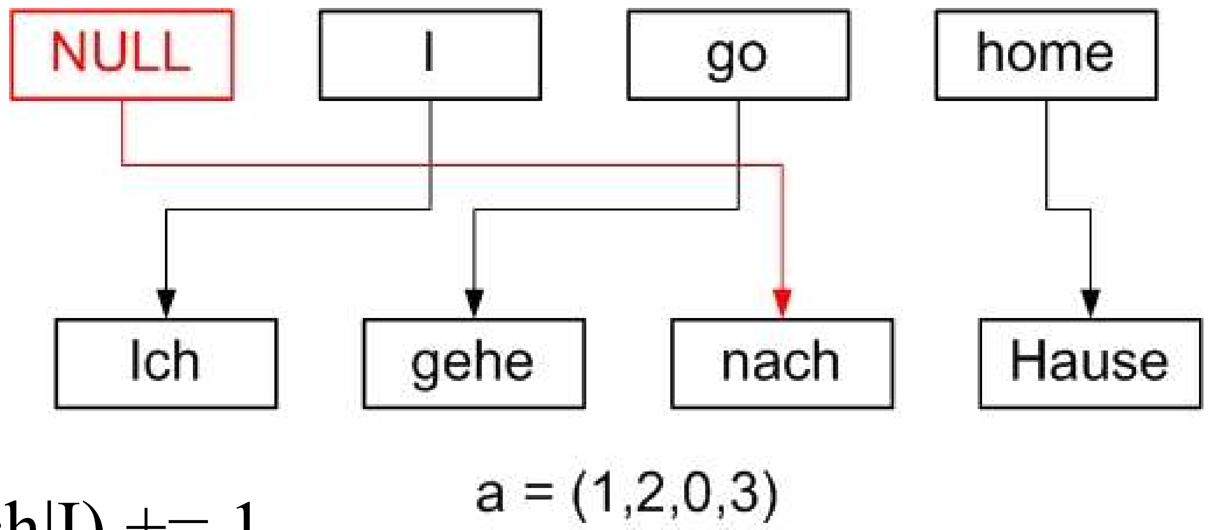
Fertility Estimation



- $c_n(1|I)$ += 1
- $c_n(1|go)$ += 1
- $c_n(1|home)$ += 1
- $c_n(1|NULL)$ += 1

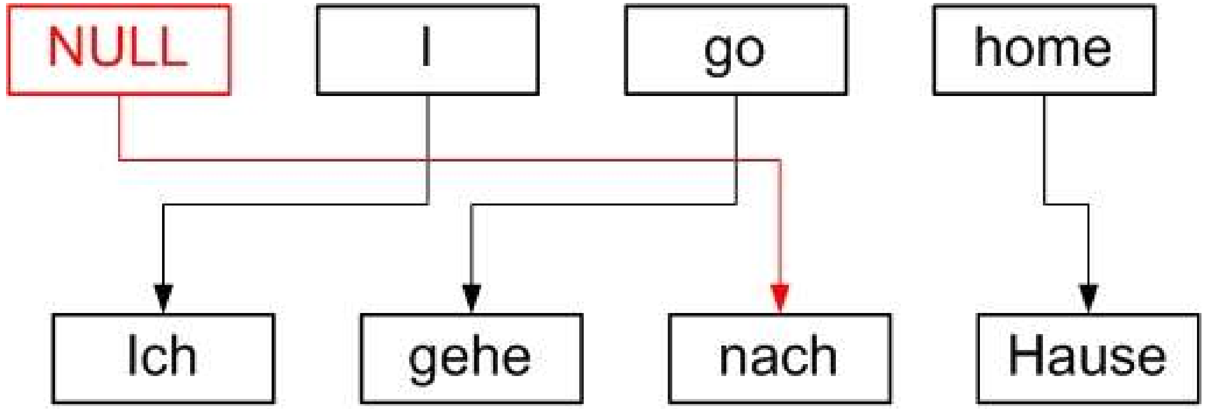
$$p_1 = \frac{\#\text{spurious words in translations}}{\#\text{normal words in translations}}$$

Translation Estimation



- c_t(Ich|I) += 1
- c_t(gehe|go) += 1
- c_t(hause|home) += 1
- c_t(nach|NULL) += 1

Positioning Estimation



a = (1,2,0,3)

- $c_p(1|1,3,4) \mathrel{+}= 1$
- $c_p(2|2,3,4) \mathrel{+}= 1$
- $c_p(4|3,3,4) \mathrel{+}= 1$

Parameter Estimation

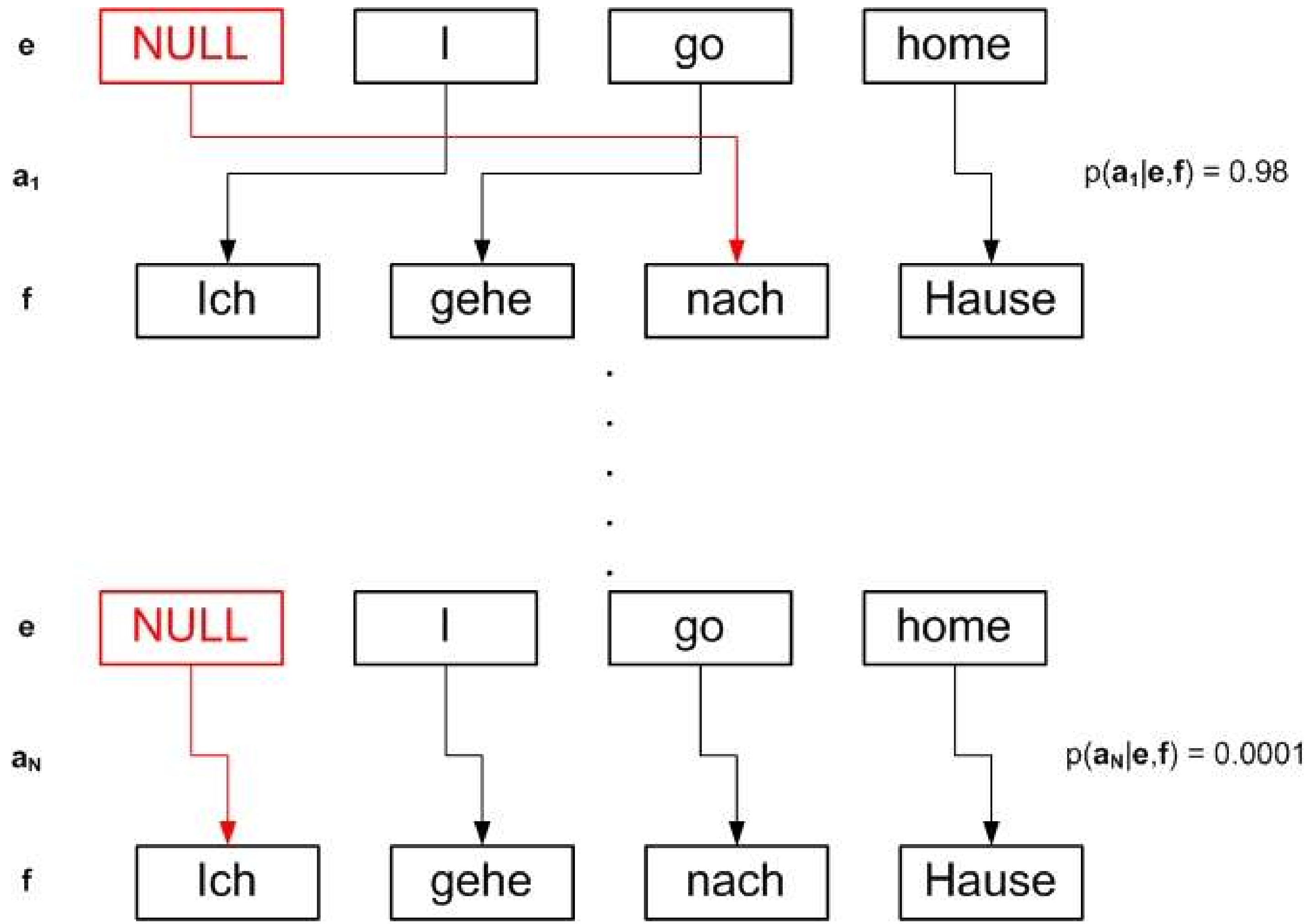
- After collecting these counts over the whole corpus, normalize them to form a proper probability distribution

- **PROBLEM:** We don‘t have alignment information

Alignment Probability

- For each (**e**.**f**), there are $(L+1)^M$ possible alignments
- Some alignments are more probable than others
- Alignment probability p(**a**|**e**,**f**)

Alignment Probability



Parameter Estimation

$$c_n(\varphi \mid e_i) += \sum_{\mathbf{a}:\{e_i \leftrightarrow (f_1, f_2, \ldots, f_\varphi)\}} p(\mathbf{a} \mid \mathbf{e}, \mathbf{f})$$

$$c_t(f_j \mid e_i) += \sum_{\mathbf{a}:\{e_i \leftrightarrow f_j\}} p(\mathbf{a} \mid \mathbf{e}, \mathbf{f})$$

$$c_p(j \mid i, L, M) += \sum_{\mathbf{a}:\{e_i \leftrightarrow f_j\}} p(\mathbf{a} \mid \mathbf{e}, \mathbf{f})$$

Parameter Estimation

- With alignment probabilies, parameter estimation is possible too.
- Again the same problem: How do we get alignment probabilities?

Alignment Probability

- Reformulate:

$$p(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) = \frac{p(\mathbf{a}, \mathbf{f} \mid \mathbf{e})}{p(\mathbf{f} \mid \mathbf{e})} = \frac{p(\mathbf{a}, \mathbf{f} \mid \mathbf{e})}{\sum_{\mathbf{a}} p(\mathbf{a}, \mathbf{f} \mid \mathbf{e})}$$

- With p(**a**,**f**|**e**), we can compute:
  - alignment probabilities p(**a**|**e**,**f**)
  - p(**f**|**e**)

p(**a**,**f**|**e**)

- …the probability that a certain translation **f** with alignment **a** will be generated, given **e**.
- Assume we have all model parameters. Then we can compute p(**a**,**f**|**e**)

p(**a**,**f**|**e**)

- Approximation:

$$p(\mathbf{a},\mathbf{f} \mid \mathbf{e}) \approx \prod_{i=1}^{L} p_n(\varphi_i \mid e_i) \prod_{j=1}^{M} p_t(f_j \mid e_{a(j)}) \prod_{\{j \mid a(j) \neq 0\}} p_p(j \mid a(j), L, M)$$

- Still missing:
  - probability for φ_0
  - φ_0! ways to arrange spurious words in free slots
  - φ_i! ways to end up in alignment $e_i \leftrightarrow (f_1, f_2, \ldots, f_{\varphi i})$

p(**a**,**f**|**e**)

- Finally:

$$p(\mathbf{a},\mathbf{f} \mid \mathbf{e}) = \prod_{i=1}^{L} p_n(\varphi_i \mid e_i) \prod_{j=1}^{M} p_t(f_j \mid e_{a(j)}) \prod_{j:a(j)\neq 0} p_p(j \mid a(j), L, M)$$

$$\binom{M-\varphi_0}{\varphi_0} (1-p_1)^{M-2\varphi_0} \, p_1^{\varphi_0} \prod_{i=0}^{L} \varphi_i! \frac{1}{\varphi_0!}$$

- …depends on model parameters only.

And now???

- To estimate the model parameters $\boldsymbol{\theta}$, we need p(**a**,**f**|**e**)
- To estimate p(**a**,**f**|**e**), we need the model parameters $\boldsymbol{\theta}$

Expectation Maximization

- The EM-Algorithm is an iterative procedure to compute the Maximum Likelihood (ML) estimate in the presence of hidden or missing data
- We search the ML for:

$$p(\mathbf{f} \mid \theta, \mathbf{e}) = \sum_{\mathbf{a}} p(\mathbf{a}, \mathbf{f} \mid \theta, \mathbf{e})$$

- Hidden data **a**

Expectation Maximization

$$\theta = \arg\max_{\theta} \sum_{\mathbf{a}} p(\mathbf{a} \mid \mathbf{f}, \theta_n, \mathbf{e}) \ln p(\mathbf{a}, \mathbf{f} \mid \theta, \mathbf{e})$$

- Expectation: Estimate p(**a**|**f**,**e**) using model parameters θ_n from the previous iteration
- Maximization: Estimate optimal model parameters θ_{n+1} using the estimated alignment probabilities

Problems with Model 3 Training

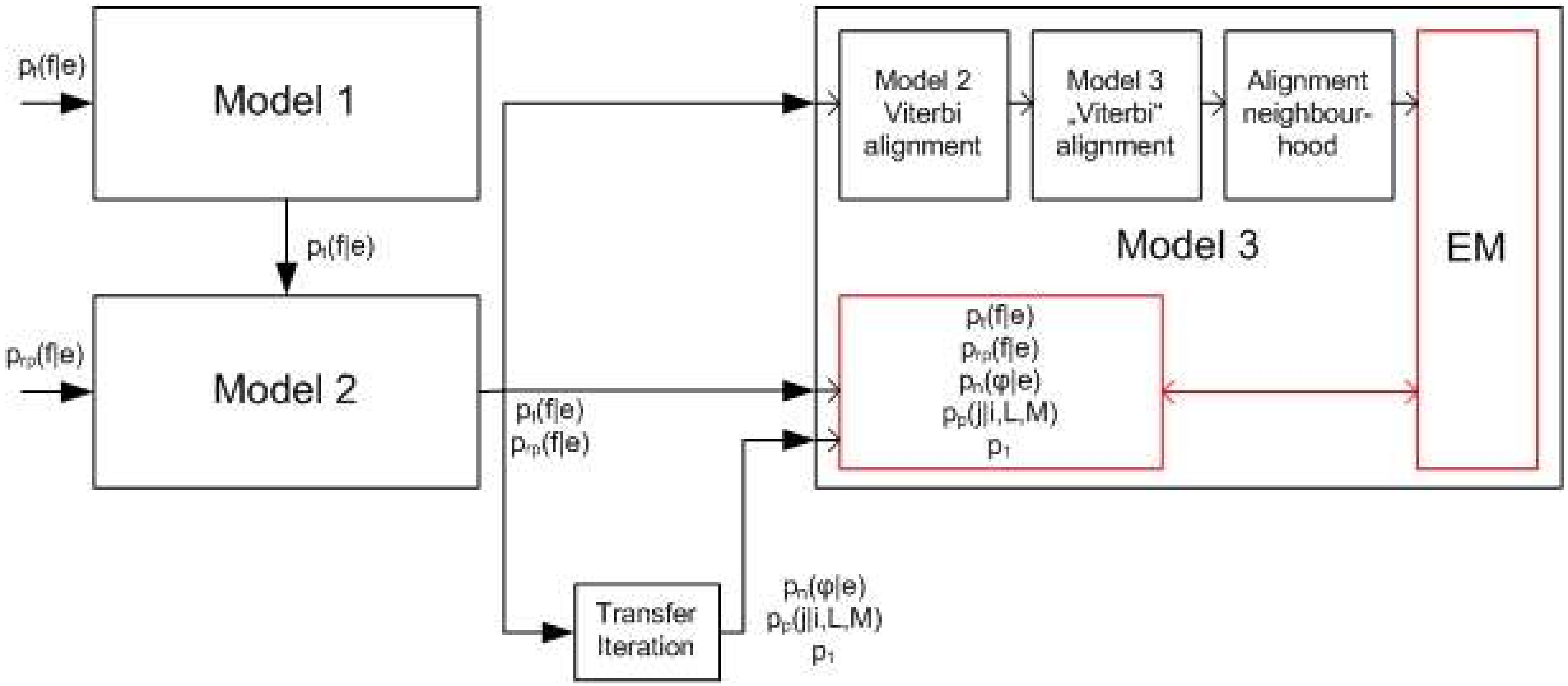
- EM depends on parameter initialization and will find local optimum only
- EM needs to iterate over all possible alignments of a sentence pair. A sentence pair consisting each of 20 words has $(21)^{20} = 2.7822e+026$ possible alignments

Solution for Model 3 Training

- Initialize θ_1 with a *good guess*
- Don‘t iterate over all possible alignments, but only over a *subset of likely alignments*

- Use simpler models (model 1 and 2) to get these guesses

Solution for Model 3 Training



Model 1

- Forget about fertility and distortion:

$$p(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) = \prod_{j=1}^{M} p_t(f_j \mid e_{a(j)})$$

- Computational advantage:

$$p(\mathbf{f} \mid \mathbf{e}) = \sum_{\mathbf{a}} \prod_{j=1}^{M} p_t(f_j \mid e_{a(j)}) = \prod_{j=1}^{M} \sum_{i=0}^{L} p_t(f_j \mid e_i)$$

Model 1

- Using the same trick, $p_t(f|e)$ can be trained with EM without iterating over all possible alignments

Model 2

- Employ only translation and *reverse positioning* probability:

$$p(\mathbf{a},\mathbf{f}\mid\mathbf{e})=\prod_{j=1}^{M} p_t(f_j \mid e_{a(j)})\prod_{j=1}^{M} p_{rp}(\mathbf{a}(j)\mid j,L,M)$$

- Again, we can use the same trick:

$$p(\mathbf{f}\mid\mathbf{e})=\sum_{\mathbf{a}}\prod_{j=1}^{M} p_t(f_j \mid e_{a(j)})p_{rp}(\mathbf{a}(j)\mid j,L,M)=$$

$$\prod_{j=1}^{M}\sum_{i=0}^{L} p_t(f_j \mid e_i)p_{rp}(i\mid j,L,M)$$

Model 2

- Efficient estimation of p_t and p_{rp} possible

- Most probable (Viterbi) alignment of a sentence:

$$\mathbf{a}(j) = \arg\max_i p_t(f_j \mid e_i) p_{rp}(i \mid j, L, M) \quad \forall j$$

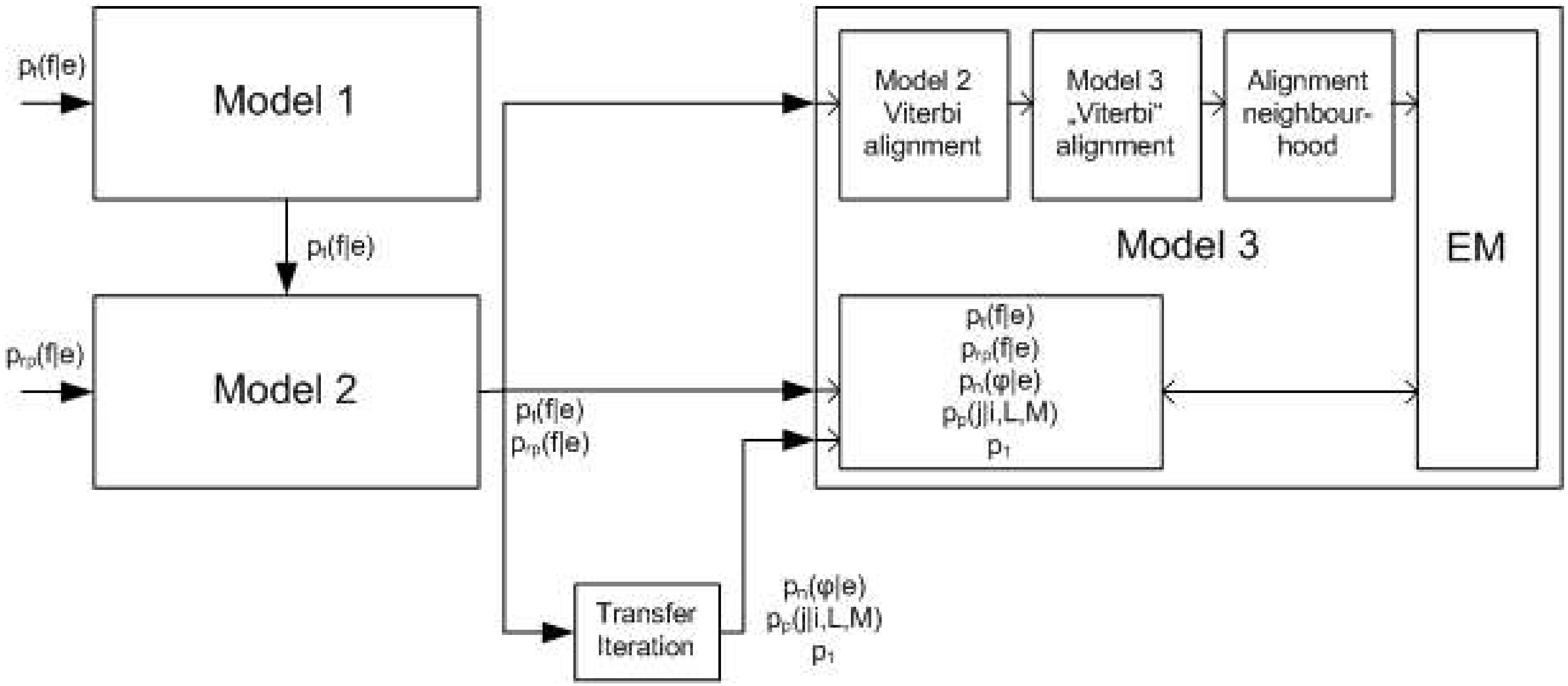
Model 3 „Viterbi“ Alignment

- Model 2 Viterbi Alignment might be suboptimal in the sense of model 3
- Hillclimbing: iterative and quick method to find a better alignment. Replace **a** with neighbour b(**a**) which maximizes p(b(**a**)|**e**,**f**).
- Solution might still be suboptimal
- e.g.: **a** = (1, 2, 3)

  b(**a**) = (1,2,1) ...*differs by one move*

  b(**a**) = (1,3,2) ...*differs by one swap*

IBM Model 1-3



p_t(f|e)
Model 1
p_t(f|e)
p_rp(f|e)
Model 2
p_t(f|e)
p_rp(f|e)
Model 2 Viterbi alignment
Model 3 „Viterbi" alignment
Alignment neighbour-hood
Model 3
EM
p_t(f|e)
p_rp(f|e)
p_n(φ|e)
p_p(j|i,L,M)
p_1
Transfer Iteration
p_n(φ|e)
p_p(j|i,L,M)
p_1

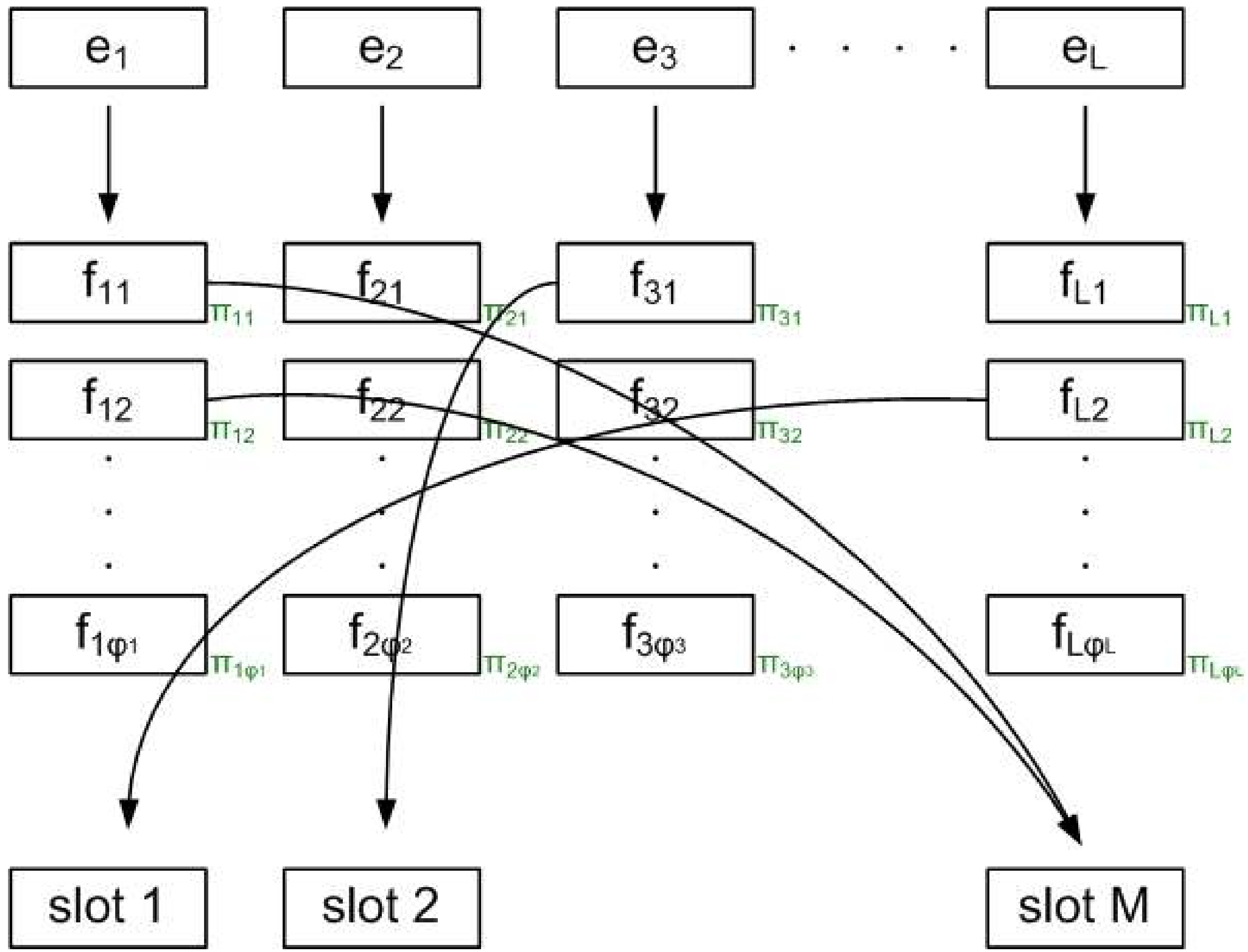
2 More Problems with Model 3

- $p_p(\pi|i,L,M)$ does not depend on words, just on positions. *Unrealistic*
- $p_p(\pi|i,L,M)$ has no memory. *Deficiency*

Deficiency

- $p_p(\pi|i,L,M)$ does not consider values assigned to earlier words -> multiple words might get same position
- $p(\mathbf{a},\mathbf{f}|\mathbf{e})$ of Model 3 wastes some of its probability mass on impossible events, i.e. generalized strings

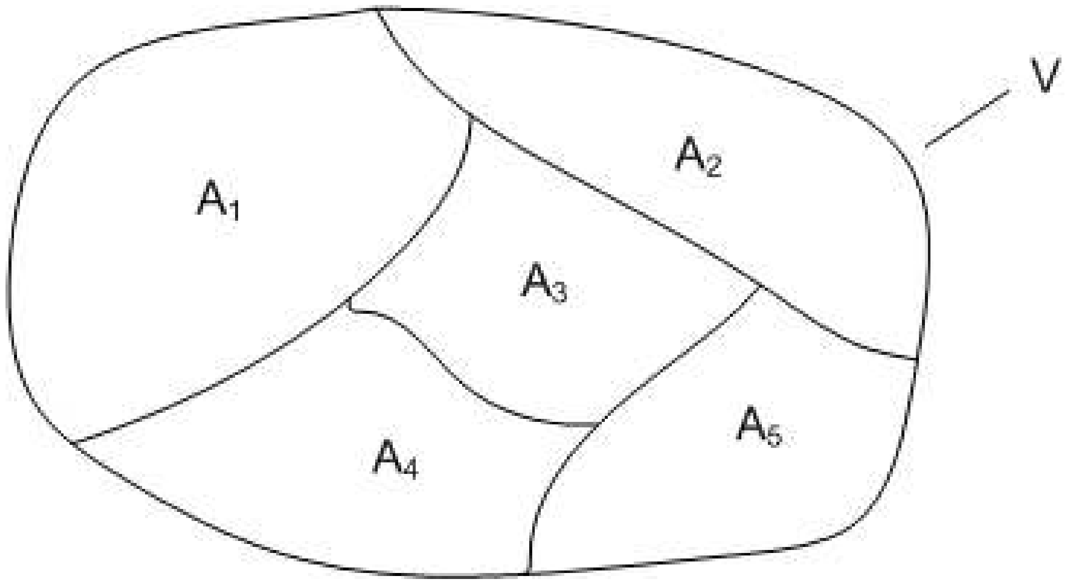
Deficiency



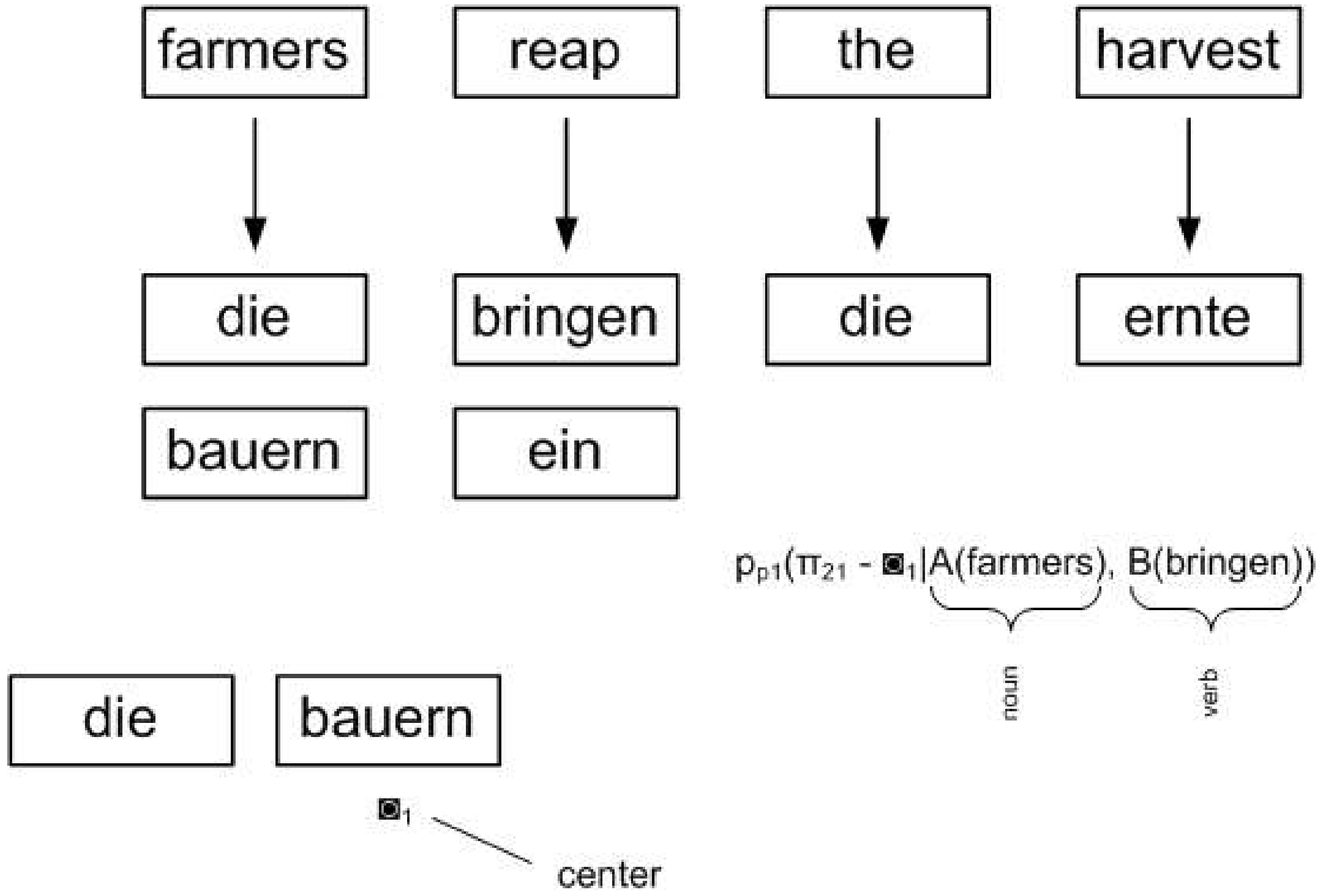
e1
e2
e3
eL
f11
f21
f31
fL1
f12
f22
f32
fL2
f1φ1
f2φ2
f3φ3
fLφL
π11
π21
π31
πL1
π12
π22
π32
πL2
π1φ1
π2φ2
π3φ3
πLφL
slot 1
slot 2
slot M

Model 4

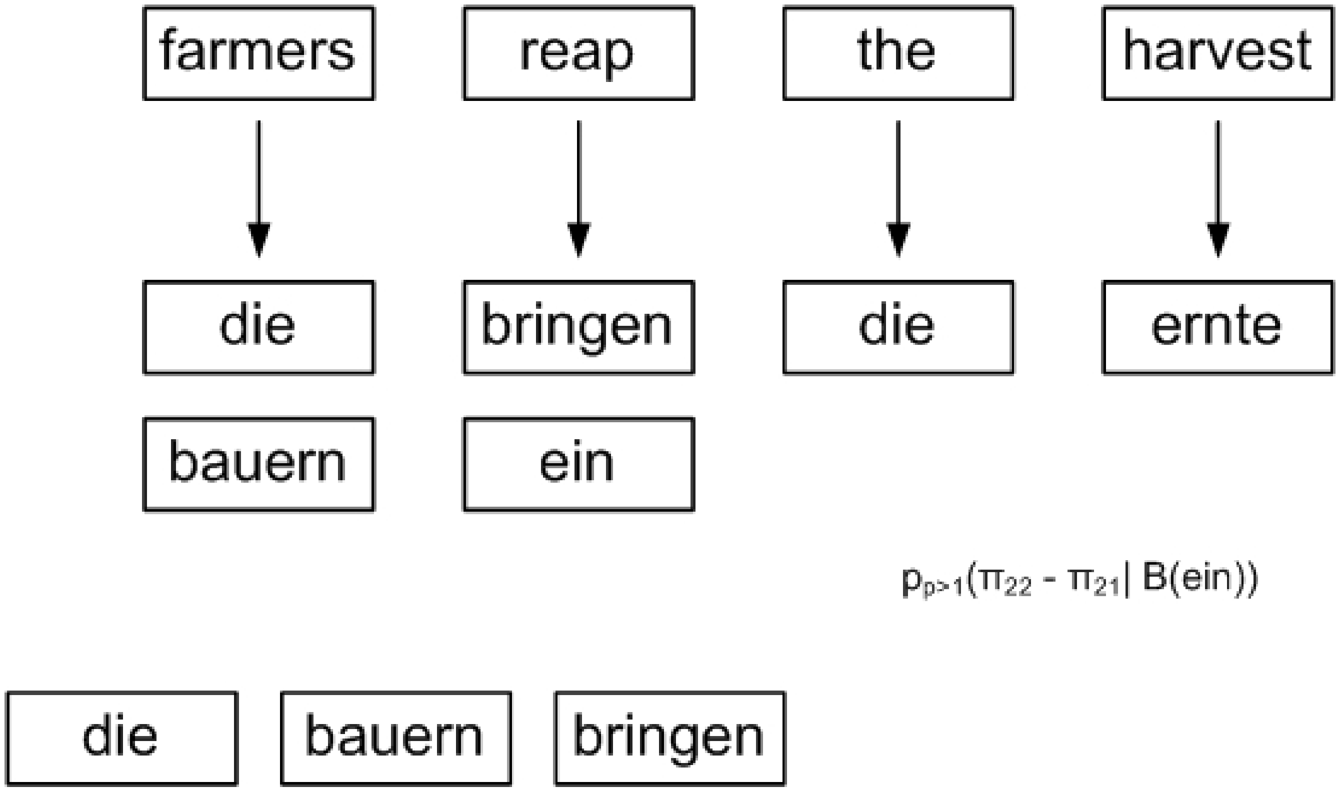
- Class based positioning of words
- e.g.: A_1 verb, A_2 noun, A_3 adjective etc…



Model4 Positioning



Model4 Positioning



Model 5

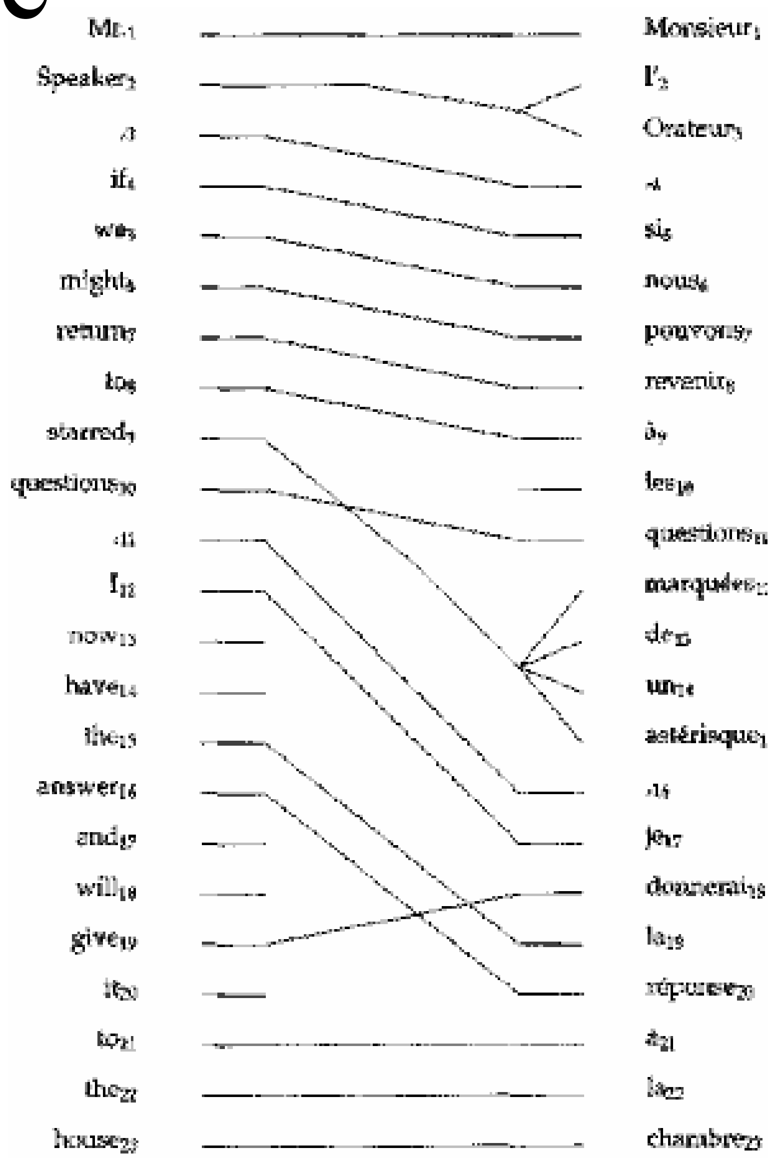
- Similar to Model 4, but prevents generation of invalid strings

Conclusion

- Model 1-5 provide effective means for obtaining word-by-word alignments of translation

Example

- Best out of 5.6×10^{31} alignments



Conclusion

- Model 1-5 provide effective means for obtaining word-by-word alignments of translation
- Single parent constraint in alignments too simplistic
- Relation between several grammatical forms of a word is ignored.

Example

- Different conjugations of one verb are treated separately

should

f	$t(f \mid e)$	ϕ	$n(\phi \mid e)$
devrait	0.330	1	0.649
devraient	0.123	0	0.336
devrions	0.109	2	0.014
faudrait	0.073		
faut	0.058		
doit	0.058		
aurait	0.041		
doivent	0.024		
devons	0.017		
devrais	0.013		

References

- Berger, Brown et al., *The Candide System for Machine Translation*, 1994
- Knight, *A Statistical MT Tutorial Workbook*, 1999
- Brown, Pietra et al., *The Mathematics of Statistical Machine Translation: Parameter Estimation*, 1993
- Borman, *The Expectation Maximization Algorithm - A Short Tutorial*, 2004