Memory-Based Parsing for German

Sandra Kübler

kuebler@sfs.uni-tuebingen.de

Seminar für Sprachwissenschaft

University of Tübingen

Germany

German Tree

(1)

ich	habe	hier	übrigens	auch	schon	mir	Unterlagen	zuschicken	lassen	von
I	have	here	by the way	also	already	to me	brochures	sent	let	of

verschiedenen	Hotels
different	hotels

’by the way here I have also had brochures sent to me about different hotels already’

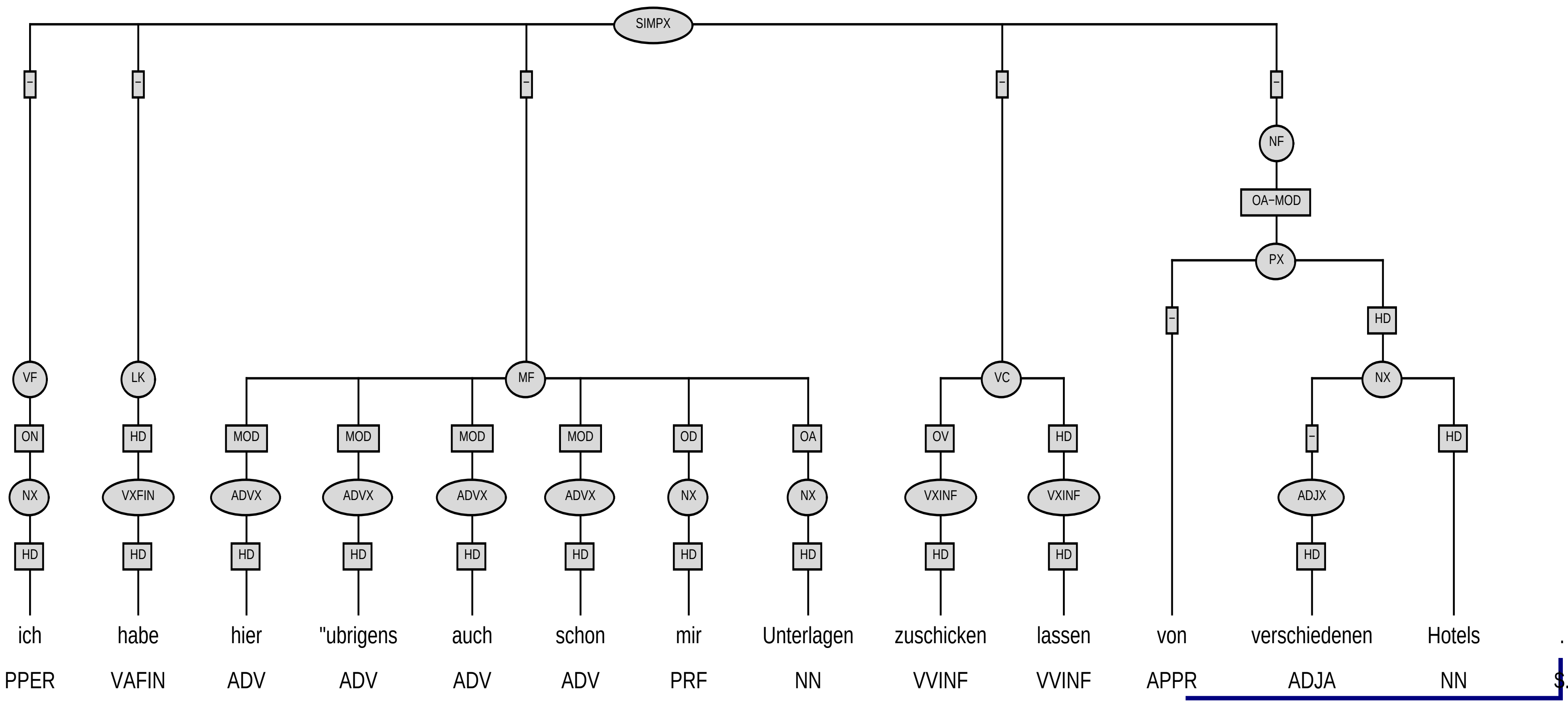
German Tree

(1)

ich	habe	hier	übrigens	auch	schon	mir	Unterlagen	zuschicken	lassen	von
I	have	here	by the way	also	already	to me	brochures	sent	let	of

verschiedenen	Hotels
different	hotels

’by the way here I have also had brochures sent to me about different hotels already’



Memory-Based Learning

- selection of the most likely class: finding the *most similar* instance in the instance base (collection of training instances)

Memory-Based Learning

- selection of the most likely class: finding the *most similar* instance in the instance base (collection of training instances)
- **no** abstraction from data: original data is always accessible

Memory-Based Learning

- selection of the most likely class: finding the *most similar* instance in the instance base (collection of training instances)
- **no** abstraction from data: original data is always accessible
- very appropriate for language learning: can deal with irregularities, subregularities, etc.

Memory-Based Learning

- selection of the most likely class: finding the *most similar* instance in the instance base (collection of training instances)
- **no** abstraction from data: original data is always accessible
- very appropriate for language learning: can deal with irregularities, subregularities, etc.
- intelligence = good similarity metric, good weighting of features

NP detection

NP detection

classification task: decide for each word: NP or not

instance: word + context

Memory-Based Learning – Example

NP detection

classification task: decide for each word: NP or not
instance: word + context

instance base (= training data):

prev. w.	**word**	next w.	prev. POS	POS	next POS	class
will	book	two	md	vb	cd	no-NP
book	two	flights	vb	cd	nns	NP
two	flights	for	cd	nns	in	NP

NP detection

classification task: decide for each word: NP or not
instance: word + context

instance base (= training data):

prev. w.	**word**	next w.	prev. POS	POS	next POS	class
will	book	two	md	vb	cd	no-NP
book	two	flights	vb	cd	nns	NP
two	flights	for	cd	nns	in	NP

test instance:

prev. w.	**word**	next w.	prev. POS	POS	next POS	class
two	cars	for	cd	nns	in	**???**

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure
- example:

I saw the man with the white hat

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure
- example:

NP:	[NP]		[NP	NP]		[NP		NP]
	I	saw	the	man	with	the	white	hat

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure
- example:

PP:					[PP			PP]
NP:	[NP]		[NP	NP]		[NP		NP]
	I	saw	the	man	with	the	white	hat

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure

- example:

VP:		[VP						VP]
PP:					[PP			PP]
NP:	[NP]		[NP	NP]		[NP		NP]
	I	saw	the	man	with	the	white	hat

Seminar für Sprachwissenschaft
EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure
- example:

CL.:	[S							S]
VP:		[VP						VP]
PP:					[PP			PP]
NP:	[NP]		[NP	NP]		[NP		NP]
	I	saw	the	man	with	the	white	hat

Typical MBL Parsing

- cascaded classifiers: NP level, PP level, VP level, clause level, function argument structure
- example:

func:	SB			DO	–			
CL.:	[S							S]
VP:		[VP						VP]
PP:					[PP			PP]
NP:	[NP]		[NP	NP]		[NP		NP]
	I	saw	the	man	with	the	white	hat

Problems with Cascades

- recursive structures such as complex clauses:

S_1:	[S							S]
S_2:			[S		S]			
	the	man	who	bought	everything	made	a	fortune

Problems with Cascades

- recursive structures such as complex clauses:

S_1:	[S							S]
S_2:			[S		S]			
	the	man	who	bought	everything	made	a	fortune

- independence assumption:

func:	SB			SB	–			
	I	saw	the	man	with	the	white	hat

Problems with Cascades

- recursive structures such as complex clauses:

S_1:	[S							S]
S_2:			[S		S]			
	the	man	who	bought	everything	made	a	fortune

- independence assumption:

func:	SB			SB	–			
	I	saw	the	man	with	the	white	hat

- in German: long-distance relations:

ON		OD	OA				OA-MOD
ich	habe	mir	Unterlagen	zuschicken	lassen	von	Hotels
I	have	to me	brochures	sent	let	of	hotels

New Approach

- new idea: find most similar tree in instance base *in one step*

New Approach

- new idea: find most similar tree in instance base *in one step*
- for new sentence: `grundsätzlich habe ich Zeit` (basically I have time)
  find training sentence: `da habe ich Zeit` (I have time then)

New Approach

- new idea: find most similar tree in instance base *in one step*
- for new sentence: `grundsätzlich habe ich Zeit` (basically I have time)
  find training sentence: `da habe ich Zeit` (I have time then)
- problem: how define similarity?

New Approach

- new idea: find most similar tree in instance base *in one step*
- for new sentence: `grundsätzlich habe ich Zeit` (basically I have time)
  find training sentence: `da habe ich Zeit` (I have time then)
- problem: how define similarity?
- problem: what if structure of most similar tree is not identical?

Adapting the Most Similar Tree

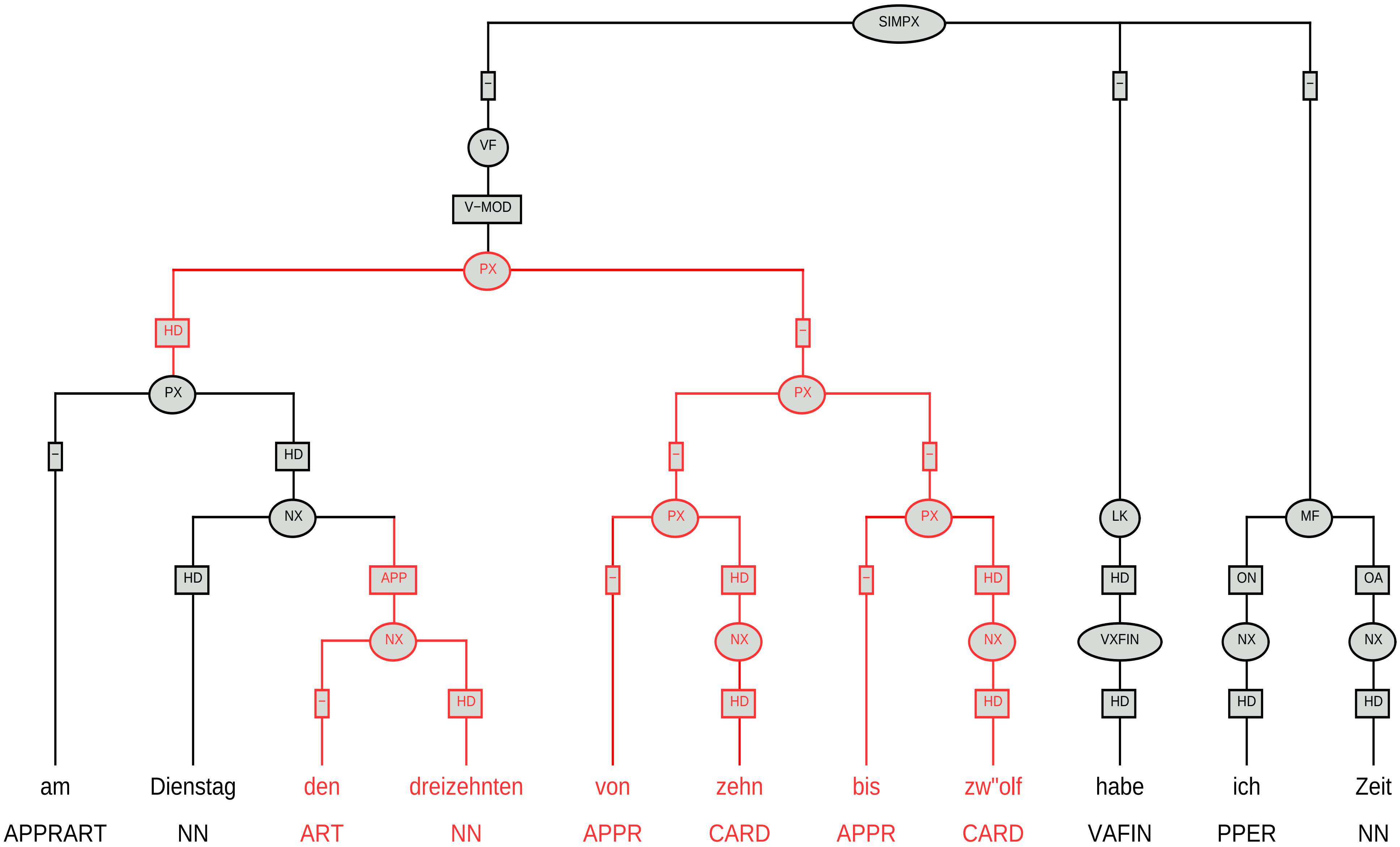
- very conservative approach: only *delete* parts from retrieved tree, *never add*!

Adapting the Most Similar Tree

- very conservative approach: only *delete* parts from retrieved tree, *never add*!
- example: new sentence `am Dienstag habe ich Zeit` (on Tuesday I have time)
  training sentence: `am Dienstag den dreizehnten von zehn bis zwölf habe ich Zeit` (on Tuesday the thirteenth from ten to twelve I have time)

Adapting the Most Similar Tree

- tree:



Preprocessing – Example

- sentence: `da muß ich leider zu einem Treffen nach Köln` (unfortunately I have to go to Cologne for a meeting)

Preprocessing – Example

- sentence: `da muß ich leider zu einem Treffen nach Köln` (unfortunately I have to go to Cologne for a meeting)

```
[simpx
    [da da]
    [vmfi nmuß]
    [nx4
        [pper ich]]
    [advx
        [adv leider]]
    [px
        [zu zu]
        [nx1
            [art einem]
            [nn Treffen]]]
    [px
        [appr nach]
        [nx1
            [ne Köln]]]]
```

TnT (Thorsten Brants)
tagfi xing (Steve Abney
CASS (Steve Abney)

Weighting Features?

Standard weighting techniques are impossible:

Weighting Features?

Standard weighting techniques are impossible:

- sequential information more important:
  `DET N V ADJ` vs. `ADJ, DET, N, V`

Weighting Features?

Standard weighting techniques are impossible:

- sequential information more important:
  `DET N V ADJ` vs. `ADJ, DET, N, V`
- no windowing approach: find tree in one step $\Rightarrow$ use all features $\Rightarrow$ different number of features

Weighting Features?

Standard weighting techniques are impossible:

- sequential information more important: `DET N V ADJ` vs. `ADJ, DET, N, V`
- no windowing approach: find tree in one step $\Rightarrow$ use all features $\Rightarrow$ different number of features
- selecting a complete tree: very difficult task $\Rightarrow$ need **all words** and **all other types of information** as features

Weighting Features?

Standard weighting techniques are impossible:

- sequential information more important:
  `DET N V ADJ` vs. `ADJ, DET, N, V`
- no windowing approach: find tree in one step ⇒ use all features ⇒ different number of features
- selecting a complete tree: very difficult task ⇒ need **all words** and **all other types of information** as features
- suggested solution: backing off strategy instead of weighting

The Parsing System

backing off: 2 main modules:

The Parsing System

backing off: 2 main modules:

- search in a word prefix trie
  allowing the omission of words or phrases / chunks in input sentence as well as training sentences

The Parsing System

backing off: 2 main modules:

- search in a word prefix trie
  allowing the omission of words or phrases / chunks in input sentence as well as training sentences
- backing off to less reliable information:

  1. search for POS sequence

The Parsing System

backing off: 2 main modules:

- search in a word prefix trie
  allowing the omission of words or phrases / chunks in input sentence as well as training sentences
- backing off to less reliable information:
  1. search for POS sequence
  2. search for longer trees and shorten them

The Parsing System

backing off: 2 main modules:

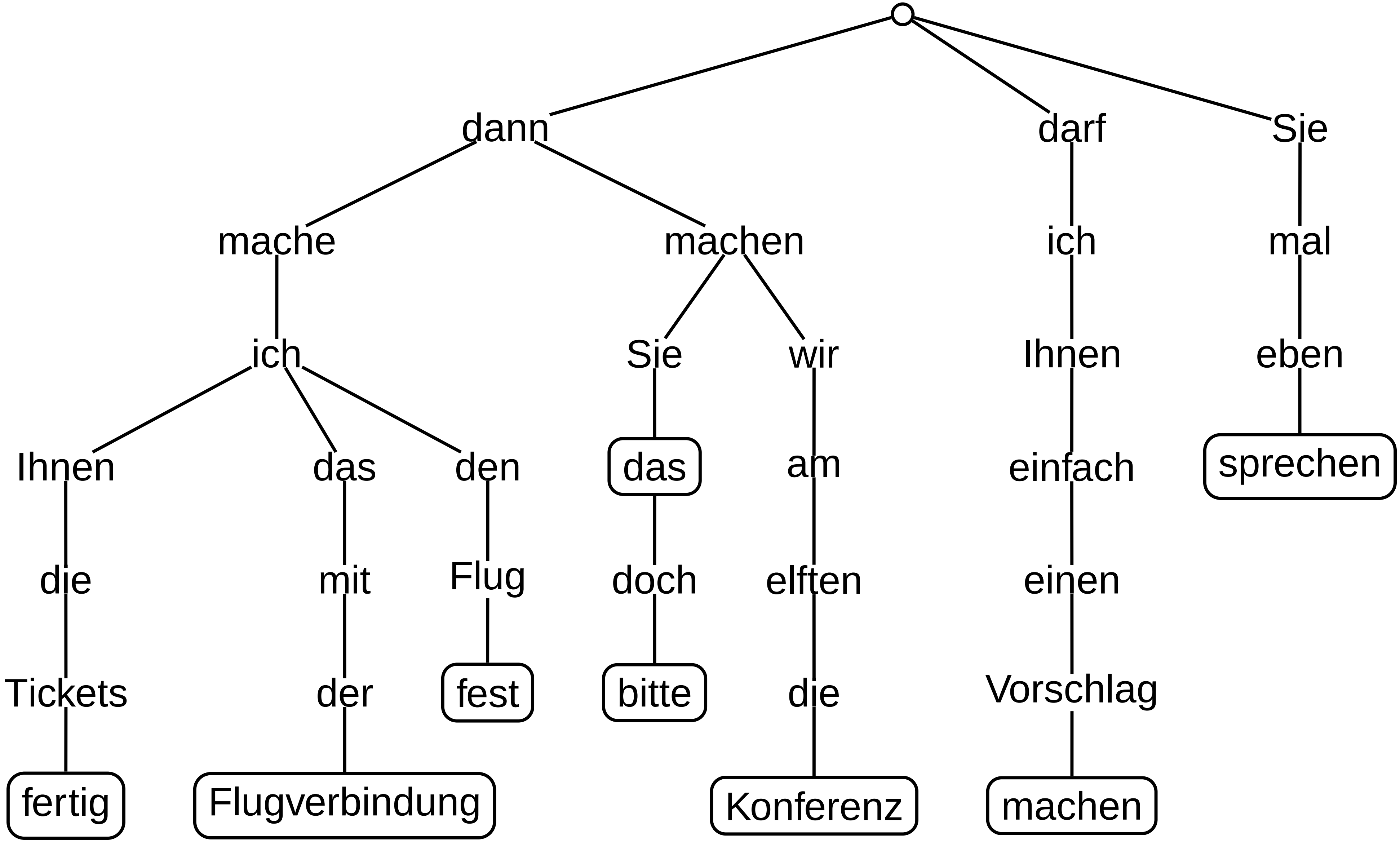
- search in a word prefix trie
  allowing the omission of words or phrases / chunks in input sentence as well as training sentences
- backing off to less reliable information:
  1. search for POS sequence
  2. search for longer trees and shorten them
  3. search for chunk sequences with matching heads

The Parsing System

backing off: 2 main modules:

- search in a word prefix trie
  allowing the omission of words or phrases / chunks in input sentence as well as training sentences
- backing off to less reliable information:
  1. search for POS sequence
  2. search for longer trees and shorten them
  3. search for chunk sequences with matching heads
  4. search for chunk sequences (without matching heads)

The Word Trie

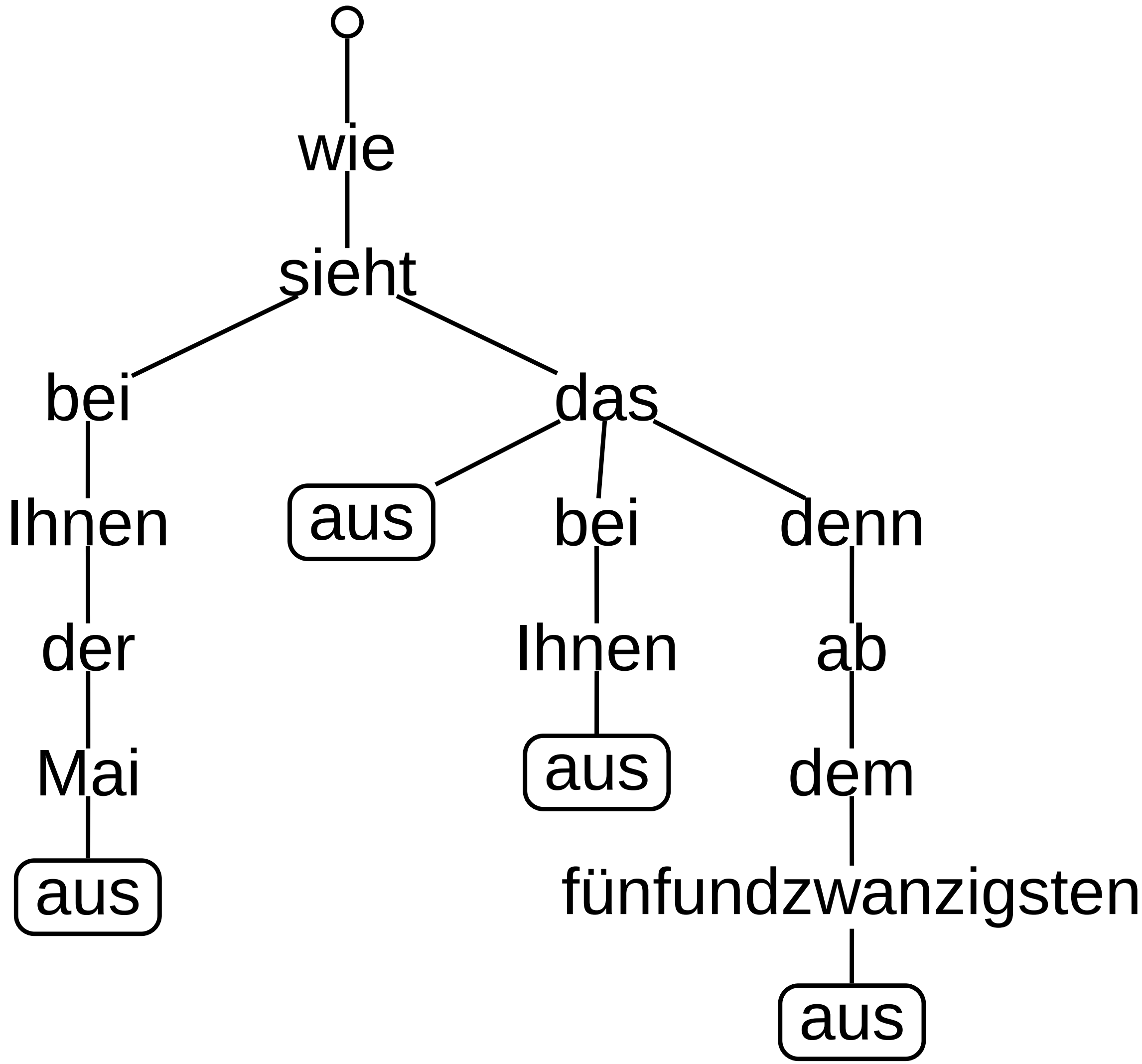


The Omission of Words in the Trie

sentence: `wie sieht das ab dem fünfundzwanzigsten aus` (how does that look from the twenty fifth on)

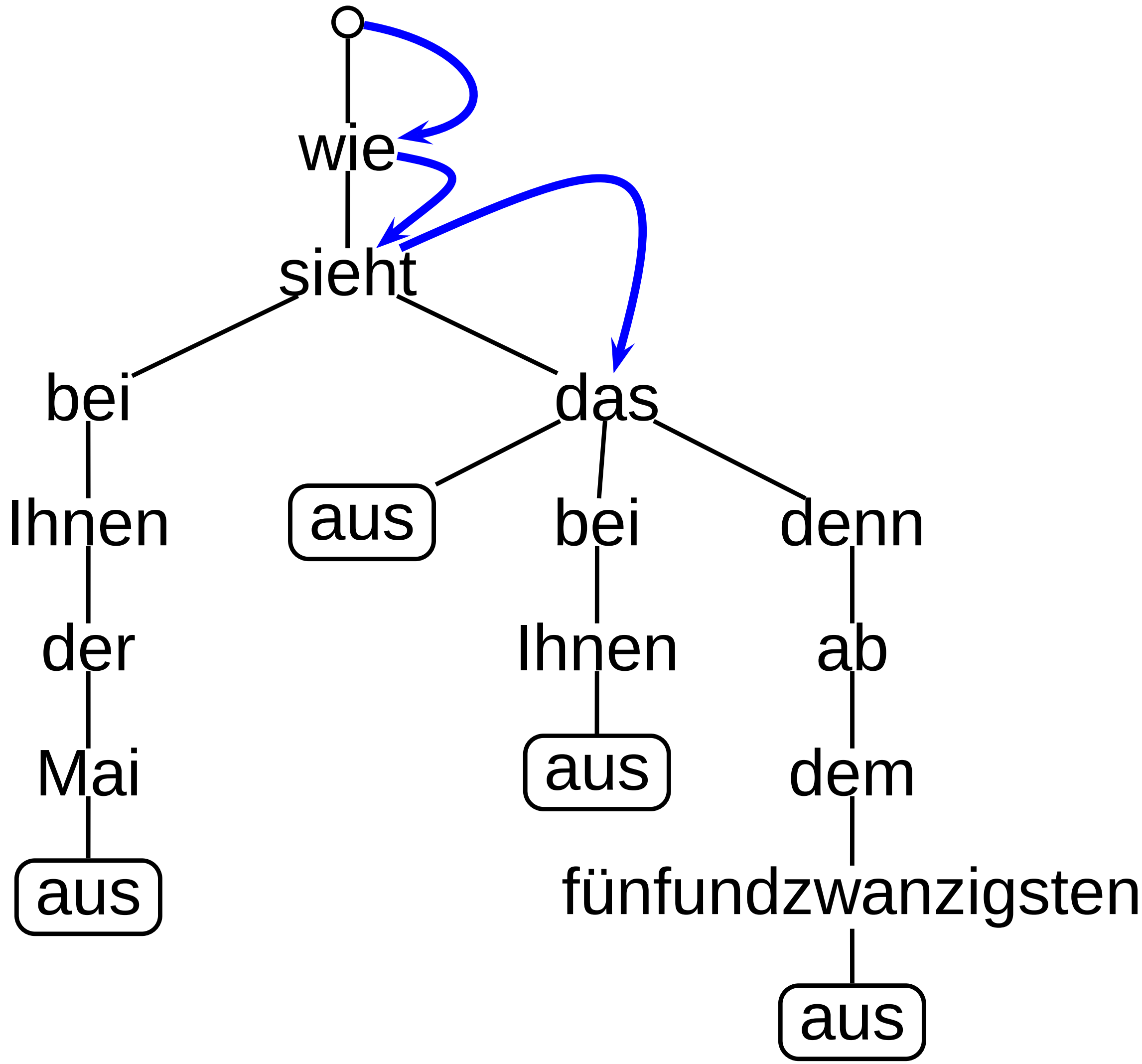
The Omission of Words in the Trie

sentence: `wie sieht das ab dem fünfundzwanzigsten aus` (how does that look from the twenty fifth on)



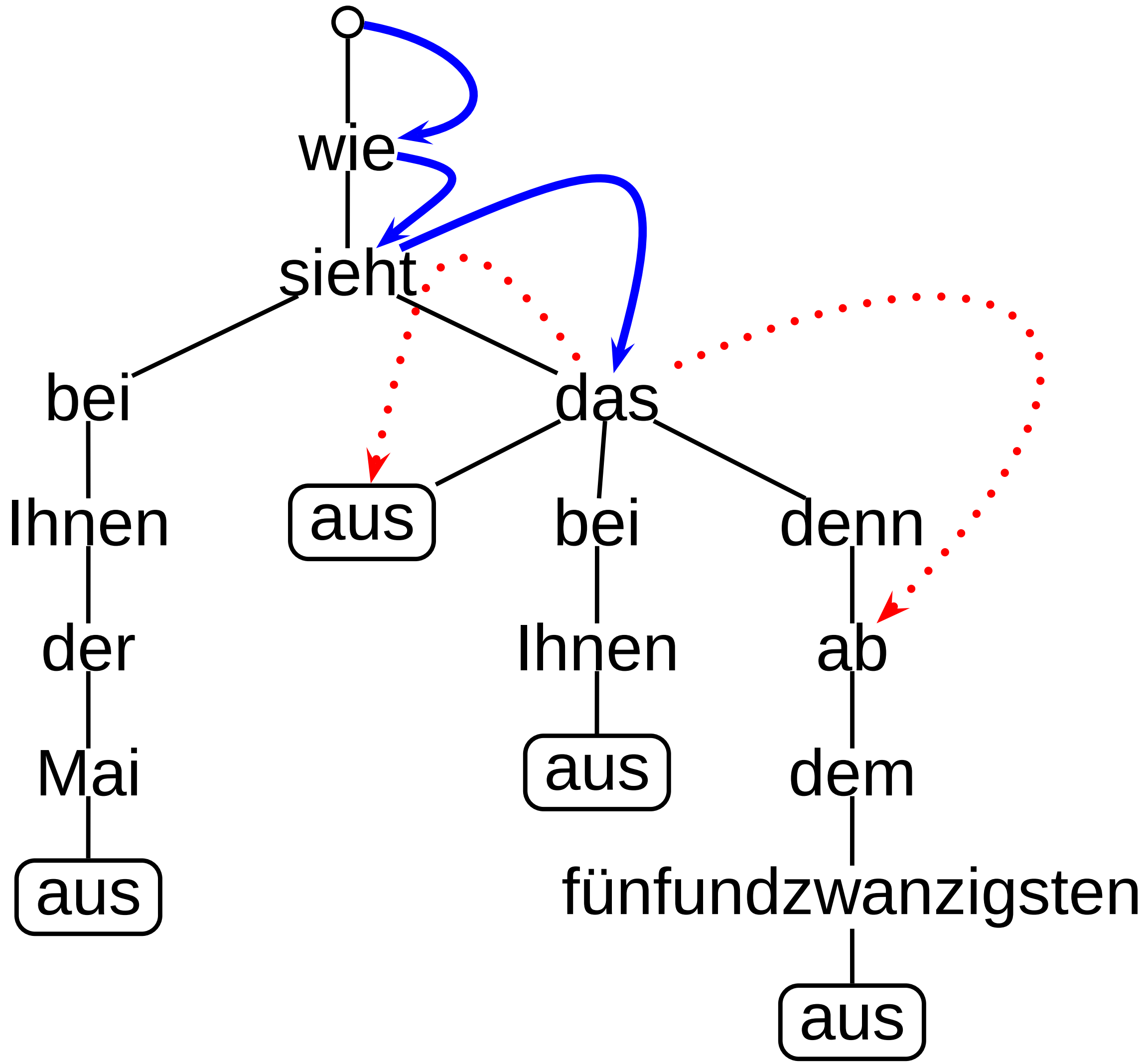
The Omission of Words in the Trie

sentence: `wie sieht das ab dem fünfundzwanzigsten aus` (how does that look from the twenty fifth on)



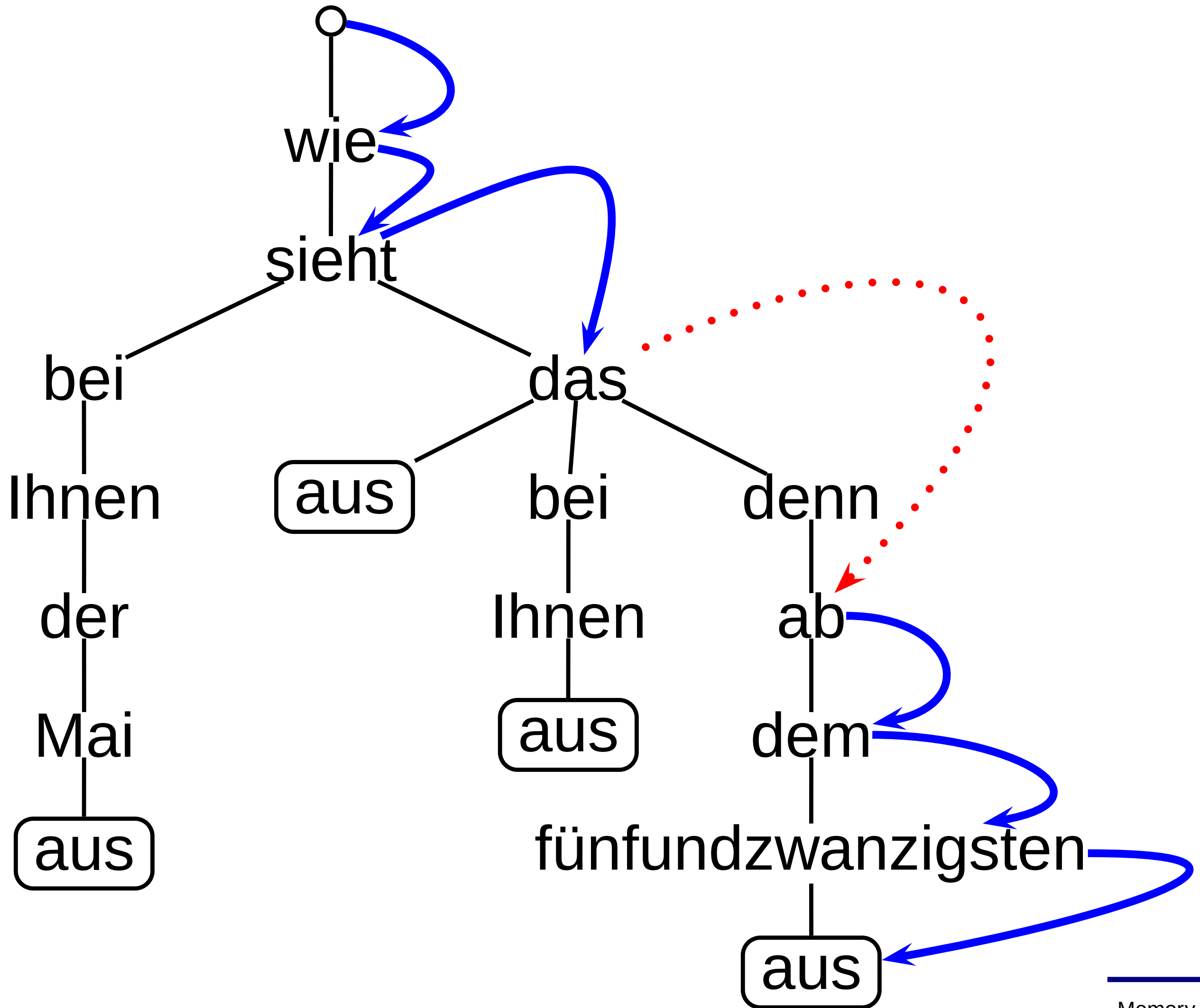
The Omission of Words in the Trie

sentence: `wie sieht das ab dem fünfundzwanzigsten aus` (how does that look from the twenty fifth on)



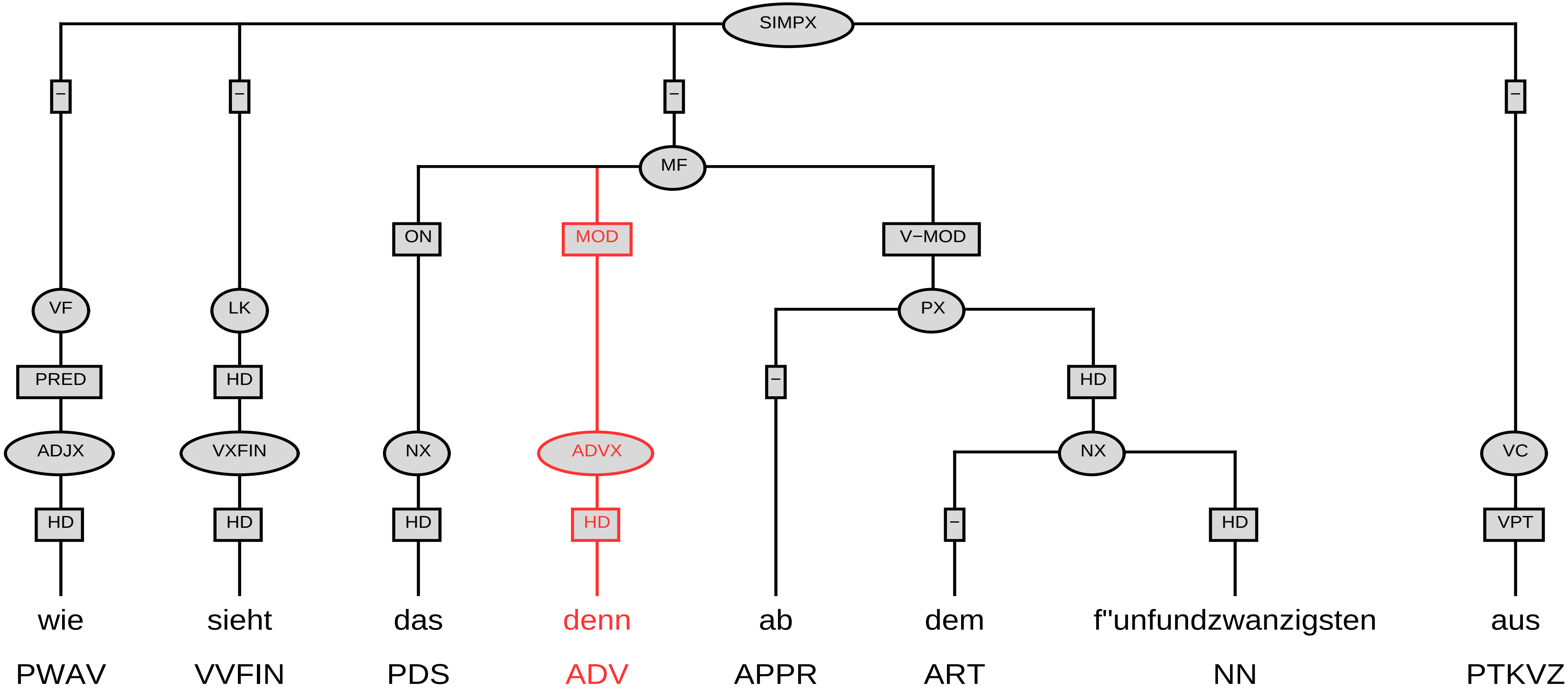
The Omission of Words in the Trie

sentence: `wie sieht das ab dem fünfundzwanzigsten aus` (how does that look from the twenty fifth on)



Seminar für Sprachwissenschaft
EBERHARD KARLS UNIVERSITÄT TÜBINGEN

The Resulting Parse



Backing Off – Matching Chunks

- input sentence:
  [simpx [px `ab Donnerstag`] [fcop `bin`] [nx4 `ich`] [advx `wieder`] [advx `hier`]]

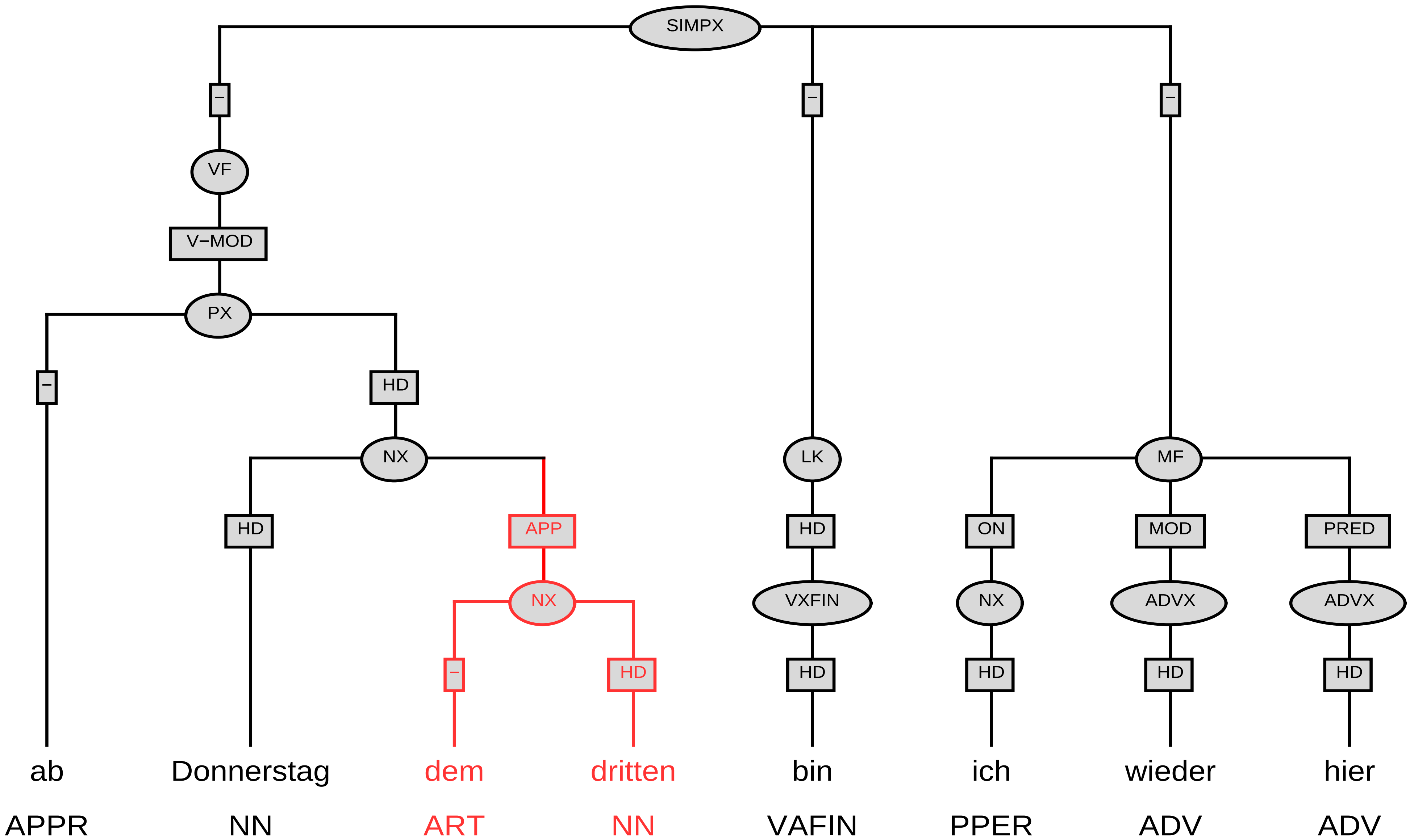
Backing Off – Matching Chunks

- input sentence:
  [simpx [px ab Donnerstag] [fcop bin] [nx4 ich] [advx wieder] [advx hier]]
- identical chunk structure from training data:
  [simpx [px ab Donnerstag dem dritten] [fcop bin] [nx4 ich] [advx wieder] [advx hier]]

Backing Off – Matching Chunks

- input sentence:
  [simpx [px ab Donnerstag] [fcop bin] [nx4 ich] [advx wieder] [advx hier]]
- identical chunk structure from training data:
  [simpx [px ab Donnerstag dem dritten] [fcop bin] [nx4 ich] [advx wieder] [advx hier]]
- identical chunk structure from training data:
  [simpx [px nach einer langen Woche] [fcop sind] [nx4 Sie] [advx wieder] [advx zurück]]
  (after a long week you will be back again)

Tree Modification



Evaluation

recall (syntactic)	82.45%
precision (syntactic)	87.25%
F_1	84.78
recall (+ func. cat.)	71.72%
precision (+ func. cat.)	75.79%
F_1	73.70
unattached const. in recall	7.14%
unattached const. in precision	7.60%
func. recall (att. const.)	95.31%
func. precision (att. const.)	95.21%

Leave-One-Out Evaluation

using **5 000 test sentences:**

recall (syntactic)	85.15%
precision (syntactic)	89.34%
F_1	87.19
recall (+ func. cat.)	76.00%
precision (+ func. cat.)	79.65%
F_1	77.78
func. recall (att. const.)	96.56%
func. precision (att. const.)	96.48%

Conclusion

- new definition of parsing task: find complete most similar tree; adapt this tree to input

Conclusion

- new definition of parsing task: find complete most similar tree; adapt this tree to input
- needs: POS tagger, chunk parser, treebank

Conclusion

- new definition of parsing task: find complete most similar tree; adapt this tree to input
- needs: POS tagger, chunk parser, treebank
- uses a backing off strategy instead of (standard) feature weighting

Conclusion

- new definition of parsing task: find complete most similar tree; adapt this tree to input
- needs: POS tagger, chunk parser, treebank
- uses a backing off strategy instead of (standard) feature weighting
- results still worse results than state of the art statistical parsers

Conclusion

- new definition of parsing task: find complete most similar tree; adapt this tree to input
- needs: POS tagger, chunk parser, treebank
- uses a backing off strategy instead of (standard) feature weighting
- results still worse results than state of the art statistical parsers
- future work: increase training data, include morphological information, use different (ML) chunk parser, evaluate on different data sets