Institut für Computerlinguistik, Uni Zürich: Effiziente Analyse unbeschränkter Texte

Vorlesung 8: Pruning and Beam Search

Gerold Schneider

Institute of Computational Linguistics, University of Zurich
Department of Linguistics, University of Geneva
gschneid@ifi.unizh.ch

December 22, 2005

Contents

1 The Need for Pruning

- Ambiguity means that thousands of partial structures can be built up for real-world sentences
- Continuing all partial structures leads to an immense search space
- Very improbable partial structures rarely lead to the best global structure
- The local max. does often not lead to the global max. → some ambiguity needs to be kept
- Extremely long structures are very error-prone, so that discarding liberally does not harm but allows the parser to find some analyses in a reasonabel amount of time
- If too restrictive pruning makes it impossible to find a global solution, partial structures still contain much information

2 Probabilities and Back-Offs

The basic fully lexicalized model (slightly adapted for most relation):

$$p(R|right, a, b) \cong \frac{\#(R, right, a, b)}{\#(right, a, b)} \tag{1}$$

Sparse data $\longrightarrow$ zero counts in most cases $\longrightarrow$?

Hindle and Rooth's Smoothing (for R=noun/verb-attach): Noun attachment estimator from unambiguous noun (n) and prep (p) occurrences

$$p(R = noun - attach, p|v, n) \approx p(p|n) \cong \frac{\#(n, p)}{\#(n)} \tag{2}$$

Their smoothed noun attachment estimator

$$p(R = noun - attach, p|v, n) \approx p(p|n) \cong \frac{\#(n,p) + \frac{\#\sum n,p}{\#\sum n}}{\#(n) + 1} \tag{3}$$

The smoothing, prep-only based estimate is weighted by $\#\sum n$:

$$\frac{\frac{\#\sum n,p}{\#\sum n}}{\frac{\#\sum n}{\#\sum n}} \tag{4}$$

$\#n, p$ large $\longrightarrow$ smoothing estimate has very little influence

$\#n, p$ small $\longrightarrow$ smoothing estimate has main influence

Collins 1996 back-off:

$$
\begin{aligned}
& p(R|\langle w_j, wtag_j\rangle \wedge \langle h_j, htag_j\rangle) = \\
& \qquad COUNT(\langle w_j, wtag_j\rangle \wedge \langle h_j, htag_j\rangle) if > 0, else \\
& \qquad COUNT(\langle w_j, wtag_j\rangle \wedge \langle htag_j\rangle) + \\
& \qquad\quad COUNT(\langle wtag_j\rangle \wedge \langle h_j, htag_j\rangle) if > 0, else \\
& \qquad COUNT(\langle wtag_j\rangle \wedge \langle htag_j\rangle)
\end{aligned}
$$

- Either only full or only back-off counts
- low counts are unsmoothed
- Collins stresses that even very low counts have more confidence than the next back-off level

Current back-off hierarchy (to be improved):

```
%%%% PP-Attachment
%% v<pp
stats2(_HTag,FH,_SH,_DTag,DescN,Prep,pobj,P,NP,D,_HC,_OG) :-
    ((vppp(FH,Prep,DescN,Count), coocvppp(FH,Prep,DescN,CoCount),baklev(vppp,0));  % full, backoffs below
     ((nountoclass(DescN,_,DescNClass) -> true;DClass=18),
       vppp_bak_nclass(FH,Prep,DescNClass,Count), coocvppp_bak_nclass(FH,Prep,DescNClass,CoCount), baklev(vppp,1)
     ); % verb & prep & nounclass
     (vppp_bak_verb_prep(FH,Prep,Count), coocvppp_bak_verb_prep(FH,Prep,CoCount), baklev(vppp,2)
     ); % verb & prep
     (verbtoclass(FH,HClass),
       vppp_bak_vclass(HClass,Prep,DescN,Count), coocvppp_bak_vclass(HClass,Prep,DescN,CoCount), baklev(vppp,3)
     ); % verbclass & prep & noun
     ((nountoclass(DescN,_,DescNClass) -> true;DClass=18),verbtoclass(FH,HClass),
       vppp_bak_class(HClass,Prep,DescNClass,Count), coocvppp_bak_class(HClass,Prep,DescNClass,CoCount), baklev(vppp,4)
     ); % verbclass & prep & nounclass
     (vppp_bak_prep_descnoun(Prep,DescN,Count), coocvppp_bak_prep_descnoun(Prep,DescN,CoCount), baklev(vppp,5)
     ); % prep & descnoun
     (vppp_bak_prep(Prep,Count), coocvppp_bak_prep(Prep,CoCount),baklev(vppp,6)
     ); % prep only
     (Count is 0.05, CoCount is 1, baklev(vppp,7)) %% Smoother
    ),
    dist(vppp,D,C,_,TotC), DP is 0.8 + ((C/TotC)*2),
    P is (Count/CoCount),
    NP is ((Count/CoCount)*2)*DP. %% PROB: ATTACHMENT / COOCCURRENCE * # of poss. attachments
```

Sparseness and Decision Points (preview: little pruning)

Verb-PP Backoff decision points			
pobj	0	full	124
	1	verb & prep & nounclass	2624
	2	verb & prep	2631
	3	verbclass & prep & noun	337
	4	verbclass & prep & nounclass	5004
	5	prep & noun	995
	6	prep	4762
	7	NONE	4747

Table 1: Verb-PP Backoff decision points for the Fully Lexicalized, Backed-Off System on Carroll's test suite

Noun-PP Backoff decision points			
modpp	0	full	30
	1	noun & prep & descnounclass	197
	2	nounclass & prep & descnoun	100
	3	noun & prep	208
	4	nounclass & prep	696
	5	prep & descnoun	73
	6	prep	227
	7	NONE	281

Table 2: Noun-PP Backoff decision points for the Fully Lexicalized, Backed-Off System on Carroll's test suite

Sparseness and Decision Points (current)

Verb/Noun-PP Backoff decision points		*pobj*	*modpp*
0	full	79	37
1	verb/noun & prep & nounclass	78	3
2	verb/noun & prep	1503	497
3	verb/nounclass & prep & noun	102	56
4	verb/nounclass & prep & nounclass	831	365
5	prep & noun	117	56
6	prep	490	180
7	NONE	373	120

Table 3: Current Verb/Noun-PP Backoff decision points for the aggressively pruned System on Carroll's test suite

Late backoff decisions are disappointingly frequent ...

But which decisions are successful in the sense that they actually feature in the globally first reading?

"Successful" Verb/Noun-PP Backoff decisions		*pobj*	*modpp*
0	full	26	25
1	verb/noun & prep & nounclass	0	0
2	verb/noun & prep	292	260
3	verb/nounclass & prep & noun	19	20
4	verb/nounclass & prep & nounclass	83	106
5	prep & noun	3	7
6	prep	20	23
7	NONE	0	0

Table 4: Current Verb/Noun-PP Backoff "successful" decision points for the aggressively pruned System on Carroll's test suite

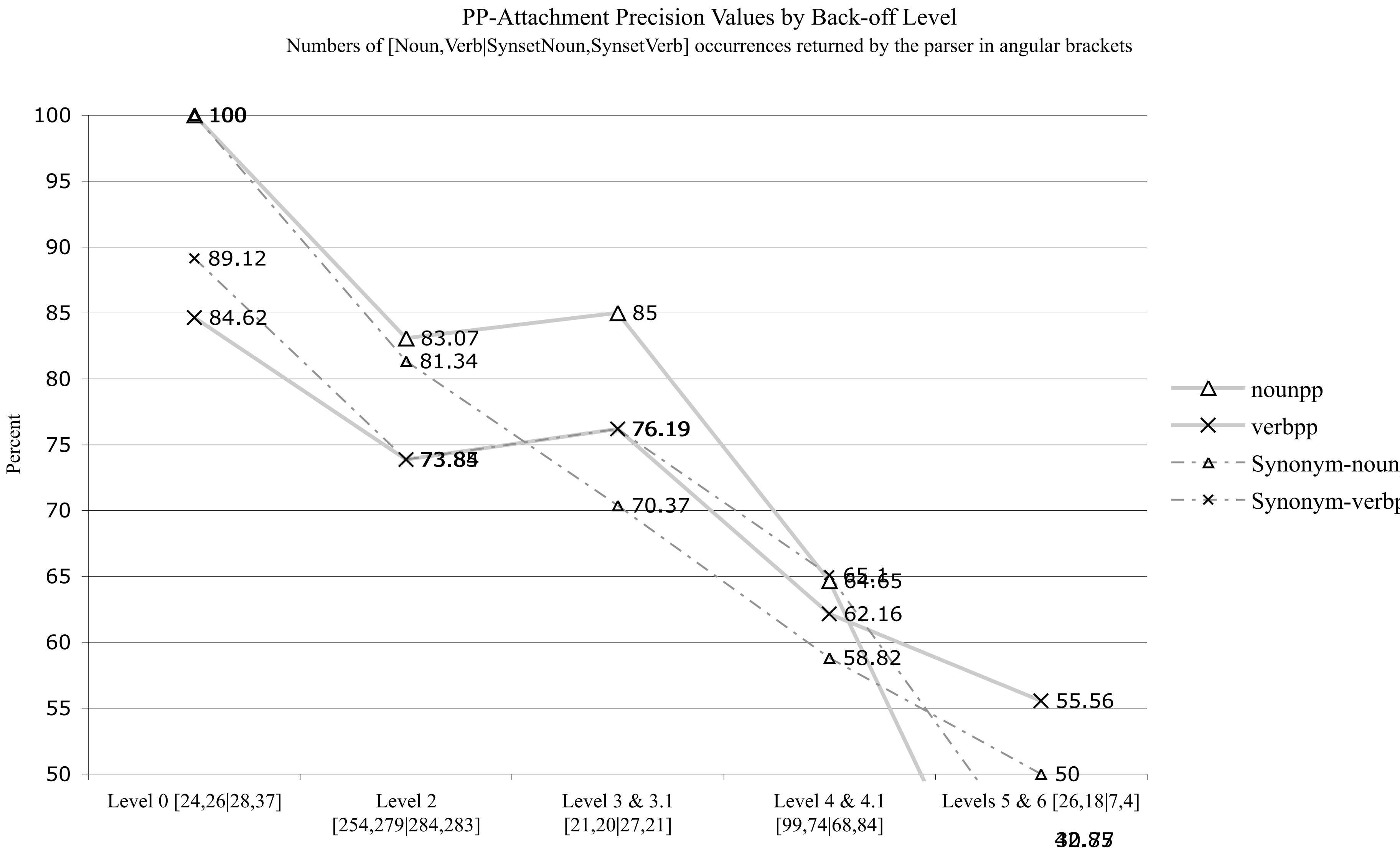

Figure 1: Evaluation Preview: Quality of Backoff

3 Decision-Based Parsing

Unlike Collins etc., this Parser is not probabilistic, but based on decision probabilitities.

Not the sum of p of possible parses, bu the sum of p of possible decisions at a decision point add to 1. Whether to attach or not (in shift/reduce parlance: to reduce or to shift) is e.g. a decision.

The probability-based score of a parse is the product of the (normalized) decisions taken during parsing.

4 Hard Local Cut

- Very unlikely local structures rarely form part of the most likely global structure
- If a very unlikely local structure forms part of the correct global structure, it chances of getting among the most likely parses are very low
- Very simple to implement
- Biggest gain in complexity reduction, no partial structures at all are built

```
sparse(FID,[FPos,Ffrom-Fto,FScore],[[F,Ftag,FType]],FuncF,
       GID,[GPos,Gfrom-Gto,GScore],[[G,Gtag,GType]],FuncG) :-
  (tried(FID,GID) -> !, fail; assert(tried(FID,GID))), % already tried
  ... head(Ftag,Gtag,l,Type,Transtag,[FChunk,GChunk,FF,FG,OF,OG],FPos-GPos),
  ... % stats
  (Prob < 0.01 -> fail; true), %% early exclusion
  ... asserta(chart(ID,[[FF,Ftag,FChunk,FID,FScore],[FG,Gtag,GChunk,GID,GScore]],
    [FPos,GPos,Gfrom-Fto,PScore,DLen],[[FF,Transtag,Type,ID]],FuncFTRes,Level)),
  retract(perlevel(X)), X1 is X+1, assert(perlevel(X1)),
  (Prob > 0.98 -> !; true), %% early commitment
  fail.
```

5 Fixed Beam Pruning

- Keep a maximum amount of readings for every span
- Only chart entries with equal span can be compared
- “Vorsicht ist die Mutter der Porzellankiste”: do not discard too much

```
prune(L,XFact) :- XFact > 3,
    Beam is 3,
    %% foreach stretch A-Z : only keep 3 most likely spans, if there are at least 3 pos
    chart(_,_,[_,_,Ffrom-Fto,_Score,_],_,_,_),
    findall((Score,ID), chart(ID,_,[_,_,Ffrom-Fto,Score,_],_,_,_),List),
    len(List,Len),
    (Len < 4 -> fail ;
      (sort(List,SList),
       Till is Len-Beam,
       prunechart(0,Till,SList),
       fail)).

prune(_,_).
```

6 Complexity-Dependent Pruning

- Pruning is only useful from a certain level of ambiguity on. Prune more and more strongly
- Again, only chart entries with equal span can be compared
- "Vorsicht ist die Mutter der Porzellankiste": do not discard too much

```
prune(L,XFact) :- XFact > 3,
    Div is (XFact/3),
    %% foreach stretch A-Z : discard lowest prob part, if there are at least 3 possibil
    chart(_,_,[_,_,Ffrom-Fto,_Score,_],_,_,_),
    findall((Score,ID), chart(ID,_,[_,_,Ffrom-Fto,Score,_],_,_,_),List),
    len(List,Len),
    (Len < 4 -> fail ;
      (sort(List,SList),
       Till is Len-(Len/Div),
       prunechart(0,Till,SList),
       fail)).

prune(_,_).
```

Prunechart discards the first ‘Till’ chart entries

```
prunechart(C,Till,[(_Score,ID)|RList]) :-
    C < Till, !,
    %displaychart(ID),
    retract(chart(ID,_,_,_,_,_)),
    %spy_me2,
    C1 is C+1,
    prunechart(C1,Till,RList).

prunechart(_,_,_). %eorec
```

7 Large Beam Panic Mode

When there are more than say 1000 chart entries, only promising paths are pursued.

- it is accepted that some permissible spans will never be found
- increasing severity based on span length and beam size

```
...,
(ID>1000 -> (((OPScore / ((Len+(Len**sqrt(2)))+(ID/2))) < 0.01)
    -> (write(' TOO LOW!'),nl,fail) ;
    ... ) % else continue
```

8 Experiments

Pruning and Time		
	ambi4	ambi1
fix2	1:10	1:05
fix3		1:30
fix5		2:10
fix10		2:45
div2	1:20	1:15
div3	1:30	1:25
div5	2:25	2:15

Table 5: Time spent to parse the 500 sentences of Carroll's test suite

8.1 Beam Sizes

The effect of different beam sizes.

(a) Flexible Beams:

Div is (XFacts): 44 secs.		
subj_prec	813 of 895	0.908
subj_recall	772 of 956	0.807
nounpp_prec	361 of 535	0.674
verbpp_prec	322 of 422	0.763
ncmod_recall	519 of 801	0.647
iobj_recall	132 of 157	0.840
Div is (XFact/2): 51 secs.		
subj_prec	814 of 895	0.909
subj_recall	773 of 956	0.808
nounpp_prec	366 of 533	0.686
verbpp_prec	324 of 424	0.764
ncmod_recall	528 of 801	0.659
iobj_recall	132 of 157	0.840

Div is (XFact/3): 55 secs.		
subj_prec	814 of 896	0.908
subj_recall	773 of 956	0.808
nounpp_prec	366 of 536	0.682
verbpp_prec	323 of 421	0.767
ncmod_recall	527 of 801	0.657
iobj_recall	132 of 157	0.840
Div is (XFact/4): 62 secs.		
subj_prec	815 of 896	0.909
subj_recall	773 of 956	0.808
nounpp_prec	367 of 535	0.685
verbpp_prec	323 of 422	0.765
ncmod_recall	528 of 801	0.659
iobj_recall	132 of 157	0.840
Div is (XFact/8): 112 secs.		
subj_prec	815 of 895	0.910
subj_recall	773 of 956	0.808
nounpp_prec	367 of 535	0.685
verbpp_prec	323 of 422	0.765
ncmod_recall	528 of 801	0.659
iobj_recall	132 of 157	0.840

(b) Fixed Beams: Len - X means that the X+1 best alternatives per span are kept

Till is Len-0: 40 secs.		
subj_prec	803 of 890	0.902
subj_recall	767 of 956	0.802
obj_prec	433 of 486	0.890
obj_recall	317 of 391	0.810
nounpp_prec	347 of 527	0.658
verbpp_prec	309 of 413	0.748
ncmod_recall	504 of 801	0.629
iobj_recall	128 of 157	0.815
argmod_recall	26 of 41	0.634
Till is Len-1: 46 secs.		
subj_prec	814 of 896	0.908
subj_recall	771 of 956	0.806
obj_prec	438 of 490	0.893
obj_recall	322 of 391	0.823
nounpp_prec	366 of 535	0.684
verbpp_prec	323 of 421	0.767
ncmod_recall	527 of 801	0.657
iobj_recall	132 of 157	0.840
argmod_recall	31 of 41	0.756

Till is Len-2: 85 secs.		
subj_prec	815 of 896	0.909
subj_recall	772 of 956	0.807
obj_prec	438 of 490	0.893
obj_recall	322 of 391	0.823
nounpp_prec	365 of 533	0.684
verbpp_prec	323 of 423	0.763
ncmod_recall	526 of 801	0.656
iobj_recall	132 of 157	0.840
argmod_recall	31 of 41	0.756
Till is Len-3: 89 secs.		
subj_prec	815 of 895	0.910
subj_recall	773 of 956	0.808
obj_prec	438 of 490	0.893
obj_recall	322 of 391	0.823
nounpp_prec	367 of 534	0.687
verbpp_prec	323 of 423	0.763
ncmod_recall	528 of 801	0.659
iobj_recall	132 of 157	0.840
argmod_recall	31 of 41	0.756
Till is Len-4: 97 secs.		
no changes		
Till is Len-10: 230 secs.		
no changes		