CS224N/Ling237 – Generalized CKY Parsing (partial table)

Initial grammar

S	→	NP VP	N	→	*cats*
VP	→	V NP	N	→	*claws*
VP	→	V NP PP	N	→	*people*
NP	→	NP PP	N	→	*scratch*
NP	→	N	V	→	*scratch*
NP	→	e	P	→	*with*
NP	→	N N			
PP	→	P NP			

Epsilon removal

S	→	NP VP	N	→	*cats*
VP	→	V NP	N	→	*claws*
VP	→	V NP PP	N	→	*people*
NP	→	NP PP	N	→	*scratch*
NP	→	N	V	→	*scratch*
NP	→	N N	P	→	*with*
PP	→	P NP			
PP	→	P			
NP	→	PP			
VP	→	V			
VP	→	V PP			
S	→	VP			

Binarization

S	→	NP VP	N	→	*cats*
VP	→	V NP	N	→	*claws*
VP	→	V VP→V.	N	→	*people*
NP	→	NP PP	N	→	*scratch*
NP	→	N	V	→	*scratch*
NP	→	N N	P	→	*with*
PP	→	P NP			
PP	→	P			
NP	→	PP			
VP	→	V			
VP	→	V PP			
S	→	VP			
VP→V.	→	NP PP			

There are many ways to do binarization. The assignment code does a simple left to right binarization that turns the grammar into a trie data structure for each category. This is a sensible thing to do because it compacts the grammar search space.

Unaries

The standard Chomsky Normal Form transform would fold unaries. That's okay if you never want to see them again. But it might be better to keep them. For two reasons: you want to preserve the nodes in unary chains, and you want to keep your grammar smaller. But it is then important to do closure over unary chains. For instance, the grammar also licenses S → V.

S	→	NP VP	N	→	*cats*
VP	→	V NP	N	→	*claws*
VP	→	V VP→V.	N	→	*people*
NP	→	NP PP	N	→	*scratch*
NP	→	N	V	→	*scratch*
NP	→	N N	P	→	*with*
PP	→	P NP			
PP	→	P			
NP	→	PP			
VP	→	V			
VP	→	V PP			
S	→	VP			
VP→V.	→	NP PP			
S	→	V			
NP	→	P			

Parse Triangle

	0	1 *cats*	2 *scratch*	3 *people*	4 *with*	5 *claws*
0		N NP		S		S
1			N V NP VP S	VP S		VP S
2				N NP		NP VP→V.
3					P PP NP	PP NP
4						N NP
5						