Parsing Algorithms

Reference Sudkamp, Chapter 18

**Algorithm 18.2.1**

**Breadth First Top Down Parser**

**input**

Context Free Grammar *G* = (*V,* Σ*, P, S*)

**data structure**

queue *Q*

1.initialize *T* with root *S*

INSERT (S, Q);

string *p* ∈ Σ∗

# repeat

* 1. *q* = *REMOV E*(*Q*); /\*node to be expanded
	2. *i* = 0; /\*number of last rule used\*/
	3. *done* = *false*; /\*set flag\*/

Let *q* = *uAv A* is the leftmost variable in *q*

# repeat

* + 1. If there is no A-rule numbered *> i*

then *done* = *true*

* + 1. **if** not **done** then

Let *A* → *w* be the first A-rule numbered greater than *i* and *j* be the number of this

* + - 1. if *uwv* /∈ Σ∗ and terminal

prefix of *uwv* matches a prefix of *p* then

* + - * 1. INSERT (*uwv*,Q)
				2. Add node *uwv* to *T*

endif endif

* + 1. *i* = *j*

**until** done or *p* = *uwv* **end of step-2.4 until** *EMPT Y* (*Q*) or *p* = *uwv* **end of step-2**

1. if *p* = *uwv* then accept else reject

**end**.

+ 2

# Algorithm 18.4.1

**Breadth First Bottom-Up Parser**

# input

Context Free Grammar *G* = (*V,* Σ*, P, S*) string *p* ∈ Σ∗

# data structure

queue *Q*

1. initialize *T* with root *p*

INSERT (p, Q);

# repeat

*q* = *REMOV E*(*Q*); /\*node to be expanded\*/

* 1. for each rule *A* → *w* in *P* do
		1. for each decomposition *uwv* of *q*

with *v* ∈ Σ∗

* + - 1. INSERT (*uAv*,Q)
			2. Add node *uAv* to *T*

# endfor endfor

**until** *q* == *S* or EMPTY(*Q*))

1. if *q* == *S* then accept else reject

**end**.