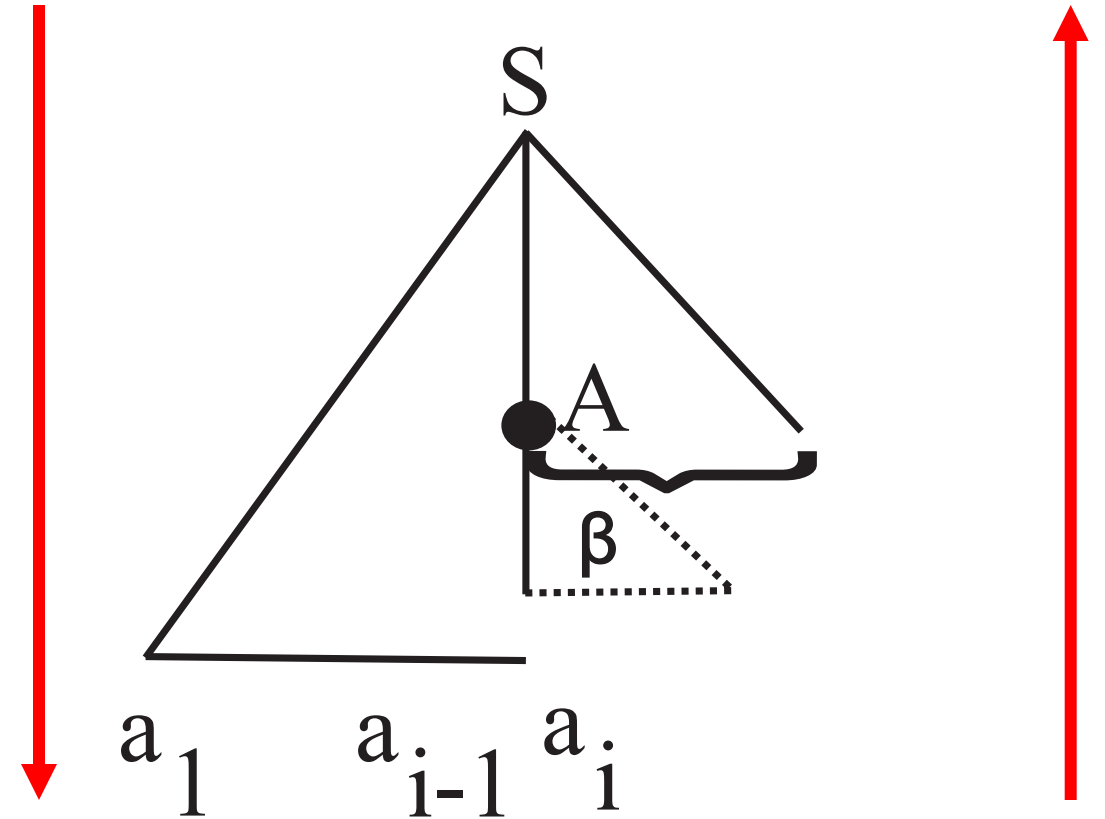# Course 6

Parsing

- Cfg $G = (N, \Sigma, P, S)$ check if $w \in L(G)$

- Construct parse tree

- How:
  1. Top-down vs. Bottom-up
  2. Recursive vs. linear

	Descendent	Ascendent
Recursive	Descendent recursive parser	Ascendent recursive parser
Linear	LL(k): LL(1)	LR(k): LR(0), SLR, LR(1), LALR

Result – parse tree -representation

- Arbitrary tree – child sybling representation

- Sequence of derivations S => α_1 => α_2 =>… => α_n = w

- String of production – index associated to prod – which prod is used at each derivation step

Descendent recursive parser

- Example

Formal model

- Configuration

$$(s, i, \alpha, \beta)$$

where:

- s = state of the parsing, can be:
    - q = normal state
    - b = back state
    - f = final state - corresponding to success: $w \in L(G)$
    - e = error state – corresponding to insuccess: $w \notin L(G)$
- i – position of current symbol in input sequence

$$w = a1a2...an, i \in \{1,...,n+1\}$$

- α = working stack, stores the way the parse is built
- β = input stack, part of the tree to be built

Initial configuration:
$(q,1,\varepsilon, S)$

Define moves between configurations

Final configuration:
$(f,n+1, \alpha,\varepsilon)$

Expand

WHEN: head of input stack is a nonterminal

$(q, i, \boldsymbol{\alpha}, A\boldsymbol{\beta}) \vdash (q, i, \boldsymbol{\alpha}A_1, \boldsymbol{\gamma}_1\boldsymbol{\beta})$

where:

$A \rightarrow \boldsymbol{\gamma}_1 \mid \boldsymbol{\gamma}_2 \mid \ldots$ represents the productions corresponding to A

1 = first prod of A

Advance

WHEN: head of input stack is a terminal = current symbol from input

$(q, i, \boldsymbol{\alpha}, a_i\boldsymbol{\beta}) \vdash (q, \textcolor{red}{i+1}, \boldsymbol{\alpha}\textcolor{red}{a_i}, \boldsymbol{\beta})$

Momentary insuccess

WHEN: head of input stack is a terminal ≠ current symbol from input

$(q, i, \boldsymbol{\alpha}, a_i\boldsymbol{\beta}) \vdash (b, i, \boldsymbol{\alpha}, \boldsymbol{\beta})$

Back

WHEN: head of working stack is a terminal

$(b, i, \alpha a, \beta) \vdash (b, i\text{-}1, \alpha, a\beta)$

Another try

WHEN: head of working stack is a nonterminal

$(b, i, \boldsymbol{\alpha} A_j, \boldsymbol{\gamma}_j \boldsymbol{\beta}) \vdash$ $(q, i, \boldsymbol{\alpha} A_{j+1}, \boldsymbol{\gamma}_{j+1} \boldsymbol{\beta})$, if $\exists\ A \rightarrow \boldsymbol{\gamma}_{j+1}$

                           $(b, i, \boldsymbol{\alpha}, A \boldsymbol{\beta})$, otherwise with the exception

                           $(e, i, \boldsymbol{\alpha}, \boldsymbol{\beta})$, if i=1, A =S, **ERROR**

Success

$(q,n+1, \boldsymbol{\alpha}, \varepsilon) \vdash (f,n+1, \boldsymbol{\alpha}, \varepsilon)$

Algorithm

w ∈ L(G) - HOW

- Process α:
  - From left to right (reverse if stored as stack)
  - Skip terminal symbols
  - Nonterminals – index of prod

- Example: $\alpha = S_1\ a\ S_2\ a\ S_3\ c\ b\ S_3\ c$