Configuration and CFSM

Xiaokang Qiu
Purdue University

Parsing using an LR(0) parser

- How to construct an LR(0) parser?
- How to determine the states and the goto/action tables?
- Basic idea: a state keeps track, simultaneously, of all possible productions that *could be matched* given what it's seen so far. When it sees a full production, match it.

Terminology for LR parsers

- Configuration: a production augmented with a "•"
- $A \rightarrow X_1 \ldots X_i \bullet X_{i+1} \ldots X_j$
- The "•" marks the point to which the production has been recognized. In this case, we have recognized $X_1 \ldots X_i$
- Configuration set: all the configurations that can apply at a given point during the parse:
- $A \rightarrow B \bullet CD$
- $A \rightarrow B \bullet GH$
- $T \rightarrow B \bullet Z$
- Idea: every configuration in a configuration set is a production that we could be in the process of matching

Configuration closure set

- Include all the configurations necessary to recognize the next symbol after the •
- For each configuration in set:
  - If next symbol is terminal, no new configuration added
  - If next symbol is non-terminal X, for each production of the form X → α, add configuration X → •α

S → E $
E → E + T | T
T → ID | (E)

closure0({S → • E $}) =
{
S → • E $
E → • E + T
E → • T
T → • ID
T → • (E)
}

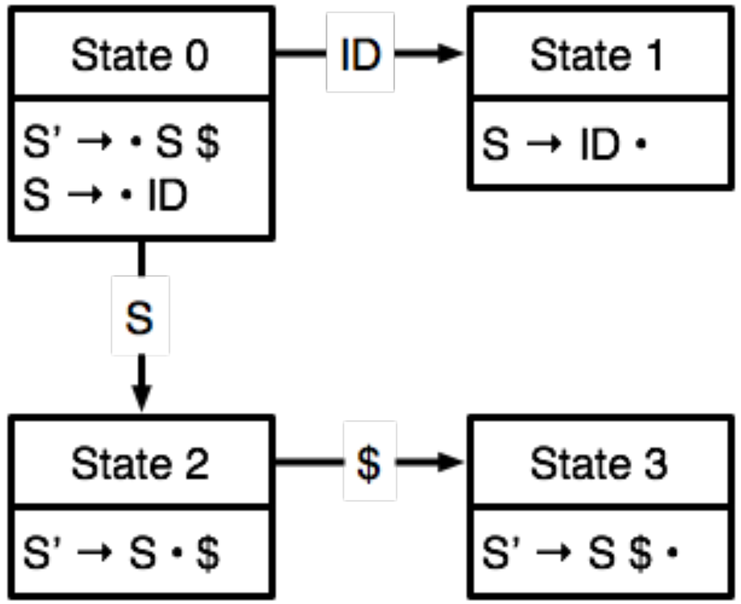
Successor configuration set

- Starting with the initial configuration set
- s0 = closure0({S → • α $})
- an LR(0) parser will find the successor given the next symbol X
- X can be either a terminal (the next token from the scanner) or a non-terminal (the result of applying a reduction)
- Determining the successor s' = go_to0(s, X):
  - For each configuration in s of the form A → β • X γ add A → β X • γ to t
  - s' = closure0(t)

CFSM

- CFSM = Characteristic Finite State Machine
- Nodes are configuration sets (starting from s0)
- Arcs are go_to relationships

S’ → S $
S → ID



Building the goto table

- We can just read this off from the CFSM

		Symbol		
		ID	$	S
State	0	1		2
	1			
	2		3	
	3			

Building the action table

- Given the configuration set s:
  - We shift if the next token matches a terminal after the • in some configuration
  - $A \rightarrow \alpha \bullet a\ \beta \in s$ and $a \in V_t$, else error
  - We reduce production P if the • is at the end of a production
  - $B \rightarrow \alpha \bullet \in s$ where production P is $B \rightarrow \alpha$
  - Extra actions:
    - shift if goto table transitions between states on a non-terminal
    - accept if we have matched the goal production

Action table

State		
	0	Shift
	1	Reduce 2
	2	Shift
	3	Accept