LR(1) Item Sets

Definitions

An "LR(1) item" is a production with a mark on the right hand side (I use the symbol "ˆ" for the mark), together with a terminal.

A "complete item" is an item in which the mark is at the far right hand side of the item.

The set of viable prefixes for a context free language form a regular set; hence, there is a finite state machine that recognizes them. The tramsition symbols are language symbols (terminal and non-terminal); the states are identified by items or sets of items. There are two ways to produce it:

(I assume that the start symbol appears on the left of only one production; to make that happen, we could introduce a new start symbol S' and a production S' →S, where S is the old start symbol.)

non-deterministic: (the states are items)

1. The start state is the state [S' →ˆ S, ⊣].
2. For any state [A →α ˆ B β, a], B a non-terminal, α and β strings of terminals and non-terminals (possibly empty), "a" a terminal, we have a transition on ϵ to all states [B →ˆ γ, b], γ a string of language symbols, "b" an element of FIRST(βa).
3. For any state [A →α ˆ Γ β, a], α and β strings of terminals and non-terminals, Γ a single symbol (terminal or non-terminal), "a" a terminal, we have a transition on Γ to the state [A →α Γ ˆ β, a].

Then we can transform into the deterministic FSM.

deterministic (directly): (the states are sets of items)

1. **Start Rule** The start state contains the item [S' →ˆS, ⊣].
2. **Competion Rule** If a state contains an item [A→α ˆ B β, a], B a non-terminal, α and β strings of terminals and non-terminals (possibly empty), "a" a terminal, then the state also contains all items of the form [B →ˆ γ, b], γ a string of language symbols, b an element of FIRST(βa).
3. **Read Rule** If a state contains a subset of items [A_i →α_iˆ$\Gamma\beta_i$, a_i], Γ a language symbol (the same for all such states), α_i and β_i strings of terminals and non-terminals (possibly empty), a_i terminals, then there is a transition of Γ to a state containing all [A_i →$\alpha_i\Gamma$ˆβ_i, a_i] (the mark moves one position).

If the deterministic machine built as above for the language from its grammar has the property that all the Shift/Reduce and Reduce/Reduce conflicts are resolved, then the grammer is said to be an LR(1) grammar.