**Script** generated by TTT

Title: Simon: Programmiersprachen (05.05.2014)

Date: Mon May 05 14:23:27 CEST 2014

Duration: 84:44 min

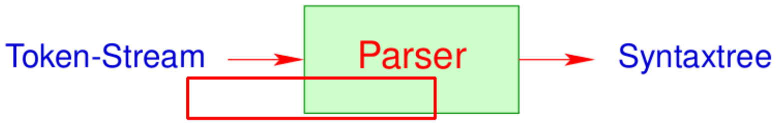
Pages: 42

# Topic:

# Syntactic Analysis



## Syntactic Analysis



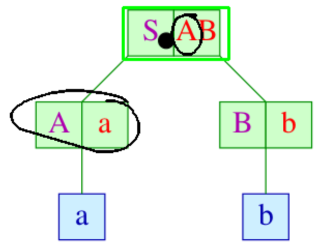

- Syntactic analysis tries to integrate Tokens into larger program units.



## Item Pushdown Automaton – Example

Our example:

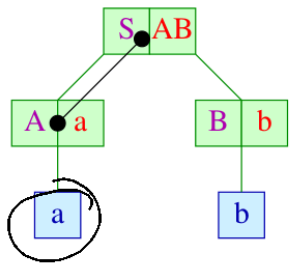
$S \rightarrow A\,B \qquad A \rightarrow a \qquad B \rightarrow b$



## Item Pushdown Automaton – Example

Our example:

$$S \to A\,B \qquad A \to a \qquad B \to b$$

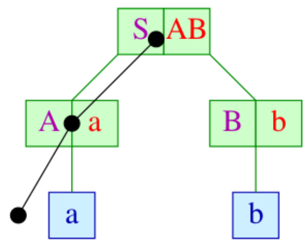





## Item Pushdown Automaton – Example

Our example:

$$S \to A\,B \qquad A \to a \qquad B \to b$$

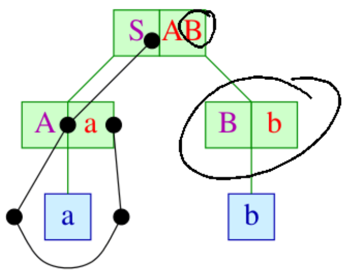





## Item Pushdown Automaton – Example

Our example:

$$S \to A\,B \qquad A \to a \qquad B \to b$$

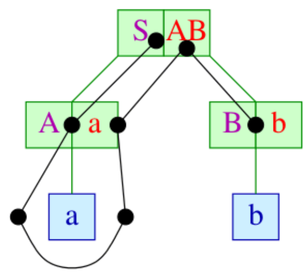





## Item Pushdown Automaton – Example

Our example:

$$S \to A\,B \qquad A \to a \qquad B \to b$$



## Item Pushdown Automaton – Example

We add another rule $\boxed{S' \to S}$ for initialising the construction:

**Start state:** $\boxed{[S' \to \bullet\, S]}$
**End state:** $[S' \to S\, \bullet]$
**Transition relations:**

| | | |
|---|---|---|
| $[S' \to \bullet\, S]$ | $\epsilon$ | $[S' \to \bullet\, S]\,[S \to \bullet\, A\,B]$ |
| $[S \to \bullet\, A\,B]$ | $\epsilon$ | $[S \to \bullet\, A\,B]\,[A \to \bullet\, a]$ |
| $[A \to \bullet\, a]$ | $a$ | $[A \to a\, \bullet]$ |
| $[S \to \bullet\, A\,B]\,[A \to a\, \bullet]$ | $\epsilon$ | $[S \to A\, \bullet\, B]$ |
| $[S \to A\, \bullet\, B]$ | $\epsilon$ | $[S \to A\, \bullet\, B]\,[B \to \bullet\, b]$ |
| $[B \to \bullet\, b]$ | $b$ | $[B \to b\, \bullet]$ |
| $[S \to A\, \bullet\, B]\,[B \to b\, \bullet]$ | $\epsilon$ | $[S \to A\,B\, \bullet]$ |
| $[S' \to \bullet\, S]\,[S \to A\,B\, \bullet]$ | $\epsilon$ | $[S' \to S\, \bullet]$ |



## Item Pushdown Automaton

The item pushdown automaton $M_G^L$ has three kinds of transitions:

**Expansions:** $([A \to \alpha \bullet B\,\beta], \epsilon, [A \to \alpha \bullet B\,\beta]\,[B \to \bullet\,\gamma])$ for $A \to \alpha\,B\,\beta,\; B \to \gamma \in P$
**Shifts:** $([A \to \alpha \bullet a\,\beta], a, [A \to \alpha\,a \bullet \beta])$ for $A \to \alpha\,a\,\beta \in P$
**Reduces:** $([A \to \alpha \bullet B\,\beta]\,[B \to \gamma \bullet], \epsilon, [A \to \alpha\,B \bullet \beta])$ for $A \to \alpha\,B\,\beta,\; B \to \gamma \in P$

Items of the form: $\boxed{[A \to \alpha\,\bullet]}$ are also called complete
The item pushdown automaton shifts the dot once around the derivation tree ...



## Item Pushdown Automaton

### Discussion:

- The expansions of a computation form a leftmost derivation
- Unfortunately, the expansions are chosen nondeterministically
- For proving correctness of the construction, we show that for every Item $[A \to \alpha \bullet B\,\beta]$ the following holds:

  $$([A \to \alpha \bullet B\,\beta], w) \vdash^* ([A \to \alpha\,B \bullet \beta], \epsilon) \quad \text{iff} \quad B \to^* w$$

- LL-Parsing is based on the item pushdown automaton and tries to make the expansions deterministic ...



## Item Pushdown Automaton

The item pushdown automaton $M_G^L$ has three kinds of transitions:

**Expansions:** $([A \to \alpha \bullet B\,\beta], \epsilon, [A \to \alpha \bullet B\,\beta]\,[B \to \bullet\,\gamma])$ for $A \to \alpha\,B\,\beta,\; B \to \gamma \in P$
**Shifts:** $([A \to \alpha \bullet a\,\beta], a, [A \to \alpha\,a \bullet \beta])$ for $A \to \alpha\,a\,\beta \in P$
**Reduces:** $([A \to \alpha \bullet B\,\beta]\,[B \to \gamma \bullet], \epsilon, [A \to \alpha\,B \bullet \beta])$ for $A \to \alpha\,B\,\beta,\; B \to \gamma \in P$

Items of the form: $[A \to \alpha\,\bullet]$ are also called complete
The item pushdown automaton shifts the dot once around the derivation tree ...

## Item Pushdown Automaton

Beispiel: $S \to \epsilon \mid a\, S\, b$

The transitions of the according Item Pushdown Automaton:

| | | | |
|---|---|---|---|
| 0 | $[S' \to \bullet\, S]$ | $\epsilon$ | $[S' \to \bullet\, S]\ [S \to \bullet]$ |
| 1 | $[S' \to \bullet\, S]$ | $\epsilon$ | $[S' \to \bullet\, S]\ [S \to \bullet\, a\, S\, b]$ |
| 2 | $[S \to \bullet\, a\, S\, b]$ | $a$ | $[S \to a \bullet S\, b]$ |
| 3 | $[S \to a \bullet S\, b]$ | $\epsilon$ | $[S \to a \bullet S\, b]\ [S \to \bullet]$ |
| 4 | $[S \to a \bullet S\, b]$ | $\epsilon$ | $[S \to a \bullet S\, b]\ [S \to \bullet\, a\, S\, b]$ |
| 5 | $[S \to a \bullet S\, b]\ [S \to \bullet]$ | $\epsilon$ | $[S \to a\, S \bullet b]$ |
| 6 | $[S \to a \bullet S\, b]\ [S \to a\, S\, b \bullet]$ | $\epsilon$ | $[S \to a\, S \bullet b]$ |
| 7 | $[S \to a\, S \bullet b]$ | $b$ | $[S \to a\, S\, b \bullet]$ |
| 8 | $[S' \to \bullet\, S]\ [S \to \bullet]$ | $\epsilon$ | $[S' \to S \bullet]$ |
| 9 | $[S' \to \bullet\, S]\ [S \to a\, S\, b \bullet]$ | $\epsilon$ | $[S' \to S \bullet]$ |

Conflicts arise between the transitions $(0, 1)$ and $(3, 4)$, resp..



## Topdown Parsing

**Problem:**

Conflicts between the transitions prohibit an implementation of the item pushdown automaton as deterministic pushdown automaton.



## Topdown Parsing

**Problem:**

Conflicts between the transitions prohibit an implementation of the item pushdown automaton as deterministic pushdown automaton.



## Topdown Parsing

**Problem:**

Conflicts between the transitions prohibit an implementation of the item pushdown automaton as deterministic pushdown automaton.

## Topdown Parsing

**Problem:**
Conflicts between the transitions prohibit an implementation of the item pushdown automaton as deterministic pushdown automaton.

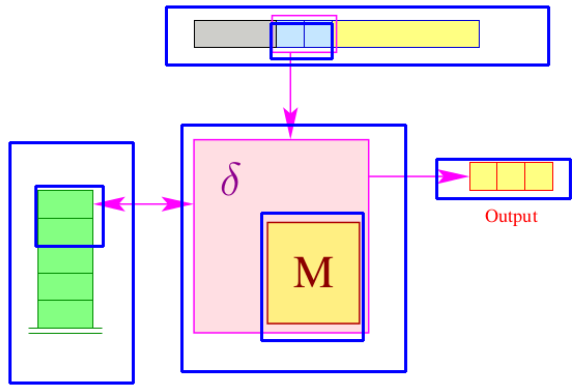
**Idee 1: GLL Parsing**
For each conflict, we create a virtual copy of the complete stack and continue computing in parallel.

**Idee 2: Recursive Descent & Backtracking**
Depth-first search for an appropriate solution.

**Idee 3: Recursive Descent & Lookahead**
Conflicts are resolved by considering a lookup of the next input symbol.



## Structure of the $LL(1)$-Parser:




- The parser accesses a frame of length $1$ of the input;
- it corresponds to an item pushdown automaton, essentially;
- table $M[q, w]$ contains the rule of choice.



## Topdown Parsing

Idee:

- Emanate from the item pushdown automaton
- Consider the next symbol to determine the appropriate rule for the next expansion
- A grammar is called $LL(1)$ if a unique choice is always possible



## Topdown Parsing

Idee:

- Emanate from the item pushdown automaton
- Consider the next symbol to determine the appropriate rule for the next expansion
- A grammar is called $LL(1)$ if a unique choice is always possible

**Definition:**
A reduced grammar is called $LL(1)$ if for each two distinct rules $A \to \alpha$, $A \to \alpha' \in P$ and each derivation $S \to^*_L u\, A\, \beta$ with $u \in T^*$ the following is valid:

$$\text{First}_1(\alpha\,\beta) \cap \text{First}_1(\alpha'\,\beta) = \emptyset$$

## Topdown Parsing

Example 1:

$$\begin{array}{lll} S & \to & \text{if ( } E \text{ ) } S \text{ else } S \quad | \\ & & \text{while ( } E \text{ ) } S \quad | \\ & & E \,; \\ E & \to & \text{id} \end{array}$$

is $LL(1)$, since First$_1(E) = \{\text{id}\}$



## Topdown Parsing

Example 1:

$$\begin{array}{lll} S & \to & \text{if ( } E \text{ ) } S \text{ else } S \quad | \\ & & \text{while ( } E \text{ ) } S \quad | \\ & & E \,; \\ E & \to & \text{id} \end{array}$$

is $LL(1)$, since First$_1(E) = \{\text{id}\}$

Example 2:

$$\begin{array}{lll} \boxed{S} & \to & \boxed{\text{if ( } E \text{ )}}\,\boxed{S} \text{ else } S \quad | \\ & & \boxed{\text{if ( } E \text{ )}}\,\boxed{S} \quad | \\ & & \text{while ( } E \text{ ) } S \quad | \\ & & E \,; \\ E & \to & \text{id} \end{array}$$

... is not $\boxed{LL(k)}$ for any $\boxed{k > 0}$.



## Lookahead Sets

**Definition:**

For a set $L \subseteq T^*$ we define:

$$\text{First}_1(L) = \{\epsilon \mid \epsilon \in L\} \cup \{u \in T \mid \exists\, v \in T^* : \; uv \in L\}$$

Example: $S \to \varepsilon \mid S \to aSb$ First$_1(S)$

| |
|---|
| $\epsilon$ → $\varepsilon$ |
| $a\,b$ → $a$ |
| $a\,a\,b\,b$ → $a$ |
| $a\,a\,a\,b\,b\,b$ → $a$ |

⋮

First$_1(S) = \{\varepsilon, a\}$



## Lookahead Sets

**Definition:**

For a set $L \subseteq T^*$ we define:

$$\text{First}_1(L) = \{\epsilon \mid \epsilon \in L\} \cup \{u \in T \mid \exists\, v \in T^* : \; uv \in L\}$$

Example:

| |
|---|
| $\epsilon$ |
| $a\,b$ |
| $a\,a\,b\,b$ |
| $a\,a\,a\,b\,b\,b$ |

the prefixes of length 1

## Lookahead Sets

Arithmetics:

$\text{First}_1(\_)$ is compatible with union and concatenation:

$$\begin{array}{lll} \text{First}_1(\emptyset) & = & \emptyset \\ \text{First}_1(L_1 \cup L_2) & = & \text{First}_1(L_1) \cup \text{First}_1(L_2) \\ \text{First}_1(L_1 \cdot L_2) & = & \text{First}_1(\text{First}_1(L_1) \cdot \text{First}_1(L_2)) \\ & := & \text{First}_1(L_1) \odot \text{First}_1(L_2) \end{array}$$

$1$ – concatenation

Observation:

Let $L_1, L_2 \subseteq T \cup \{\epsilon\}$ with $L_1 \neq \emptyset \neq L_2$. Then:

$$L_1 \odot L_2 = \begin{cases} L_1 & \text{if } \epsilon \notin L_1 \\ (L_1 \backslash \{\epsilon\}) \cup L_2 & \text{otherwise} \end{cases}$$

If all rules of $G$ are productive, then all sets $\text{First}_1(A)$ are non-empty.



## Lookahead Sets

For $\alpha \in (N \cup T)^*$ we are interested in the set:

$$\text{First}_1(\alpha) = \text{First}_1(\{w \in T^* \mid \alpha \to^* w\})$$

Idea: Treat $\epsilon$ separately: $F_\epsilon$

- Let $\text{empty}(X) = \text{true}$ iff $X \to^* \epsilon$ .
- $F_\epsilon(\boxed{X_1 \dots X_m}) = \bigcup_{i=1}^{\boxed{j}} \boxed{F_\epsilon(X_i)}$ if $\boxed{\text{empty}(X_1) \wedge \dots \wedge \text{empty}(\boxed{X_{j-1}})}$



## Lookahead Sets

For $\alpha \in (N \cup T)^*$ we are interested in the set:

$$\text{First}_1(\alpha) = \text{First}_1(\{w \in T^* \mid \alpha \to^* w\})$$

Idea: Treat $\epsilon$ separately: $F_\epsilon$

- Let $\text{empty}(X) = \text{true}$ iff $X \to^* \epsilon$ .
- $F_\epsilon(X_1 \dots X_m) = \boxed{\bigcup_{i=1}^{j} F_\epsilon(X_i)}$ if $\text{empty}(X_1) \wedge \dots \wedge \text{empty}(X_{j-1})$

We characterize the $\epsilon$-free $\text{First}_1$-sets with an inequality system:

$$\begin{array}{llll} F_\epsilon(a) = \{a\} & \text{if} & a \in T \\ \boxed{F_\epsilon(A) \supseteq F_\epsilon(X_j)} & \text{if} & \boxed{\begin{array}{l} A \to X_1 \dots X_m \in P, \\ \text{empty}(X_1) \wedge \dots \wedge \text{empty}(X_{j-1}) \end{array}} \end{array}$$



## Lookahead Sets

for example...

$$\begin{array}{lllll} E & \to & E \boxed{+} T & \mid & T \\ T & \to & T \boxed{*} F & \mid & F \\ F & \to & \boxed{(} E ) & \mid \boxed{\text{name}} & \mid \boxed{\text{int}} \end{array}$$

with $\boxed{\text{empty}(E) = \text{empty}(T) = \text{empty}(F) = \text{false}}$

## Lookahead Sets

for example...

$$\begin{array}{lcllll} E & \to & E+T & \mid & T & \\ T & \to & T*F & \mid & F & \\ F & \to & (\ E\ ) & \mid & \text{name} & \mid \ \text{int} \end{array}$$

with $\text{empty}(E) = \text{empty}(T) = \text{empty}(F) = \text{false}$

... we obtain:

$$\begin{array}{lcllcl} F_\epsilon(S') & \supseteq & F_\epsilon(E) & F_\epsilon(E) & \supseteq & F_\epsilon(E) \\ F_\epsilon(E) & \supseteq & F_\epsilon(T) & F_\epsilon(T) & \supseteq & F_\epsilon(T) \\ F_\epsilon(T) & \supseteq & F_\epsilon(F) & F_\epsilon(F) & \supseteq & \{\,(\,,\text{name},\text{int}\} \end{array}$$



## Fast Computation of Lookahead Sets

Observation:

- The form of each inequality of these systems is:

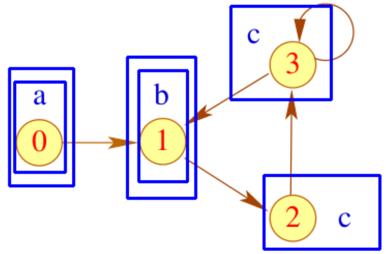
$$x \sqsupseteq y \qquad \text{resp.} \qquad x \sqsupseteq d$$

for variables $x, y$ und $d \in \mathbb{D}$.

- Such systems are called pure unification problems
- Such problems can be solved in linear space/time.

for example: $\mathbb{D} = 2^{\{a,b,c\}}$

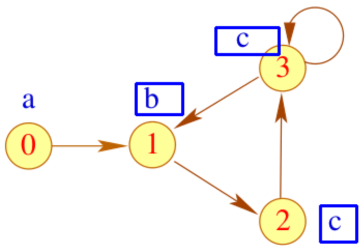
$$\begin{array}{lll} x_0 \supseteq \{a\} & & \\ x_1 \supseteq \{b\} & x_1 \supseteq x_0 & x_1 \supseteq x_3 \\ x_2 \supseteq \{c\} & x_2 \supseteq x_1 & \\ x_3 \supseteq \{c\} & x_3 \supseteq x_2 & x_3 \supseteq x_3 \end{array}$$






## Fast Computation of Lookahead Sets

Frank DeRemer
& Tom Pennello



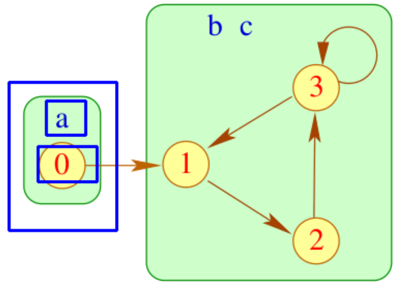

Proceeding:

- Create the Variable dependency graph for the inequality system.



## Fast Computation of Lookahead Sets

Frank DeRemer
& Tom Pennello

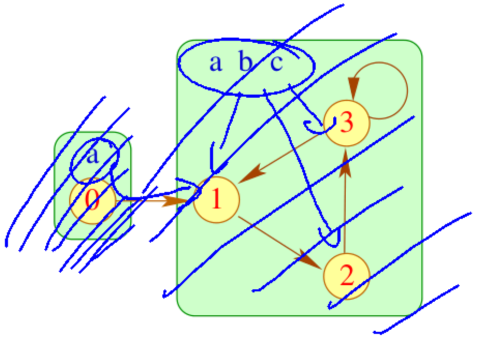



Proceeding:

- Create the Variable dependency graph for the inequality system.
- Whithin a strongly connected component ($\to$ Tarjan) all variables have the same value
- Is there no ingoing edge for an SCC, its value is computed via the smallest upper bound of all values within the SCC

## Fast Computation of Lookahead Sets

Frank DeRemer
& Tom Pennello




### Proceeding:

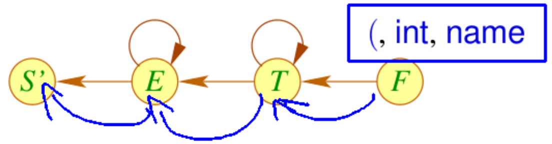
- Create the Variable dependency graph for the inequality system.
- Whithin a strongly connected component ($\rightarrow$ Tarjan) all variables have the same value
- Is there no ingoing edge for an SCC, its value is computed via the smallest upper bound of all values within the SCC
- In case of ingoing edges, their values are also to be considered for the upper bound



## Fast Computation of Lookahead Sets

... for our example grammar:

$\mathsf{First}_1$ :






## Item Pushdown Automaton as LL(1)-Parser

back to the example: $S \rightarrow \boxed{\epsilon} \mid \boxed{a\,S\,b}$

The transitions in the according Item Pushdown Automaton:

| | | | |
|---|---|---|---|
| 0 | $[S' \rightarrow \bullet\, S]$ | $\epsilon$ | $[S' \rightarrow \bullet\, S]\ [S \rightarrow \bullet]$ |
| 1 | $[S' \rightarrow \bullet\, S]$ | $\epsilon$ | $[S' \rightarrow \bullet\, S]\ [S \rightarrow \bullet\, a\,S\,b]$ |
| 2 | $[S \rightarrow \bullet\, a\,S\,b]$ | $a$ | $[S \rightarrow a \bullet S\,b]$ |
| 3 | $[S \rightarrow a \bullet S\,b]$ | $\epsilon$ | $[S \rightarrow a \bullet S\,b]\ [S \rightarrow \bullet]$ |
| 4 | $[S \rightarrow a \bullet S\,b]$ | $\epsilon$ | $[S \rightarrow a \bullet S\,b]\ [S \rightarrow \bullet\, a\,S\,b]$ |
| 5 | $[S \rightarrow a \bullet S\,b]\ [S \rightarrow \bullet]$ | $\epsilon$ | $[S \rightarrow a\,S \bullet b]$ |
| 6 | $[S \rightarrow a \bullet S\,b]\ [S \rightarrow a\,S\,b \bullet]$ | $\epsilon$ | $[S \rightarrow a\,S \bullet b]$ |
| 7 | $[S \rightarrow a\,S \bullet b]$ | $b$ | $[S \rightarrow a\,S\,b \bullet]$ |
| 8 | $[S' \rightarrow \bullet\, S]\ [S \rightarrow \bullet]$ | $\epsilon$ | $[S' \rightarrow S \bullet]$ |
| 9 | $[S' \rightarrow \bullet\, S]\ [S \rightarrow a\,S\,b \bullet]$ | $\epsilon$ | $[S' \rightarrow S \bullet]$ |

Conflicts arise between transations $(0, 1)$ or $(3, 4)$ resp..

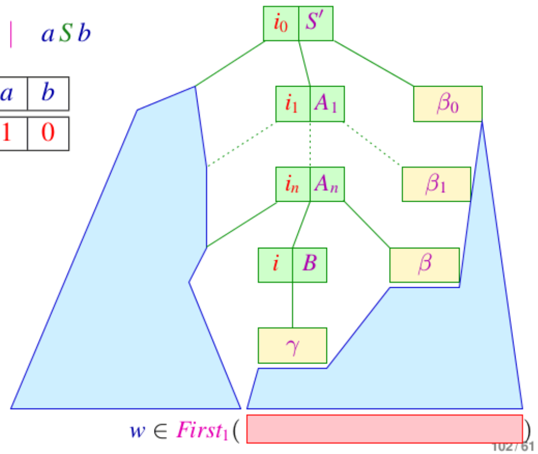


## Item Pushdown Automaton as LL(1)-Parser

Is $G$ an $LL(1)$-grammar, we can index a lookahead-table with items and nonterminals:
We set $M[B,\ w] = i$ exactly if $(B, i)$ is the rule $B \rightarrow \gamma$ and:
$w \in \mathsf{First}_1(\gamma) \odot \bigcup\{\mathsf{First}_1(\beta) \mid S' \rightarrow_L^* u\,B\,\beta\}$ .

... for example: $S \rightarrow \epsilon \mid a\,S\,b$

| | $\epsilon$ | $a$ | $b$ |
|---|---|---|---|
| $S$ | 0 | 1 | 0 |



## Item Pushdown Automaton as LL(1)-Parser

back to the example: $S \to \epsilon \mid a\,S\,b$

The transitions in the according Item Pushdown Automaton:

| | | | |
|---|---|---|---|
| 0 | $[S' \to \bullet\, \boxed{S}]$ | $\epsilon$ | $[S' \to \bullet\, S]\ [S \to \bullet]$ |
| 1 | $[S' \to \bullet\, \boxed{S}]$ | $\epsilon$ | $[S' \to \bullet\, S]\ [S \to \bullet\, a\,S\,b]$ |
| 2 | $[S \to \bullet\, a\,S\,b]$ | $a$ | $[S \to a \bullet S\,b]$ |
| 3 | $[S \to a \bullet S\,b]$ | $\epsilon$ | $[S \to a \bullet S\,b]\ [S \to \bullet]$ |
| 4 | $[S \to a \bullet S\,b]$ | $\epsilon$ | $[S \to a \bullet S\,b]\ [S \to \bullet\, a\,S\,b]$ |
| 5 | $[S \to a \bullet S\,b]\ [S \to \bullet]$ | $\epsilon$ | $[S \to a\,S \bullet b]$ |
| 6 | $[S \to a \bullet S\,b]\ [S \to a\,S\,b \bullet]$ | $\epsilon$ | $[S \to a\,S \bullet b]$ |
| 7 | $[S \to a\,S \bullet b]$ | $b$ | $[S \to a\,S\,b \bullet]$ |
| 8 | $[S' \to \bullet\, S]\ [S \to \bullet]$ | $\epsilon$ | $[S' \to S \bullet]$ |
| 9 | $[S' \to \bullet\, S]\ [S \to a\,S\,b \bullet]$ | $\epsilon$ | $[S' \to S \bullet]$ |

Conflicts arise between transations $(0,1)$ or $(3,4)$ resp..

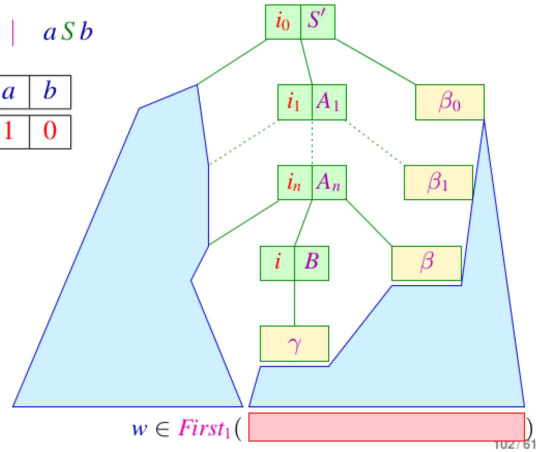


## Item Pushdown Automaton as LL(1)-Parser

Is $G$ an $LL(1)$-grammar, we can index a lookahead-table with items and nonterminals:
We set $M[B,\ w] = i$ exactly if $(B, i)$ is the rule $B \to \gamma$ and:
$w \in \text{First}_1(\gamma) \odot \boxed{\bigcup\{\text{First}_1(\beta) \mid S' \to_L^* u\,B\,\beta\}}$.

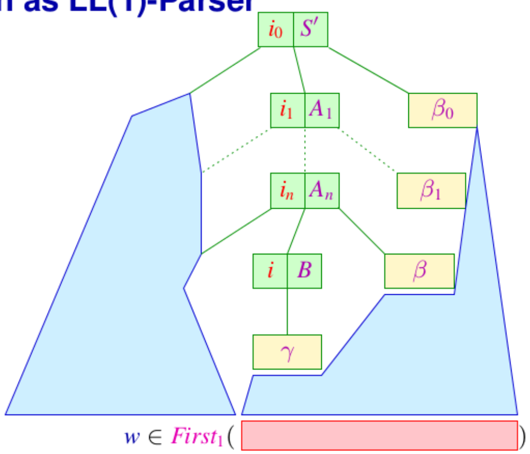
... for example: $S \to \epsilon \mid a\,S\,b$

| | $\epsilon$ | $a$ | $b$ |
|---|---|---|---|
| $S$ | 0 | 1 | 0 |






## Item Pushdown Automaton as LL(1)-Parser




Inequality system for $\text{Follow}_1(B) = \bigcup\{\text{First}_1(\beta) \mid S' \to_L^* u\,B\,\beta\}$

| | | | | |
|---|---|---|---|---|
| $\text{Follow}_1(S)$ | $\supseteq$ | $\{\epsilon\}$ | | |
| $\text{Follow}_1(B)$ | $\supseteq$ | $F_\epsilon(X_j)$ | if | $A \to \alpha\,B\,X_1 \ldots X_m \in P$, $\text{empty}(X_1) \wedge \ldots \wedge \text{empty}(X_{j-1})$ |
| $\text{Follow}_1(B)$ | $\supseteq$ | $\text{Follow}_1(A)$ | if | $A \to \alpha\,B\,X_1 \ldots X_m \in P$, $\text{empty}(X_1) \wedge \ldots \wedge \text{empty}(X_m)$ |



## Item Pushdown Automaton as LL(1)-Parser

For example: $S \to \epsilon \mid a\,S\,b$

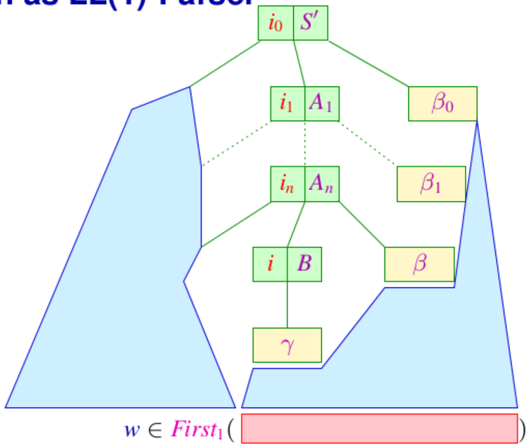
The transitions of the according Item Pushdown Automaton:

| | | | |
|---|---|---|---|
| 0 | $[S' \to \bullet\, \boxed{S}]$ | $\epsilon$ | $[S' \to \bullet\, S]\ [S \to \bullet]$ |
| 1 | $[S' \to \bullet\, \boxed{S}]$ | $\epsilon$ | $[S' \to \bullet\, S]\ [S \to \bullet\, a\,S\,b]$ |
| 2 | $[S \to \bullet\, a\,S\,b]$ | $a$ | $[S \to a \bullet S\,b]$ |
| 3 | $[S \to a \bullet \boxed{S\,b}]$ | $\epsilon$ | $[S \to a \bullet S\,b]\ [S \to \bullet]$ |
| 4 | $[S \to a \bullet \boxed{S\,b}]$ | $\epsilon$ | $[S \to a \bullet S\,b]\ [S \to \bullet\, a\,S\,b]$ |
| 5 | $[S \to a \bullet S\,b]\ [S \to \bullet]$ | $\epsilon$ | $[S \to a\,S \bullet b]$ |
| 6 | $[S \to a \bullet S\,b]\ [S \to a\,S\,b \bullet]$ | $\epsilon$ | $[S \to a\,S \bullet b]$ |
| 7 | $[S \to a\,S \bullet b]$ | $b$ | $[S \to a\,S\,b \bullet]$ |
| 8 | $[S' \to \bullet\, S]\ [S \to \bullet]$ | $\epsilon$ | $[S' \to S \bullet]$ |
| 9 | $[S' \to \bullet\, S]\ [S \to a\,S\,b \bullet]$ | $\epsilon$ | $[S' \to S \bullet]$ |

Lookahead table:

| | $\epsilon$ | $a$ | $b$ |
|---|---|---|---|
| $S$ | 0 | 1 | 0 |

## Item Pushdown Automaton as LL(1)-Parser




Inequality system for $\text{Follow}_1(B) = \bigcup\{\text{First}_1(\beta) \mid S' \rightarrow_L^* u\, B\, \beta\}$

$$
\begin{array}{llll}
\text{Follow}_1(S) & \supseteq & \{\epsilon\} & \\
\text{Follow}_1(B) & \supseteq & F_\epsilon(X_j) & \text{if} \quad A \to \alpha\, B\, X_1 \ldots X_m \ \in P, \\
 & & & \quad \text{empty}(X_1) \wedge \ldots \wedge \text{empty}(X_{j-1}) \\
\text{Follow}_1(B) & \supseteq & \text{Follow}_1(A) & \text{if} \quad A \to \alpha\, B\, X_1 \ldots X_m \ \in P, \\
 & & & \quad \text{empty}(X_1) \wedge \ldots \wedge \text{empty}(X_m)
\end{array}
$$



## Topdown-Parsing

### Discussion

- A practical implementation of an $LL(1)$-parser via recursive Descent is a straight-forward idea
- However, only a subset of the deterministic contextfree languages can be read this way.
- Solution: Going from $LL(1)$ to $LL(k)$
- The size of the occuring sets is rapidly increasing with larger $k$
- Unfortunately, even $LL(k)$ parsers are not sufficient to accept all deterministic contextfree languages.
- In practical systems, this often motivates the implementation of $k = 1$ only ...



End of presentation. Click to exit.