SLR(1) Parsing
Properties of Applicability

Prop1: (No shift-reduce)
$(\forall I_k \in Coll(0)), (\forall A ::= \alpha . a \gamma, B ::= \beta . \in I_k)$
$a \notin follow(B)$

Prop2: (No reduce-reduce)
$(\forall I_k \in Coll(0)), (\forall A ::= \alpha ., B ::= \beta . \in I_k)$
$follow(A) \cap follow(B) = \{\}$

Theorem. Let G be a grammar.
- $G \in SLR(1)$ if and only if both Prop1 and Prop2 hold.
- G has shift-reduce, 1 symbol of lookahead, SLR parser if and only if both Prop1 and Prop2 hold.

SLR
Concluding Remarks

1. In what sense, the analyzer "bottom-up", is not predictive ?

Bottom-up decides the derivation on the basis of the maximum VP to the handle and this VP entirely, includes the string to derive.

2. Let G be a SLR(1) grammar.

- Can contain its ACTION more than one action per entry ?
- Can contain its GOTO more than one state for a same entry ?

3. What means that G∈SLR(1) has Table:

ACTION such that: for each I, for each pair a≠b,

$$ACTION(I,a)=ACTION(I,b)$$

when $ACTION(I,a) \neq \perp \neq ACTION(I,b)$?

To answer 3, let us consider the **SLR(0)** grammar below:

E::= E+T | T
T::= num

SLR
Concluding Remarks - 2

4. Is G ∈SLR(1) decidable for all Context Free grammars ?

5. Is **L** ∈SLR(1) decidable for all Context free languages ?

Recall that: L ∈SLR(1) ≡ ∃ G ∈LL(1): L(G) = **L**

6. Is the class of SLR(1) grammars including that of LL(1) grammars ?

To answer 6, let us consider the grammar below:

S::= Bc | b | A
A::= aBb
B::=ε

SLR
Concluding Remarks - 3

7. Is the class of SLR(1) languages, including that of LL(1) languages ?

To answer 7, let us consider the role of non-terminal B:

S::= Bc | b | A
A::= aBb
B::=ε

8. Consider the condition C:

C ≡ A::=α.∈I_k ∈Coll(0) *iff* A::=α is the handhe for all a∈follow(A).

C states a necessary and sufficient for reduction in SLR(1) Grammars.

- Is C again necessary and sufficient for reduction in botton-up, non SLR grammars ?
- Is C, only a necessary, condition for reduction in botton-up, non SLR grammars ?
- Is C, only a sufficient, condition for reduction in botton-up, non SLR grammars ?

To answer 8, let us consider the grammar above and the existence of a rightmost derivation for string b. Such a derivation (in reverse order) is b<=S, but SLR(1) fails in doing it. Why?

Bottom-Up Analysis:

True and False Conflicts in SLR

$S \;_r\!\!\Rightarrow^* \gamma A \beta \;_r\!\!\Rightarrow \gamma\alpha\beta$

$S \;_r\!\!\Rightarrow^* \gamma A \beta$ is a Right Sentential Form → $First(\beta) \in follow(A)$

$\gamma A \beta \;_r\!\!\Rightarrow \gamma\alpha\beta$ is a Right Derivation → $A ::= \alpha$ is the handle of $\gamma\alpha\beta$

First(β)∈Follow(A) is only a necessary condition

Th. (necessary condition)

$\forall\, G, \forall\, \gamma\alpha \in VP_G$

if $A ::= \alpha$ handle of $\gamma\alpha\beta$

then - $First(\beta) \in follow(A)$

- $A \rightarrow \alpha.$ is Valid Item

Bottom-Up Analysis:
First(β)∈Follow(A)

Let: $S \Rightarrow_r^* \gamma\alpha\beta$ for

γα ∈viable prefix - *is maximum vp*
First(β)∈follow(A)
A::=α. is Valid Item for γα

→

Can you conclude that
A::= α is the handle of γαβ ?

First(β)∈follow(A) is not a sufficient condition

Use this grammar for a proof

S::= Bc | b | A
A::= aBb
B::=ε

Proposition. (Inclusion is not a sufficient condition)
∀G, ∀γα ∈ VP_G with Valid Item A->α.
First(β) ∈ follow(A) **does not imply that** A::=α is the handle of γα*β*

Use this grammar for a proof: (proof)

Let γαβ≡b. Then, $S \Rightarrow^* b$
γα≡λ∈VP is maximum prefix; First(β)∈Follow(B); B::=ε is valid for γα BUT B::=ε is not the handle of b

An Example

S::= A u | a v
A::= a

I0:Closure({S'::=.S}) ={S'::= .S
S::= .Au, S::= .av, A::= .a}
I1:Goto(I0,S) ={S'::= S.}
I2:Goto(I0,A) ={S::= A.u}
I3:Goto(I0,a) ={S::= a.v, A::= a.}
I4:Goto(I2,u) ={S::= Au.}
I5:Goto(I3,v) ={S::=av.}

I0:Closure({S'::=.S}) ={S'::=.S
S::=.Au, S::= .av, S::= .Bv,
A::=.a, B::=.xA}
I1:Goto(I0,S) ={S'::= S.}
I2:Goto(I0,A) ={S::= A.u}
I3:Goto(I0,a) ={S::= a.v
A::= a.}
…. conflict: follow(A)={u,v}

S::= A u | a v | B v
A::= a
B::= x A

a∈VP_G has A::= a. as Valid Item, v∈follow(A)

But A::=a is not the handle of av

$S \Rightarrow_r^* a\ v$

Inclusion is Only Necessary
A proof

CounterExample. Consider: γ, α, β

S::= u A a | A b | a a
A::= a

$\gamma = \lambda$, $\alpha = a$, $\gamma\alpha = a \in VP$

A->a is the handle of "ab", with $\gamma\alpha$=a and β=b.
Hence, $\gamma\alpha$=a is maximum prefix

But, **A->a is not the handle of "aa", with $\gamma\alpha$= a** and β=a
even if $\gamma\alpha$= a is a maximum prefix and {a,b}$\in$Follow(A)

LL(1)$\not\subset_G$SLR(1)
Another Example

S::= uAa | B
B::= Ab | a
A::= ε

$S \Rightarrow_r^* a$

- $\gamma=\lambda$, $\alpha= \lambda$, $\beta= a$
- $\lambda \in VP$
- A->. valid item per λ

No: non è maniglia - $S \Rightarrow_r B \Rightarrow_r a$

A New Class of
More Powerful Items

Let $X::=\alpha.A\beta \in I_k$ and $A::=\gamma$ be a grammar (canonical) production

Then

$A::=.\gamma \in I_k$ **only for the symbols in the set U**=First(β) (for $B\neq>\lambda$)

First(β *look*($X::=\alpha.A\beta$))

Pairing each LR(0) item with the set U of all the symbols that can follow the handle string associated to the item

$A::=.\gamma,U$ $\in I_k$

Item LR(1)

U is called *look(A::=.γ)*

Canonical collection Coll(1)

A New Class of Items that Remember Prefix Follows

Let G=<S,V,Π,s>

$S'::=.S/\$ \in I_0$ by definition

$Clos(I) =_{min} I \cup Clos\{B::= .\gamma/U \mid A::=\alpha.B\beta/V \in Clos(I), B::=\gamma \in \Pi,$
$U = \{first(\beta x) \mid x \in V\}\}$

Let B a nonterminal
let u ∈U, then:
if u ∈V, A::=α.Bβ/V *propagates* on B::= .γ/U
if u ∉V, A::=α.Bβ/V *spontaneously generates* on B::= .γ/U

$Goto(I,x)=Closure\{A::= \alpha x.\beta/U \mid A::=\alpha.x\beta/U \in I\}$

LR(1) Grammars

Prop1: (No shift-reduce)

$(\forall\ I_k \in Coll(1))$, $(\forall\ A::=\alpha.a\gamma/V\ ,\ B::=\beta./U\ \in I_k)$

$a \notin U$

Prop2: (No reduce-reduce)

$(\forall\ I_k \in Coll(1))$, $(\forall\ A::=\alpha./V\ ,\ B::=\beta./U \in I_k)$

$V \cap U = \{\}$

Theorem. Let G be a grammar.

- **G∈LR(1) if and only if both Prop1 and Prop2 hold.**
- **G has shift-reduce, 1 symbol of lookahead, LR parser if and only if both Prop1 and Prop2 hold.**

LR Parsing
The Table Action for LR(1)

ACTION(i,a)=<shift,j>

if goto(Ii,a)=Ij and a∈Σ

ACTION(i,a)= <reduce,p>

if A::=α./U ∈Ii and

p≡ A::= α and a∈U

ACTION(i,$)= <accept,->

if S'::=S./$ ∈ Ii

GOTO(i,A)=j

if goto(Ii,A)=Ij and A ∈N

N=Nonterminal Set

Apply LR(1) to G∉SLR(1)

I0:Closure({S'::=.S}) ={S'::= .S,$
S::= .Au,$, S::= .av,$, S::= .Bv,$,
A::= .a,u, B::=.xA,v}
I1:Goto(I0,S) ={S'::= S.,$}
I2:Goto(I0,A)={S::= A.u,$}
I3:Goto(I0,a) ={S::= a.v,$
A::= a.,u} No conflict
I4:Goto(I0,B) ={S::= B.v,$}
I5:Goto(I0,x) ={S::= x.A,v
A::= .a,v}
I6:Goto(I2,u) ={S::= Au.,$}
I7:Goto(I3,v) ={S::= av.,$}
I8:Goto(I4,v) ={S::= Bv.,$}
I9:Goto(I5,A) ={S::= xA.,v}
I10:Goto(I5,a) ={A::= a.,v}

S::= A u | a v | B v
A::= a
B::= x A

Space Complexity

Table Size		$G = <S, V, \Pi, s>$
LL(1):	O(N*T)	(with N =|S| , T=|V|)
SLR:	O(N*P*T)	(with P=production right hand size)
LR(1):	O(N*P*T*T)	
LALR(1): the same of SLR		

Example

S::=Sb | caSd | c

Example: S::=Sb | caSd | c

I0:{S'::=.S S::=.Sb S::=.caSd S::=.c}
I1=Goto(I0,S)={S'::=S. S::=S.b}
I2=Goto(I0,c)={S::=c.aSd S::=c.}
I3=Goto(I1,b)={S::=Sb.}
I4=Goto(I2,a)={S::=ca.Sd S::=.Sb S::=.caSd S::=.c} kernel
I5=Goto(I4,S)={S::=caS.d S::=S.b}
Goto(I4,c)=I2
I6=Goto(I5,d)={S::=caSd.}

Example - cnt.

S::=Sb | caSd | c

I0:{S'::=.S,$ S::=.Sb,$/b   S::=.caSd,$/b
S::=.c,$/b}
I1=Goto(I0,S)={S'::=S.,$ S::=S.b,$/b}
I2=Goto(I0,c)={S::=c.aSd,$/b   S::=c.,$/b}
I3=Goto(I1,b)={S::=Sb.,$/b}
I4=Goto(I2,a)={S::=ca.Sd,$/b   S::=.Sb,d/b
S::=.caSd,d/b   S::=.c,d/b}
I5=Goto(I4,S)={S::=caS.d,$/b   S::=S.b,d/b}
I6=Goto(I4,c)={S::=c.aSd,d/b   S::=c.,d/b}
I7=Goto(I5,d)={S::=caSd.,$/b}=Goto(I10,b)
I8=Goto(I5,b)={S::=Sb.,d/b}=
I9=Goto(I6,a)={S::=ca.Sd,d/b   S::=.Sb,d/b
S::=.caSd,d/b   S::=.c,d/b}
I10=Goto(I9,S)={S::=caS.d,d/b   S::=S.b,d/b}
Goto(I9,c)=I6

From *LR(1)* to more compact ***LALR(1)***

Each SLR(1) state

$\{t_1\ t_2\ \ldots\ t_k\}$

may be duplicated into n LR(1) states

$\{t_1/U_{11}\ \ t_2/U_{21}\ \ldots\ \ t_k/U_{k1}\}$

......

$\{t_1/U_{1n}\ \ \ t_2/U_{2n} \ldots\ t_k/U_{kn}\}$

LALR(1)

merges state duplications

$\{t_1/U_{11} \cup \ldots \cup U_{1n}\ \ \ \ t_2/U_{21} \cup \ldots \cup U_{2n}\ \ldots\ t_k/U_{k1} \cup \ldots \cup U_{kn}\}$

Canonical Collection of LALR states: $Coll_{LALR}$

Let ⇔ be the equivalence relation on Coll(1) defined below:

$$I_j \Leftrightarrow I_k \quad \textit{iff} \quad I_j{\downarrow}0 = I_k{\downarrow}0$$

where $I{\downarrow}0 = \{A ::= \alpha.\beta \mid A ::= \alpha.\beta/U \in I\}$

$Coll_{LALR}$ = Coll(1)/⇔ = {I/⇔ | I ∈Coll(1)}
(partition Coll(1) modulo ⇔)
dove I/⇔ = h ∈{J ∈Coll(1) | J⇔I}

(∀ I_j,I_k∈Coll(1)),(∀ a ∈ N ∪ T),
I_j=Goto(I_k,a) *se e solo se* I_j/⇔ =Goto(I_k/⇔,a)

Function Goto of LALR parser

SLR(1), LR(1), LALR(1): Conflict Comparation

shift/reduce

$A ::= \alpha./U$, $B ::= \beta.a\delta/V \in Ij \in Coll_{LALR(1)}$ and $a \in U$

$U = U_1 \cup \ldots \cup U_n$

↓

The conflict persists in LR(1) for some Ij_i ($1 \leq i \leq n$)

reduce/reduce

The conflict could be absent in LR(1)

LALR(1) Grammars

Prop1: (No shift-reduce)

$(\forall\, I_k \in Coll_{LALR}), (\forall\, A ::= \alpha.a\gamma/V\ ,\ B ::= \beta./U\ \in I_k)$

$a \notin U$

Prop2: (No reduce-reduce)

$(\forall\, I_k \in Coll_{LALR}), (\forall\, A ::= \alpha./V\ ,\ B ::= \beta./U \in I_k)$

$V \cap U = \{\}$

Theorem. Let G be a grammar.

- G∈LALR(1) if and only if both Prop1 and Prop2 hold.

- G has shift-reduce, 1 symbol of lookahead, LALR parser if and only if both Prop1 and Prop2 hold.

Bottom-up:
Concluding Remarks

1. Why SLR(1) ⊂ LALR(1) ⊂ LR(1) is strict inclusion ?

2. Is for each language **L** decidable the existence of some **G** such that:
L(**G**) = **L** and **G** ∈LALR(1) [resp. **G** ∈LR(1)] ?

3. Why, looking for a 1-lookahead, bottom-up, parser for some G, people try proving G ∈SLR(1), for first ?

Bottom-up:

Concluding Remarks

4. If $G \notin LALR(1)$ for some G, due to the violation of Prop.2, nevertheless $G \in LR$ can hold. Why?

5. Why $SLR(1) \subset LALR(1)$?
   - Prove that: if $G \in SLR(1)$ then $G \in LALR(1)$
   - Prove that inclusion is really a strict one ?

6. Let G be an ambiguous grammar:
   Which of the two properties is violated, for sure: Prop.1 or Prop2 ?
   Why ?