Approximating Context-Free Grammars for Parsing and Verification

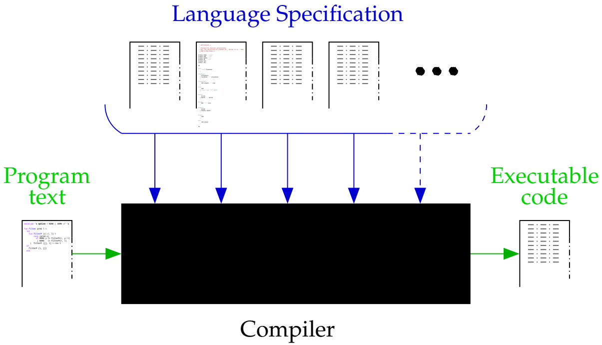
Sylvain Schmitz

Laboratoire I3S, Université de Nice - Sophia Antipolis & CNRS

September 24, 2007

Standard ML

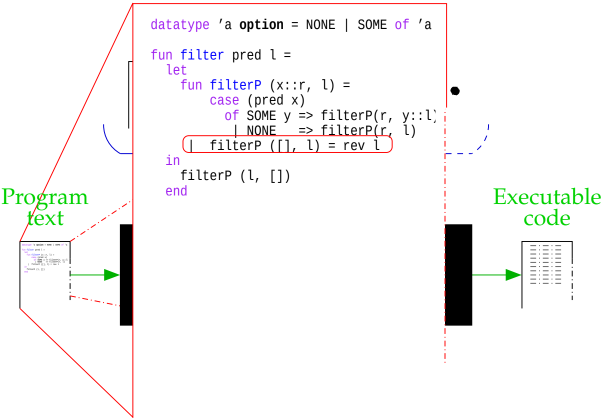
Milner et al. [1997]



Standard ML

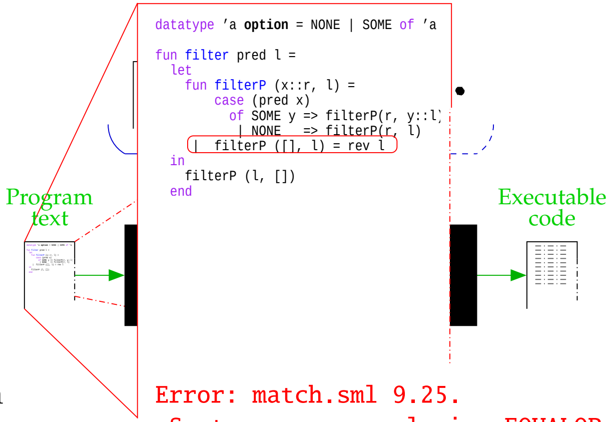
Milner et al. [1997]

```
datatype 'a option = NONE | SOME of 'a

fun filter pred l =
  let
    fun filterP (x::r, l) =
        case (pred x)
          of SOME y => filterP(r, y::l)
           | NONE   => filterP(r, l)
      |  filterP ([], l) = rev l
  in
    filterP (l, [])
  end
```



Standard ML

Milner et al. [1997]



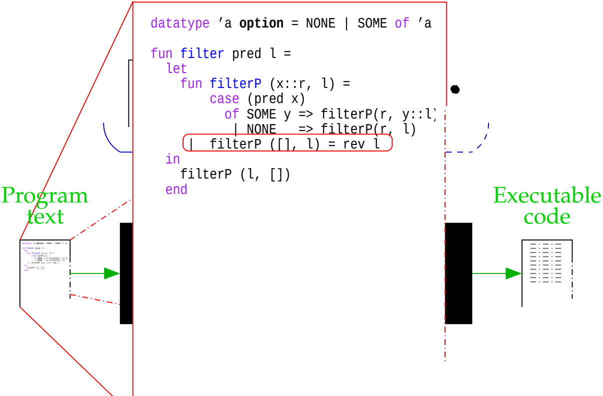

```
datatype 'a option = NONE | SOME of 'a

fun filter pred l =
  let
    fun filterP (x::r, l) =
        case (pred x)
          of SOME y => filterP(r, y::l)
           | NONE   => filterP(r, l)
     |  filterP ([], l) = rev l
  in
    filterP (l, [])
  end
```

- MLton
- Moscow ML
- Poly/ML
- SML/NJ

```
Error: match.sml 9.25.
 Syntax error: replacing EQUALOP with DARROW
```

Standard ML

Milner et al. [1997]

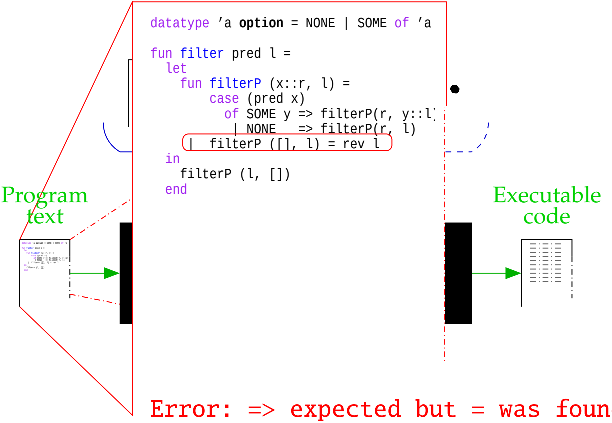


- MLton
- Moscow ML
- Poly/ML
- SML/NJ

```
! Toplevel input:
!       |  filterP ([], l) = rev l
!                            ^
! Syntax error.
```

Standard ML

Milner et al. [1997]



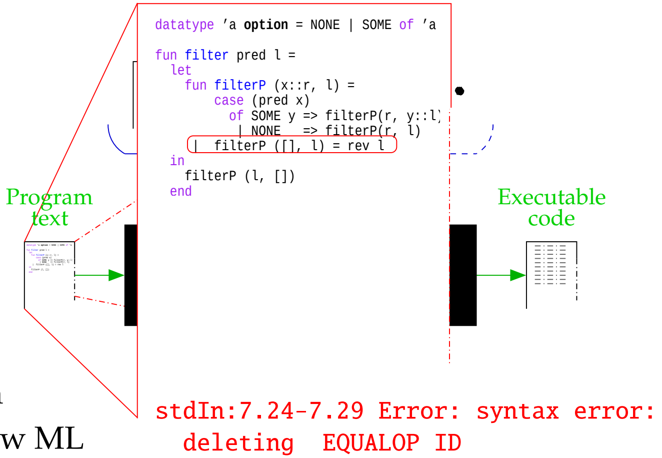

```
datatype 'a option = NONE | SOME of 'a

fun filter pred l =
  let
    fun filterP (x::r, l) =
        case (pred x)
          of SOME y => filterP(r, y::l)
           | NONE   => filterP(r, l)
    |  filterP ([], l) = rev l
  in
    filterP (l, [])
  end
```

Error: => expected but = was found

- MLton
- Moscow ML
- Poly/ML
- SML/NJ

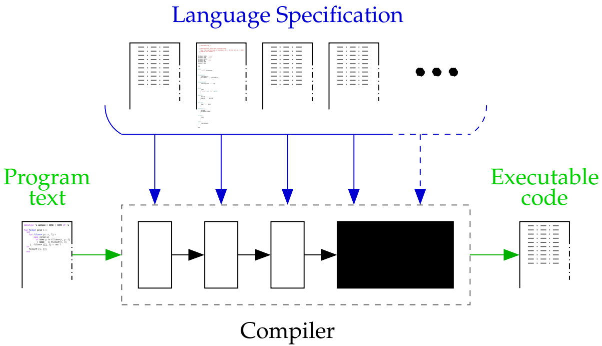
Standard ML

Milner et al. [1997]

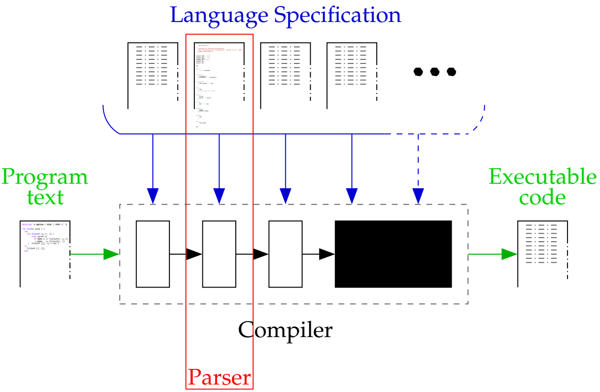


- MLton
- Moscow ML
- Poly/ML
- SML/NJ

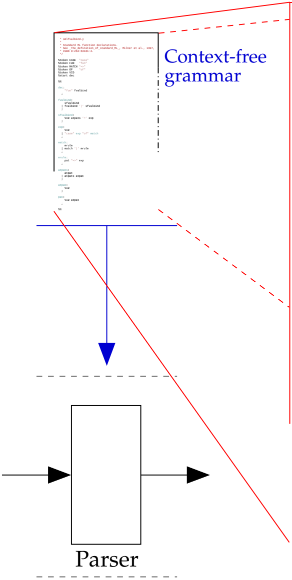
Parsers


Language Specification
Program text
Executable code
Compiler

Parsers

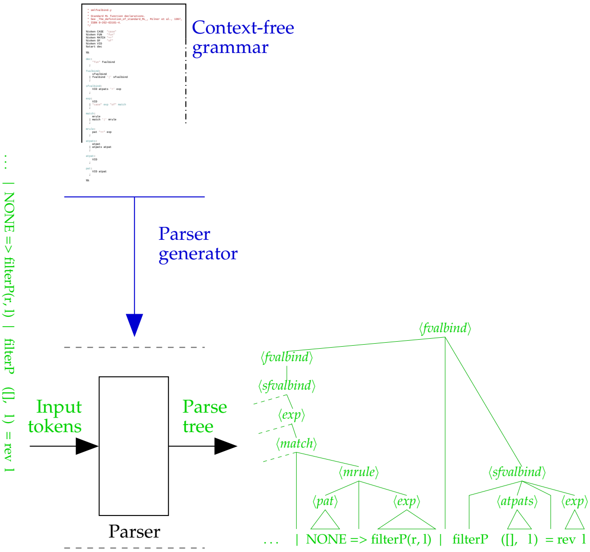

Language Specification
Program text
Executable code
Compiler
Parser

Parsers

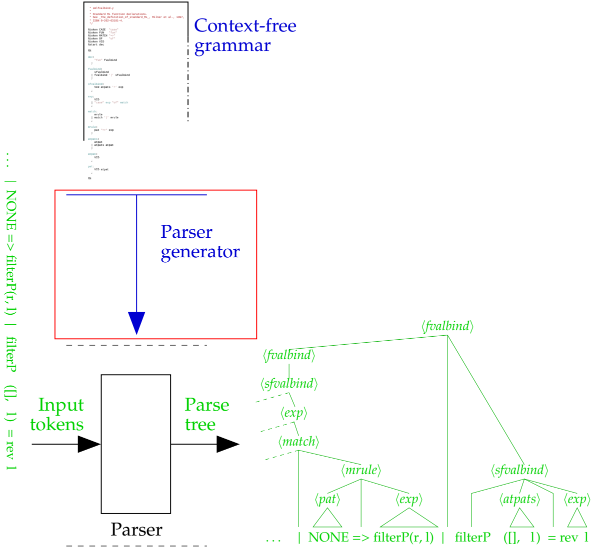



```
 * smlfvalbind.y
 *
 * Standard ML function declarations.
 * See _The_definition_of_standard_ML_, Milner et al., 1997,
 * ISBN 0-262-63181-4.
 */

%token CASE  "case"
%token FUN   "fun"
%token MATCH "=>"
%token OF    "of"
%token VID
%start dec

%%

dec:
    "fun" fvalbind
  ;

fvalbind:
    sfvalbind
  | fvalbind '|' sfvalbind
  ;

sfvalbind:
    VID atpats '=' exp
  ;

exp:
    VID
  | "case" exp "of" match
  ;

match:
    mrule
  | match '|' mrule
  ;

mrule:
    pat "=>" exp
  ;

atpats:
    atpat
  | atpats atpat
  ;

atpat:
    VID
  ;

pat:
    VID atpat
  ;

%%
```

Parsers


* smlfvalbind.y
*
* Standard ML function declarations.
* See _The_definition_of_standard_ML_, Milner et al., 1997,
* ISBN 0-262-63181-4.
*/
%token CASE "case"
%token FUN "fun"
%token MATCH "=>"
%token OF "of"
%token VID
%start dec
%%
dec:
"fun" fvalbind
;
fvalbind:
sfvalbind
| fvalbind '|' sfvalbind
;
sfvalbind:
VID atpats '=' exp
;
exp:
VID
| "case" exp "of" match
;
match:
mrule
| match '|' mrule
;
mrule:
pat "=>" exp
;
atpats:
atpat
| atpats atpat
;
atpat:
VID
;
pat:
VID atpat
;
%%
Context-free grammar
Parser generator
... | NONE => filterP(r, l) | filterP ([], l) = rev l
Input tokens
Parser
Parse tree
⟨fvalbind⟩
⟨fvalbind⟩
⟨sfvalbind⟩
⟨exp⟩
⟨match⟩
⟨mrule⟩
⟨pat⟩
⟨exp⟩
⟨sfvalbind⟩
⟨atpats⟩
⟨exp⟩
... | NONE => filterP(r, l) | filterP ([], l) = rev l

Parsers


Context-free grammar
Parser generator
Input tokens
Parser
Parse tree
⟨fvalbind⟩
⟨fvalbind⟩
⟨sfvalbind⟩
⟨exp⟩
⟨match⟩
⟨mrule⟩
⟨pat⟩
⟨exp⟩
⟨sfvalbind⟩
⟨atpats⟩
⟨exp⟩
... | NONE => filterP(r, l) | filterP ([], l) = rev l

LALR(1) Parser Generator

- GNU Bison

```
state 20
    6 exp: "case" exp "of" match .
    8 match: match . '|' mrule

    '|'  shift, and go to state 24
    '|'       [reduce using rule 6 (exp)]
```

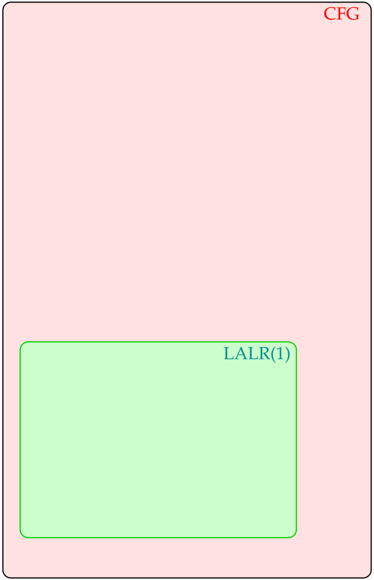
- Restricted grammar class

LALR(1) Parser Generator

- GNU Bison

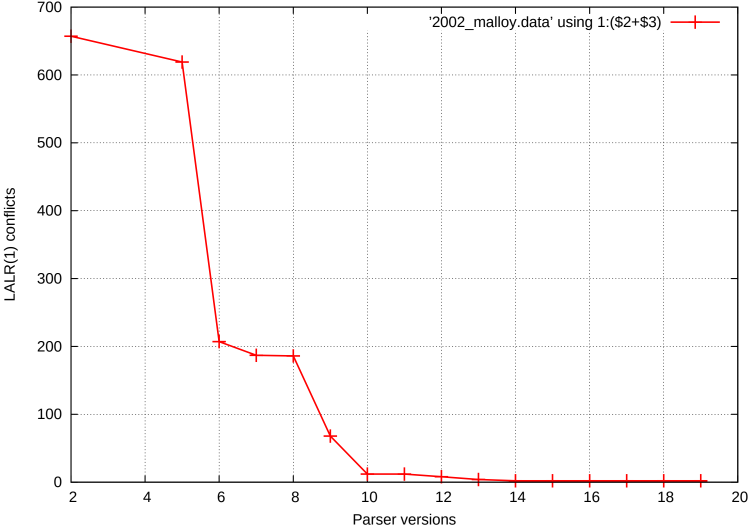
```
state 20
    6 exp: "case" exp "of" match .
    8 match: match . '|' mrule

    '|'  shift, and go to state 24
    '|'       [reduce using rule 6 (exp)]
```

- Restricted grammar class



Dealing with Conflicts

An Objective Measure [Malloy et al., 2002] on a C# Grammar



Dealing with Conflicts

A Subjective Measure

Courtesy of `http://www.phdcomics.com`.

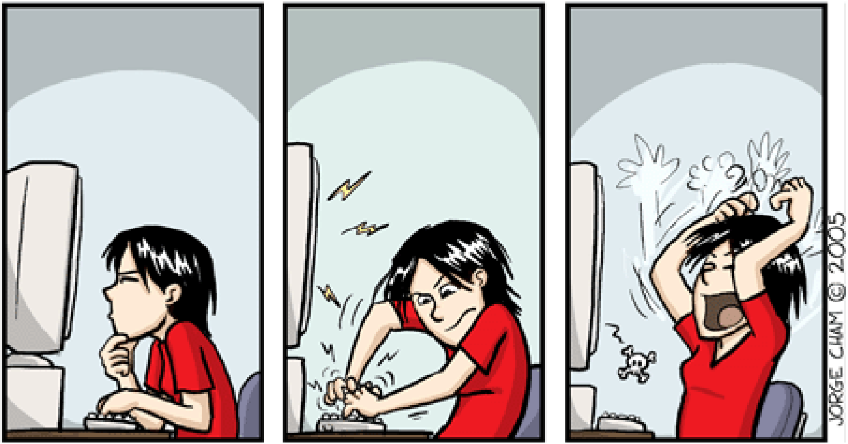
Dealing with Conflicts

A Subjective Measure

Courtesy of `http://www.phdcomics.com`.

Dealing with Conflicts

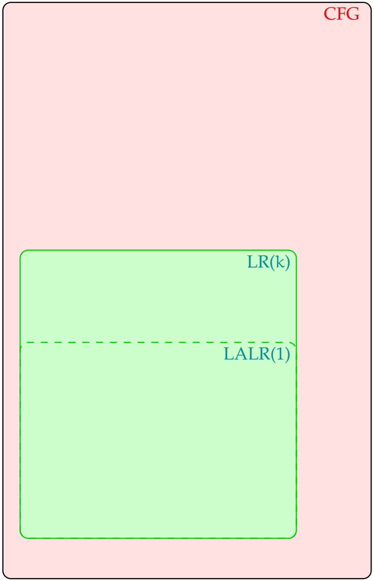
A Subjective Measure



Courtesy of `http://www.phdcomics.com`.

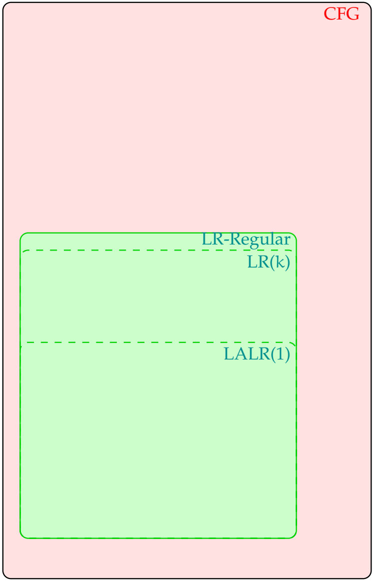
State of the Art

- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
  - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
  - Horizontal and vertical unambiguity test [Brabrand et al., 2007]



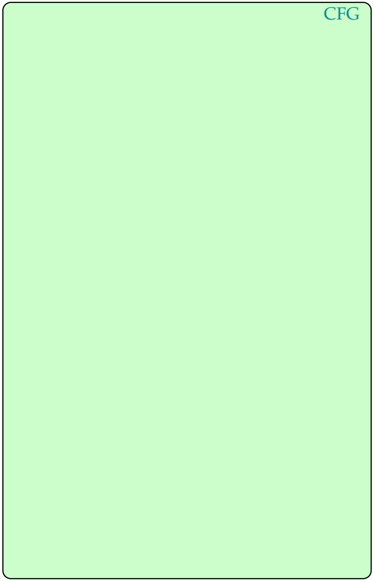
State of the Art

- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
  - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
  - Horizontal and vertical unambiguity test [Brabrand et al., 2007]

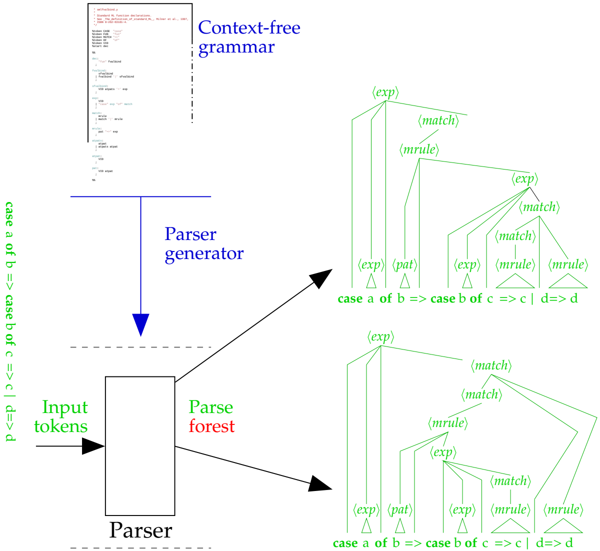


State of the Art

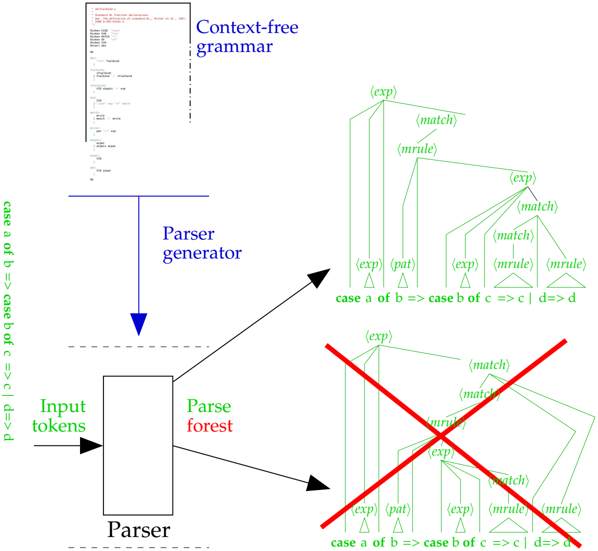
- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
  - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
  - Horizontal and vertical unambiguity test [Brabrand et al., 2007]



Ambiguity

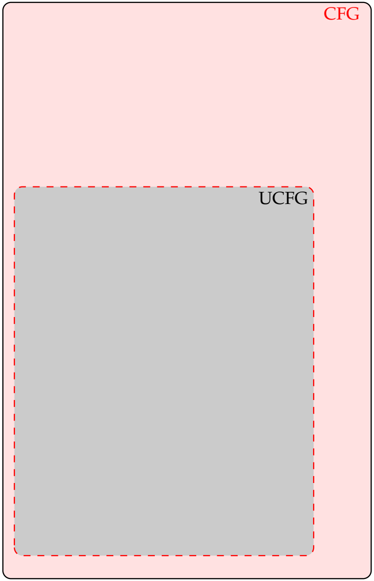

Context-free grammar
Parser generator
case a of b => case b of c => c | d=> d
Input tokens
Parser
Parse forest
⟨exp⟩
⟨match⟩
⟨mrule⟩
⟨exp⟩
⟨match⟩
⟨match⟩
⟨exp⟩ ⟨pat⟩ ⟨exp⟩ ⟨mrule⟩ ⟨mrule⟩
case a of b => case b of c => c | d=> d
⟨exp⟩
⟨match⟩
⟨match⟩
⟨mrule⟩
⟨exp⟩
⟨match⟩
⟨exp⟩ ⟨pat⟩ ⟨exp⟩ ⟨mrule⟩ ⟨mrule⟩
case a of b => case b of c => c | d=> d

Ambiguity


Context-free grammar
Parser generator
case a of b => case b of c => c | d=> d
Input tokens
Parser
Parse forest
⟨exp⟩
⟨match⟩
⟨mrule⟩
⟨exp⟩ ⟨pat⟩ ⟨exp⟩ ⟨mrule⟩ ⟨mrule⟩
case a of b => case b of c => c | d=> d
⟨exp⟩
⟨match⟩
⟨match⟩
⟨mrule⟩
⟨exp⟩
⟨match⟩
⟨exp⟩ ⟨pat⟩ ⟨exp⟩ ⟨mrule⟩ ⟨mrule⟩
case a of b => case b of c => c | d=> d

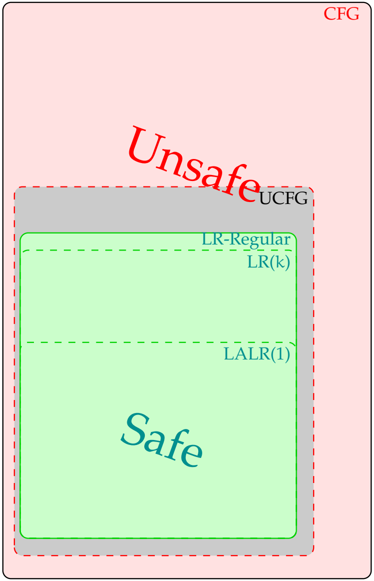
State of the Art

- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
  - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
  - Horizontal and vertical unambiguity test [Brabrand et al., 2007]



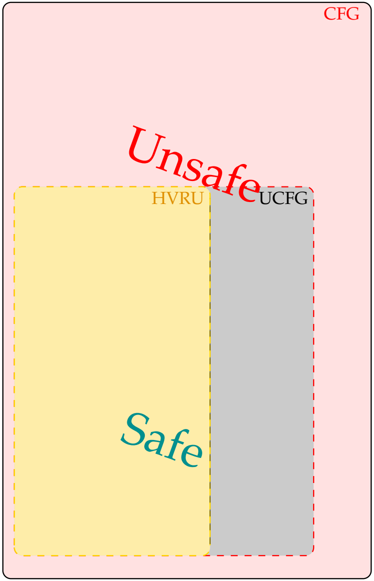
State of the Art

- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
  - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
  - Horizontal and vertical unambiguity test [Brabrand et al., 2007]



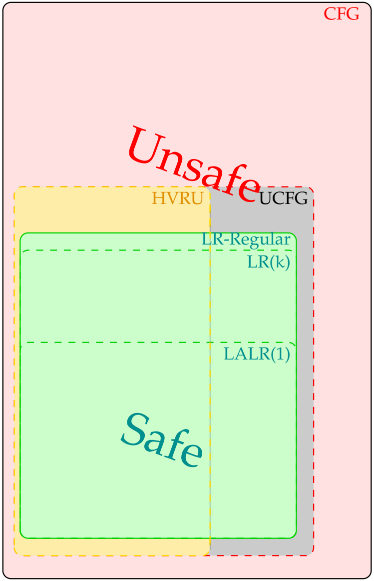
State of the Art

- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
  - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
  - Horizontal and vertical unambiguity test [Brabrand et al., 2007]



State of the Art

- LR(k) [Knuth, 1965]
- LR-Regular [Čulik and Cohen, 1973]
- Generalized LR [Tomita, 1986]
    - Unambiguous CFGs [Cantor, 1962, Chomsky and Schützenberger, 1963]
    - Horizontal and vertical unambiguity test [Brabrand et al., 2007]

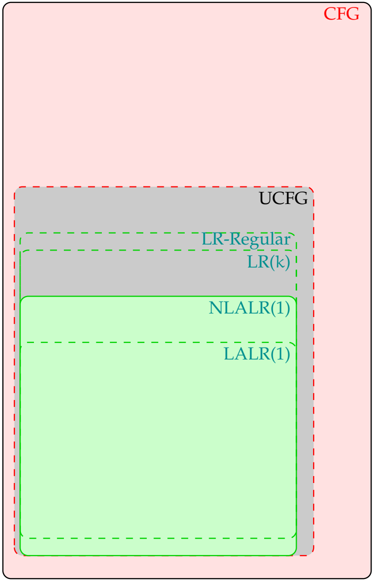


Contributions

- Noncanonical parsing methods [Szymanski and Williams, 1976, Tai, 1979]
  - Noncanonical LALR(1)
  - Shift-Resolve
- Noncanonical unambiguity test
- Framework for grammar approximations

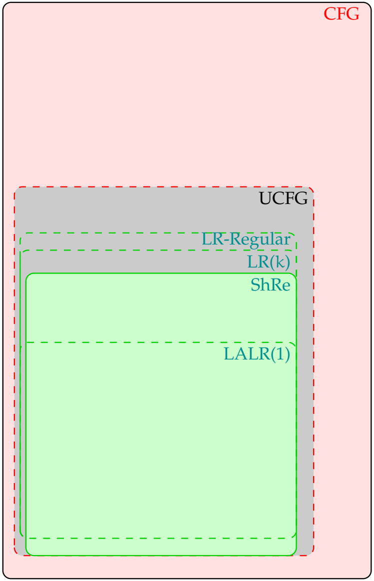
Contributions

- Noncanonical parsing methods [Szymanski and Williams, 1976, Tai, 1979]
  - Noncanonical LALR(1)
  - Shift-Resolve
- Noncanonical unambiguity test
- Framework for grammar approximations



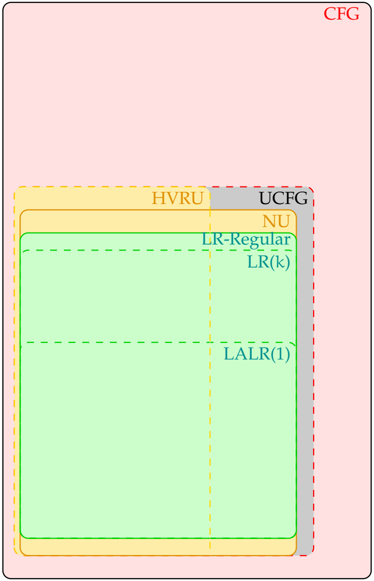
Contributions

- Noncanonical parsing methods [Szymanski and Williams, 1976, Tai, 1979]
  - Noncanonical LALR(1)
  - Shift-Resolve
- Noncanonical unambiguity test
- Framework for grammar approximations



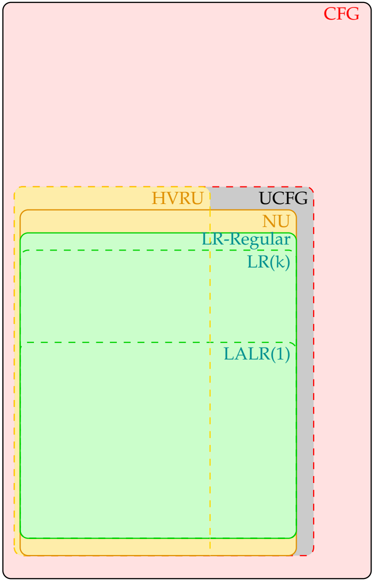
Contributions

- Noncanonical parsing methods [Szymanski and Williams, 1976, Tai, 1979]
  - Noncanonical LALR(1)
  - Shift-Resolve
- Noncanonical unambiguity test
- Framework for grammar approximations



Contributions

- Noncanonical parsing methods [Szymanski and Williams, 1976, Tai, 1979]
    - Noncanonical LALR(1)
    - Shift-Resolve
- Noncanonical unambiguity test
- Framework for grammar approximations



Bracketed Grammars

$$\mathcal{G} = \langle N, T, P, S \rangle,\ V = N \cup T$$

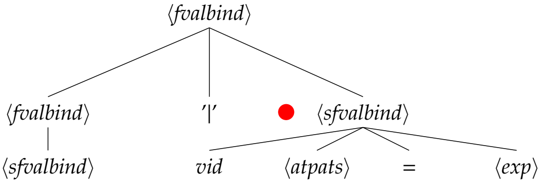
$$\begin{aligned}
\langle dec \rangle &\xrightarrow{1} \textbf{fun}\ \langle fvalbind \rangle \\
\langle fvalbind \rangle &\xrightarrow{2} \langle sfvalbind \rangle \\
\langle fvalbind \rangle &\xrightarrow{3} \langle fvalbind \rangle\ '|'\ \langle sfvalbind \rangle \\
\langle sfvalbind \rangle &\xrightarrow{4} vid\ \langle atpats \rangle\ =\ \langle exp \rangle \\
\langle exp \rangle &\xrightarrow{5} \textbf{case}\ \langle exp \rangle\ \textbf{of}\ \langle match \rangle \\
\langle match \rangle &\xrightarrow{6} \langle mrule \rangle \\
\langle match \rangle &\xrightarrow{7} \langle match \rangle\ '|'\ \langle mrule \rangle \\
\langle mrule \rangle &\xrightarrow{8} \langle pat \rangle\ \texttt{=>}\ \langle exp \rangle \\
\langle atpats \rangle &\xrightarrow{9} \langle atpat \rangle \\
\langle atpats \rangle &\xrightarrow{10} \langle atpats \rangle\ \langle atpat \rangle \\
\langle pat \rangle &\xrightarrow{11} vid\ \langle atpat \rangle \\
\langle atpat \rangle &\xrightarrow{12} vid
\end{aligned}$$

Bracketed Grammars

$$\mathcal{G}_b = \langle N, T_b, P_b, S \rangle,\ V_b = N \cup T_b$$

$$\begin{array}{rcl}
\langle dec \rangle & \xrightarrow{1} & d_1\ \textbf{fun}\ \langle fvalbind \rangle\ r_1 \\
\langle fvalbind \rangle & \xrightarrow{2} & d_2\ \langle sfvalbind \rangle\ r_2 \\
\langle fvalbind \rangle & \xrightarrow{3} & d_3\ \langle fvalbind \rangle\ '|'\ \langle sfvalbind \rangle\ r_3 \\
\langle sfvalbind \rangle & \xrightarrow{4} & d_4\ vid\ \langle atpats \rangle\ =\ \langle exp \rangle\ r_4 \\
\langle exp \rangle & \xrightarrow{5} & d_5\ \textbf{case}\ \langle exp \rangle\ \textbf{of}\ \langle match \rangle\ r_5 \\
\langle match \rangle & \xrightarrow{6} & d_6\ \langle mrule \rangle\ r_6 \\
\langle match \rangle & \xrightarrow{7} & d_7\ \langle match \rangle\ '|'\ \langle mrule \rangle\ r_7 \\
\langle mrule \rangle & \xrightarrow{8} & d_8\ \langle pat \rangle\ \texttt{=>}\ \langle exp \rangle\ r_8 \\
\langle atpats \rangle & \xrightarrow{9} & d_9\ \langle atpat \rangle\ r_9 \\
\langle atpats \rangle & \xrightarrow{10} & d_{10}\ \langle atpats \rangle\ \langle atpat \rangle\ r_{10} \\
\langle pat \rangle & \xrightarrow{11} & d_{11}\ vid\ \langle atpat \rangle\ r_{11} \\
\langle atpat \rangle & \xrightarrow{12} & d_{12}\ vid\ r_{12}
\end{array}$$

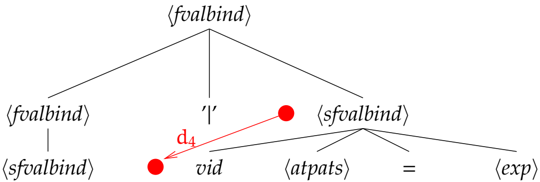
Positions



$$d_3\ d_2\ \langle \mathit{sfvalbind} \rangle\ r_2\ '|'\ \bullet\ d_4\ \mathit{vid}\ \langle \mathit{atpats} \rangle\ =\ \langle \mathit{exp} \rangle\ r_4\ r_3$$

Position Graph Γ

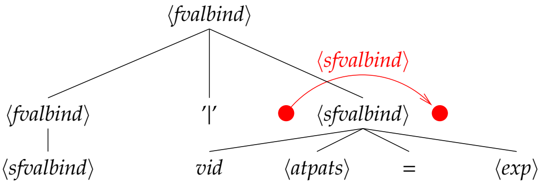
Left-to-right Walks in Trees



$$d_3\ d_2\ \langle \mathit{sfvalbind} \rangle\ r_2\ '|'\ d_4 \bullet\ \mathit{vid}\ \langle \mathit{atpats} \rangle\ =\ \langle \mathit{exp} \rangle\ r_4\ r_3$$

Position Graph Γ

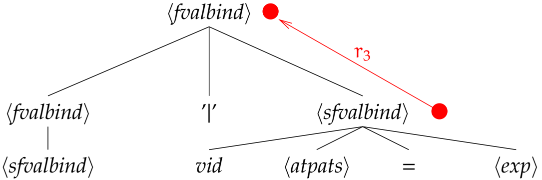
Left-to-right Walks in Trees



$$\mathrm{d}_3\ \mathrm{d}_2\ \langle \mathit{sfvalbind} \rangle\ \mathrm{r}_2\ '|'\ \mathrm{d}_4\ \mathit{vid}\ \langle \mathit{atpats} \rangle\ =\ \langle \mathit{exp} \rangle\ \mathrm{r}_4 \bullet\ \mathrm{r}_3$$

Position Graph Γ

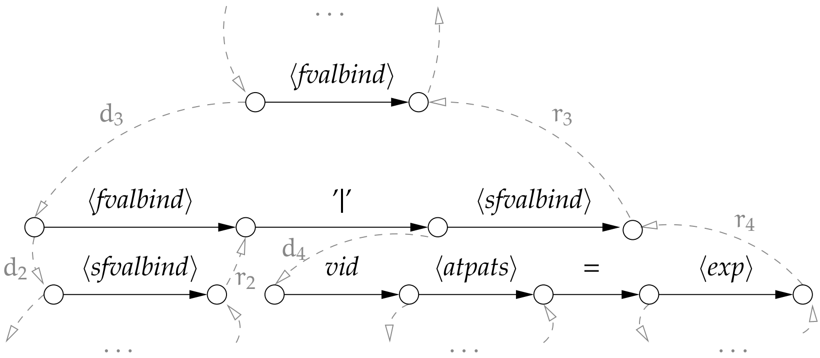
Left-to-right Walks in Trees



$$\mathrm{d}_3\ \mathrm{d}_2\ \langle \mathit{sfvalbind} \rangle\ \mathrm{r}_2\ {}'|'\ \mathrm{d}_4\ \mathit{vid}\ \langle \mathit{atpats} \rangle\ =\ \langle \mathit{exp} \rangle\ \mathrm{r}_4\ \mathrm{r}_3 \bullet$$

Position Graph Γ

Left-to-right Walks in Trees



Position Automaton $\Gamma/\equiv$

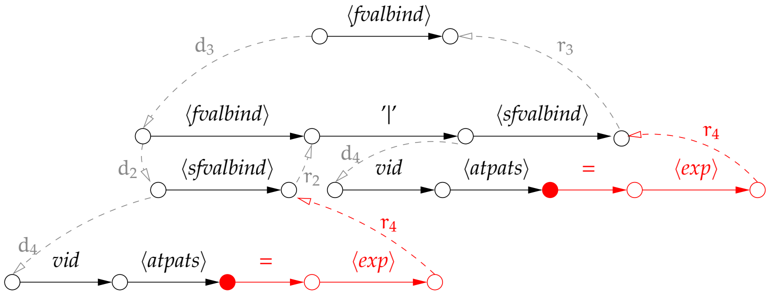
Definition

$\Gamma/\equiv$ is the quotient of Γ by an equivalence relation $\equiv$ between positions.

Theorem (Language over-approximation)

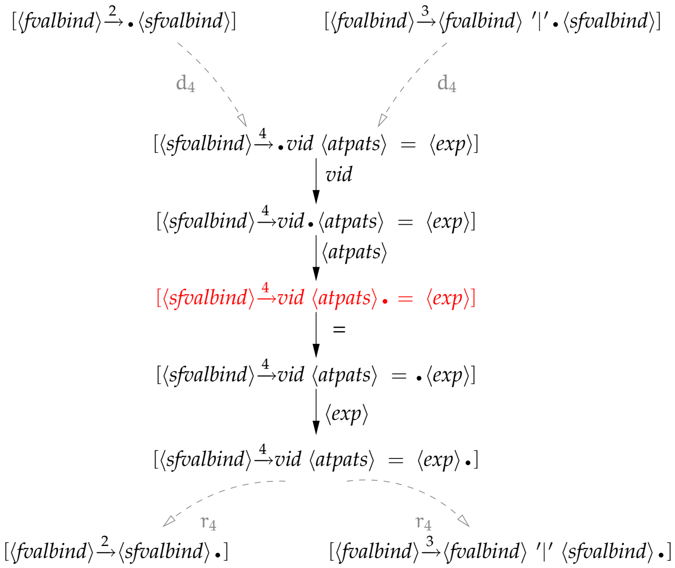
$$\mathcal{L}(\mathcal{G}_b) \subseteq \mathcal{L}(\Gamma/\equiv) \cap T_b^*$$

Example: item$_0$ Equivalence



- equivalence class $[\langle sfvalbind\rangle \xrightarrow{4} vid\ \langle atpats\rangle \bullet\ =\ \langle exp\rangle]$
- LR(0) items
- Γ/item_0: nondeterministic LR(0) automaton

Example: item_0 Equivalence



Summary

- general framework for approximations
- applications:
  - parser construction
  - ambiguity detection
  - XML validation [Segoufin and Vianu, 2002]?
  - symbolic supertagging [Boullier, 2003]?

Summary

- general framework for approximations
- applications:
  - parser construction
  - ambiguity detection
  - XML validation [Segoufin and Vianu, 2002]?
  - symbolic supertagging [Boullier, 2003]?

Shift-Resolve Parsing

- noncanonical
- $k = 1$ reduced lookahead symbol
- resolve = reduce + pushback: emulates a bounded reduced lookahead without any preset bound

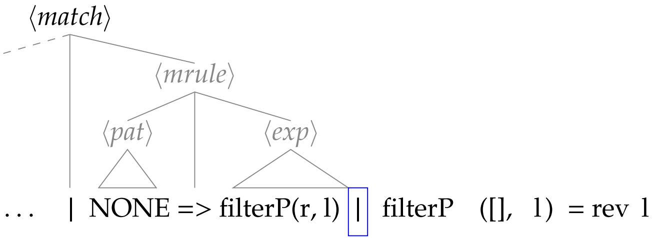
Shift-Resolve Parsing

- noncanonical
- $k = 1$ reduced lookahead symbol
- resolve = reduce + pushback: emulates a bounded reduced lookahead without any preset bound

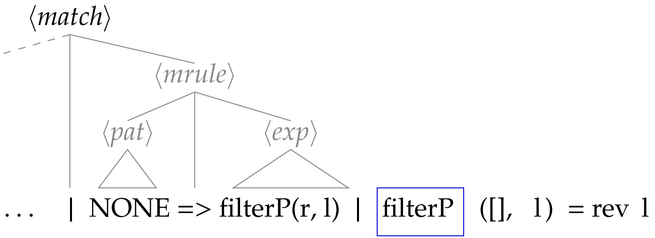
Shift-Resolve Parse

. . . | NONE => filterP(r, l) | filterP ([], l) = rev l

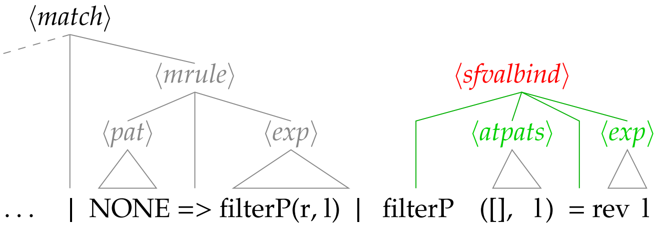
Shift-Resolve Parse



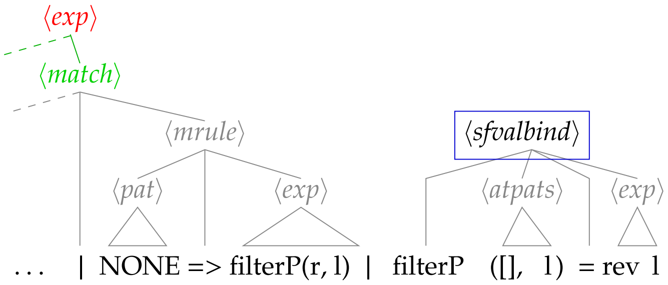
Shift-Resolve Parse



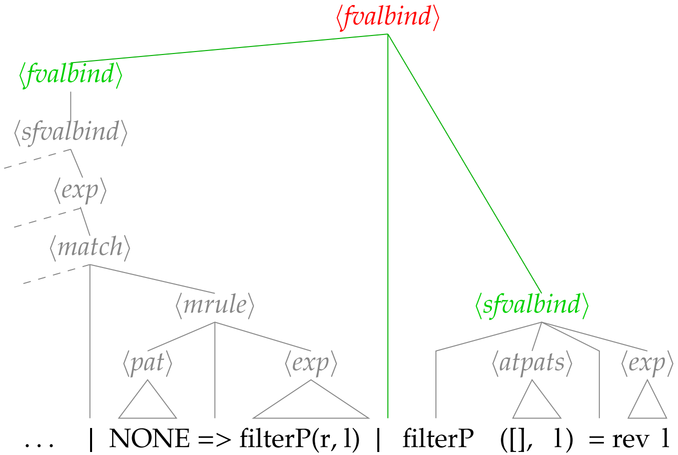
Shift-Resolve Parse



Shift-Resolve Parse



Shift-Resolve Parse



Generating the Parser

1. position automaton
2. determinization by subset construction

Subset Construction

Principle

- d_i transitions denote traditional item closures
- r_i transitions denote a phrase that should be reduced
- other transitions denote shifts
- items in the construction hold
  1. a state of the position automaton
  2. a parsing action
  3. a pushback length

Subset Construction

Principle

- d_i transitions denote traditional item closures
- r_i transitions denote a phrase that should be reduced
- other transitions denote shifts
- items in the construction hold
    1. a state of the position automaton
    2. a parsing action
    3. a pushback length

Subset Construction

Example

$$\langle exp \rangle \rightarrow \textbf{case } \langle exp \rangle \textbf{ of } \langle match \rangle \bullet, 0, 0$$
$$\langle match \rangle \rightarrow \langle match \rangle \bullet \text{ '|' } \langle mrule \rangle, 0, 0$$

Subset Construction

Example

r_5

$\langle exp\rangle \rightarrow \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle \bullet, 0, 0$

$\langle match\rangle \rightarrow \langle match\rangle \bullet\ \text{'|'}\ \langle mrule\rangle, 0, 0$

$\langle sfvalbind\rangle \rightarrow vid\ \langle atpats\rangle = \langle exp\rangle \bullet, 5, 0$

Subset Construction

Example

$$\langle exp\rangle \to \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle\,\bullet, 0, 0$$

$$\langle match\rangle \to \langle match\rangle\,\bullet\ \text{'|'}\ \langle mrule\rangle, 0, 0$$

$$\langle sfvalbind\rangle \to vid\ \langle atpats\rangle = \langle exp\rangle\,\bullet, 5, 0$$

r_4

$$\langle fvalbind\rangle \to \langle fvalbind\rangle\ \text{'|'}\ \langle sfvalbind\rangle\,\bullet, 5, 0$$

$$\langle fvalbind\rangle \to \langle sfvalbind\rangle\,\bullet, 5, 0$$

Subset Construction

Example

$\langle exp\rangle \rightarrow \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle \bullet, 0, 0$

$\langle match\rangle \rightarrow \langle match\rangle \bullet\ \text{'|'}\ \langle mrule\rangle, 0, 0$

$\langle sfvalbind\rangle \rightarrow vid\ \langle atpats\rangle = \langle exp\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\ \text{'|'}\ \langle sfvalbind\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle sfvalbind\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle \bullet\ \text{'|'}\ \langle sfvalbind\rangle, 5, 0$

$\langle dec\rangle \rightarrow \textbf{fun}\ \langle fvalbind\rangle \bullet, 5, 0$

$S' \rightarrow \langle dec\rangle \bullet \$, 5, 0$

Subset Construction

Example

$\langle exp \rangle \rightarrow \textbf{case}\ \langle exp \rangle\ \textbf{of}\ \langle match \rangle \bullet, 0, 0$

$\langle match \rangle \rightarrow \langle match \rangle \bullet\ '|'\ \langle mrule \rangle, 0, 0$

$\langle sfvalbind \rangle \rightarrow vid\ \langle atpats \rangle = \langle exp \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle\ '|'\ \langle sfvalbind \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle sfvalbind \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle \bullet\ '|'\ \langle sfvalbind \rangle, 5, 0$

$\langle dec \rangle \rightarrow \textbf{fun}\ \langle fvalbind \rangle \bullet, 5, 0$

$S' \rightarrow \langle dec \rangle \bullet \$, 5, 0$

Subset Construction

Example

$\langle exp\rangle \rightarrow \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle \bullet, 0, 0$

$\langle match\rangle \rightarrow \langle match\rangle \bullet\ '|'\ \langle mrule\rangle, 0, 0$

$\langle sfvalbind\rangle \rightarrow vid\ \langle atpats\rangle = \langle exp\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\ '|'\ \langle sfvalbind\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle sfvalbind\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle \bullet\ '|'\ \langle sfvalbind\rangle, 5, 0$

$\langle dec\rangle \rightarrow \textbf{fun}\ \langle fvalbind\rangle \bullet, 5, 0$

$S' \rightarrow \langle dec\rangle \bullet \$, 5, 0$

$\downarrow\ '|'$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\ '|'\ \bullet \langle sfvalbind\rangle, 5, 1$

$\langle match\rangle \rightarrow \langle match\rangle\ '|'\ \bullet \langle mrule\rangle, 0, 0$

Subset Construction

Example

$\langle exp \rangle \rightarrow \textbf{case}\ \langle exp \rangle\ \textbf{of}\ \langle match \rangle \bullet, 0, 0$

$\langle match \rangle \rightarrow \langle match \rangle \bullet\ \text{'|'}\ \langle mrule \rangle, 0, 0$

$\langle sfvalbind \rangle \rightarrow vid\ \langle atpats \rangle = \langle exp \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle\ \text{'|'}\ \langle sfvalbind \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle sfvalbind \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle \bullet\ \text{'|'}\ \langle sfvalbind \rangle, 5, 0$

$\langle dec \rangle \rightarrow \textbf{fun}\ \langle fvalbind \rangle \bullet, 5, 0$

$S' \rightarrow \langle dec \rangle \bullet \$, 5, 0$

↓ '|'

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle\ \text{'|'} \bullet \langle sfvalbind \rangle, 5, 1$

d_8 $\langle match \rangle \rightarrow \langle match \rangle\ \text{'|'} \bullet \langle mrule \rangle, 0, 0$

$\langle mrule \rangle \rightarrow \bullet \langle pat \rangle \Rightarrow \langle exp \rangle, 0, 0$

Subset Construction

Example

$\langle exp\rangle \to \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle \bullet, 0, 0$

$\langle match\rangle \to \langle match\rangle \bullet\ '|'\ \langle mrule\rangle, 0, 0$

$\langle sfvalbind\rangle \to vid\ \langle atpats\rangle = \langle exp\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \to \langle fvalbind\rangle\ '|'\ \langle sfvalbind\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \to \langle sfvalbind\rangle \bullet, 5, 0$

$\langle fvalbind\rangle \to \langle fvalbind\rangle \bullet\ '|'\ \langle sfvalbind\rangle, 5, 0$

$\langle dec\rangle \to \textbf{fun}\ \langle fvalbind\rangle \bullet, 5, 0$

$S' \to \langle dec\rangle \bullet \$, 5, 0$

$\downarrow\ '|'$

$\langle fvalbind\rangle \to \langle fvalbind\rangle\ '|' \bullet \langle sfvalbind\rangle, 5, 1$

$\langle match\rangle \to \langle match\rangle\ '|' \bullet \langle mrule\rangle, 0, 0$

$\langle mrule\rangle \to \bullet \langle pat\rangle \Rightarrow \langle exp\rangle, 0, 0$

$\langle pat\rangle \to \bullet vid\ \langle atpat\rangle, 0, 0$

$\langle sfvalbind\rangle \to \bullet vid\ \langle atpats\rangle = \langle exp\rangle, 0, 0$

Construction Failure

$\langle exp\rangle \rightarrow \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle\,\bullet, 0, 0$

$\langle match\rangle \rightarrow \langle match\rangle\,\bullet\ '|'\ \langle mrule\rangle, 0, 0$

$\langle sfvalbind\rangle \rightarrow vid\ \langle atpats\rangle = \langle exp\rangle\,\bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\ '|'\ \langle sfvalbind\rangle\,\bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle sfvalbind\rangle\,\bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\,\bullet\ '|'\ \langle sfvalbind\rangle, 5, 0$

$\langle dec\rangle \rightarrow \textbf{fun}\ \langle fvalbind\rangle\,\bullet, 5, 0$

$S' \rightarrow \langle dec\rangle\,\bullet \$, 5, 0$

Construction Failure

$\langle exp \rangle \rightarrow \textbf{case}\ \langle exp \rangle\ \textbf{of}\ \langle match \rangle \bullet, 0, 0$

$\langle match \rangle \rightarrow \langle match \rangle \bullet\ \text{'|'}\ \langle mrule \rangle, 0, 0$

$\langle sfvalbind \rangle \rightarrow vid\ \langle atpats \rangle = \langle exp \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle\ \text{'|'}\ \langle sfvalbind \rangle \bullet, 5, 0$

r_5 $\langle fvalbind \rangle \rightarrow \langle sfvalbind \rangle \bullet, 5, 0$

$\langle fvalbind \rangle \rightarrow \langle fvalbind \rangle \bullet\ \text{'|'}\ \langle sfvalbind \rangle, 5, 0$

$\langle dec \rangle \rightarrow \textbf{fun}\ \langle fvalbind \rangle \bullet, 5, 0$

$S' \rightarrow \langle dec \rangle \bullet \$, 5, 0$

$\langle mrule \rangle \rightarrow \langle pat \rangle\ \text{'|'}\ \langle exp \rangle \bullet, 5, 0$

Construction Failure

$\langle exp\rangle \rightarrow \textbf{case}\ \langle exp\rangle\ \textbf{of}\ \langle match\rangle\,\bullet, 0, 0$

$\langle match\rangle \rightarrow \langle match\rangle\,\bullet\ '|'\ \langle mrule\rangle, 0, 0$

$\langle sfvalbind\rangle \rightarrow vid\ \langle atpats\rangle = \langle exp\rangle\,\bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\ '|'\ \langle sfvalbind\rangle\,\bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle sfvalbind\rangle\,\bullet, 5, 0$

$\langle fvalbind\rangle \rightarrow \langle fvalbind\rangle\,\bullet\ '|'\ \langle sfvalbind\rangle, 5, 0$

$\langle dec\rangle \rightarrow \textbf{fun}\ \langle fvalbind\rangle\,\bullet, 5, 0$

$S' \rightarrow \langle dec\rangle\,\bullet \$, 5, 0$

$\langle mrule\rangle \rightarrow \langle pat\rangle\ '|'\ \langle exp\rangle\,\bullet, 5, 0$

$\langle match\rangle \rightarrow \langle mrule\rangle\,\bullet, 5, 0$

$\langle match\rangle \rightarrow \langle match\rangle\,\bullet\ '|'\ \langle mrule\rangle, 5, 0$

Complexity

- $|\Gamma/{\equiv}|$: size of the position automaton
- $|\mathcal{A}|$: size of the parser: $\mathcal{O}(2^{|\Gamma/{\equiv}|\,|P|})$
- parsing time complexity for input w: $\mathcal{O}(|w|)$

Complexity

- $|\Gamma/\equiv|$: size of the position automaton
  $|\Gamma/\mathsf{item}_0| = \mathcal{O}(|\mathcal{G}|)$
- $|\mathcal{A}|$: size of the parser: $\mathcal{O}(2^{|\Gamma/\equiv|\,|P|})$
- parsing time complexity for input w: $\mathcal{O}(|w|)$

Limitations

– incomparable with classical parsing techniques

+ subset construction mendable

Limitations

- − incomparable with classical parsing techniques
- + subset construction mendable

Summary

- Shift Resolve parsers
  1. Large class of grammars accepted
  2. Unambiguity
  3. Linear time parsing
- 2-steps construction
  1. Simple
  2. Flexible

Principles

- a bracketed sentence = a derivation tree
- ambiguity = more than one tree with the same yield

$d_6\, d_8\, d_{13}$ *vid* r_{13} => d_5 **case** d_{14} *vid* r_{14} **of** $d_7\, d_6\, d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_6$ '|' $d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_7 r_5 r_8 r_6$

$d_7\, d_6\, d_8\, d_{13}$ *vid* r_{13} => d_5 **case** d_{14} *vid* r_{14} **of** $d_7\, d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_7 r_5 r_8 r_6$ '|' $d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_7$

- construct a FSA $\mathcal{A}$ such that $\mathcal{L}(\mathcal{G}_b) \subseteq \mathcal{L}(\mathcal{A})$, and look for bracketed sentences with the same yield

Principles

- a bracketed sentence = a derivation tree
- ambiguity = more than one tree with the same yield

$d_6\, d_8\, d_{13}$ *vid* r_{13} => d_5 **case** d_{14} *vid* r_{14} **of** $d_7\, d_6\, d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14}\, r_8\, r_6$ '|' $d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14}\, r_8\, r_7\, r_5\, r_8\, r_6$

$d_7\, d_6\, d_8\, d_{13}$ *vid* r_{13} => d_5 **case** d_{14} *vid* r_{14} **of** $d_7\, d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14}\, r_8\, r_7\, r_5\, r_8\, r_6$ '|' $d_8\, d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14}\, r_8\, r_7$

- construct a FSA $\mathcal{A}$ such that $\mathcal{L}(\mathcal{G}_b) \subseteq \mathcal{L}(\mathcal{A})$, and look for bracketed sentences with the same yield

Principles

- a bracketed sentence = a derivation tree
- ambiguity = more than one tree with the same yield

$d_6 d_8 d_{13}$ *vid* r_{13} => d_5 **case** d_{14} *vid* r_{14} **of** $d_7 d_6 d_8 d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_6$ '|' $d_8 d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_7 r_5 r_8 r_6$

$d_7 d_6 d_8 d_{13}$ *vid* r_{13} => d_5 **case** d_{14} *vid* r_{14} **of** $d_7 d_8 d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_7 r_5 r_8 r_6$ '|' $d_8 d_{13}$ *vid* r_{13} => d_{14} *vid* $r_{14} r_8 r_7$

- construct a FSA $\mathcal{A}$ such that $\mathcal{L}(\mathcal{G}_b) \subseteq \mathcal{L}(\mathcal{A})$, and look for bracketed sentences with the same yield

RU($\equiv$)

- $\mathcal{G}$ is regular unambiguous for $\equiv$ of finite index, if there does not exist $w_b \neq w'_b$ in $\mathcal{L}(\Gamma/\equiv) \cap T_b^*$ with $h(w_b) = h(w'_b)$
- $\mathrm{LR}(0) \not\subseteq \mathrm{RU}(\mathrm{item}_0)$
- regular approximations are too weak

RU($\equiv$)

- $\mathcal{G}$ is regular unambiguous for $\equiv$ of finite index, if there does not exist $w_b \neq w'_b$ in $\mathcal{L}(\Gamma/\equiv) \cap T_b^*$ with $h(w_b) = h(w'_b)$
- LR(0) $\not\subseteq$ RU(item_0)
- regular approximations are too weak

Nonterminal Transitions

- $\mathcal{SF}(\mathcal{G}_b) \subseteq \mathcal{L}(\Gamma/\equiv)$
- look for two different bracketed sentential forms in $\mathcal{L}(\Gamma/\equiv)$

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

- a nonterminal transition represents exactly its derived context-free language

Nonterminal Transitions

- $\mathcal{SF}(\mathcal{G}_b) \subseteq \mathcal{L}(\Gamma/\equiv)$

- look for two different bracketed sentential forms in $\mathcal{L}(\Gamma/\equiv)$

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

- a nonterminal transition represents exactly its derived context-free language

Nonterminal Transitions

- $\mathcal{SF}(\mathcal{G}_b) \subseteq \mathcal{L}(\Gamma/\equiv)$
- look for two different bracketed sentential forms in $\mathcal{L}(\Gamma/\equiv)$

  $d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

  $d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

- a nonterminal transition represents exactly its derived context-free language

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8\ d_{14}$ *vid* r_{14} => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8\ d_{14}$ *vid* r_{14} => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

epsilon: mae

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ d_{14} *vid* r_{14} => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ d_{14} *vid* r_{14} => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

epsilon: mae

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8\ d_{14}\ \textit{vid}\ r_{14}$ => d_5 **case** $\langle \textit{exp} \rangle$ **of** $d_7\ \langle \textit{match} \rangle$ '|' $\langle \textit{mrules} \rangle r_7 r_5 r_8 r_6$

$d_7 d_6 d_8\ d_{14}\ \textit{vid}\ r_{14}$ => d_5 **case** $\langle \textit{exp} \rangle$ **of** $\langle \textit{match} \rangle r_5 r_8 r_6$ '|' $\langle \textit{mrules} \rangle\ r_7$

epsilon: mae

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8\ d_{14}$ *vid* r_{14} => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8\ d_{14}$ *vid* r_{14} => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

shift: mas

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8\ d_{14}$ *vid* r_{14} => d_5 **case** $\langle exp\rangle$ **of** d_7 $\langle match\rangle$ '|' $\langle mrules\rangle r_7 r_5 r_8 r_6$

$d_7 d_6 d_8\ d_{14}$ *vid* r_{14} => d_5 **case** $\langle exp\rangle$ **of** $\langle match\rangle r_5 r_8 r_6$ '|' $\langle mrules\rangle$ r_7

nothing!

Mutual Accessibility Relations

- between pairs of states of $\Gamma/{\equiv}$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

shift: mas

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

conflict: mac

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

conflict: mac

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

conflict: mac

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

shift: mas

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ $\langle pat \rangle$ => d_5 **case** $\langle exp \rangle$ **of** d_7 $\langle match \rangle$ '|' $\langle mrules \rangle r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ $\langle pat \rangle$ => d_5 **case** $\langle exp \rangle$ **of** $\langle match \rangle r_5 r_8 r_6$ '|' $\langle mrules \rangle$ r_7

reduce: mar

Mutual Accessibility Relations

- between pairs of states of $\Gamma/\equiv$, (q_1, q_2)
- synchronized left-to-right walks from an initial pair (q_s, q_s)

$d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** d_7 ⟨*match*⟩ '|' ⟨*mrules*⟩$r_7 r_5 r_8 r_6$

$d_7 d_6 d_8$ ⟨*pat*⟩ => d_5 **case** ⟨*exp*⟩ **of** ⟨*match*⟩$r_5 r_8 r_6$ '|' ⟨*mrules*⟩ r_7

conflict: mac

NU(≡)

- $\mathsf{ma} = \mathsf{mas} \cup \mathsf{mae} \cup \mathsf{mac} \cup \mathsf{mar}$
- $\mathcal{G}$ is noncanonically unambiguous if there does not exist a relation $(q_s, q_s)\ \mathsf{ma}^*\ (q_f, q_f)$ that uses mac at some step
- Computation in $\mathcal{O}(|\Gamma/\equiv|^2)$ in space

Comparisons

- Regular Unambiguity RU($\equiv$)
- Bounded-length detection schemes
- LR(k) and LR-Regular (LR(Π))
- Horizontal and vertical ambiguity (HVRU($\equiv$))

Bounded-length detection

[Gorn, 1963, Cheung and Uzgalis, 1995, Schröer, 2001, Jampana, 2005]

- generate sentences
- not conservative
- prefix_m prevents from false positives in sentences of length $< m$
- need to generate a^{2^n+1} to find $\mathcal{G}_4^n$ ambiguous, but $\mathcal{G}_4^n \notin \mathrm{NU}(\text{item}_0)$

$$S \rightarrow A \mid B_n a,\ A \rightarrow Aaa \mid a,\ B_1 \rightarrow aa,\ B_2 \rightarrow B_1 B_1,\ \ldots,\ B_n \rightarrow B_{n-1} B_{n-1} \qquad (\mathcal{G}_4^n)$$

LR(k) and LR-Regular

[Knuth, 1965, Hunt III et al., 1975, Čulik and Cohen, 1973, Heilbrunner, 1983]

- conservative tests
- define item_Π s.t. $\mathrm{LR}(\Pi) \subset \mathrm{NU}(\mathsf{item}_\Pi)$
- need a LR(2^n) test to prove $\mathcal{G}_3^n$ unambiguous, but $\mathcal{G}_3^n \in \mathrm{NU}(\mathsf{item}_0)$

$$S \rightarrow A \mid B_n,\ A \rightarrow Aaa \mid a,\ B_1 \rightarrow aa,\ B_2 \rightarrow B_1 B_1,\ \ldots,\ B_n \rightarrow B_{n-1} B_{n-1} \quad (\mathcal{G}_3^n)$$

Implementation

- For the whole SML grammar:
  - conflicts in the LALR(1) parser

    `sml.y: conflicts: 223 shift/reduce, 35 reduce/reduce`
  - Our tool:

    `89 potential ambiguities with LR(1) precision detected`

- For the SML grammar fragment:

```
2 potential ambiguities with LR(0) precision detected:
   (match -> mrule . , match -> match . '|' mrule )
   (match -> match . '|' mrule , match -> match '|' mrule . )
```

- NU($item_1$) correctly identifies 87% of our unambiguous grammars—73% of the non-LALR(1) ones

Summary

- conservative ambiguity detection
- provably better than several other techniques
- also experimentally better

Conclusion

- Main issues in parser development:
  - nondeterminism
  - ambiguity in particular
- Deterministic parsers for larger classes of grammars
- Ambiguity detection algorithm

Directions for Future Work

- Linear time parsing for NU(≡) grammars?
- Improved implementation
- Noncanonical languages
- Regular approximations

Thanks!

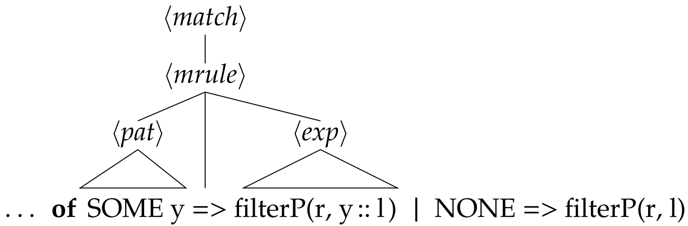
Our Issue
Shift/Reduce Conflict

GNU Bison

```
state 20
    6 exp: "case" exp "of" match .
    8 match: match . '|' mrule

    '|'  shift, and go to state 24
    '|'       [reduce using rule 6 (exp)]
```

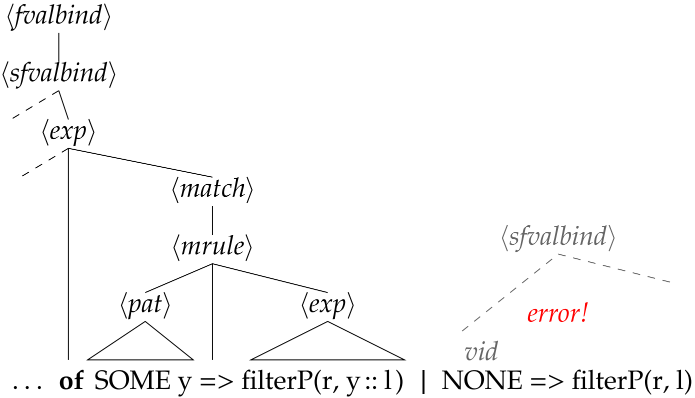
Our Issue
Shift/Reduce Conflict

Which action to choose?



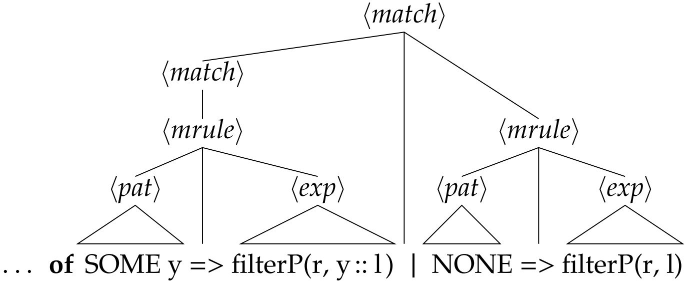
Our Issue
Shift/Reduce Conflict

Which action to choose? Reduce?



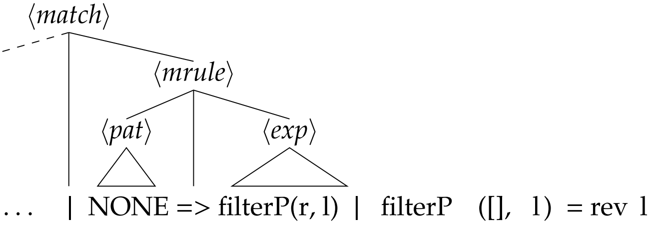
Our Issue
Shift/Reduce Conflict

Which action to choose? Shift?



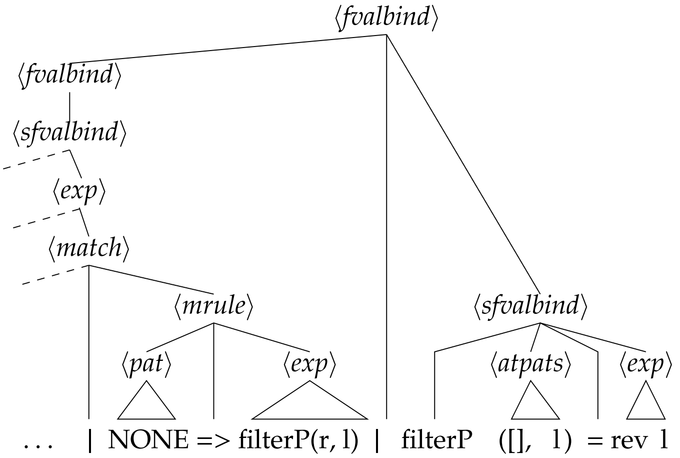
Our Issue
Shift/Reduce Conflict

Which action to choose?



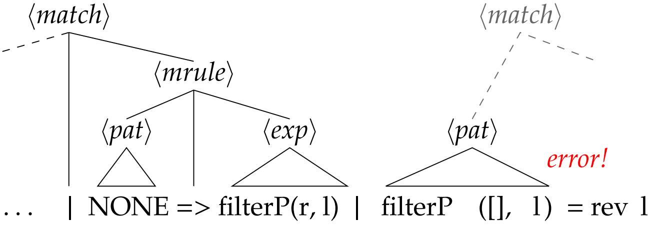
Our Issue
Shift/Reduce Conflict

Which action to choose? Reduce?

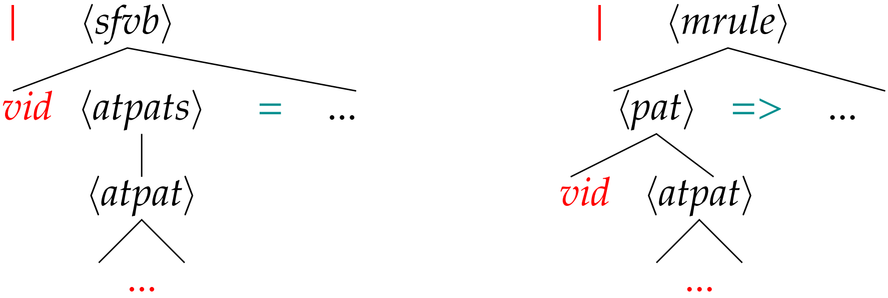


Our Issue
Shift/Reduce Conflict

Which action to choose? Shift?



Unbounded Lookahead



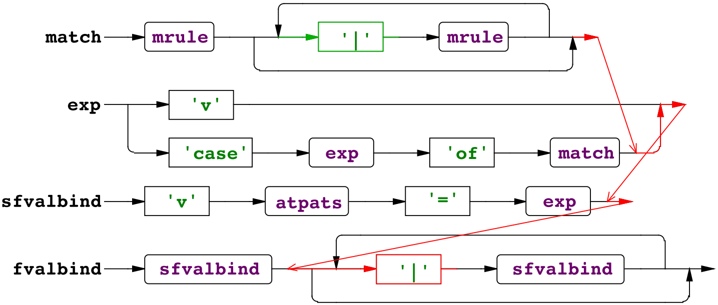
Limitations

Ambiguity Report

- grambiguity [Brabrand et al., 2007]

```
*** horizontal ambiguity at E[plus]: Exp <--> '+' Exp
    ambiguous string: "x+x+x"
```

- ANTLRWorks [Parr, 2007]



Other Limitations

- memory requirements: a solution could be a NLALR test
- dynamic disambiguation: inverse problem, some means to deciding equivalence needed

H. J. S. Basten. Ambiguity detection methods for context-free grammars. Master's thesis, Centrum voor Wiskunde en Informatica, Universiteit van Amsterdam, Aug. 2007.

P. Boullier. Supertagging: A non-statistical parsing-based approach. In IWPT'03, pages 55–65, 2003. URL ftp://ftp.inria.fr/INRIA/Projects/Atoll/Pierre.Boullier/supertaggeur_final.pdf.

C. Brabrand, R. Giegerich, and A. Møller. Analyzing ambiguity of context-free grammars. In J. Holub and J. Žďárek, editors, CIAA'07, 2007. URL http://www.brics.dk/~brabrand/grambiguity/. To appear in Lecture Notes in Computer Science.

D. G. Cantor. On the ambiguity problem of Backus