Context Free Grammars and Induction

Second Inductive Theorem Proving Festival, 2015

Dan Rosén

Chalmers University of Technology

Context Free Grammars and Induction

- Unambiguity proving of a CFG is an induction problem
- Recursion only by simple structural induction
- Can require very complicated lemmas

Expression grammar

$$E ::= (E + E) \mid x \mid y$$

Expression grammar

$$E ::= (E + E) \mid x \mid y$$

$$\begin{aligned}
&\textbf{data}\ E = Plus\ E\ E \mid EX \mid EY \\
&\textbf{data}\ Token = C \mid D \mid P \mid X \mid Y
\end{aligned}$$

$$\begin{aligned}
&show\ :: E \rightarrow [\,Token]\\
&show\ (Plus\ a\ b) = [\,C] +\!\!+\ show\ a +\!\!+ [\,P] +\!\!+\ show\ b +\!\!+ [\,D]\\
&show\ EX \qquad\quad\ = [\,X]\\
&show\ EY \qquad\quad\ = [\,Y]
\end{aligned}$$

Expression grammar

$$E ::= (E + E) \mid x \mid y$$

$$
\begin{aligned}
&\textbf{data } E = \mathit{Plus}\ E\ E \mid EX \mid EY \\
&\textbf{data } \mathit{Token} = C \mid D \mid P \mid X \mid Y
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{show}\ :: E \to [\mathit{Token}] \\
&\mathit{show}\ (\mathit{Plus}\ a\ b) = [C] \mathbin{+\!\!+} \mathit{show}\ a \mathbin{+\!\!+} [P] \mathbin{+\!\!+} \mathit{show}\ b \mathbin{+\!\!+} [D] \\
&\mathit{show}\ EX \qquad\qquad = [X] \\
&\mathit{show}\ EY \qquad\qquad = [Y]
\end{aligned}
$$

$$
\begin{aligned}
&\forall\ s\ t\ .\ \mathit{show}\ s = \mathit{show}\ t \implies s = t \\
&\forall\ s\ t\ .\ s \neq t \implies \mathit{show}\ s \neq \mathit{show}\ t
\end{aligned}
$$

Expression unambiguity, step case

$$\begin{aligned}
show\ (Plus\ a\ b) &= [\,C\,] +\!\!+\ show\ a +\!\!+ [\,P\,] +\!\!+\ show\ b +\!\!+ [\,D\,] \\
show\ EX &= [\,X\,] \\
show\ EY &= [\,Y\,]
\end{aligned}$$

$$\forall\ s\ t\ .\ show\ s = show\ t \Longrightarrow s = t$$

$$\begin{aligned}
assumption: &\ show\ (Plus\ s_1\ s_2) = show\ (Plus\ t_1\ t_2) \\
goal: &\ Plus\ s_1\ s_2 = Plus\ t_1\ t_2
\end{aligned}$$

Expression unambiguity, step case

$$\begin{aligned}
show\ (Plus\ a\ b) &= [C] ++ show\ a ++ [P] ++ show\ b ++ [D] \\
show\ EX &= [X] \\
show\ EY &= [Y]
\end{aligned}$$

$$\forall\ s\ t\ .\ show\ s = show\ t \Longrightarrow s = t$$

$$\begin{aligned}
assumption: &\ show\ (Plus\ s_1\ s_2) = show\ (Plus\ t_1\ t_2) \\
goal: &\ Plus\ s_1\ s_2 = Plus\ t_1\ t_2
\end{aligned}$$

$$show\ s_1 ++ [P] ++ show\ s_2 = show\ t_1 ++ [P] ++ show\ t_2$$

Expression unambiguity, step case

$$
\begin{aligned}
&\mathit{show}\ (\mathit{Plus}\ a\ b) &&= [C] +\!\!+\ \mathit{show}\ a +\!\!+ [P] +\!\!+\ \mathit{show}\ b +\!\!+ [D] \\
&\mathit{show}\ \mathit{EX} &&= [X] \\
&\mathit{show}\ \mathit{EY} &&= [Y]
\end{aligned}
$$

$$\forall\ s\ t\ .\ \mathit{show}\ s = \mathit{show}\ t \Longrightarrow s = t$$

$$
\begin{aligned}
&\mathit{assumption}: && \mathit{show}\ (\mathit{Plus}\ s_1\ s_2) = \mathit{show}\ (\mathit{Plus}\ t_1\ t_2) \\
&\mathit{goal}: && \mathit{Plus}\ s_1\ s_2 = \mathit{Plus}\ t_1\ t_2
\end{aligned}
$$

$$\mathit{show}\ s_1 +\!\!+ [P] +\!\!+\ \mathit{show}\ s_2 = \mathit{show}\ t_1 +\!\!+ [P] +\!\!+\ \mathit{show}\ t_2$$

$$\forall\ a\ b\ u\ v\ .\ \mathit{show}\ a +\!\!+\ u = \mathit{show}\ b +\!\!+\ v \Longrightarrow a = b \wedge u = v$$

Expression unambiguity, lemma

$$\forall\ a\ b\ u\ v\ .\ show\ a \mathbin{+\!\!+} u = show\ b \mathbin{+\!\!+} v \Longrightarrow a = b \land u = v$$

Expression unambiguity, lemma

$$\forall\ a\ b\ u\ v\ .\ show\ a +\!\!+\ u = show\ b +\!\!+\ v \Longrightarrow a = b \wedge u = v$$

$$\begin{aligned}
IH_1 &: \forall\ u'\ v'\ .\ show\ a_1 +\!\!+\ u' = show\ b_1 +\!\!+\ v' \Longrightarrow a_1 = b_1 \wedge u' = v' \\
assumption &: show\ (Plus\ a_1\ a_2) +\!\!+\ u = show\ (Plus\ b_1\ b_2) +\!\!+\ v \\
goal &: \quad Plus\ a_1\ b_1 = Plus\ a_2\ b_2 \wedge u = v
\end{aligned}$$

Expression unambiguity, lemma

$$\forall\ a\ b\ u\ v\ .\ show\ a \mathbin{+\!\!+} u = show\ b \mathbin{+\!\!+} v \Longrightarrow a = b \wedge u = v$$

$$\begin{array}{ll} IH_1 : & \forall\ u'\ v'\ .\ show\ a_1 \mathbin{+\!\!+} u' = show\ b_1 \mathbin{+\!\!+} v' \Longrightarrow a_1 = b_1 \wedge u' = v' \\ assumption : & show\ (Plus\ a_1\ a_2) \mathbin{+\!\!+} u = show\ (Plus\ b_1\ b_2) \mathbin{+\!\!+} v \\ goal : & Plus\ a_1\ b_1 = Plus\ a_2\ b_2 \wedge u = v \end{array}$$

$$\begin{array}{rl} & [C] \mathbin{+\!\!+} show\ a_1 \mathbin{+\!\!+} [P] \mathbin{+\!\!+} show\ a_2 \mathbin{+\!\!+} [D] \mathbin{+\!\!+} u \\ = & [C] \mathbin{+\!\!+} show\ b_1 \mathbin{+\!\!+} [P] \mathbin{+\!\!+} show\ b_2 \mathbin{+\!\!+} [D] \mathbin{+\!\!+} v \end{array}$$

Expression unambiguity, lemma

$$\forall\ a\ b\ u\ v\ .\ show\ a +\!\!+\ u = show\ b +\!\!+\ v \Longrightarrow a = b \wedge u = v$$

$$\begin{aligned}
IH_1 &: \forall\ u'\ v'\ .\ show\ a_1 +\!\!+\ u' = show\ b_1 +\!\!+\ v' \Longrightarrow a_1 = b_1 \wedge u' = v' \\
assumption &: show\ (Plus\ a_1\ a_2) +\!\!+\ u = show\ (Plus\ b_1\ b_2) +\!\!+\ v \\
goal &: \qquad Plus\ a_1\ b_1 = Plus\ a_2\ b_2 \wedge u = v
\end{aligned}$$

$$\begin{aligned}
& [C] +\!\!+\ show\ a_1 +\!\!+ [P] +\!\!+\ show\ a_2 +\!\!+ [D] +\!\!+\ u \\
= & [C] +\!\!+\ show\ b_1 +\!\!+ [P] +\!\!+\ show\ b_2 +\!\!+ [D] +\!\!+\ v
\end{aligned}$$

$$show\ a_2 +\!\!+ [D] +\!\!+\ u = show\ b_2 +\!\!+ [D] +\!\!+\ v$$

A more difficult example

$$S ::= A \mid B$$
$$A ::= x\ A\ y \quad \mid z$$
$$B ::= x\ B\ y\ y \mid z$$
$$\{x^n\ z\ y^n \mid n > 0\} \cup \{x^n\ z\ y^{2n} \mid n > 0\}$$

Not LR(k) for any k

Injectivity digression

easy:

$$\forall\ xs\ ys\ zs\ .\ xs +\!\!+\ ys = xs +\!\!+\ zs \Longrightarrow ys = zs$$

Injectivity digression

"hard":

$$\forall\ xs\ ys\ zs\ .\ xs +\!\!+\ zs = ys +\!\!+\ zs \Longrightarrow xs = ys$$

Injectivity digression

"hard":

$$\forall\ xs\ ys\ zs\ .\ xs +\!\!+ zs = ys +\!\!+ zs \Longrightarrow xs = ys$$

$$IH : \forall\ xs\ ys\ .\ xs +\!\!+ cs = ys +\!\!+ cs \Longrightarrow xs = ys$$

$$assume : as +\!\!+ c : cs = bs +\!\!+ c : cs$$

$$show : \quad as +\!\!+ bs$$

Injectivity digression

"hard":

$$\forall\ xs\ ys\ zs\ .\ xs +\!\!+\ zs = ys +\!\!+\ zs \Longrightarrow xs = ys$$

$$IH : \forall\ xs\ ys\ .\ xs +\!\!+\ cs = ys +\!\!+\ cs \Longrightarrow xs = ys$$

$$assume : as +\!\!+\ c : cs = bs +\!\!+\ c : cs$$

$$show : \quad as +\!\!+\ bs$$

$$\begin{aligned} assumption &: (as +\!\!+ [c]) +\!\!+ cs = (bs +\!\!+ [c]) +\!\!+ cs \\ by\ IH &: \quad as +\!\!+ [c] \qquad\qquad = bs +\!\!+ [c] \end{aligned}$$

Injectivity digression

"hard":

$$\forall\ xs\ ys\ zs\ .\ xs +\!\!+\ zs = ys +\!\!+\ zs \Longrightarrow xs = ys$$

$$IH : \forall\ xs\ ys\ .\ xs +\!\!+\ cs = ys +\!\!+\ cs \Longrightarrow xs = ys$$

$$assume : as +\!\!+\ c : cs = bs +\!\!+\ c : cs$$

$$show : \quad as +\!\!+\ bs$$

$$\begin{array}{ll} assumption : (as +\!\!+ [c]) +\!\!+ cs & = (bs +\!\!+ [c]) +\!\!+ cs \\ by\ IH : \quad as +\!\!+ [c] & = bs +\!\!+ [c] \end{array}$$

$$\forall\ xs\ ys\ z\ .\ xs +\!\!+ [z] = ys +\!\!+ [z] \Longrightarrow xs = ys$$

Injectivity lemma

$$\begin{aligned} &assume : (a : as) +\!\!+ [c] = (b : bs) +\!\!+ [c] \\ &show : \quad a : as = b : bs \end{aligned}$$

$$IH : as +\!\!+ [c] = bs +\!\!+ [c] \Longrightarrow as = bs$$

$$\begin{aligned} &(a : as) +\!\!+ [c] = (b : bs) +\!\!+ [c] \\ &a : (as +\!\!+ [c]) = b : (bs +\!\!+ [c]) \\ &a = b \wedge as +\!\!+ [c] = bs +\!\!+ [c] \\ &a = b \wedge as = bs \end{aligned}$$

Required Lemmas (besides injectivity and trivialities)

$$S ::= A \mid B$$
$$A ::= x\ A\ y \quad \mid z$$
$$B ::= x\ A\ y\ y \mid z$$
$$\{x^n\ z\ y^n \mid n > 0\} \cup \{x^n\ z\ y^{2n} \mid n > 0\}$$

$$count\ x\ (xs +\!\!+\ ys) = count\ x\ xs + count\ x\ ys$$
$$count\ x\ A = count\ y\ A$$
$$double\ (count\ x\ B) = count\ y\ B$$

$$count\ x\ A > 0$$
$$count\ y\ A > 0$$
$$count\ x\ B > 0$$
$$count\ y\ B > 0$$

$$double\ x \neq x \text{ for } x > 0, using : x + y = x + z \Rightarrow y = z$$
$$double\ x = x + x$$

Successful run

```
Proved:
    count Z (showB x) = S Zero
    count Z (showA x) = S Zero
    count Y (showA x) = count X (showA x)
    double (count X (showB x)) = count Y (showB x)
    nonZero (count x (showB y)) = True
    nonZero (count x (showA y)) = True
    count x xs + count x ys = count x (xs ++ ys)
    double (count x xs) = count x (xs ++ xs)
    count x (xs ++ ys) = count x (ys ++ xs)
    (xs ++ ys) ++ zs = xs ++ (ys ++ zs)
    (x + y) + z = x + (y + z)
    double x = x + x
    x + y = y + x
    xs ++ [] = xs
    x + Zero = x
    unambigS {- showS u == showS v => u == v -}
    unambigB {- showB u == showB v => u == v -}
    plusInjL {- y+x == z+x => y == z -}
    injR {- v++u == w++u => v == w -}
    unambigA {- showA u == showA v => u == v -}
    plusInjR {- x+y == x+z => y == z -}
    injL {- u++v == u++w => v == w -}
    inj1 {- v++(x:[]) == w++(x:[]) => v == w -}


real    1m41.581s
user    3m1.933s
sys     0m3.747s
```

Some other (simple!) grammars

$$
\begin{aligned}
&\textit{Balanced nonparentheses}:\\
&B ::= A\ A\\
&A ::= x\ A\ x\\
&\qquad \mid y
\end{aligned}
$$

$$
\begin{aligned}
&\textit{Dyck language}:\\
&D ::= (D)\ D\\
&\qquad \mid\ (D)\\
&\qquad \mid\ ()
\end{aligned}
$$

$$
\begin{aligned}
&\textit{Palindromes}:\\
&P ::= a\ P\ a\\
&\qquad \mid\ b\ P\ b\\
&\qquad \mid\ a\\
&\qquad \mid\ b\\
&\qquad \mid\ \epsilon
\end{aligned}
$$

Post Correspondence Problem

a_1	a_2	a_3	$\ldots$	a_n
b_1	b_2	b_3	$\ldots$	b_n

Post Correspondence Problem

$$| \; a_1 \; | \; a_2 \; | \; a_3 \; | \; \ldots \; | \; a_n \; |$$
$$| \; b_1 \; | \; b_2 \; | \; b_3 \; | \; \ldots \; | \; b_n \; |$$

$$S ::= A \mid B$$
$$A ::= x_0 \mid a_1 \; A \; x_1 \mid a_2 \; A \; x_2 \mid \ldots \mid a_n \; A \; x_n$$
$$B ::= x_0 \mid b_1 \; B \; x_1 \mid b_2 \; B \; x_2 \mid \ldots \mid b_n \; B \; x_n$$
$$showS \; (A \; a) = showA \; a$$
$$showS \; (B \; b) = showB \; b$$

$$showA \; (A_1 \; a) = a_1 +\!\!+ \; showA \; a +\!\!+ [X_1]$$
$$\ldots$$
$$showA \; (A_n \; a) = a_n +\!\!+ \; showA \; a +\!\!+ [X_n]$$
$$\ldots$$
$$showB \; (B_n \; b) = b_n +\!\!+ \; showB \; b +\!\!+ [X_n]$$

Post Correspondence Problem

a_1	a_2	a_3	...	a_n
b_1	b_2	b_3	...	b_n

$$\mathbf{data}\ X = X_1 \mid X_2 \mid \ldots \mid X_n$$
$$upper :: X \to [Tok]$$
$$lower :: X \to [Tok]$$
$$\forall\ (xs :: [X])\ .\ concatMap\ upper\ xs \neq concatMap\ lower\ xs \vee null\ xs$$

$$concatMap :: (a \to [b]) \to [a] \to [b]$$

Conclusions

- Interesting class of problems
- Very simple programs, very difficult proofs
- How can we synthesise those functions for lemmas?