Defining Parser Combinator using Attribute Grammars

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Parser Combinators

- A parser combinator library is like an axiomatic system.
- Simple parsers are combined to derive complex parsers.
  1. Domain Specific Language.
  2. Strong relation between *grammar* and *parser*.
  3. Highly **compositional**.
  4. **type-safe** semantic actions.

Language of all (binarised) BNF Grammars

$$\begin{aligned} \mathit{Cexpr} &::= \epsilon \\ &\mid \ \textbf{char}\ \mathit{Char} \\ &\mid \ \mathit{Cexpr}\ \langle|\rangle\ \mathit{Cexpr} \\ &\mid \ \mathit{Cexpr}\ \langle*\rangle\ \mathit{Cexpr} \\ &\mid \ \mathit{ID} \end{aligned}$$

$$\begin{aligned} \mathit{Doc} &::= \mathit{Rule}^{*} \\ \mathit{Rule} &::= \mathit{ID}\ \texttt{'='}\ \mathit{Cexpr} \\ \mathit{Char} &::= \texttt{'a'} \\ &\mid \ \texttt{'b'} \\ &\mid \ \ldots \end{aligned}$$

Abstract Syntax (Recog.)

$$\begin{aligned} \mathit{Cexpr} &::= \epsilon \\ &\mid \ \textbf{char}\ \mathit{Char} \\ &\mid \ \mathit{Cexpr}\ \langle|\rangle\ \mathit{Cexpr} \\ &\mid \ \mathit{Cexpr}\ \langle*\rangle\ \mathit{Cexpr} \end{aligned}$$

Attributes

attr *Cexpr*
  inh *str* :: *String*
  inh *inp* :: *Int*
  syn *out* :: [*Int*]

Abstract Syntax (Parser)

$$\begin{aligned} &\textbf{data}\ \mathit{Cexpr}\ x\ \textbf{where} \\ &\textbf{satisfy} :: a \rightarrow \mathit{Cexpr}\ a \\ &\textbf{char} :: \mathit{Char} \rightarrow \mathit{Cexpr}\ \mathit{Char} \\ &\langle*\rangle :: \mathit{Cexpr}\ (b \rightarrow a) \rightarrow \mathit{Cexpr}\ b \rightarrow \mathit{Cexpr}\ a \\ &\langle|\rangle :: \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ a \end{aligned}$$

Derived Combinators

$\langle\$\rangle :: (b \rightarrow a) \rightarrow \mathit{Cexpr}\ b \rightarrow \mathit{Cexpr}\ a$
$f\ \langle\$\rangle\ p = \textbf{satisfy}\ f\ \langle*\rangle\ p$

$\langle* :: \mathit{Cexpr}\ b \rightarrow \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ b$
$l\ \langle*\ r = \mathit{const}\ \langle\$\rangle\ l\ \langle*\rangle\ r$

$\langle\$ :: b \rightarrow \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ b$
$f\ \langle\$\ p = \mathit{const}\ f\ \langle\$\rangle\ p$

Extended BNF

$\mathit{optional} :: \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ (\mathit{Maybe}\ a)$
$\mathit{optional}\ p = \mathit{Just}\ \langle\$\rangle\ p\ \langle|\rangle\ \textbf{satisfy}\ \mathit{Nothing}$

$\mathit{many} :: \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ [a]$
$\mathit{many}\ p = (:)\ \langle\$\rangle\ p\ \langle*\rangle\ \mathit{many}\ p\ \langle|\rangle\ \textbf{satisfy}\ [\,]$

$\mathit{some} :: \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ [a]$
$\mathit{some}\ p = (:)\ \langle\$\rangle\ p\ \langle*\rangle\ \mathit{many}\ p$

$\mathit{sepBy} :: \mathit{Cexpr}\ a \rightarrow \mathit{Cexpr}\ b \rightarrow \mathit{Cexpr}\ [a]$
$\mathit{sepBy}\ p\ \mathit{sep} = (:)\ \langle\$\rangle\ p\ \langle*\rangle\ \mathit{many}\ (\mathit{sep}\ *\rangle\ p)$

Attribute Grammars

- AGs describe programming language semantics.
- High-level programming in terms of trees and attributes.
  1. Abstract Syntax + Attributes + Semantic Definitions.
  2. **concern-separation**: isolated code-fragments.
  3. **aspect-oriented**: isolated computations.
  4. **modular**: implicit propagation of information.

Parser Combinators as Embedded DSL

- From the Glasgow Haskell Compiler we obtain:
  1. The parser.
  2. Name-binding.
  3. Char type.
  4. Type system for providing type safety of combinator expressions.

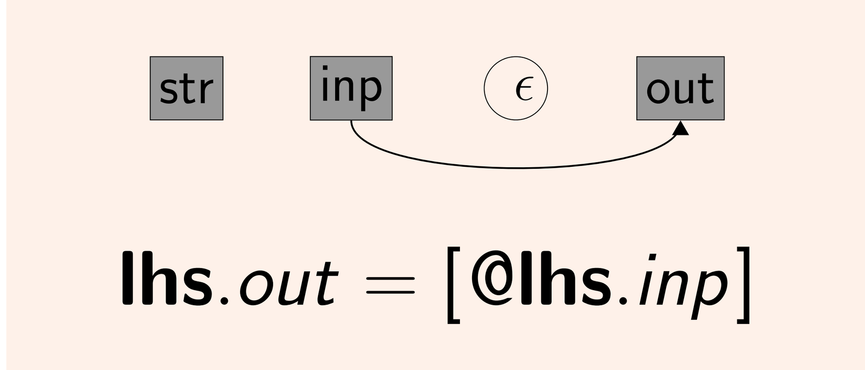
Count occurrences of "ab" in a string of "ab"s?

$\mathit{pX} = (1+)\ \langle\$\ \textbf{char}\ \texttt{'a'}\ \langle*\ \textbf{char}\ \texttt{'b'}\ \langle*\rangle\ \mathit{pX}\ \langle|\rangle\ \textbf{satisfy}\ 0$

Semantic evaluation as recogniser

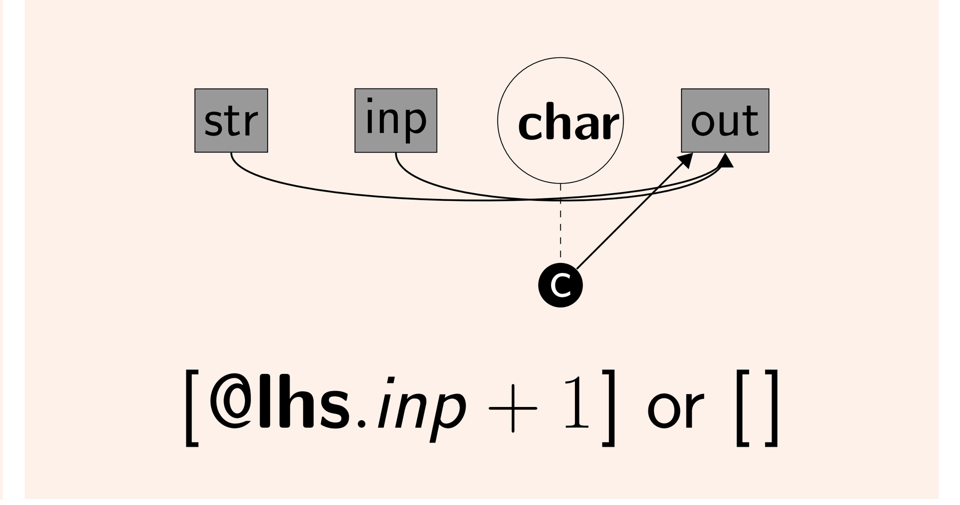
- Every combinator expressions computes the substrings of *str* (starting at *inp*) it recognises.
- For each constructor we define part of the computation.
- The pieces are combined to complete the computation.

Semantics of epsilon



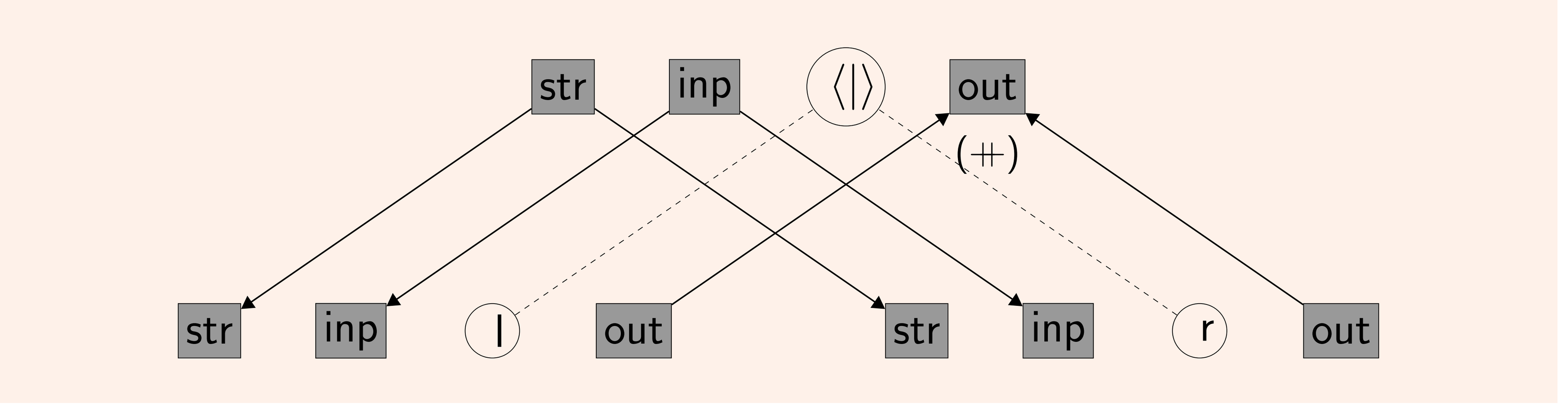
$\textbf{lhs}.\mathit{out} = [\texttt{@}\textbf{lhs}.\mathit{inp}]$

Semantics of char

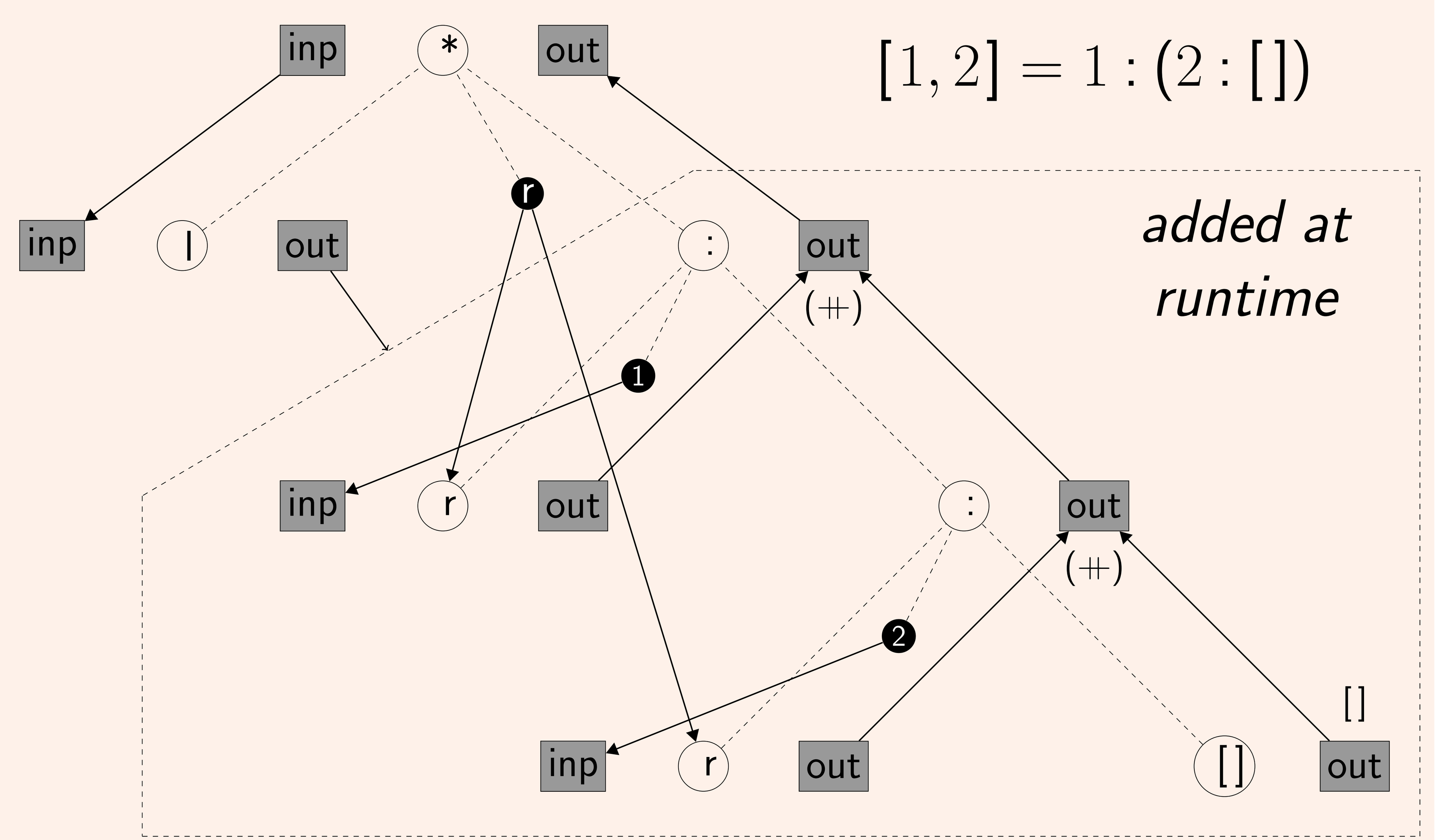


$[\texttt{@}\textbf{lhs}.\mathit{inp} + 1]$ or $[\,]$

Semantics of $\langle|\rangle$



Semantics of $\langle*\rangle$



L. Thomas van Binsbergen
Royal Holloway, University of London, ltvanbinsbergen@acm.org