–contents of Input-Output Grammars–

–requires–

Context-free Grammars
Regular Expressions
Regular Expression Grammars

Input-Output Grammars

Definition of Input-Output Grammar

One last form of grammar needs to be defined. The new grammar, named an Input/Output grammar (IOG), has a new set of symbols for output.[1] Extending the formalisms for context-free grammars, we have:

$$\mathcal{G} \stackrel{\text{def}}{=} \langle V_I, V_O, V_N, V_G, \Pi\rangle \tag{1}$$

consisting of a set of input symbols, a set of output symbols, a set of phrase names, a set of goals, and a set of reductions. It satisfies the following constraints:

$$\begin{aligned}
V &\stackrel{\text{def}}{=} V_I \cup V_O \cup V_N \\
V_I \cap V_O &\stackrel{\text{def}}{=} \{\} \\
V_I \cap V_N &\stackrel{\text{def}}{=} \{\} \\
V_O \cap V_N &\stackrel{\text{def}}{=} \{\} \\
V_G &\stackrel{\text{def}}{\subseteq} V_N \\
\Pi &\stackrel{\text{def}}{\subseteq} V_N \times V^*
\end{aligned}$$

There is no abstract way to distinguish V_I and V_O. In fact in some uses they will be interchanged. The CFG is just an IOG with $V_O = \{\}$.

[1]The output symbols are analogous to the YACC actions.

The Proposition Grammar (again)

$$
\begin{array}{lcl}
\mathit{Proposition} & \leftarrow & \mathit{Disjunction}\ 0 \\
\mathit{Disjunction} & \leftarrow & \mathit{Disjunction} \vee \mathit{Conjunction}\ 1 \\
\mathit{Disjunction} & \leftarrow & \mathit{Conjunction}\ 2 \\
\mathit{Conjunction} & \leftarrow & \mathit{Conjunction} \wedge \mathit{Negation}\ 3 \\
\mathit{Conjunction} & \leftarrow & \mathit{Negation}\ 4 \\
\mathit{Negation} & \leftarrow & \neg\ \mathit{Boolean}\ 5 \\
\mathit{Negation} & \leftarrow & \mathit{Boolean}\ 6 \\
\mathit{Boolean} & \leftarrow & \mathrm{t}\ 7 \\
\mathit{Boolean} & \leftarrow & \mathrm{f}\ 8 \\
\mathit{Boolean} & \leftarrow & (\ \mathit{Disjunction}\ )\ 9
\end{array}
$$

Table 0: Proposition Input-Output Grammar

Interpretation of Input-Output Grammar

Left-to-right order is required by the following alternative definition; in this case it will not only consume the input τ, but also construct the output ρ. The predicate $\mathcal{C}$ is used, as in context-free grammars, to prove a string is in the language defined by the grammar. It is sufficient for the purposes here to restrict the IOG to rules with the output symbol at the end of each rule.

$$
\begin{array}{rcl}
G \in V_G & \Rightarrow & \mathcal{C}(G, \lambda, \lambda) \\
B{\leftarrow}\beta\mathrm{r} \in \Pi \wedge \mathcal{C}(\sigma B, \tau, \rho) & \Rightarrow & \mathcal{C}(\sigma\beta, \tau, \mathrm{r}\rho) \\
\mathrm{a} \in V_I \wedge \mathcal{C}(\sigma \mathrm{a}, \tau, \rho) & \Rightarrow & \mathcal{C}(\sigma, \mathrm{a}\tau, \rho) \\
\mathcal{C}(\lambda, \tau, \rho) & \Rightarrow & G \overset{\rho}{\leftarrow} \tau
\end{array}
$$

Table 1: Left-to-right Reduction Application

The first implication defines the goal. The second implication applies a reduction on the top of the parse stack. The third implication shifts a symbol from the input text and pushes it on the top of the stack. The *parse* is the third argument to $\mathcal{C}$.

Application of Input-Output Grammar

Repeating the parse of f $\vee$ t $\wedge$ $\neg$f:

$$
\begin{array}{l}
\mathcal{C}(Proposition, \lambda, \lambda) \\
\mathcal{C}(Disjunction, \lambda, 0) \\
\mathcal{C}(Disjunction \vee Conjunction, \lambda, 10) \\
\mathcal{C}(Disjunction \vee Conjunction \wedge Negation, \lambda, 310) \\
\mathcal{C}(Disjunction \vee Conjunction \wedge \neg Boolean, \lambda, 5310) \\
\mathcal{C}(Disjunction \vee Conjunction \wedge \neg \mathrm{f}, \lambda, 85310) \\
\mathcal{C}(Disjunction \vee Conjunction \wedge \neg, \mathrm{f}, 85310)
\end{array}
$$

$\mathcal{C}(Disjunction \vee Conjunction \wedge, \neg \mathrm{f}, 85310)$
$\mathcal{C}(Disjunction \vee Conjunction, \wedge \neg \mathrm{f}, 85310)$
$\mathcal{C}(Disjunction \vee Negation, \wedge \neg \mathrm{f}, 485310)$
$\mathcal{C}(Disjunction \vee Boolean, \wedge \neg \mathrm{f}, 6485310)$
$\mathcal{C}(Disjunction \vee \mathrm{t}, \wedge \neg \mathrm{f}, 76485310)$
$\mathcal{C}(Disjunction \vee, \mathrm{t} \wedge \neg \mathrm{f}, 76485310)$
$\mathcal{C}(Disjunction, \vee \mathrm{t} \wedge \neg \mathrm{f}, 76485310)$
$\mathcal{C}(Conjunction, \vee \mathrm{t} \wedge \neg \mathrm{f}, 276485310)$
$\mathcal{C}(Negation, \vee \mathrm{t} \wedge \neg \mathrm{f}, 4276485310)$
$\mathcal{C}(Boolean, \vee \mathrm{t} \wedge \neg \mathrm{f}, 64276485310)$
$\mathcal{C}(\mathrm{f}, \vee \mathrm{t} \wedge \neg \mathrm{f}, 864276485310)$
$\mathcal{C}(\lambda, \mathrm{f} \vee \mathrm{t} \wedge \neg \mathrm{f}, 864276485310)$

Table 2: Proof of and parse for '$\mathrm{f} \vee \mathrm{t} \wedge \neg \mathrm{f}$' $\in \mathcal{L}(\mathcal{G})$