Lecture 13: Parsing & Logic

CSCI 81
Spring, 2015

Kim Bruce

Arithmetic grammar

```
  <exp> ::= <exp> <addop> <term>
          | <term>
 <term> ::= <term> <mulop> <factor>
          | <factor>
<factor> ::= ( <exp> )
          | NUM
          | ID
 <addop> ::= + | -
 <mulop> ::= * | /
```

*Look at parse tree & abstract syntax tree for 2 * 3 + 7*

Recursive Descent Parser

Base recognizer (*ignore building tree*) on productions:

```
<exp> ::= <exp> <addop> <term>

addop (fst:rest) = if fst=='+' or fst=='-' then rest
                   else error ...

exp input = let
       inputAfterExp = exp input
       inputAfterAddop = addOp inputAfterExp
       rest = term inputAfterAddop
     in
       rest
```

or

```
exp input = term(addOp(exp input));
```

Problems

- How do we select which production to use when alternatives?
- Left-recursive - never terminates

Rewrite Grammar

```
    <exp> ::= <term> {<addop> <term>}*
   <term> ::= <factor> {<mulop> <factor>}*
 <factor> ::= ( <exp> ) | NUM | ID
  <addop> ::= + | -
  <mulop> ::= * | /
```

Rewrite Grammar

```
        <exp> ::= <term> <termTail>                       (1)
   <termTail> ::= <addop> <term> <termTail>               (2)
                | ε                                       (3)
       <term> ::= <factor> <factorTail>                   (4)
 <factorTail> ::= <mulop> <factor> <factorTail>           (5)
                | ε                                       (6)
     <factor> ::= ( <exp> )                               (7)
                | NUM                                     (8)
                | ID                                      (9)
      <addop> ::= + | -                                   (10)
      <mulop> ::= * | /                                   (11)
```

No left recursion
How do we know which production to take?

Predictive Parsing (LL(1))

Goal: $a_1 a_2 \ldots a_n$

$S \rightarrow \alpha$
...
$\rightarrow a_1 a_2 X \beta$

Want next terminal character derived to be a_3

Need to apply a production $X ::= \gamma$ where
1) γ can eventually derive a string starting with a_3 or ← *a_3 in First(γ)*
2) If X can derive the empty string, then see if β can derive a string starting with a_3. ← *a_3 in Follow(X)*

Non-terminals	*ID*	*NUM*	*Addop*	*Mulop*	*(*	*)*	*EOF*
<exp>	1	1			1		
<termTail>			2			3	3
<term>	4	4			4		
<factTail>			6	5		6	6
<factor>	9	8			7		
<addop>			10				
<mulop>				11			

Read off from table which production to apply!
*Ex: Parse 2 * 3 + 7*

Logic(s)

Logic

- Context free language designed for expressing Boolean-valued statements
- Goal is to investigate when logic statements are
  - True in some or all models
  - Provable according to rules for proving
  - ... and to see if there is a connection between the two
- Start simple & work up in complexity.

Propositional Logic

- Definition of well-formed formulas of prop logic:
  - $S ::= P \mid (S \lor S) \mid (S \land S) \mid (\neg S) \mid (S \rightarrow S)$
  - $P ::= p \mid q \mid r \mid \ldots$

Use "::=" in place of "→" for productions to avoid confusion

- Often (informally) drop parentheses around terms
  - Precedence: $\neg$, $\land$, $\lor$, $\rightarrow$
  - $\land$ and $\lor$ are left associative; $\rightarrow$ is right associative.
- Sometimes add $\top$ for true and $\bot$ for false.

Semantics of Propositional Logic

- Meaning of formula depends on meaning of propositional letters.
  - Start with valuation fcn V: Prop Letters → {true,false}
  - Extend to V*: Prop Logic Formulas → {true,false} by
    - $V^*(p) = V(p)$ if p is propositional letter
    - $V^*(\neg\phi)$ = false iff $V^*(\phi)$ = true
    - $V^*(\phi \lor \gamma)$ = true iff $V^*(\phi)$ = true or $V^*(\gamma)$ = true (or both)
    - $V^*(\phi \land \gamma)$ = true iff $V^*(\phi)$ = true and $V^*(\gamma)$ = true
    - $V^*(\phi \rightarrow \gamma)$ = false iff $V^*(\phi)$ = true and $V^*(\gamma)$ = false

Truth Tables

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \rightarrow Q$
T	T	F	T	T	T
T	F	F	F	T	F
F	T	T	F	T	T
F	F	T	F	F	T

Each row corresponds to different valuation

Categories of WFFs

- A formula ϕ is *valid*, or a *tautology*, if for all valuations V, we have $V^*(\phi)$ = true.
- A formula ϕ is *satisfiable* if for some valuation V, we have $V^*(\phi)$ = true.
- A formula ϕ is *falsifiable* if for some valuation V, we have $V^*(\phi)$ = false.
- A formula ϕ is *unsatisfiable*, or a *contradiction*, if for all valuations V , we have $V^*(\phi)$ = false.

Semantic Entailment

- $\phi_1, \ldots, \phi_n \models \psi$ iff for every valuation V s.t. $V^*(\phi_1) = \ldots = V^*(\phi_n)$ = true, then $V^*(\psi)$ = true
  - Example: $P \models Q \rightarrow P$
  - Read $\phi_1, \ldots, \phi_n \models \psi$ as $\phi_1, \ldots, \phi_n$ *semantically entails* ψ
- Hence, $\models \psi$ iff ψ is a tautology.
- Show: $\phi_1, \ldots, \phi_n, \phi \models \psi$ iff $\phi_1, \ldots, \phi_n \models \phi \rightarrow \psi$

Proof Rules

- Syntactically determined set of rules for inferring conclusion from hypotheses.
- Rules provide kind of meaning for connectives
  - Different texts use different rules -- all equivalent!
- $\phi_1, \ldots, \phi_n \vdash \psi$
- Constructing proof is creative
  - Not clear what rules to apply

Rules for ∧

$$\frac{\phi \qquad \psi}{\phi \wedge \psi} \wedge i$$

$$\frac{\phi \wedge \psi}{\phi} \wedge e_1 \qquad\qquad \frac{\phi \wedge \psi}{\psi} \wedge e_2$$

Rules for ¬, →

$$\frac{\neg\neg\phi}{\phi} \neg\neg e \qquad\qquad \frac{\phi}{\neg\neg\phi} \neg\neg i^*$$

$$\frac{\phi \quad \phi \to \psi}{\psi} \to e \qquad\qquad \frac{\neg\psi \quad \phi \to \psi}{\neg\phi} MT^*$$

modus ponens *modus tolens*

**derived rules*

→ Introduction

$$\frac{\boxed{\begin{array}{c}\phi \\ \vdots \\ \psi\end{array}}}{\phi \to \psi} \to i$$

If from an assumption of ϕ, one can deduce ψ,
then one can deduce $\phi \to \psi$
I.e., hypothesis of ϕ is discharged in proof.

∨ Rules

$$\frac{\phi}{\phi \vee \psi} \vee i_1$$

$$\frac{\psi}{\phi \vee \psi} \vee i_2$$

$$\frac{\phi \vee \psi \quad \boxed{\begin{array}{c}\phi \\ \vdots \\ \chi\end{array}} \quad \boxed{\begin{array}{c}\psi \\ \vdots \\ \chi\end{array}}}{\chi} \vee e$$

Negation & ⊥-Rules

$$\frac{\phi \wedge \neg\phi}{\bot}\ \neg e$$

$$\frac{\bot}{\phi}\ \bot e$$

$$\frac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \bot\end{array}}}{\neg\phi}\ \neg i$$

The basic rules of natural deduction:

	introduction	*elimination*
$\wedge$	$\frac{\phi \quad \psi}{\phi \wedge \psi}\ \wedge i$	$\frac{\phi \wedge \psi}{\phi}\ \wedge e_1 \quad \frac{\phi \wedge \psi}{\psi}\ \wedge e_2$
$\vee$	$\frac{\phi}{\phi \vee \psi}\ \vee i_1 \quad \frac{\psi}{\phi \vee \psi}\ \vee i_2$	$\frac{\phi \vee \psi \quad \boxed{\begin{array}{c}\phi\\ \vdots\\ \chi\end{array}} \quad \boxed{\begin{array}{c}\psi\\ \vdots\\ \chi\end{array}}}{\chi}\ \vee e$
$\rightarrow$	$\frac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \psi\end{array}}}{\phi \rightarrow \psi}\ \rightarrow i$	$\frac{\phi \quad \phi \rightarrow \psi}{\psi}\ \rightarrow e$
$\neg$	$\frac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \bot\end{array}}}{\neg\phi}\ \neg i$	$\frac{\phi \quad \neg\phi}{\bot}\ \neg e$
$\bot$	(no introduction rule for $\bot$)	$\frac{\bot}{\phi}\ \bot e$
$\neg\neg$		$\frac{\neg\neg\phi}{\phi}\ \neg\neg e$ *Classical only*

Proofs

- Ordered list of steps where each step justified as premise or by proof rule from earlier steps.
- Show ⊢ ¬(φ ∧ ¬φ)

1.	φ ∧ ¬φ	*assumption*
2.	φ	1, ∧e
3.	¬φ	1, ∧e
4.	⊥	2,3, ¬e
5.	¬(φ ∧ ¬φ)	1-4, ¬i

Always indicate proof rule and steps used to get new wff Use boxes for subproofs to be discharged

Distinction between hypothesis and assumption

Example Proofs

- Be careful with proof boxes:
  - Can't use internal steps when reasoning outside the box.
- Typically work backwards!
- Show
  - φ ∨ ψ ⊢ ψ ∨ φ
  - ⊢ φ → (ψ →φ)
  - φ →(ψ → χ) ⊢ ψ →(φ → χ)

Some useful derived rules:

$$\frac{\phi \to \psi \quad \neg\psi}{\neg\phi}\ \mathrm{MT} \qquad \frac{\phi}{\neg\neg\phi}\ \neg\neg\mathrm{i}$$

$$\frac{\boxed{\begin{array}{c}\neg\phi\\ \vdots\\ \bot\end{array}}}{\phi}\ \mathrm{PBC} \qquad \frac{}{\phi \vee \neg\phi}\ \mathrm{LEM}$$

Classical only (PBC and LEM)

Constructivist vs Classical Logics

- Constructivists don't believe in ¬¬e rule:
  - $\neg\neg\phi \vdash \phi$
- Don't believe $\vdash \phi \vee \neg\phi$ except in special cases.
  - Don't accept proof by contradiction!
- Give constructive proof of $\phi \vee \psi \vdash \neg(\neg\phi \wedge \neg\psi)$

Troublesome Proof

- Are there two irrational numbers, *a* and *b*, such that a^b is rational?
  - Notice $(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^2 = 2$
  - Case 1: $\sqrt{2}^{\sqrt{2}}$ is rational and take $a = b = \sqrt{2}$
  - Case 2: $\sqrt{2}^{\sqrt{2}}$ is irrational and take $a = \sqrt{2}^{\sqrt{2}}$, $b = \sqrt{2}$
- Constructivist rejects because can't tell which alternative is true.