Plan for Today

LR parsing as pushdown automata

Intro to PA4: Parsing and AST generation
- restricted grammar in example code
- parse trees versus ASTs
- using dot to visualize ASTs

Abstract syntax tree construction
- building them with nonterminal attributes
- examples: Figures 2.39, 5.10, 5.11
- variable declarations included?
- sequences of statements

Example LR Parse Table

<table>
<thead>
<tr>
<th>State</th>
<th>Action</th>
<th>Goto</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>r2</td>
<td>r2</td>
</tr>
<tr>
<td>1</td>
<td>s2</td>
<td>accept</td>
</tr>
<tr>
<td>2</td>
<td>r2</td>
<td>r2</td>
</tr>
<tr>
<td>3</td>
<td>s2</td>
<td>s4</td>
</tr>
<tr>
<td>4</td>
<td>r1</td>
<td>r1</td>
</tr>
</tbody>
</table>

Pushdown Automata for Grammar

Structure of the MiniJava Compiler

Analysis
- character stream
- tokens “words”
- syntactic analysis
- semantic analysis
- AST “sentences”

Synthesis
- IR code generation
- optimization
- code gen
- MIPS
Abstract Syntax Tree for Memory Layout Example