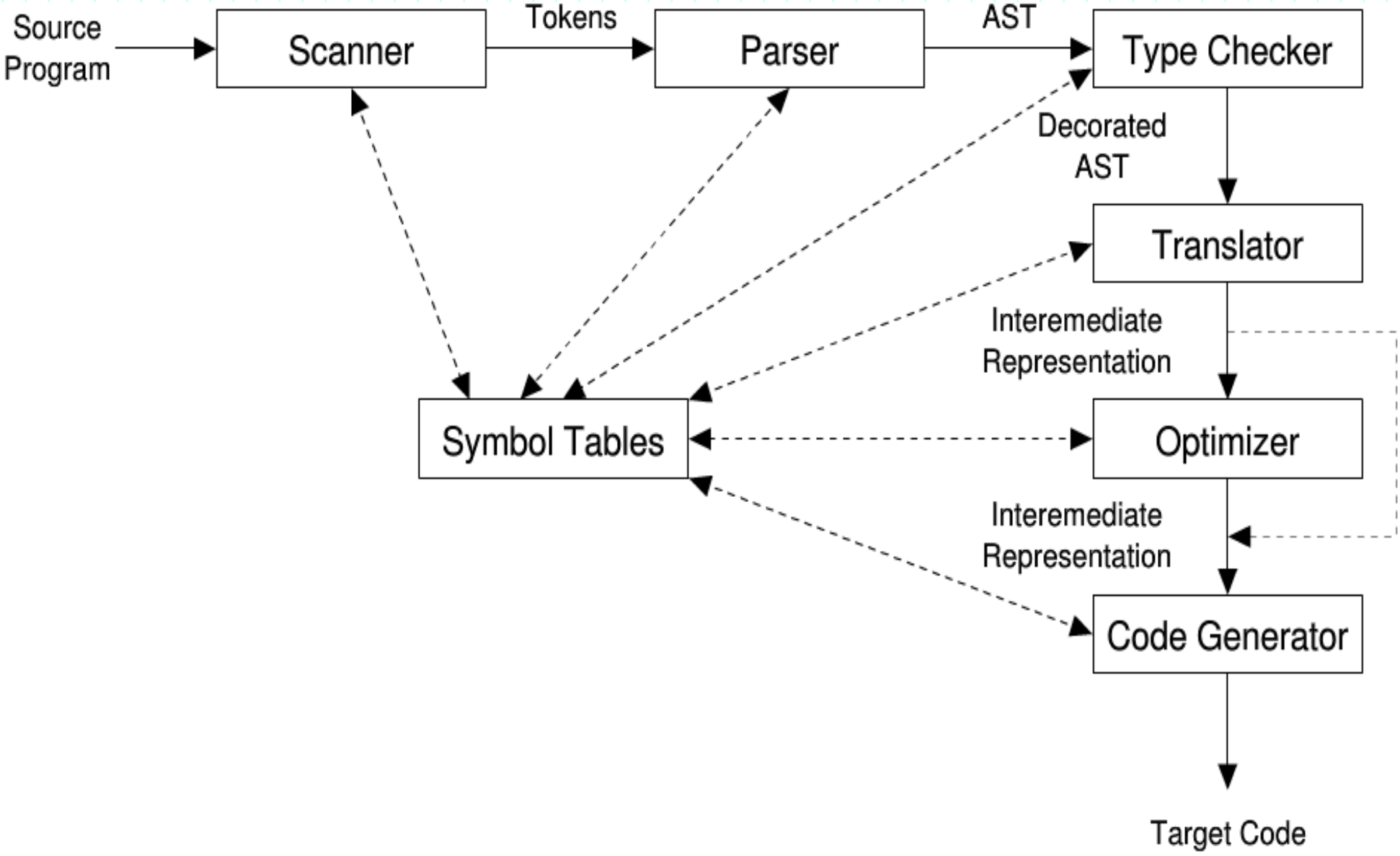
CSE431 – Translation of Computer Languages

- Code Generation

Doug Shook

Roadmap


Source Program
Scanner
Tokens
Parser
AST
Type Checker
Decorated AST
Translator
Interemediate Representation
Symbol Tables
Optimizer
Interemediate Representation
Code Generator
Target Code

Code Generation

- Essentially the inverse of parsing
  - Parsing = Grammar -> AST
  - Code Generation = AST -> Code

- Our target code here is JVM
  - This process would be different if we targeted native machine code
    - Structure
    - Registers

Code Generation - Visitors

- TopVisitor
  - Base class – starts at the root, works down the tree
  - Handles class/method declarations
  - Calls other visitors when necessary
- MethodBodyVisitor
  - The workhorse
  - Calls other visitors when necessary

Code Generation - Visitors

- LHSVisitor
  - Generates code for LHS of assignments
  - May call other visitors if necessary (like what?)
- SignatureVisitor
  - Handles method definitions and invocations
    - What do we need to process?
  - Called by MethodBodyVisitor for invocations

Code Emission

- Emission is the act of generating instructions
  - CodeGenVisitors emit instructions

- Consider what this would look like for addition:

```
visit(left)
visit(right)
emit("iadd\n")
```

Postludes

- Sometimes a single emission isn't enough
  - Or there's nothing to emit when a node is first visited
- Consider assignment statements
  - Must visit LHS first (why?)
  - Visit RHS to compute value
  - Must revisit LHS (why?)
- Better
  - LHS saves operation to a postlude
  - Parent requires postlude emission

TopVisitor

- Handles class and method declarations
- ClassDeclaring
  - Emit the class skeleton (jasmin)

```
.class public foo
.super java/lang/Object
.field public myField I
```

- Do we need a postlude?

TopVisitor

- MethodDeclaring
  - Emit method skeleton

```
.method public static bar(S)I
.limit locals 2
.limit stack 4
```

- We need a postlude here:

```
.end method
```

- Where can we get the limit values?

MethodBodyVisitor

- Generates code for any node that can be found in a method body
  - Like what?
- Don’t forget about the other visitors called from MethodBodyVisitor
  - LHSVisitor
  - SignatureVisitor

LHSVisitor

- Generates the correct register and postlude
- May have to call other visitors
  - MethodBodyVisitor, for example (why?)
- How do we know the correct register?

LHSVisitor

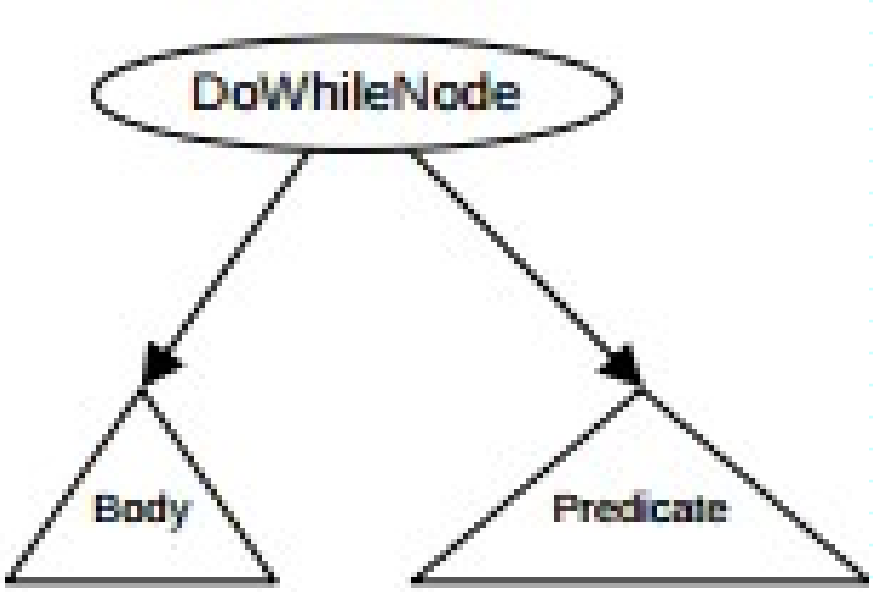
- Locals
  - Emission?
  - Postlude?
- Statics
  - Emission?
  - Postlude?
- Fields
  - Emission?
  - Postlude?
- Arrays?

Lab 5

- Let’s go over the requirements for lab 5

Exercises

- Write a visit statement for a do while node:



- Assume that the predicate will leave a 0(false) or 1(true) on the top of the stack

Exercises

- Determine the emission and postlude for LHSVisitor when the LHS of an assignment is:
  - A static field of class C
  - A local variable in register 12
  - A field x of instance o in register 4 of class C
  - A value in array b (register 7) at index i (register 8)

Exercises

- Write a visit method for an AND node. Assume that booleans are implemented as integers (0 = false, 1 = true)

Putting it all together

- For the following:
  - AST -> Type Check -> Reachability -> Jasmin

```
public float withdraw(float amount) {
    if (this.balance – amount >= 0) {
      this.balance = this.balance – amount;
    }
    return this.balance;
}
```