That other team

Quinn Mckown
Chris Gregory

Basic flow

String -> Lexer -> Tokens

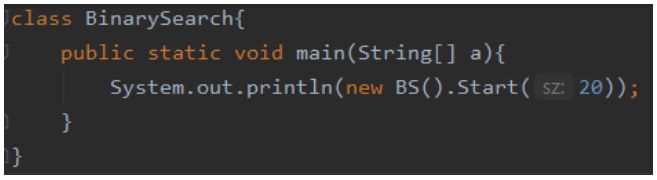
Tokens -> Parser -> AST

AST -> Emitter -> IRInstructions

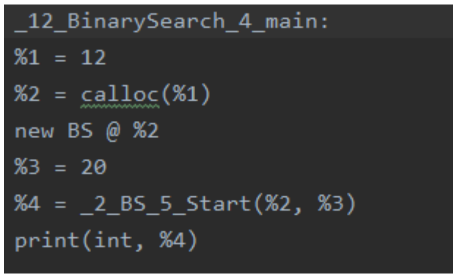
IR -> X86Emitter -> X86Instructions

Intermediate Language

- Everything is a virtual register
- 3 argument instructions
- Mangling
- Pseudo assembly with load and store instructions (no array indexing)
- Print, Calloc, and function calls are not compiled down

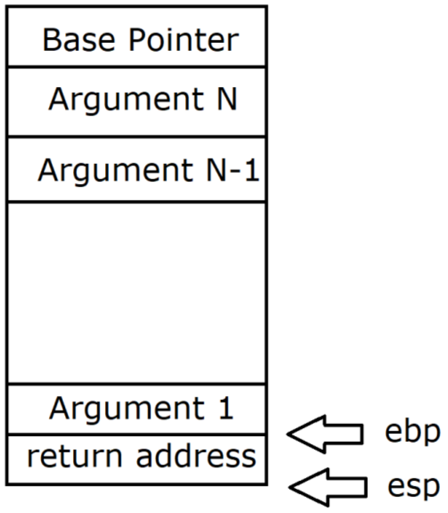


```
class BinarySearch{
    public static void main(String[] a){
        System.out.println(new BS().Start( sz: 20));
    }
}
```



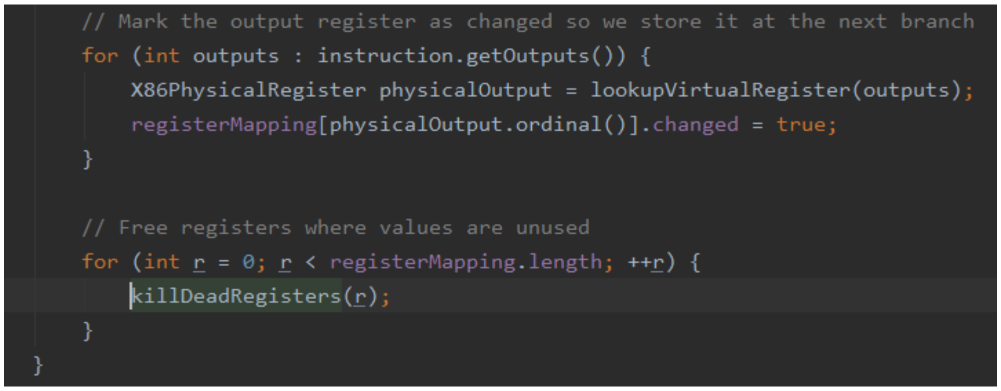
```
_12_BinarySearch_4_main:
%1 = 12
%2 = calloc(%1)
new BS @ %2
%3 = 20
%4 = _2_BS_5_Start(%2, %3)
print(int, %4)
```

Activation Records



X86 Optimization

- Register allocation is optimized using live variable analysis
- Dead virtual registers are marked as cleared
- Peephole optimization to remove unnecessary loads and stores

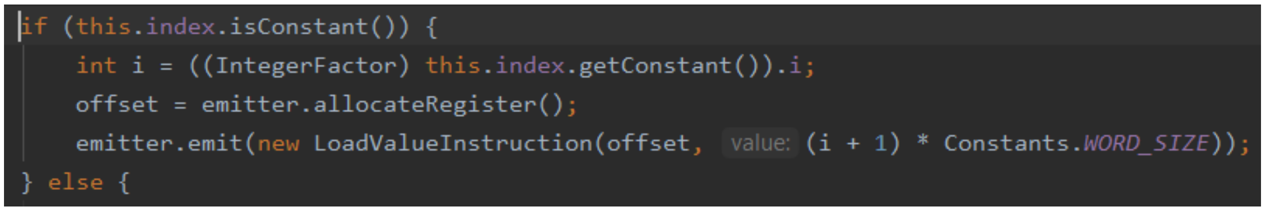


```
    // Mark the output register as changed so we store it at the next branch
    for (int outputs : instruction.getOutputs()) {
        X86PhysicalRegister physicalOutput = lookupVirtualRegister(outputs);
        registerMapping[physicalOutput.ordinal()].changed = true;
    }

    // Free registers where values are unused
    for (int r = 0; r < registerMapping.length; ++r) {
        killDeadRegisters(r);
    }
}
```

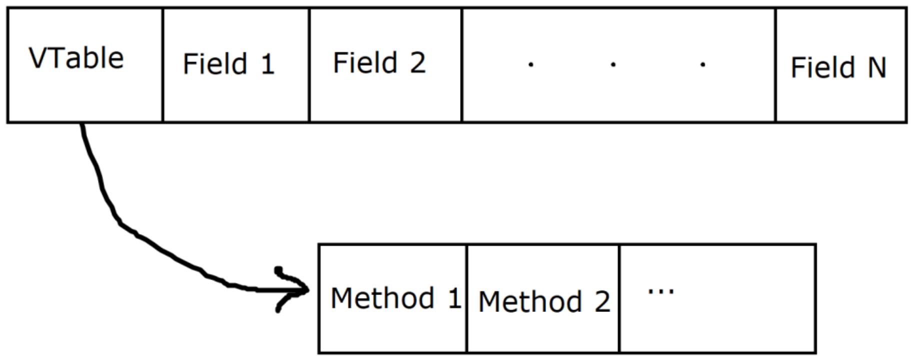
Parse tree optimization

- Constant expressions are reduced
- Constant propagation through statements
- Unreachable code elimination



```
if (this.index.isConstant()) {
    int i = ((IntegerFactor) this.index.getConstant()).i;
    offset = emitter.allocateRegister();
    emitter.emit(new LoadValueInstruction(offset, value: (i + 1) * Constants.WORD_SIZE));
} else {
```

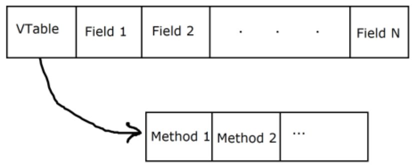
VTables



VTables

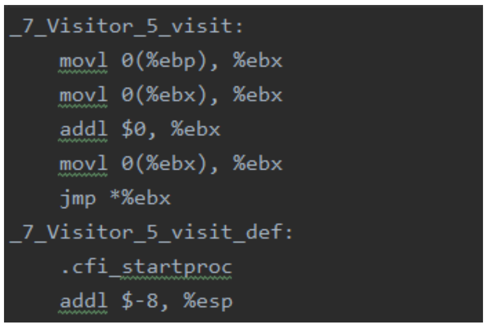
```
    .globl _7_Visitor_vtable
    .align 4
_7_Visitor_vtable:
    .long _7_Visitor_5_visit_def
```

```
    .globl _9_MyVisitor_vtable
    .align 4
_9_MyVisitor_vtable:
    .long _9_MyVisitor_5_visit_def
```

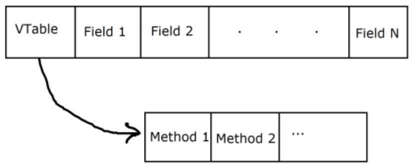


VTables

```
call _get_pc_ebx
addl $_GLOBAL_OFFSET_TABLE_, %ebx
leal _9_MyVisitor_vtable@GOTOFF(%ebx), %ebx
movl %ebx, 0(%eax)
```



```
_7_Visitor_5_visit:
    movl 0(%ebp), %ebx
    movl 0(%ebx), %ebx
    addl $0, %ebx
    movl 0(%ebx), %ebx
    jmp *%ebx
_7_Visitor_5_visit_def:
    .cfi_startproc
    addl $-8, %esp
```


VTable
Field 1
Field 2
. . .
Field N
Method 1
Method 2
...