

Compiler Construction

Recap

Rina Zviel-Girshin and Ohad Shacham
School of Computer Science
Tel-Aviv University

Admin

- You may submit PA4 until Jan 27
 - FIRM

2

Exam

- 31/01/2010 at 9:00
- Look for previous exams on website
- Possible questions
 - Parsing
 - Extend IC
 - Activation records
 - ...

3

Register allocation

- Sethi Ullman
 - can only handle expressions without side effect
- Global register allocation
- IR registers are treated as local variables
- When we have an actual spill we use the stack

4

Weighted register allocation

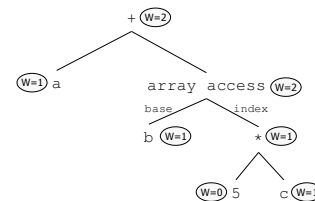
- Can save registers by re-ordering subtree computations
- Label each node with its weight
 - Weight = number of registers needed
 - Leaf weight known
 - Internal node weight
 - $w(\text{left}) > w(\text{right})$ then $w = \text{left}$
 - $w(\text{right}) > w(\text{left})$ then $w = \text{right}$
 - $w(\text{right}) = w(\text{left})$ then $w = \text{left} + 1$
- Choose heavier child as first to be translated
- Have to check that no side-effects exist

5

Weighted reg. alloc. example

$R0 := TR[a+b[5*c]]$

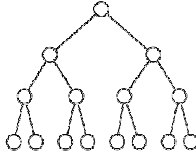
Phase 1: - check absence of side-effects in expression tree
- assign weight to each AST node



6

Sethi Ullman

- On what type of tree SU achieved the **worst case** respect to the tree's height?



7

Sethi Ullman

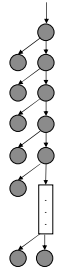
- On what type of tree SU achieved the **best case** respect to the tree's height?



8

Sethi Ullman

- On what type of tree, the maximal ratio between registers allocated by traversing the tree from left to right or from right to left

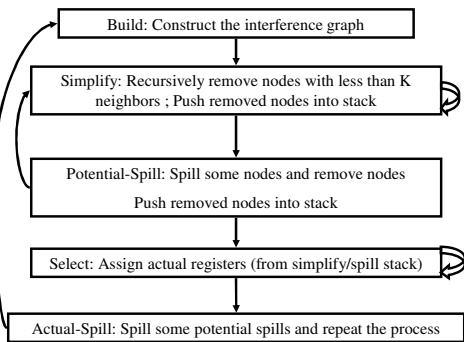


9

Global Register Allocation

- Register allocation technique based on graph coloring
- Graph coloring – NP complete
 - Uses heuristics to approximate

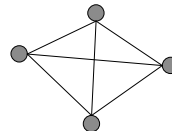
10



11

Global Register Allocation

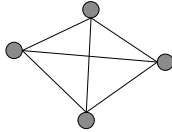
- Given k registers
 - on which graph the algorithm will generate an actual spill?



12

Global Register Allocation

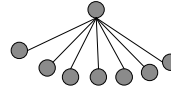
- Given an interference graph
 - How can we compute the **best case** of the global register allocation on the graph?



13

Global Register Allocation

- Given an interference graph
 - How can we compute the **worst case** of the global register allocation on the graph?

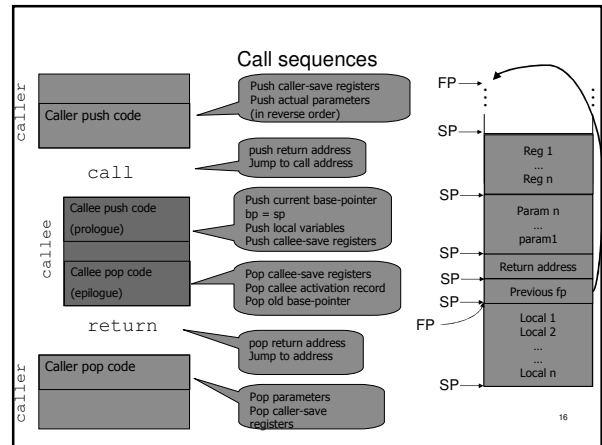


14

Activation records

- New environment = activation record (a.k.a. frame)
- Activation record = data of current function / method call
 - User data
 - Local variables
 - Parameters
 - Return values
 - Register contents
 - Administration data
 - Code addresses

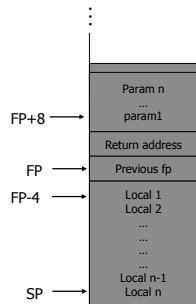
15



16

Accessing stack variables

- Use offset from EBP
- Stack grows downwards
- Above EBP = parameters
- Below EBP = locals
- Examples
 - `%ebp + 4` = return address
 - `%ebp + 8` = first parameter
 - `%ebp - 4` = first local



17

Question – Activation record

- Run the program on the input and show the stack at after ...
- Add a “newS” operation that allocates on the stack
 - How to do it?

18

Question – Activation record

Lexical analyzer

- Add a new token "NEWS"
- "newS" { return token(sym.NEWS); }

Parser

```
Expr ::= NEWS CLASS_ID:name LP RP
      { RESULT = new NewSClass(parser.getLine(),name); ; } |
      NEWS Type:type LB Expr:size RB
      { RESULT = new NewSArray(type, size); ; } |
      ...
```

19

Question – Activation record

Semantic analysis

- Use the types of "newS" as the ones of "new"

Intermediate representation

- Introduce new allocation methods
- `__allocateStackObject(s), R1`
- `__allocateStackArray(s), R1`
- Translate newS exactly as new while using the above methods
- Assume that s is size in bytes for object
- Assume that s is the size of array for arrays

20

Question – Activation record

Code generation

- Need to implement allocation methods in assembly

- `__allocateStackObject(s)`
- `__allocateStackArray(s)`

- `__allocateStackObject(s), R1`

```
mov -4(%ebp), %eax // assume s is the first local var
sub %eax,%esp
mov %esp, -8(%ebp) // assume R1 is the second local var (spill)
```

21

Question – Activation record

- `__allocateStackArray(s), R1`

```
mov -4(%ebp), %eax // assume s is the first local var
mov %eax, %ebx
add $1, %ebx
mul $4, %ebx
sub %ebx,%esp
mov %eax, (%esp)
mov %esp, %eax
add $4, %eax
mov %eax, -8(%ebp) // assume R1 is the second local var (spill)
```

22

2006 מועד ב, שאלה 3

```
class ByRefExample {
  static void main(string [] args) {
    int x = 5;
    increment(x);
    Library.println(x);
  }

  static void increment(int & y) {
    y = y + 1;
  }
}
```

23

2006 מועד ב, שאלה 3

- Treat by reference as address
- Lexer
 - Add token for &
- Parser
 - Add reference type
 - Can appear only as a formal parameter
- Semantic analysis
 - New rules for type checking
 - Compare the actual parameter type and the reference type
 - Treat the scope checking

24

3 מועד ב, שאלה 3

- IR
 - Add support for by reference
 - Usage of a referenced var should be marked
- Code generation
 - Caller
 - Add the pointer of the actual parameter to the stack instead of the actual parameter itself
 - Callee
 - Treat the parameter as a pointer

25

Question

- Add constructors to IC
- Treat the constructor as a function and call the function on allocation
- Lexical analysis
 - Nothing
- Parsing
 - Add a method without return type and allocate a new constructor type for the AST
 - Allow Object allocation with parameters inside the parenthesis
 - NEW CLASS_ID:name LP argumentsList RP
 - Extend class allocation node to support include arguments list

26

Question

- Semantic analysis
 - Check that the constructor name and the class name are the same
 - Check that the actual parameter's type and the formal parameter are the same
- Intermediate representation
 - Add a new allocation function that keeps parameters for the constructor
 - Allocate the first entry in the DV for the constructor
- Code generation
 - Add a call to the constructor after allocation
 - Of course, add the parameters to the stack and save registers as needed in every function call

27

Question

- After implementing constructors add "super" to IC
- ```
class A {
 A(int a) {...}
}
class B extends A {
 B(int b)
 {
 super(b);
 ...
 }
}
```
- Super should be the first call inside the constructor

28

### Question

- Lexical analysis
  - Add a new token SUPER
- Parsing
  - Add a rule for super?
  - Add a new AST node for super?

29

### Question

- Semantic analysis
  - Check that super appears only inside constructors
  - Check that super is the first action
  - Check that the super class contains constructor with matching parameter
  - Bind the call to the proper constructor

30

### Question

- Intermediate representation
  - Add a new call type to a super constructor
    - Known at compile time
    - Should get "this" at runtime
- Code generation
  - Translate the super call to call that gets this as the first argument

31

### Questions

- Extend IC with function overloading
- Show the problem that can be achieved by doing static function overloading
- Do we know a solution for this problem?
- Add runtime handling for function overloading in order to solve this problem
- Can we use some compile time optimization to reduce the runtime overhead?

32