Plan for today

PA7 overview

Expression Tree intermediate representation

Possible approaches for performing translation

- All visitor methods return a Translate.Exp reference
- Visitor methods return void, but map AST nodes to Tree.Exp nodes and a list of Tree.Stm

PA7 Overview

TranslateToIRTree.java

- driver for PA7

ast_visitors/

- BuildSymTable.java has one new piece
- Translate.java
- Frag.java

Tree/*

Temp/

- TempMap.java interface

Frame/

- new functionality added to Frame.java interface and Access.java interface

Mips/

- MipsFrame.java implements the new functionality
- InFrame.java implements the Tree.Exp exp(Tree.Exp) method

SymTable/

- VarSTE, LocalSTE, and MemberSTE implement Tree.Exp exp(Tree.Exp)
- ClassSTE has getNumMembers() method

TempMap functionality in MipsFrame

```
    static final temp zero = new temp(); // zero reg
    static final temp v0 = new temp(); // function result
    static final temp t0 = new temp(); // callee-saved
    static final temp t1 = new temp();
    ...
    static final temp sp = new temp(); // stack pointer
    static final temp s0 = new temp(); // callee-save (frame pointer)
    static final temp ra = new temp(); // return address

private static final
    HashMap<temp,string> tempMap = new HashMap<temp,string>(32);
    static {
        tempMap.put(zero, "$0");
        tempMap.put(v0,   "$v0");
        tempMap.put(t0,   "$t0");
        ...
        tempMap.put(sp,   "$sp");
        tempMap.put(s0,   "$fp");
        tempMap.put(ra,   "$ra");
    }
```

```
public string tempMap(temp temp) {
        if (tempMap.containsKey(temp)) {
            return tempMap.get(temp);
        } else {
            return temp.toString();
        }
    }
```


TranslateToIRTree driver

```
// translate the AST to the IR Tree data structure
Translate translateVisitor = new Translate(globalST, frame);
ast.apply(translateVisitor);
...
// debug output
translateVisitor.outputTrees(new PrintWriter(IRDebug_out));

// result output
Iterator<Frag> fragIter = translateVisitor.getResults();
...
Tree.Print pr = new Tree.Print(printWriter,frame);
while (fragIter.hasNext()) {
    Frag frag = fragIter.next();
    printWriter.println("==========================================");
    printWriter.println(frag.frame.name.toString());
    pr.prStmList(frag.body);
    printWriter.println();
}
```

Key features in Translate

HashMaps

```
// mapping of nodes in AST to tree.Exp's
private Map<Node,tree.Exp> nodeToTreeExp;
// mapping of AST nodes to a list of tree.Stm
private Map<Node,List<tree.Stm>> nodeToTreeStmList;
```

Frag list

```
private LinkedList<Frag> frags = new LinkedList<Frag>();
public Iterator<Frag> getResults() {
    return frags.iterator();
}
```

outputTrees

- use for debugging

Tree intermediate representation

ExpCONST(int i) - The integer constant i

ExpNAME(Label n) - Constant address. Corresponds to label in assembly.

ExpTEMP(Temp t) - Temporary t. "Infinite" Temps in Tree IR.

ExpBINOP(int binop, Exp left, Exp right) - left binop right

ExpMEM(Exp exp) - If left child of move, then store into address calculated by exp. Otherwise, fetch value at address calculated by exp.

ExpCALL(Exp func, List<Exp> args) - evaluate func to find func address, then evaluate args left to right.

StmMOVE(Temp t, e) - Eval e and put result in t.

StmMOVE(ExpMEM(e1), e2) - Eval e2 and store into address e1..

StmEXP(Exp e) - Eval e and ignore result

StmJUMP(Label targ) - Transfer control to given label.

StmCJUMP(int rel, Exp l, Exp r, Label t, Label f) - if l rel r then goto t else goto f

StmLABEL(Label l) - Label in assembly.