Decaf PA3 实验报告

> 计64 翁家翌 2016011446

本次实验我也不知道我写的都是啥，就是面向 `output/*.tac` 编程，对着翻译，然后就做完了，没想到里面有几个给的tac还是错的！！！

DivZero

1. 修改 `decaf/error/RuntimeError.java`，添加除0的报错信息：

```java
public static final String DIVISION_BY_ZERO = "Decaf runtime error: Division by zero error.\n";
```

2. 修改 `decaf/translate/Translater.java`，首先抄一遍 `genCheckNewArraySize`

```java
public void genCheckDivisionZero(Temp src) {
    Label exit = Label.createLabel();
    Temp cond = genEqu(src, genLoadImm4(0));
    genBeqz(cond, exit);
    Temp msg = genLoadStrConst(RuntimeError.DIVISION_BY_ZERO);
    genParm(msg);
    genIntrinsicCall(Intrinsic.PRINT_STRING);
    genIntrinsicCall(Intrinsic.HALT);
    genMark(exit);
}
```

3. 然后在函数 `genDiv` 和 `genMod` 下面加一句 `genCheckDivisionZero(src2);`

scopy

1. 在typecheck里面的 `visitScopy` 写一下 `scopy` 的 `symbol`： `scopy.symbol = v;`
2. 在transpass2里面直接加入以下内容，直接翻译自 `output/q1-scopy-test1.tac`： （找class比较蛋疼）

```
@Override
public void visitScopy(Tree.Scopy scopy) {
    scopy.expr.accept(this);
    Temp t40 = tr.genDirectCall(((ClassType)
(scopy.expr.type)).getSymbol().getNewFuncLabel(), BaseType.INT);
    Temp t41 = tr.genLoad(scopy.expr.val, OffsetCounter.WORD_SIZE * 2);
    tr.genStore(t41, t40, OffsetCounter.WORD_SIZE * 2);
    Temp t42 = tr.genLoad(scopy.expr.val, OffsetCounter.WORD_SIZE);
    tr.genStore(t42, t40, OffsetCounter.WORD_SIZE);
    tr.genAssign(((Variable)(scopy.symbol)).getTemp(), t40);
}
```

GuardStmt

1. transpass2加入以下代码（抄IfStmt的）

```
@Override
public void visitGuardedES(Tree.GuardedES guardedES) {
    Label exit = Label.createLabel();
    tr.genBeqz(guardedES.expr.val, exit);
    if (guardedES.stmt != null) {
        guardedES.stmt.accept(this);
        tr.genBranch(exit);
    }
    tr.genMark(exit);
}
@Override
public void visitGuardStmt(Tree.GuardStmt guardStmt) {
    for (Tree.GuardedES i: guardStmt.guardedES) {
        i.expr.accept(this);
        Label falseLabel = Label.createLabel();
        tr.genBeqz(i.expr.val, falseLabel);
        i.stmt.accept(this);
        tr.genMark(falseLabel);
    }
}
```

VarIdent

1. 修改 `decaf/tree/Tree.java` 中的 `LValue` 类，在 `Kind` 中添加 `UNKNOWN` 类别，为了 `visitAssign` 不报错
2. 修改 `decaf/typecheck/TypeCheck.java` 中的 `visitVarIdent`，加入 `varIdent.lvKind = Tree.LValue.Kind.UNKNOWN;`
3. 修改 transpass2 的 `visitAssign` 函数

```
case UNKNOWN:
    tr.genAssign(((Tree.VarIdent) assign.left).symbol.getTemp(), assign.expr.val);
    break;
```

4. 在 transpass2 中添加 `visitVarIdent` 函数（抄 `VarDef` 的）

```java
@Override
public void visitVarIdent(Tree.VarIdent varIdent) {
    if (varIdent.symbol.isLocalVar()) {
        Temp t = Temp.createTempI4();
        t.sym = varIdent.symbol;
        varIdent.symbol.setTemp(t);
    }
}
```

%%

1. 向 `decaf/error/RuntimeError.java` 中添加报错信息：（tac的报错信息是错的）

```java
public static final String NEGSIZE2 = "Decaf runtime error: The length of the
created array should not be less than 0.\n";
```

2. 修改 `decaf/translate/Translater.java`：
    1. 给 `genCheckNewArraySize` 加一个参数:

    ```java
    public void genCheckNewArraySize(Temp size, boolean cas) {
        Label exit = Label.createLabel();
        Temp cond = genLes(size, genLoadImm4(0));
        genBeqz(cond, exit);
        Temp msg = genLoadStrConst(cas?
    RuntimeError.NEGSIZE2:RuntimeError.NEGATIVE_ARR_SIZE);
        genParm(msg);
        genIntrinsicCall(Intrinsic.PRINT_STRING);
        genIntrinsicCall(Intrinsic.HALT);
        genMark(exit);
    }
    ```

    2. 添加 `%%` 的产生函数：（对着 `q5-array-test1-bool.tac` 翻译），class new那个地方卡了很久

    ```java
    public Temp genMOMO(Temp src1, Temp src2, Tree.Expr left) {
        genCheckNewArraySize(src2, true);
        Temp unit = genLoadImm4(OffsetCounter.WORD_SIZE);
        Temp size = genAdd(unit, genMul(unit, src2));
        genParm(size);
        Temp obj = genIntrinsicCall(Intrinsic.ALLOCATE);
        genStore(src2, obj, 0);
        Label loop = Label.createLabel(); // L11
        Label exit = Label.createLabel(); // L12
        Label l13 = Label.createLabel(); // L13
        Label l14 = Label.createLabel(); // L14
        Temp zero = genLoadImm4(0);
        append(Tac.genAdd(obj, obj, size));
        genMark(loop); // L11
        append(Tac.genSub(size, size, unit));
    ```

```
        genBeqz(size, exit);
        append(Tac.genSub(obj, obj, unit));
        genStore(zero, obj, 0);
        genBranch(loop);
        genMark(exit); // L12
        Temp t14 = genLoadImm4(0);
        genMark(l14); // L14
        Temp t15 = genLes(t14, src2);
        genBeqz(t15, l13);
        Temp t16 = genLoadImm4(OffsetCounter.WORD_SIZE);
        Temp t17 = genMul(t14, t16);
        Temp t18 = genAdd(obj, t17);
        genStore(src1, t18, 0);
        // class
        if (left.type.isClassType()) {
            Temp t33 = genDirectCall(((ClassType)
    (left.type)).getSymbol().getNewFuncLabel(), BaseType.INT);
            Temp t34 = genLoad(src1, OffsetCounter.WORD_SIZE);
            genStore(t34, t33, OffsetCounter.WORD_SIZE);
            genStore(t33, t18, 0);
        }
        Temp t19 = genLoadImm4(1);
        Temp t20 = genAdd(t14, t19);
        genAssign(t14, t20);
        genBranch(l14);
        genMark(l13);
        return obj;
    }
```

3. 修改 transpass2 的 `visitBinary`， 添加对 `%%` 的判断:

```
case Tree.MOMO:
    expr.val = tr.genMOMO(expr.left.val,  expr.right.val, expr.left);
    break;
```

Default

1. 向 transpass2 中添加 `visitDefault`， 直接翻译 `q5-array-test2-default.tac`:

```
@Override
public void visitDefault(Tree.Default deft) {
    deft.array.accept(this); // val: t3
    deft.index.accept(this); // val: t22
    deft.deft.accept(this); // val: t23
    Label l15 = Label.createLabel();
    Label l16 = Label.createLabel();
    Temp t24 = Temp.createTempI4();
    Temp t25 = tr.genLoadImm4(0);
    Temp t26 = tr.genGeq(deft.index.val, t25);
    tr.genBeqz(t26, l15);
    Temp t27 = tr.genLoad(deft.array.val, -OffsetCounter.WORD_SIZE);
    Temp t28 = tr.genLes(deft.index.val, t27);
    tr.genBeqz(t28, l15);
    Temp t29 = tr.genLoadImm4(OffsetCounter.WORD_SIZE);
    Temp t30 = tr.genMul(deft.index.val, t29);
    Temp t31 = tr.genAdd(deft.array.val, t30);
    Temp t32 = tr.genLoad(t31, 0);
    tr.genAssign(t24, t32);
    tr.genBranch(l16);
    tr.genMark(l15);
    tr.genAssign(t24, deft.deft.val);
    tr.genMark(l16);
    deft.val = t24;
}
```

foreach

1. transpass2 中添加 `visitExDef` （抄 `VarDef` ） 和 `visitForEach` （翻译 `q5-array-test3-var.tac`，发现tac里面的条件跳转写错了）

```
@Override
public void visitExDef(Tree.ExDef exdef) {
    if (exdef.symbol.isLocalVar()) {
        Temp t = Temp.createTempI4();
        t.sym = exdef.symbol;
        exdef.symbol.setTemp(t);
    }
}
@Override
public void visitForEach(Tree.ForEach foreach) {
    Label l18 = Label.createLabel(); // exit
    Label l19 = Label.createLabel();
    Label l20 = Label.createLabel();
    foreach.exdef.accept(this);
    foreach.expr1.accept(this); // t13
    Temp t33 = tr.genLoadImm4(0);
    Temp t34 = tr.genLoadImm4(1);
    Temp t35 = tr.genLoad(foreach.expr1.val, -OffsetCounter.WORD_SIZE);
    tr.genBranch(l19);
    tr.genMark(l20);
    Temp t36 = tr.genAdd(t33, t34);
    tr.genAssign(t33, t36);
    tr.genMark(l19);
    Temp t37 = tr.genLes(t33, t35);
    tr.genBeqz(t37, l18);
    Temp t38 = tr.genLoadImm4(OffsetCounter.WORD_SIZE);
    Temp t39 = tr.genMul(t33, t38);
    Temp t40 = tr.genAdd(foreach.expr1.val, t39);
    Temp t41 = tr.genLoad(t40, 0);
    tr.genAssign(foreach.exdef.symbol.getTemp(), t41);
    if (foreach.istrue == false) {
        foreach.expr2.accept(this); // t43
        tr.genBeqz(foreach.expr2.val, l18);
    }
    loopExits.push(l18);
    if (foreach.stmt != null) {
        foreach.stmt.accept(this);
    }
    loopExits.pop();
    tr.genBranch(l20);
    tr.genMark(l18);
}
```