

COMPILERS

IR Optimisations

hussein suleman
uct csc3003s 2007

Types of Optimisations

- Peephole Optimisation
 - Considers only a restricted subset of the IR tree
- Global Optimisation
 - Considers the entire program
- Modular Optimisation
 - Considers each module in its entirety

Constant Folding

- ❑ Convert calculations that result in a constant value into a pre-calculated constant.

BINOP (PLUS, CONST 1, CONST 2)



CONST 3

Constant Propagation

- ❑ Convert uses of a constant's name to its value.

a = 5; b = a; c = b;

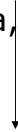


a = 5; b = 5; c = 5;

Unreachable Code Elimination

- Remove code that will never be executed because of the logic of the program.

```
SEQ (SEQ (CJUMP (LT, CONST 1, CONST  
2, LABEL T, LABEL F), LABEL F), MOVE  
(TEMP a, TEMP b))
```

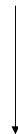


```
JUMP (NAME T)
```

Inlining

- Replace subprogram calls with the body of each subprogram.

```
sub incr { return $_+1; }  
c = incr(a) + incr(b);
```



```
c = a+1 + b + 1;
```

Loop Unrolling

- Convert short loops with a constant number of iterations into multiple static statements.

```
for ( int i=0; i<=2; i++ ) { a[i] = 0; }
```



```
a[0] = 0;  
a[1] = 0;  
a[2] = 0;
```

Common Subexpression Elimination

- Eliminate identical sub-expressions that are calculated multiple times.

```
a = b + c * d; e = b + c * d;
```



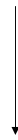
```
x = b + c * d; a = x; e = x;
```

Strength Reduction

- Convert multiplications within a loop into (possibly faster) additions.

basically ...

```
for ( int i=0; i<10; i++ ) { print i*5; }
```



```
for (int i=0; i<50; i+=5 ) { print i; }
```