CSE110A: Compilers

Topics:

- *Module 4 – LVN STITCH-CODE-BACK*

Stitching optimized code back into program

How to stitch optimized code back into the program

```
a = b + c;
d = e + f;
g = b + c;

label_0:
h = g + a;
k = a + g;
```

How to stitch optimized code back into the program

split into basic blocks

```
a = b + c;
d = e + f;
g = b + c;

label_0:
h = g + a;
k = a + g;
```

```
a = b + c;
d = e + f;
g = b + c;
```

```
label_0:
h = g + a;
k = a + g;
```

How to stitch optimized code back into the program

```
a = b + c;
d = e + f;
g = b + c;

label_0:
h = g + a;
k = a + g;
```

```
a = b + c;
d = e + f;
g = b + c;
```

```
label_0:
h = g + a;
k = a + g;
```

number

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = b0 + c1;
```

```
label_0:
h2 = g0 + a1;
k3 = a1 + g0;
```

How to stitch optimized code back into the program

move numbered code on slide to make room

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = b0 + c1;
```

```
label_0:
h2 = g0 + a1;
k3 = a1 + g0;
```

How to stitch optimized code back into the program

optimize

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = b0 + c1;
```

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = a1 + g0;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
```

How to stitch optimized code back into the program

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = b0 + c1;
```

```
label_0:
h2 = g0 + a1;
k3 = a1 + g0;
```

optimize

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
```

put together?

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
label_0:
h2 = g0 + a1;
k3 = h2;
```

What are the resulting issues?

How to stitch optimized code back into the program

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = b0 + c1;
```

```
label_0:
h2 = g0 + a1;
k3 = a1 + g0;
```

optimize

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
```

put together?

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
label_0:
h2 = g0 + a1;
k3 = h2;
```

What are the resulting issues? undefined!

How to stitch optimized code back into the program

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = b0 + c1;
```

```
label_0:
h2 = g0 + a1;
k3 = a1 + g0;
```

optimize

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
```

stitch
part 1: *assign original variables their latest values*

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
g = g6;
d = d5
a = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
h = h2;
k = k3;
```

How to stitch optimized code back into the program

make room on slide

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
g = g6;
d = d5
a = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
h = h2;
k = k3;
```

what else needs to be done?

How to stitch optimized code back into the program

stitch part 2: drop numbers from first use of variables

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
g = g6;
d = d5
a = a2;
```

```
a2 = b + c;
d5 = e + f;
g6 = a2;
g = g6;
d = d5
a = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
h = h2;
k = k3;
```

```
label_0:
h2 = g + a;
k3 = h2;
h = h2;
k = k3;
```

How to stitch optimized code back into the program

Now they can be combined

```
a2 = b0 + c1;
d5 = e3 + f4;
g6 = a2;
g = g6;
d = d5
a = a2;
```

```
label_0:
h2 = g0 + a1;
k3 = h2;
h = h2;
k = k3;
```

```
a2 = b + c;
d5 = e + f;
g6 = a2;
g = g6;
d = d5
a = a2;
```

```
label_0:
h2 = g + a;
k3 = h2;
h = h2;
k = k3;
```

```
a2 = b + c;
d5 = e + f;
g6 = a2;
g = g6;
d = d5
a = a2;
label_0:
h2 = g + a;
k3 = h2;
h = h2;
k = k3;
```

How to stitch optimized code back into the program

original

```
a = b + c;
d = e + f;
g = b + c;

label_0:
h = g + a;
k = a + g;
```

new

```
a2 = b + c;
d5 = e + f;
g6 = a2;
g = g6;
d = d5;
a = a2;
label_0:
h2 = g + a;
k3 = h2;
h = h2;
k = k3;
```

is it really optimized?

It looks a lot longer...

How to stitch optimized code back into the program

original

```
a = b + c;
d = e + f;
g = b + c;

label_0:
h = g + a;
k = a + g;
```

new

```
a2 = b + c;
d5 = e + f;
g6 = a2;
g = g6;
d = d5
a = a2;
label_0:
h2 = g + a;
k3 = h2;
h = h2;
k = k3;
```

is it really optimized?

Common pattern for code to get larger, but it will contain patterns that are easier optimize away

later passes will minimize copies

e.g. copy propagation eliminates repeats of variables.