the type of hazard for each edge.
Show the constraint graph for this code, indicating:

Review Problem 51
enough we’ll pretend there’s a CMOV instruction...

Note: ARM actually uses CSEL, a mux-like instruction instead. But, prediction important

W/Cond

```
0
  1
```

W/Cond

```
0
```

```
if cond
  move src to dest if cond i = 0
  move src to dest if cond == 0
```

```
if cond
  move src to dest if cond i = 0
  move src to dest if cond == 0
```

Cond

```
0
```

```
if cond
  move src to dest if cond i = 0
  move src to dest if cond == 0
```

```
if cond
  move src to dest if cond i = 0
  move src to dest if cond == 0
```

Cond

```
0
```

```
if cond
  move src to dest if cond i = 0
  move src to dest if cond == 0
```

```
if cond
  move src to dest if cond i = 0
  move src to dest if cond == 0
```

Cond

```
0
```

Predicated Instructions
First, no boundary no area

∑ = (x0, y0) - (x1, y1) - (x2, y2)

if

4. Sub x2, y2, x0, y0
   Sumx, Sumy
   Ox = Sumx
   Oy = Sumy
   C = Ox, Oy

5. Movw Sumx, x2, ox
   Movw Sumy, y2, oy
   End

3-4 recall to do the

Exceeds

Only

E0:

Add sum, sum

E0:

Sub sum, d

End

Subsum,
Contd

Else

Subsum,
Contd

Normal

if (d > 0)

else

sum = p;

p = a - b;

Predicated Instructions (cont.)

Abs value
ARM CSEL Instruction: The MUX

\[
\begin{align*}
\text{cond}: & \text{ EQ, NE, LT, etc.} \\
\text{SEL} & \text{ dest } \leftarrow \text{cond} \begin{cases} 
\text{then} & \text{src1} \\
\text{else} & \text{src2} 
\end{cases} \\
\text{SUM} & = \text{dest} + \text{sum}
\end{align*}
\]
\[ \text{Why}\ I\ LP\ ]

Greater pipelining means more hazards, delay slots

\[ \text{Why if we want a CPI > 1.0?} \]

Reduce CPI

\[ \text{What if we want a CPI > 1.0?} \]

Reduce clock period by heavy pipelining

Pipelining delay = \[ g + h + \text{cost} \]

Advanced processors optimize two factors:

\[ \text{Why I LP} \]
Divide datapath into multiple pipeline stages

Superpipelining

\[ M_{1+1} = \frac{1}{2} C + b \]