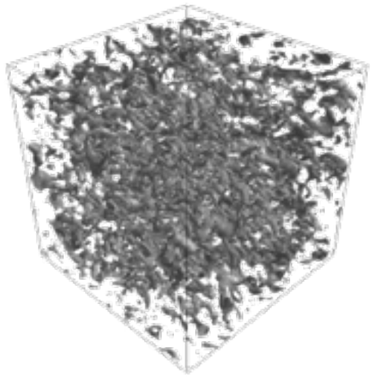
# Diagnosing Performance Limitations of the Microarchitecture

Georgia Tech College of Computing
Computational Science and Engineering

Kenneth Czechowski
Richard Vuduc

# Problem Statement

Identifying performance limitations



Turbulence Simulation

+


Intel Xeon 7560

= 6 hours

Why??

# Active Performance Analysis

Passive (Observational) vs Active (Experimental)

# Active Performance Analysis

Passive (Observational) vs **Active (Experimental)**

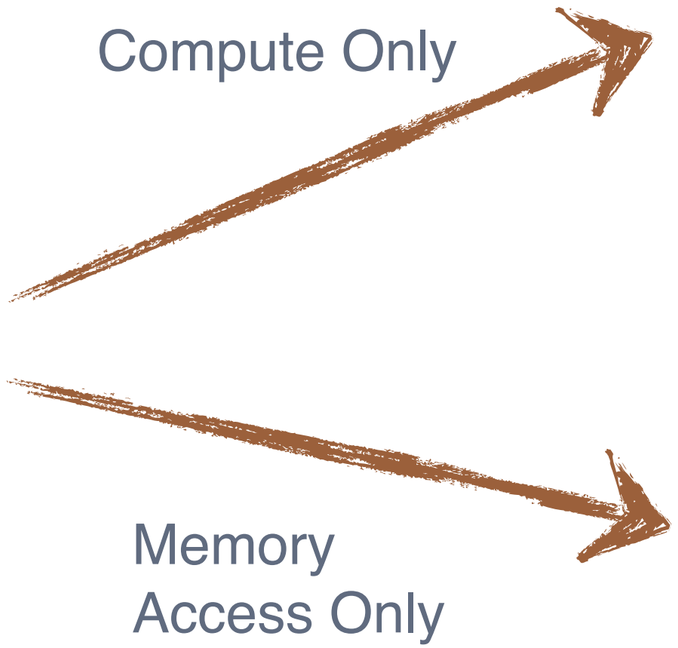
**Pressure Point Analysis** (PPA) rigorously tests the impact of microarchitectural bottlenecks

# PPA: Conceptual Example

Tri-Diagonal Elimination

```
for ( i=1 ; i<n ; i++ ) {
    x[i] = z[i]*( y[i] - x[i-1] );
}
```

```
vmovsd    xmm1, [8+rsi+r12]
vmovsd    xmm2, [16+rsi+r12]
vsubsd    xmm0, xmm1, xmm0
vmulsd    xmm3, xmm0, [8+rsi+rbp]
vmovsd    [8+rsi+r13], xmm3
vsubsd    xmm4, xmm2, xmm3
vmulsd    xmm0, xmm4, [16+rsi+rbp]
vmovsd    [16+rsi+r13], xmm0
```

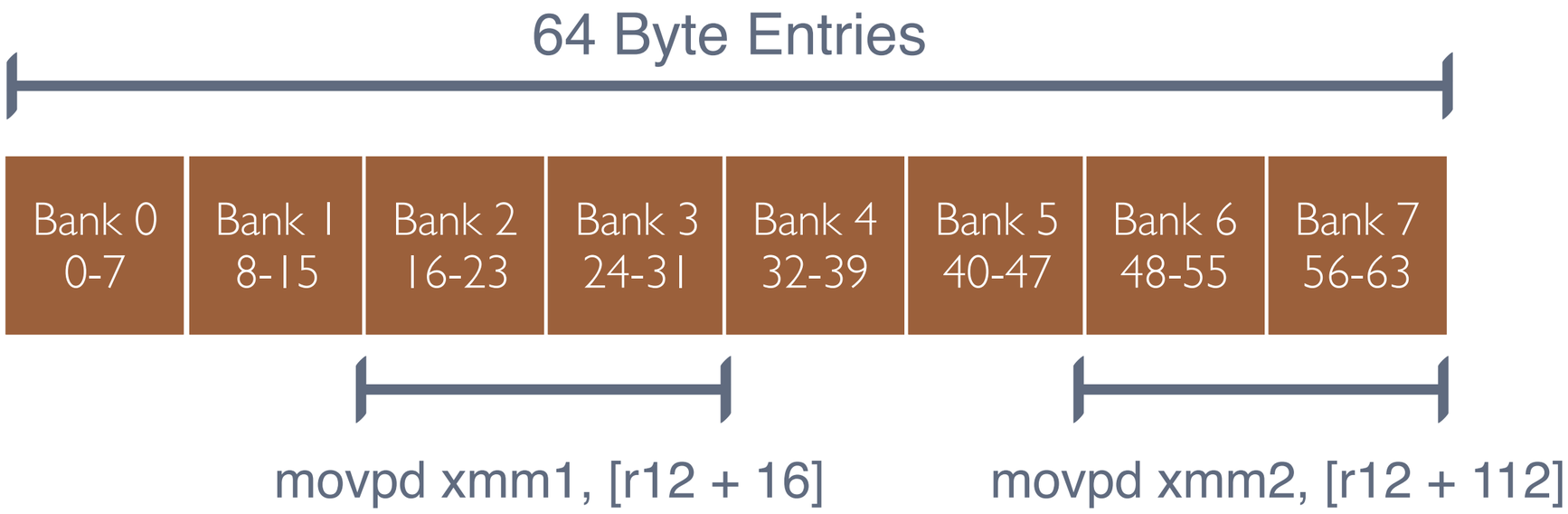



```
nop
nop
vsubsd    xmm0, xmm1, xmm0
vmulsd    xmm3, xmm0, xmm10
nop
vsubsd    xmm4, xmm2, xmm3
vmulsd    xmm0, xmm4, xmm12
nop
```

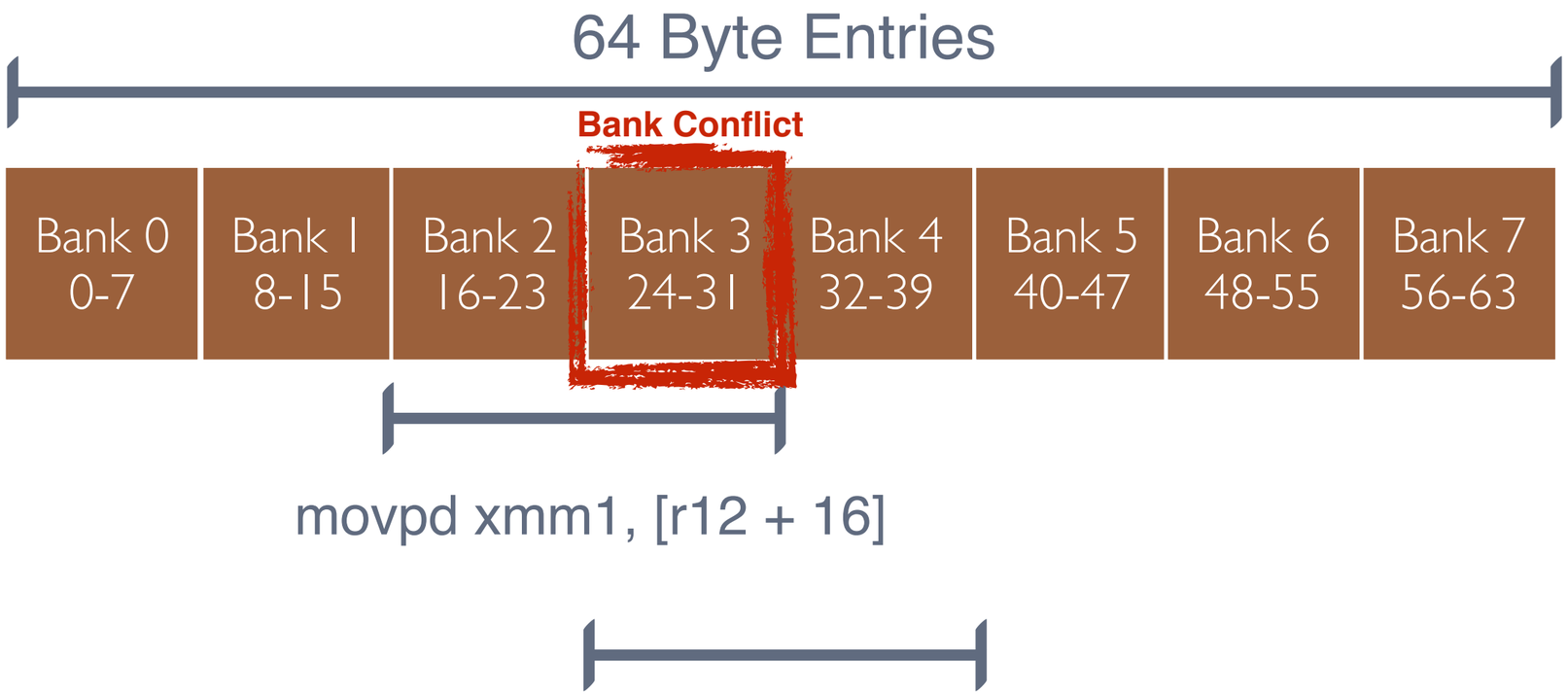
```
vmovsd    xmm1, [8+rsi+r12]
vmovsd    xmm2, [16+rsi+r12]
nop
vmovsd    xmm3, [8+rsi+rbp]
vmovsd    [8+rsi+r13], xmm3
nop
vmovsd    xmm0, [16+rsi+rbp]
vmovsd    [16+rsi+r13], xmm0
```

Perturbations do not need to preserve the semantic meaning

# Concrete Example: L1D Bank Conflicts



# Concrete Example: L1D Bank Conflicts



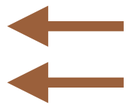
# CONCRETE EXAMPLE: L1D BANK CONFLICTS

| | | |
|---|---|---|
| [ **8** +rsi+r12] | -> | [**X**+rsi+r12] |
| [ **8** +rsi+r14] | -> | [**X**+rsi+r14] |
| [ **8** +rsi+rbp] | -> | [**X**+rsi+rbp] |
| [ **8** +rsi+r13] | -> | [**X**+rsi+r13] |
| [**16**+rsi+rbp] | -> | [**X**+rsi+rbp] |
| [**16**+rsi+r13] | -> | [**X**+rsi+r13] |

*Assume rsi, r12, r13, r14, and rbp are 64-byte aligned

Original

```
vmovsd   xmm1, [8+rsi+r12]
vmovsd   xmm2, [8+rsi+r14]
vsubsd   xmm0, xmm1, xmm0
vmulsd   xmm3, xmm0, [8+rsi+rbp]
vmovsd   [8+rsi+r13], xmm3
vsubsd   xmm4, xmm2, xmm3
vmulsd   xmm0, xmm4, [16+rsi+rbp]
vmovsd   [16+rsi+r13], xmm0
```



Bank Conflicts ??

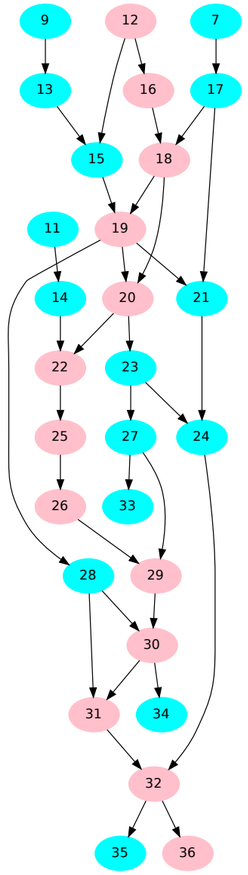
Perturbed Version

```
vmovsd   xmm1, [8+rsi+r12]
vmovsd   xmm2, [16+rsi+r14]
vsubsd   xmm0, xmm1, xmm0
vmulsd   xmm3, xmm0, [8+rsi+rbp]
vmovsd   [8+rsi+r13], xmm3
vsubsd   xmm4, xmm2, xmm3
vmulsd   xmm0, xmm4, [16+rsi+rbp]
vmovsd   [16+rsi+r13], xmm0
```

# IDENTIFYING OOO-DEFICIENCIES

## Original

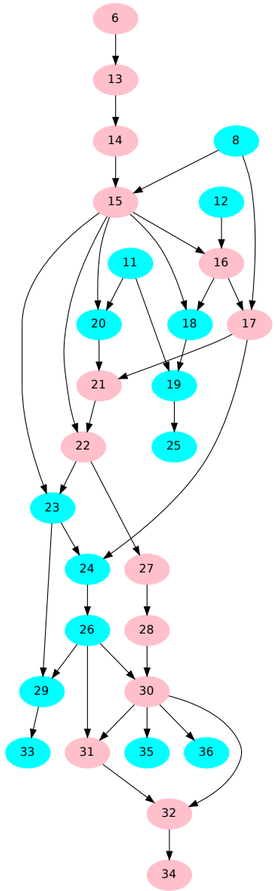
```
0  inloop:
1  movsd     xmm1, [88+r12+r9*8]
2  movsd     xmm1, [104+r12+r9*8]
3  movsd     xmm2, [120+r12+r9*8]
4  movsd     xmm2, [136+r12+r9*8]
5  movaps    xmm0, [80+r12+r9*8]
6  movhpd    xmm1, [96+r12+r9*8]
7  movaps    xmm2, [96+r12+r9*8]
8  movhpd    xmm3, [112+r12+r9*8]
9  movaps    xmm1, [112+r12+r9*8]
10 movhpd    xmm0, [128+r12+r9*8]
11 movaps    xmm0, [128+r12+r9*8]
12 movhpd    xmm3, [144+r12+r9*8]
13 mulpd     xmm1, xmm1
14 mulpd     xmm0, xmm0
15 mulpd     xmm1, xmm3
16 mulpd     xmm3, xmm3
17 mulpd     xmm2, xmm2
18 mulpd     xmm3, xmm2
19 mulpd     xmm1, xmm3
20 mulpd     xmm3, xmm1
21 addpd     xmm2, xmm1
22 addpd     xmm0, xmm3
23 addpd     xmm3, xmm3
24 addpd     xmm2, xmm3
25 mulpd     xmm0, [r15+r9*8]
26 mulpd     xmm0, [16+r15+r9*8]
27 mulpd     xmm3, [32+r15+r9*8]
28 mulpd     xmm1, [48+r15+r9*8]
29 addpd     xmm0, xmm3
30 addpd     xmm0, xmm1
31 addpd     xmm1, xmm0
32 addpd     xmm1, xmm2
33 movaps    [r11+r9*8], xmm3
34 movaps    [16+r11+r9*8], xmm0
35 movaps    [32+r11+r9*8], xmm1
36 movaps    [48+r11+r9*8], xmm1
37 add       r8, 1
38 cmp       r8, rbx
39 jb        inloop
```




Cycles per Iteration: **31.51 cycles**

## Scrambled

```
0  inloop:
1  movsd     xmm2, [88+r12+r9*8]
2  movsd     xmm0, [104+r12+r9*8]
3  movsd     xmm0, [120+r12+r9*8]
4  movsd     xmm3, [136+r12+r9*8]
5  movaps    xmm0, [80+r12+r9*8]
6  movhpd    xmm3, [96+r12+r9*8]
7  movaps    xmm0, [96+r12+r9*8]
8  movhpd    xmm2, [112+r12+r9*8]
9  movaps    xmm1, [112+r12+r9*8]
10 movhpd    xmm0, [128+r12+r9*8]
11 movaps    xmm1, [128+r12+r9*8]
12 movhpd    xmm0, [144+r12+r9*8]
13 mulpd     xmm3, xmm3
14 mulpd     xmm3, xmm3
15 mulpd     xmm3, xmm2
16 mulpd     xmm0, xmm3
17 mulpd     xmm2, xmm0
18 mulpd     xmm0, xmm3
19 mulpd     xmm0, xmm1
20 mulpd     xmm1, xmm3
21 addpd     xmm1, xmm2
22 addpd     xmm1, xmm3
23 addpd     xmm3, xmm1
24 addpd     xmm2, xmm3
25 mulpd     xmm0, [r15+r9*8]
26 mulpd     xmm2, [16+r15+r9*8]
27 mulpd     xmm1, [32+r15+r9*8]
28 mulpd     xmm1, [48+r15+r9*8]
29 addpd     xmm3, xmm2
30 addpd     xmm1, xmm2
31 addpd     xmm2, xmm1
32 addpd     xmm2, xmm1
33 movaps    [r11+r9*8], xmm3
34 movaps    [16+r11+r9*8], xmm2
35 movaps    [32+r11+r9*8], xmm1
36 movaps    [48+r11+r9*8], xmm1
37 add       r8, 1
39 cmp       r8, rbx
39 jb        inloop
```

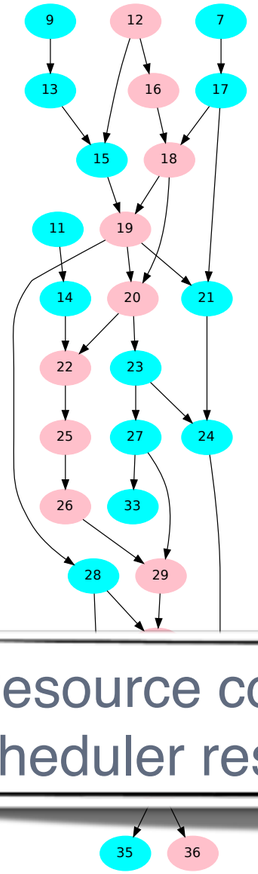



Cycles per Iteration: **19.65 cycles**

# IDENTIFYING OOO-DEFICIENCIES

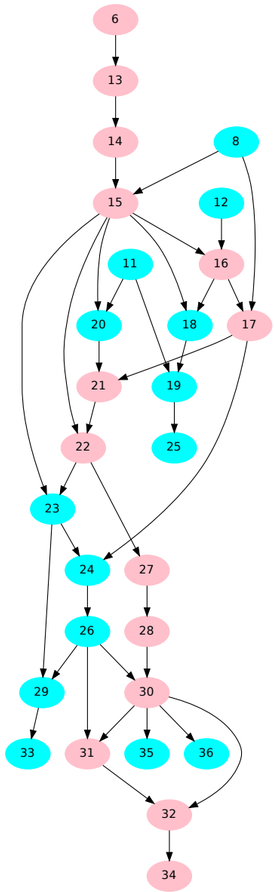
## Original

```
0  inloop:
1  movsd     xmm1, [88+r12+r9*8]
2  movsd     xmm1, [104+r12+r9*8]
3  movsd     xmm2, [120+r12+r9*8]
4  movsd     xmm2, [136+r12+r9*8]
5  movaps    xmm0, [80+r12+r9*8]
6  movhpd    xmm1, [96+r12+r9*8]
7  movaps    xmm2, [96+r12+r9*8]
8  movhpd    xmm3, [112+r12+r9*8]
9  movaps    xmm1, [112+r12+r9*8]
10 movhpd    xmm0, [128+r12+r9*8]
11 movaps    xmm0, [128+r12+r9*8]
12 movhpd    xmm3, [144+r12+r9*8]
13 mulpd     xmm1, xmm1
14 mulpd     xmm0, xmm0
15 mulpd     xmm1, xmm3
16 mulpd     xmm3, xmm3
17 mulpd     xmm2, xmm2
18 mulpd     xmm3, xmm2
19 mulpd     xmm1, xmm3
20 mulpd     xmm3, xmm1
21 addpd     xmm2, xmm1
22 addpd     xmm0, xmm3
23 addpd     xmm3, xmm3
24 addpd     xmm2, xmm3
25 mulpd     xmm0, [r15+r9*8]
26 mulpd     xmm0, [16+r15+r9*8]
27 mulpd     xmm3, [32+r15+r9*8]
28 mulpd     xmm1, [48+r15+r9*8]
29 addpd     xmm0, xmm3
30 addpd     xmm0, xmm1
31 addpd     xmm1, xmm0
32 addpd     xmm1, xmm2
33 movaps    [r11+r9*8],
34 movaps    [16+r11+r9*
35 movaps    [32+r11+r9*
36 movaps    [48+r11+r9*
37 add       r8, 1
38 cmp       r8, rbx
39 jb        inloop
```




## Scrambled

```
0  inloop:
1  movsd     xmm2, [88+r12+r9*8]
2  movsd     xmm0, [104+r12+r9*8]
3  movsd     xmm0, [120+r12+r9*8]
4  movsd     xmm3, [136+r12+r9*8]
5  movaps    xmm0, [80+r12+r9*8]
6  movhpd    xmm3, [96+r12+r9*8]
7  movaps    xmm0, [96+r12+r9*8]
8  movhpd    xmm2, [112+r12+r9*8]
9  movaps    xmm1, [112+r12+r9*8]
10 movhpd    xmm0, [128+r12+r9*8]
11 movaps    xmm1, [128+r12+r9*8]
12 movhpd    xmm0, [144+r12+r9*8]
13 mulpd     xmm3, xmm3
14 mulpd     xmm3, xmm3
15 mulpd     xmm3, xmm2
16 mulpd     xmm0, xmm3
17 mulpd     xmm2, xmm0
18 mulpd     xmm0, xmm3
19 mulpd     xmm0, xmm1
20 mulpd     xmm1, xmm3
21 addpd     xmm1, xmm2
22 addpd     xmm1, xmm3
23 addpd     xmm3, xmm1
24 addpd     xmm2, xmm3
25 mulpd     xmm0, [r15+r9*8]
26 mulpd     xmm2, [16+r15+r9*8]
27 mulpd     xmm1, [32+r15+r9*8]
28 mulpd     xmm1, [48+r15+r9*8]
37 add       r8, 1
39 cmp       r8, rbx
39 jb        inloop
```




Resource conflicts in the Out-of-Order scheduler result in a 12 cycle slowdown

Cycles per Iteration: **31.51 cycles**

Cycles per Iteration: **19.65 cycles**

# Our Vision for Performance Analysis

Can we account for all lost cycles?

```
for ( k=0 ; k<n ; k++ ) {
    x[k] = u[k] + r*( z[k] + r*y[k] ) +
        t*( u[k+3] + r*( u[k+2] + r*u[k+1] ) +
        t*( u[k+6] + r*( u[k+5] + r*u[k+4] ) ) );
}
```

Automated battery of experiments

- Frontend bottlenecks
- Scheduling resource conflicts
- Data bypass delays
- Cache latency stalls
- Memory disambiguation conflicts
- Retirement bandwidth

# Conclusion / Summary

**Major Contribution**: Active Performance Analysis

**Status**: Proof of concept

**Gaps**:

- Comprehensive set of experiments
- Scale beyond the core
- Generalize to additional microarchitectures

**Cross-Pollination**:

- Software optimization
- Autotuning and super-optimizing compilers
- Hardware-software codesign

# Questions