

Visualization of Data Movements and Accesses

Til Mohr

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Matrix in Memory:

1	2	3	4
---	---	---	---

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

--	--

Number of cache misses: 0

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: 1

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: 1

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

3	4
---	---

Number of cache misses: **2**

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	1
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

3	4
---	---

Number of cache misses: **2**

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	1
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: **3**

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	1
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: **3**

Sum of Matrix Elements

Listing 1: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for column in 0..2 {
4     for row in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	1
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

3	4
---	---

Number of cache misses: 4

Sum of Matrix Elements - Reordered Loops

Listing 2: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for row in 0..2 {
4     for column in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

--	--

Number of cache misses: 0

Sum of Matrix Elements - Reordered Loops

Listing 2: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for row in 0..2 {
4     for column in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: 1

Sum of Matrix Elements - Reordered Loops

Listing 2: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for row in 0..2 {
4     for column in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

0	1
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: 1

Sum of Matrix Elements - Reordered Loops

Listing 2: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for row in 0..2 {
4     for column in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

1	2
---	---

Number of cache misses: 1

Sum of Matrix Elements - Reordered Loops

Listing 2: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for row in 0..2 {
4     for column in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	0
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

3	4
---	---

Number of cache misses: **2**

Sum of Matrix Elements - Reordered Loops

Listing 2: Matrix Summation

```
1 let matrix = Matrix::random(2, 2);
2 let mut sum = 0;
3 for row in 0..2 {
4     for column in 0..2 {
5         sum += matrix.get(row, column);
6     }
7 }
8 sum
```

Matrix:

	0	1
0	1	2
1	3	4

Current Item:

1	1
---	---

Matrix in Memory:

1	2	3	4
---	---	---	---

Cache:

3	4
---	---

Number of cache misses: **2**

Outline

- Memory-Related Performance Problems
 - Data Locality
 - Processor-Memory Performance Gap
- Overview of the Optimization Workflow

Memory-Related Performance Problems | Data Locality

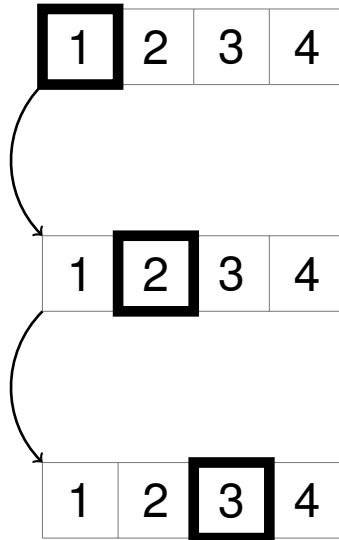
$$t_{avg} = p \cdot t_c + (1 - p) \cdot t_m \quad (1)$$

t_{avg} :	average access time
p :	cache hit percentage
t_c :	cache access time
t_m :	memory access time

$$t_c \ll t_m \quad (2)$$

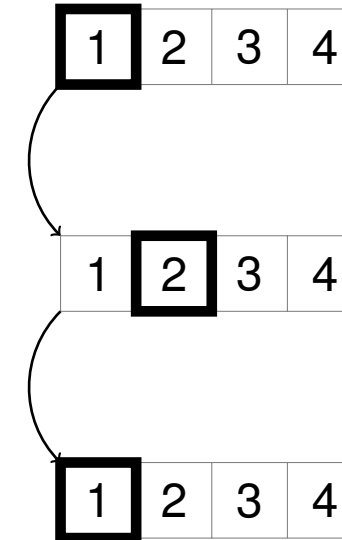
Spatial Locality

- Data that is referenced spatially close together is likely to be accessed in the near future

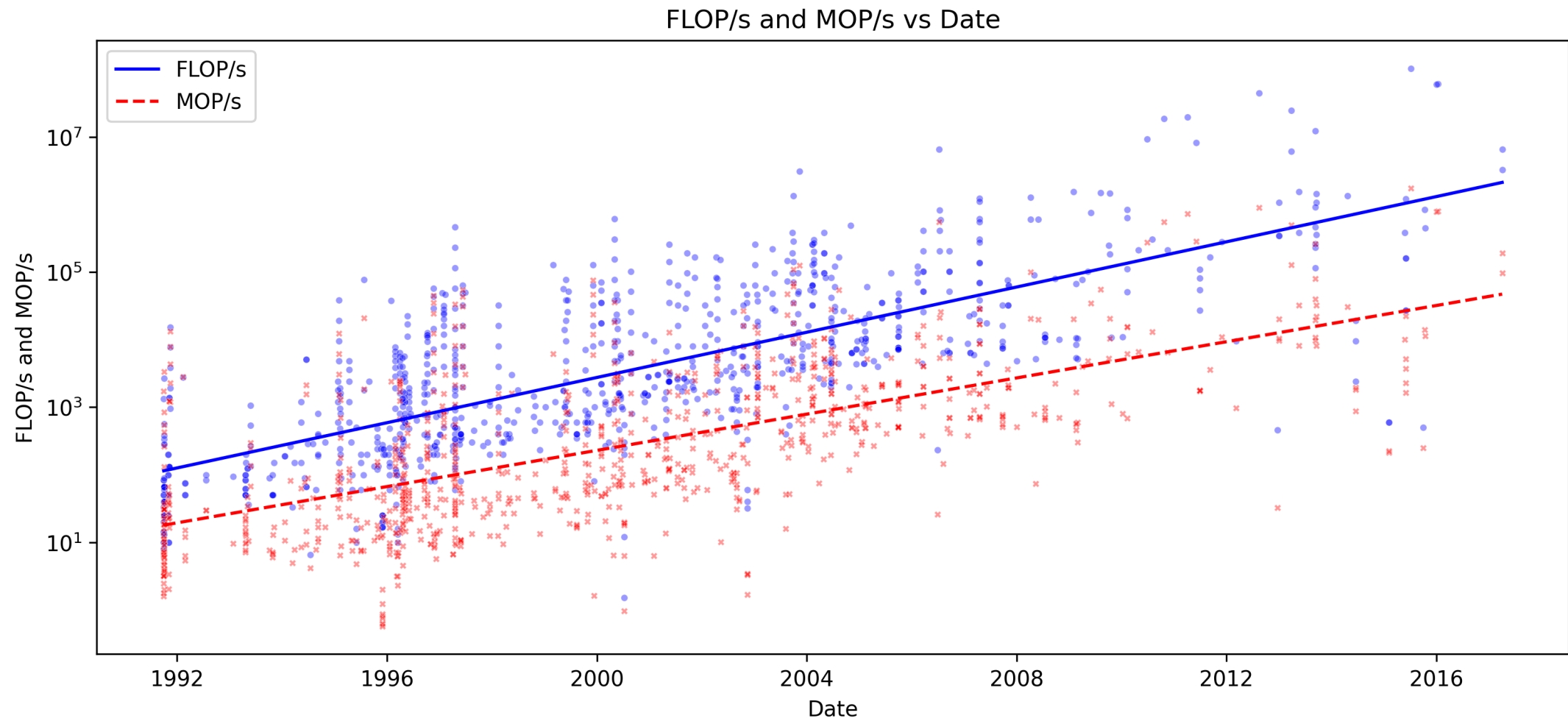


Temporal Locality

- Data that is referenced in the near past is likely to be accessed in the near future



Memory-Related Performance Problems | Processor-Memory Performance Gap



Visualization-Guided Optimization