Project 1.1
Analytical Models of Locality within Matrix Multiply Algorithms

Emily Bragg, Cagri Eryilmaz, Ali Fakhrzadehgan

January 30, 2015

1 Introduction

2 Single Level Analytical Models

2.1 Assumptions

In this model there is only level of cache (L1) and there is no Register File in the architecture, so that the processor only operates on the data in memory, which obviously if it is present in L1 (Hit) it can be used directly and if it is not (Miss) it should be gathered first.
Cache organization is considered to be a *fully associative*, ideal *LRU* replacement policy, with cache-line/blocks size of L and total cache size is Z.
The analytical metric used here is the Arithmetic Intensity of the Matrix Multiplication. Basically, the multiplication algorithm is the well-known $O(N^3)$ algorithm and in each step we are trying to optimize this procedure. Notice that, optimization can be done both on computation complexity and cache complexity of the algorithm, but here the main focus is to improve the cache complexity.

2.2 Multiplication Models

2.2.1 Baseline Matrix Multiplication Algorithm

As it is provided in the attachments of this report, baseline core computation of matrix multiply looks like below.

Algorithm 1 Baseline Matrix Multiplication

```
for i = 0; i < N; i++ do
    for j = 0; j < N; j++ do
        C[i][j] = 0;
        for k = 0; k < N; k++ do
            C[i][j] += A[i][k] × B[k][j];
        end for
    end for
end for
```

By looking the Algorithm 1, the asymptotic computation complexity could be figured out, which is $O(N^3)$. However, since we need a more accurate metric here, we consider total number of arithmetic operations as $2N^3$ [1]. For this point on, since core of the computation is not changed, we consider number of operations to be the same.
Now we focus on the cache complexity of the computation. If we start from the third loop (the most inner one), we can see there are three data access. However, between these three, hopefully $C[i][j]$ is almost always present in the cache. This statement is true because before entering this loop, $C[i][j]$ is cached and it will resident for this whole loop according to the

[1] One + and one ×.

replacement policy[2].

An important thing about the A is, it is being traced in a row-major order, hence we can utilize the locality in each cache-line. Since each row of A is N words, it will require $\frac{N}{L}$ transfers of cache-lines.

Matrix B is being trace in the column-major in this algorithm and since we are considering matrixes to be large in comparison with cache-line and cache size, there is almost no chance to observe any locality between consecutive access. Therefor, in each iteration B will experience almost N misses.

Putting it all together, B will produce total number of N^3 and A will produce $\frac{N^3}{L}$ misses. The important thing about C is that it being accessed in the second loop (the middle one) and begin traced in row-major order. Consequently, the total number of misses from C would be $\frac{N^2}{L}$. Using the arithmetic intensity of the operations, we will get:

$$\text{Arithmetic Intensity} = \frac{2N^3}{\frac{N^3}{L} + N^3 + \frac{N^2}{L}}$$

2.2.2 Reordered Multiplication and Using Transposition

2.2.3 Partitioning the Matrix to specified sub-matrixes

2.2.4 Divide & Conquer

2.3 Graphs

3 Model with Two-Level Cache Hierarchy

3.1 Assumptions

3.2 Model

3.3 Graphs

4 Model with Two-Level Cache Hierarchy and Register File

4.1 Assumptions

4.2 Model

4.3 Graphs

$$\begin{aligned}
(x+y)^3 &= (x+y)^2(x+y) \\
&= (x^2+2xy+y^2)(x+y) \\
&= (x^3+2x^2y+xy^2)+(x^2y+2xy^2+y^3) \\
&= x^3+3x^2y+3xy^2+y^3
\end{aligned} \tag{4.1}$$

[2] Since every assignment in this loop is addressing $C[i][j]$ repeatedly, so LRU will not remove it from the cache.

Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.

$$A = \begin{bmatrix} A_{11} & A_{21} \\ A_{21} & A_{22} \end{bmatrix} \tag{4.2}$$

Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.

4.3.1 Heading on level 3 (subsubsection)

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Heading on level 4 (paragraph) Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

5 Lists

5.1 Example of list (3*itemize)

- First item in a list
  - First item in a list
    - First item in a list
    - Second item in a list
  - Second item in a list
- Second item in a list

5.2 Example of list (enumerate)

1. First item in a list
2. Second item in a list
3. Third item in a list