Lab2 Performance Lab And Bonus Lab

DDL: 10/4/2021

Part I

Strassen-Winograd 算法

实现一个串行的 Strassen-Winograd 算法（Strassen 矩阵乘的一个变形）。要求，从命令行通过 argc/argv 读入矩阵规模 n，计算结果与一个普通矩阵乘进行结果比较和检验。

算法描述如下：

0.1

将输入矩阵 A、B 和输出矩阵 C 分割成 4 块:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

0.2

用 A、B 子矩阵计算 T_i 和 S_i，T_i 是 A 的 4 个子矩阵的线性组合，S_i 是 B 的 4 个子矩阵的线性组合；将 T_i 和 S_i 逐个相乘，得到 Q_i，C 的 4 个

子矩阵就是 Q_i 的线性组合：

$$
\begin{array}{llll}
T_0 = A_{11} & S_0 = B_{11} & Q_0 = T_0 \cdot S_0 & U_1 = Q_0 + Q_3 \\
T_1 = A_{12} & S_1 = B_{21} & Q_1 = T_1 \cdot S_1 & U_2 = U_1 + Q_4 \\
T_2 = A_{21} + A_{22} & S_2 = B_{12} - B_{11} & Q_2 = T_2 \cdot S_2 & U_3 = U_1 + Q_2 \\
T_3 = T_2 - A_{11} & S_3 = B_{22} - S_2 & Q_3 = T_3 \cdot S_3 & C_{11} = Q_0 + Q_1 \\
T_4 = A_{11} - A_{21} & S_4 = B_{22} - B_{12} & Q_4 = T_4 \cdot S_4 & C_{12} = U_3 + Q_5 \\
T_5 = A_{12} - T_3 & S_5 = B_{22} & Q_5 = T_5 \cdot S_5 & C_{21} = U_2 - Q_6 \\
T_6 = A_{22} & S_6 = S_3 - B_{21} & Q_6 = T_6 \cdot S_6 & C_{22} = U_2 + Q_2
\end{array}
$$

0.3

Strassen-Winograd 算法是一个递归算法，对 7 个小矩阵的乘积，递归调用以上步骤。

算法框架： 附件: *cppstrawino.cpp*

要求 在给出的算法框架中，修改//to do... 部分，其余部分不变。

评分 1. 保证正确实现 2. 在正确实现的基础上，使用尽可能多的优化。以最后性能的高低排名并给分.

Part II

bonus lab（选做）DDL:3/5/2021

实现一个能够自动测试以下表格中所有 L/K 组合的程序。可以考虑用 C++ template meta-programming 实现。

Unrolling & Accumulating: Double *

- **Case**
  - Intel Haswell
  - Double FP Multiplication
  - Latency bound: 5.00. Throughput bound: 0.50

FP *	Unrolling Factor L							
K	1	2	3	4	6	8	10	12
1	5.01	5.01	5.01	5.01	5.01	5.01	5.01	
2		2.51		2.51		2.51		
3			1.67					
4				1.25		1.26		
6					0.84			0.88
8						0.63		
10							0.51	
12								0.52

Accumulators