CEG5201 Hardware Technologies, Principles, & Platforms
(Semester I, AY2023/2024)
CA-2 Statement

Release date: 25-27 Sept, 2023
Consultation date: **16th Oct 2023**
Submission deadline: 13rd Nov 2023 11:59pm

Instructions to candidates:

The CA2 Canvas submission deadline is midnight as shown above. No late submission is considered, try NOT to do last-minute submissions to prevent traffic congestion.

I. Objective

In this assignment, we will attempt to benchmark various matrix multiplication algorithms across sequential and multi-processinf methods. To achieve this goal, you need to familiarize yourself with the Python multiprocessing package. Please read through the following materials first, which introduce the basics of the package.

The pool class - https://superfastpython.com/multiprocessing-pool-python/

Multiprocessing pool example - https://superfastpython.com/multiprocessing-pool-example/

Threadpool vs. pool class differences - https://superfastpython.com/threadpool-vs-pool-in-python

You may also access other useful sites, if any, to learn. Do share such new links by posting on the Discussion on Canvas so that it will be useful for your classmates.

II. Project overview

We will attempt to implement matrix multiplication through four different algorithms (**Strassen's algorithm** [1], **Coppersmith–Winograd algorithm** [2], **Cannon's algorithm** [3], and **Fox's algorithm** [4]) in this project, and each team member focuses on only one algorithm and implement in both sequential and multi-process ways. The project is composed of individual tasks and joint tasks as shown in Table I. Blue shaded parts in the table are individual tasks and yellow shaded parts are joint tasks.

Table I. Overview of the tasks

Task ID	Task description
A	Matrices generation
B-n	Sequential implementation
C-n	Multiprocessing implementation
D-n	Determining the ultimate speed-up

E	Comparison of the algorithms

"n" in Task B, C, and D stands for each team member is calculated as **"(group ID + member ID) %4 +1"** ("%" here stands for the modulo operator), and the algorithm corresponding to "n" is listed in Table II. Please refer to Appendix A for general ideas of the four algorithms.

Table II. List of the algorithms used

S/N (n)	Algorithm
1	Strassen's algorithm
2	Coppersmith–Winograd algorithm
3	Cannon's algorithm
4	Fox's algorithm

III. Task description

[O] Getting started

As a team, first fix the hardware platform that you are going to use – multi-core CPU / GPU before doing individual coding.

[A] Matrix generation

Randomly generate **8 pairs** of matrices $(A_i,\ B_i)$, $i \in [0, 1, \dots 7]$ with $M_i \times M_i$ size where M= [16, 32, 64, 128, 256, 512, 1024, 2048]. The value of each element should be between 0 and 255 (integers only). Generate 10 groups of such matrix pairs G_j, $j \in [0, 1, \dots 9]$. All the team members need to use the same set of G for their subsequent tasks.

[B-n] Sequential Processing

[B-n.1] Perform multiplications for all 8 matrix pairs in G_0 sequentially. Record the time taken for processing each pair and the total time. This is the sequential processing time of G_0 and please present the obtained results in Table B-n.1 shown below. Please put the table in your report exactly as it appears.

Table B-n.1: **Processing time of G_0 under sequential implementation**

Pair Index	Measured Sequential Time	Measured Cumulative Sequential time
0	t_0	t_0
1	t_1	$t_0 + t_1$
2	t_2	$t_0 + t_1 + t_2$
…	…	…
7	t_7	$t_0 + t_1 + \cdots + t_7$

[B-n.2] Repeat the above processes for all the 10 groups. This is the sequential processing time of all groups and please present the results obtained in Table B-n.2 shown below. Please put the table in your report exactly as it appears.

Table B-n.2: **Processing time of all groups under sequential implementation**

Group Index	Measured Sequential Time	Measured Cumulative Sequential time
0	t_0	t_0
1	t_1	$t_0 + t_1$
2	t_2	$t_0 + t_1 + t_2$
…	…	…
9	t_9	$t_0 + t_1 + \cdots + t_9$

[C-n] Multiprocessing

Identify the bottleneck in the sequential processing and try to accelerate it using multiple processes.

[C-n.1] Record the time it takes to complete the processing of G_0 by varying the number of processes. Please present the obtained results in Table C-n.1 shown below. Please put the table in your report exactly as it appears.

Table C-n.1: Processing time of G_0 under multiprocessing implementation

	Measured MP time						Measured Cumulative MP Time					
Process → Pair Index↓	1	2	3	4	…	…	1	2	3	4	…	…
0	y1	z1					y1					
1	y2						y1+y2					
2												
3												
…												
7	y8	z8					x	m				

[C-n.2] Record the time it takes to complete the processing of all groups by varying the number of processes. Please present the obtained results in Table C-n.2 shown below. Please put the table in your report exactly as it appears.

Table C-n.2: Processing time of all groups under multiprocessing implementation

	Measured MP time						Measured Cumulative MP Time					
Process → Grp Index↓	1	2	3	4	…	…	1	2	3	4	…	…
0	x	m					x	m				
1	p	q					x+p	m+q				
2												
3												
…												
9							a	b	c	d		

[D-n] Determine the speed-up

[D-n.1] Determine the speed-up of processing G_0 under multiprocessing implementation (C-n.1) versus sequential implementation (B-n.1).

[D-n.2] Determine the speed-up of processing all groups under multiprocessing implementation (C-n.2) versus sequential implementation (B-n.2).

Plot the speed-up curves w.r.t the number of processes and the number of workloads (pairs/groups) and write a detailed discussion.

[E] Compare the algorithms

Use the results obtained by all team members and present a detailed comparison of the various algorithms in the report.

IV. Submission

What to submit?

Each group should upload **only one compressed file (.zip)** to *Canvas*, including three types of files: the report (.pdf), the source code file(s) (.py), and the readme text file (.txt).

Follow the rules below to name your files:

• For the compressed file:

CA2_(Group ID)_(Matriculation number of all team members separated by an underscore).zip

Example: CA2_12_A0213331Z_A0010101Y_A0340127X_A0711001X.zip

• For the report:

CA2_Report_(Group ID)_(Matriculation number of all team members separated by an underscore).pdf

Example: CA2_Report_12_ A0213331Z_A0010101Y_A0340127X_A0711001X.pdf

Report format

Report Formatting Guidelines – Please use single column format. The font of the main body should be Times New Roman 12pt.

Page Limit – Maximum 12 pages (excluding the first page and the last page)

First Page of your report – Title + Full Names + Matriculation numbers (this page is not counted in the report page limit mentioned below).

Last Page of your report – Title: Coding effort (this page is not counted in the report page limit mentioned below). Please see the next subsection on what to write here.

Code file(s) – Please put useful comments generously throughout in your code file for readers to understand the gist of computation taking place.

Coding effort

Attach ONE separate page titled "Coding Effort - <your name>" at the end of report. Describe how much is your coding effort (quantify in %). If you had either directly or partly used any CODE(s) from *github* or from any other site, list those sites and point us exactly where you obtained those codes. If you have used others codes and did not list or cite the reference directly no marks will be awarded. If you have generated your own data, describe how you generated your data needed for your experiments. If you are using any tools for your experiments (not for data), then point to them. These also can be part of your references listing after conclusions.

Readme file

This file must contain clear instructions to facilitate running your program and use of data and parameters. If you are using any specific packages, list them and also indicate the URLs from where the readers can download. If you are using any specific DATA from any URL, indicate the URL. If the readers need to set any input parameters (hard coded way) explicitly state that requirement. Do NOT ASSUME anything and omit details. Step-by-step instructions will be very useful.

V. Assessment

Grading

Your assignment will be graded out of 60 marks, and the final weight of this assignment is **35%.** For the joint effort, all members will be awarded identical marks. The mark breakdown is shown in Table III.

Table III. Grade breakdown

Joint effort (50%)	Effort in (A) and (E). Title, abstract, introduction, joint discussions, and conclusion.
Individual effort (50%)	Effort in (B), (C), and (D). Sections on interpreting individual results.

Plagiarism Penalty

You are allowed to use only 15% of any written material collectively from all other sources. Anything more than this 15% will be penalized. References cited may show up in your plagiarism checker and you can ignore this percentage. **Copying of coding and simulation results directly from others will be awarded 0 marks for both Source and Copier(s)**.

Rubrics

Criteria	Remarks	Points
[Joint] Abstract	Capture the algorithms and implementation approaches used and highlight of any significant results in not more than 12 lines.	2
[Joint] Introduction	Brief introduction of the algorithms and implementation approaches. Clear problem statement in plain English and also using technical description.	5
[Joint] Description of input data [A]	Describe your input matrix set, and how they are generated. If you are using any special packages, please indicate that clearly.	6
[Indiv] Description of algorithm and implementation approach [B, C]	Each one of you can open a separate sub-section in this part. Describe the algorithms in a concise fashion. Highlight any non-trivial decisions/steps that algorithm takes and use proper citations. You may use a simple numerical example to explain, if you wish. Describe clearly how you implement the multi-processing method. Elaborate how you identify the bottleneck and maximize the parallelism and acceleration enabled by the multi-processing	10

	approach.	
[Indiv] Individual results and discussions [B, C]	Each one of you can open a separate sub-section in this part. Put all your simulation results in a systematic fashion under each subsection, point to the results and argue on the trends, behaviour, results, etc, clearly. Clearly interpret and discuss your table / graphs and try to relate to algorithms described in the earlier section.	15
[Joint] Joint discussions [E]	This is the joint discussion under Task E. You may wish to open a common section and put your arguments/discussions/comments as a comparison of the algorithms, if it is valid. This demonstrates your group effort. Describe strengths and pitfalls of the algorithms, if any, clearly. Argue why algo(s) fails in certain conditions and what it does not consider, etc. You may describe here what you feel is right and wrong that is within the scope of formulations used in the paper.	12
[Joint] Conclusions & reference(s) used [E]	Conclude how the algorithms benefits/suffers from the multi-processing, and the potentials/limitations to the number of processes/workloads in not more than 12 lines. Formally list the paper(s) & sites you have directly used.	5
[Indiv] Coding effort	Report here your coding effort.	5
TOTAL MARKS		**60**

Appendix A

The following gives the general ideas of four common algorithms used to perform matrix multiplication. Here we consider $Z = Y \cdot X$.

1. Strassen's algorithm

Divide matrices X, Y, and Z into four parts,

$$\begin{bmatrix} Z_{11} & Z_{21} \\ Z_{21} & Z_{22} \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} \cdot \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix}$$

Compute the following,

$$M_1 = (X_{11} + X_{21}) \cdot (Y_{11} + Y_{12})$$
$$M_2 = (X_{12} + X_{22}) \cdot (Y_{21} + Y_{22})$$
$$M_3 = (X_{11} - X_{22}) \cdot (Y_{11} + Y_{22})$$
$$M_4 = X_{11} \cdot (Y_{12} - Y_{22})$$
$$M_5 = (X_{21} + X_{22}) \cdot Y_{11}$$
$$M_6 = (X_{11} + X_{12}) \cdot (Y_{22})$$
$$M_7 = X_{22} \cdot (Y_{21} - Y_{11})$$

Finally,

$$Z_{11} = M_2 + M_3 - M_6 - M_7$$
$$Z_{12} = M_4 + M_6$$
$$Z_{21} = M_5 + M_7$$
$$Z_{22} = M_1 - M_3 - M_4 - M_5$$

2. Coppersmith–Winograd algorithm

Divide matrices X, Y, and Z into four parts,

$$\begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} \cdot \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix}$$

Compute the following,

$$S_1 = X_{21} + X_{22}$$
$$S_2 = S_1 - X_{11}$$
$$S_3 = X_{11} - X_{21}$$
$$S_4 = X_{12} + S_2$$

$$T_1 = Y_{12} - Y_{11}$$
$$T_2 = Y_{22} - T_1$$
$$T_3 = Y_{22} - Y_{12}$$
$$T_4 = T_2 - Y_{21}$$

$$M_1 = X_{11} \cdot Y_{11}$$
$$M_2 = X_{12} \cdot Y_{21}$$
$$M_3 = S_4 \cdot Y_{22}$$
$$M_4 = X_{22} \cdot T_4$$
$$M_5 = S_1 \cdot T_1$$
$$M_6 = S_2 \cdot T_2$$
$$M_7 = S_3 \cdot T_3$$

Finally,

$$Z_{11} = M_1 + M_2$$
$$Z_{12} = M_1 + M_3 + M_5 + M_6$$
$$Z_{21} = M_7 - M_5$$
$$Z_{12} = M_1 + M_5 + M_6 + M_7$$

3. Cannon's algorithm

Here we divide matrices X, Y, and Z into nine parts for example, it will take three stages to perform the multiplication. Note that $\circ$ stand for element-wise product.

$$\begin{bmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix} \cdot \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} \\ Y_{21} & Y_{22} & Y_{23} \\ Y_{31} & Y_{32} & Y_{33} \end{bmatrix}$$

In stage 1,

$$M_1 = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{22} & X_{23} & X_{21} \\ X_{33} & X_{31} & X_{32} \end{bmatrix} \circ \begin{bmatrix} Y_{11} & Y_{22} & Y_{33} \\ Y_{21} & Y_{32} & Y_{13} \\ Y_{31} & Y_{12} & Y_{23} \end{bmatrix} = \begin{bmatrix} X_{11}Y_{11} & X_{12}Y_{22} & X_{13}Y_{33} \\ X_{22}Y_{21} & X_{23}Y_{32} & X_{21}Y_{13} \\ X_{33}Y_{31} & X_{31}Y_{12} & X_{32}Y_{23} \end{bmatrix}$$

In stage 2,

$$M_2 = \begin{bmatrix} X_{12} & X_{13} & X_{11} \\ X_{23} & X_{21} & X_{22} \\ X_{31} & X_{32} & X_{33} \end{bmatrix} \circ \begin{bmatrix} Y_{21} & Y_{32} & Y_{13} \\ Y_{31} & Y_{12} & Y_{23} \\ Y_{11} & Y_{22} & Y_{33} \end{bmatrix}$$

Stage 3,

$$M_3 = \begin{bmatrix} X_{13} & X_{11} & X_{12} \\ X_{21} & X_{22} & X_{23} \\ X_{32} & X_{33} & X_{31} \end{bmatrix} \circ \begin{bmatrix} Y_{31} & Y_{12} & Y_{23} \\ Y_{11} & Y_{22} & Y_{33} \\ Y_{21} & Y_{32} & Y_{13} \end{bmatrix}$$

Finally,

$$Z = M_1 + M_2 + M_3$$

4. Fox's algorithm

Here we divide matrices X, Y, and Z into nine parts for example, it will take three stages to perform the multiplication.

$$\begin{bmatrix} Z_{11} & Z_{12} & Z_{13} \\ Z_{21} & Z_{22} & Z_{23} \\ Z_{31} & Z_{32} & Z_{33} \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix} \cdot \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} \\ Y_{21} & Y_{22} & Y_{23} \\ Y_{31} & Y_{32} & Y_{33} \end{bmatrix}$$

In stage 1,

$$M_1 = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix} \circ \begin{bmatrix} Y_{11} & Y_{22} & Y_{33} \\ Y_{11} & Y_{22} & Y_{33} \\ Y_{11} & Y_{22} & Y_{33} \end{bmatrix} = \begin{bmatrix} X_{11}Y_{11} & X_{12}Y_{22} & X_{13}Y_{33} \\ X_{21}Y_{11} & X_{22}Y_{22} & X_{23}Y_{33} \\ X_{31}Y_{11} & X_{32}Y_{22} & X_{33}Y_{33} \end{bmatrix}$$

In stage 2,

$$M_2 = \begin{bmatrix} X_{12} & X_{13} & X_{11} \\ X_{22} & X_{23} & X_{21} \\ X_{32} & X_{33} & X_{31} \end{bmatrix} \circ \begin{bmatrix} Y_{21} & Y_{32} & Y_{13} \\ Y_{21} & Y_{32} & Y_{13} \\ Y_{21} & Y_{32} & Y_{13} \end{bmatrix}$$

In stage 3,

$$M_3 = \begin{bmatrix} X_{13} & X_{11} & X_{12} \\ X_{23} & X_{21} & X_{22} \\ X_{33} & X_{31} & X_{32} \end{bmatrix} \circ \begin{bmatrix} Y_{31} & Y_{12} & Y_{23} \\ Y_{31} & Y_{12} & Y_{23} \\ Y_{31} & Y_{12} & Y_{23} \end{bmatrix}$$

Finally,

$$Z = M_1 + M_2 + M_3$$

References [*If you wish to do any additional reading you may use these references*]

[1] V. Strassen, "Gaussian elimination is not optimal," Numer Math (Heidelb), vol. 13, no. 4, pp. 354–356, 1969, doi: 10.1007/BF02165411.

[2] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," J Symb Comput, vol. 9, no. 3, pp. 251–280, 1990, doi: https://doi.org/10.1016/S0747-7171(08)80013-2.

[3] L. E. Cannon, "A Cellular Computer to Implement the Kalman Filter Algorithm," PhD Thesis, Montana State University, Bozeman, Montana, USA, 1969, [Online]. Available: https://scholarworks.montana.edu/xmlui/bitstream/handle/1/4168/31762100054244.pdf?sequence=1.

[4] G. C. Fox, S. W. Otto, and A. J. G. Hey, "Matrix algorithms on a hypercube I: Matrix multiplication," Parallel Comput, vol. 4, no. 1, pp. 17–31, 1987, doi: https://doi.org/10.1016/0167-8191(87)90060-3.

----- END -----