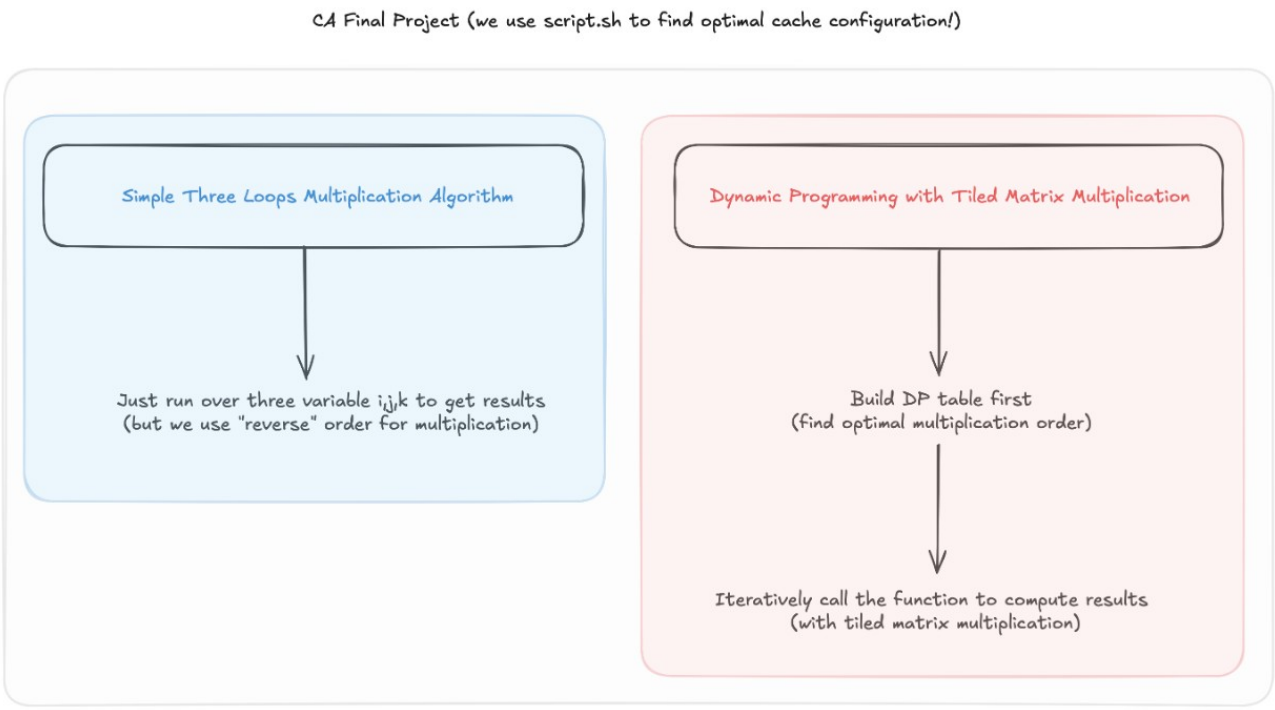
計算機結構期末報告

鄭宇彥 江履方
B12901035 B12901140

一、簡介

在這個專案中，我們提供了兩種方法去實作本次的程式碼，並且都能夠突破各項要求。我們提出了包含使用優化 malloc 次數的 Three Loops 演算法，以及有更好表現的 Dynamic Programming 加上 Tiled Matrix Multiplication 之演算法。在範例測資第五題中，藉由自動化腳本尋找，兩種方法都可以在使用最佳快取組態 (optimal cache configuration) 的情況下，取得足夠好的評分。



(a) Design overview, we propose two methods to pass the strong baseline

二、演算法設計與分析

由於本專案希望設計一個演算法，能利用較少的時間與較少的快取大小，去執行矩陣鏈乘法，所以在實作矩陣鏈乘法的最佳順序 Dynamic Programming 演算法，以及硬體友善的 Tiled Matrix Multiplication 演算法之前，我們嘗試了利用自動化腳本直接尋找 Three Loops 演算法的最佳快取組態。

我們發現，即便是在最佳組態的情況下，使用單純的 Three Loops 演算法，仍然無法突破所有要求。然而當我們將乘法順序完全反過來時，剛好在範例測資第五題中，比起直接照順序乘起來，反過來乘會更靠近最佳的矩陣鏈乘法順序。在此情況下，我們將 Three Loops 演算法程式優化成只需兩次 malloc 就可以執行的版本，可以在最佳快取組態下通過所有標準。

除此之外，爲了能夠兼容更大的輸入情況，以及自動尋找最佳的矩陣鏈乘法順序，我們也設計了利用 Dynamic Programming 與 Tiled Matrix Multiplication 的演算法程式。在我們嘗試過後，使用 Tile size $=$ 32 的最佳快取組態演算法，可以達到更加好的表現，以下爲兩種算法分別取得的成果：

```
Testbench executed
python3 testbench.py
P0 succeed
P1 succeed
P2 succeed
P3 succeed
P4 succeed
P5 succeed
Scoring with performance testcases
python3 score.py
L1DCache Size: 4096
L1ICache Size: 4096
L2Cache Size: 1024
Test Case 5 Execution Time: 2298592 ns
Score: 66659168.0
```

(b) Three loops algorithm with reverse order, cache associativity $= 8$

```
Testbench executed
python3 testbench.py
P0 succeed
P1 succeed
P2 succeed
P3 succeed
P4 succeed
P5 succeed
Scoring with performance testcases
python3 score.py
L1DCache Size: 2048
L1ICache Size: 4096
L2Cache Size: 1024
Test Case 5 Execution Time: 2307906 ns
Score: 64621368.0
```

(c) Dynamic programming with tiled matrix multiplication, cache associativity $= 8$

三、團隊分工

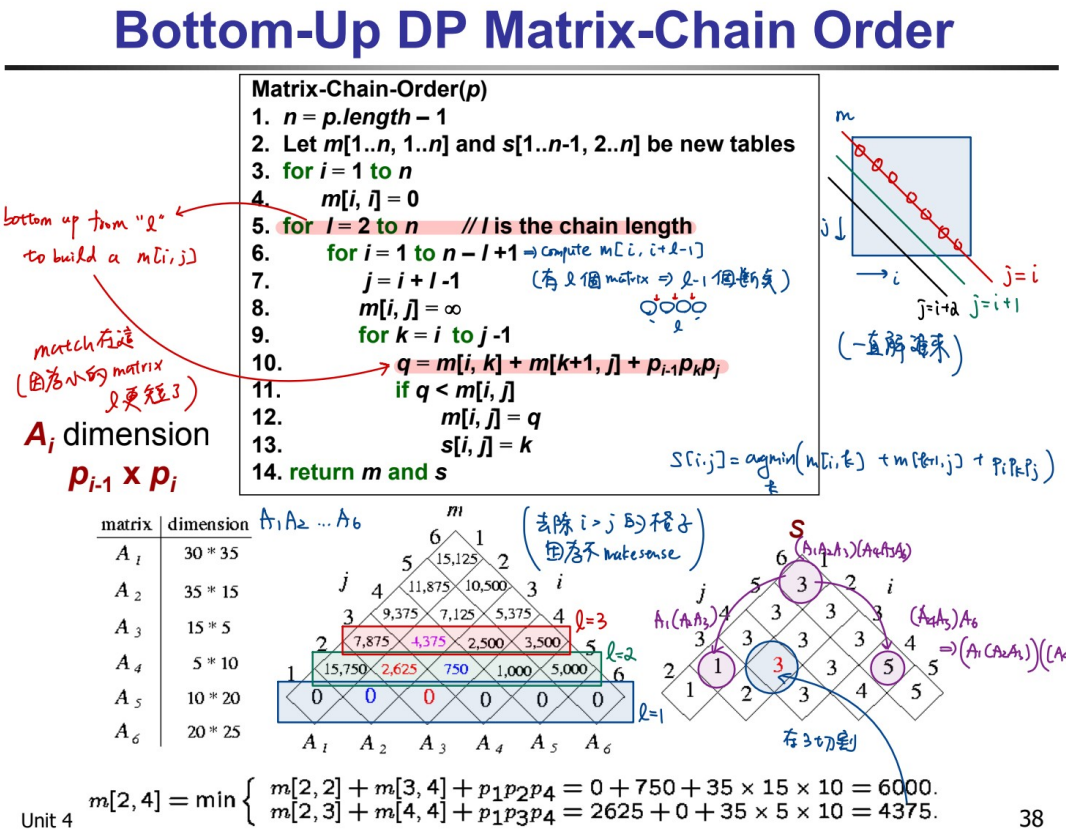
- B12901035 鄭宇彥：組合語言程式設計、程式效能優化、Debug
- B12901140 江履方：程式效能優化、Debug、Report

四、心得與感想

這個 final project 算是蠻有趣的，我們除了研究了很多組合語言優化以外，也嘗試了不少演算法在我們的 assembly code 上面。(在此特別感謝演算法課程的投影片，讓我們有 pseudo code 可以參考)

我們其實有發現不少可以取得高分的方法，以及 tile 的演算法可能在小 input size 下不那麼強大的問題，但我們仍然選擇嘗試實踐這些演算法，並利用自動化腳本跑過所有參數，試著找出它們的最佳表現爲何。(題外話，因爲鄭宇彥的電腦裝了 linux 跑太快了，所以我們才有辦法去暴力搜，找出比較好的 config 來用，聽說其他人跑一題都要好幾分鐘，多虧這件事才幫我們省下了不少時間)

所以說在這樣的精神上面，我想我們學到了很多超出課本外的知識，也更加理解 RISC-V 這個指令集架構的威力。這份作業作爲這堂課的結束，我們確實花了不少心力在上面，我想實在是有太多想提及的內容，都在這個課程裡面進行，感覺眞的算蠻充實的，希望未來再接觸這個領域的時候，我們已經非常理解這些方向了。



(d) NTUEE Algorithm, Spring 2025: Dynamic Programming Slide provided by B12901140