Final Project: Matrix Chain Multiplication

I. Design Specification

子模組	功能	特色
matrix_chain_multiplication	1. 以 **Bottom-Up DP** 決定最佳乘法順序 2. 依 DP 表遞迴建立最終乘積	一次 *malloc* cost/split/dims 三塊連續緩衝
_mcm_rec	依 split 表遞迴呼叫 _matmul_core 重建矩陣	Leaf node 直接傳回輸入矩陣指標
_matmul_core	真正的矩陣乘法核心 (8 × 8 **tile blocking**)	支援任意維度

II. Special Techniques Used

1. Dynamic Programming for MCM

資料流程

A. 先用 rows[0] / cols[] 建構 dims[]。

B. DP 表 cost[i][j] 以兩層長度-for 迴圈計算，split[i][j] 同步記錄最佳 k。

C. 完成後呼叫 _mcm_rec(0,N-1) 依 split 表遞迴乘回最終矩陣。

2. 8 × 8 Tiled (Blocked) Matrix-Multiply

目標	作法
L1D Hit	**一次只搬入 8×8 A-tile 和 8×8 B-tile（各 256 B）再累積至 C-tile；整組 ≈ 768 B 可被 8 KiB L1D 完整容納**
邊界維度	**外層 iT/jT/kT 仍以 8 為步徑，但每層都有 “>M/K/N” 檢查，確保不越界**
迴圈展開	**tile 內 3 層迴圈保持最小指令，若 M、N、K 皆為 8 的倍數可達理想 64 MAC / 192 load/store**

3.記憶體位置用指標遞增方式計算，減少乘加指令

III. Cache Tuning

Level	**Size**	**$\log_2$(Size)**	Associativity	**備註**
L1 I	8 KiB	13	8	程式碼 < 3 KiB，即使展開/unroll 也夠
L1 D	8 KiB	13	8	8 × 8×3 tiles = 0.75 KiB，加上 stack & spill 仍綽綽有餘
L2	64 KiB	16 (×½ = 8)	8	足以暫存 DP 表與遞迴工作集

IV. Reflections

1. **不斷 Debug**
    - 最常見錯誤是「暫存器覆寫」
2. **Tiling 邊界注意**
    - 當矩陣尺寸不是 8 的倍數時，務必在三層 tile 進入處加 **越界判斷**，否則最後一 tile 會讀過尾端造成 segmentation fault 或無窮迴圈。
3. **Tiling 能 improve 的效能有限**