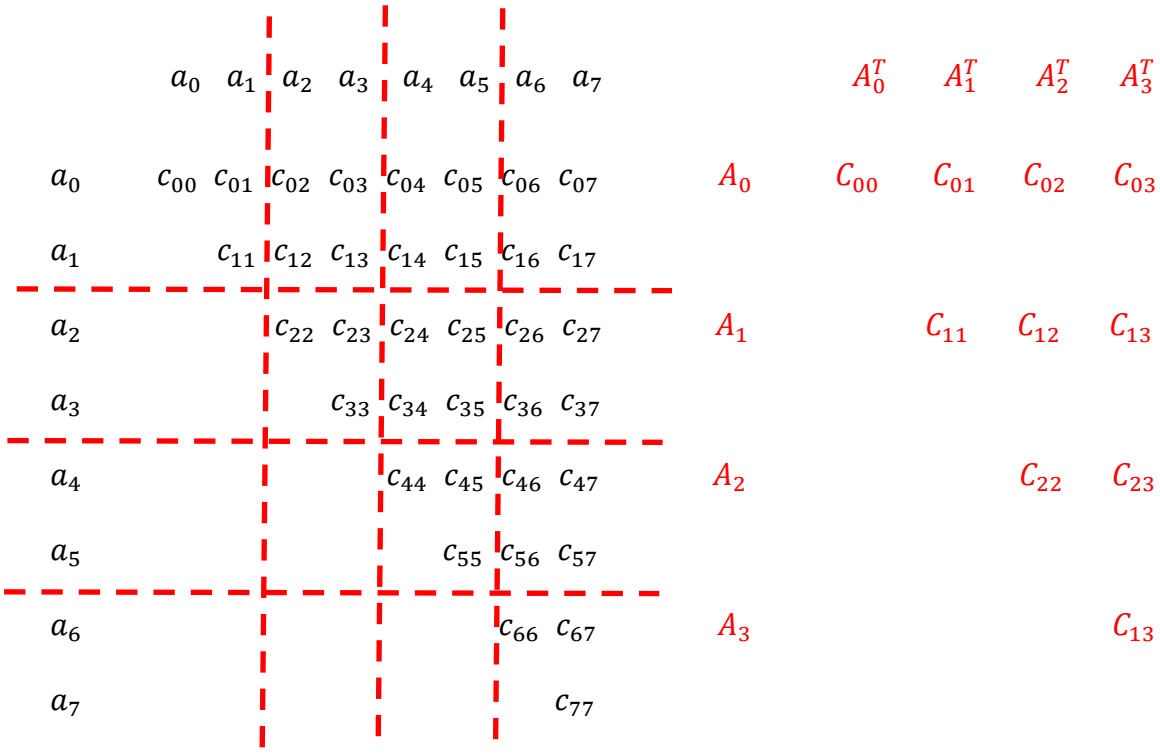
Tutorial 5

In this tutorial you will design and implement your own efficient parallel algorithm as described in the Assignment 1 description. Here are some hints on how to partition the task using the blocking technique and to assign the work to threads to balance the load across the threads.

Assume that the input 2D matrix is of size N by M, i.e., there are N sequences of size M. Use a_i to denote sequence i (or the i^{th} row of the matrix). For each pair of sequences (i, j), we perform a dot product $c_{ij} = a_i \cdot a_j$.

The baseline algorithm mainly uses vector operations and thus the computational intensity is very low. To increase the computational intensity, we must use blocking technique.



The above is an example of block partitioning. When we partition matrix C into a 2D block matrix, we also partition the sequences, or the input 2D matrix into block rows.

$$A_i = \begin{bmatrix} a_{ib} \\ \vdots \\ a_{(i+1)b-1} \end{bmatrix}, \quad A_i^T = [a_{ib}, \cdots, a_{(i+1)b-1}], \text{ and } C_{ij} = A_i A_j^T$$

Where b is the block size (assuming square blocks). The computation of C_{ij} involves two block rows of the input matrix. (You have done the exercise on how to optimize the performance of this computation using loop unrolling.) Note the computation for diagonal blocks is a bit different and you need also to consider optimize the code using loop unrolling.

In partitioning and work assignment we need to seriously consider load balancing.

1. Block sizes must be as equal as possible.
2. In partitioning we must consider how the partitioned blocks can be evenly distributed across the threads, e.g, the total number of partitioned blocks is divisible by the number of threads
3. Note we may not be able to make all the blocks be of the same size. For example, each diagonal block contains less amount of work and some blocks may have one less row/column than other blocks. In the assignment of blocks to threads we must try the best to balance the workload across the threads.

Finally, the output must be correctly stored in a 1D array in a row major order.