Parallelizing MMM:

\[ C = A \cdot B \]

\[ p_\Phi \]
\[ p_1 \]

\[ C \]

\[ p_\Phi \]
\[ p_1 \]

\[ A \]

\[ B \]

\[ 1 \]

The examples above show how to position the input and output matrices among 2 processors.

(Partitioning among n processors is trivial - the same technique applies.)

In (1), C, A, and B are positioned and sent to different processors for execution. B must be accessed in full by both processors. This is the ideal way to parallelize the MMM.

(2) uses the same idea as (1), but is not as effective, since unlike (1), the processors do not work on one big consecutive chunk (in memory) per matrix.
Parallelizing the WHT.

\[ \text{WHT}_{nm} = \left( \text{WHT}_n \otimes \text{Im} \right) \left( \text{Im} \otimes \text{WHT}_m \right) \]

Needs to be rewritten for parallelization.

Naturally parallelized for \( n \) processors.

(See rules in previous lecture.)

One way to express \( (\text{WHT}_n \otimes \text{Im}) \) is:

\[ (\text{WHT}_n \otimes \text{Im}) = L_n^m (\text{Im} \otimes \text{WHT}_n) L_m^m \]

The \( Ls \) (stride permutations) simply become data exchanges between processors in the implementation.

The following figures let us visualize this idea.
\[ WHT_2^* = (WHT_3 \otimes I_8) \; (I_2 \oplus (NHT_8)) \]
\[ \text{WHT}_2^* = L_2^{16} (I_8 \otimes \text{NHT}_2) L_2^{16} (I_8 \otimes \text{WHT}_8) \]
\( \text{WHT}_2 \otimes \text{i} \) can also be rewritten as \( (L^8 \otimes I_2)(I_4 \otimes \text{WHT}_3 \otimes I_2)(L^6 \otimes I_2) \)

Data exchanges in the \( (L \otimes I_n) \) stage exchanges \( n \) elements at a time, thus avoiding false sharing for cache line lengths of \( \leq n \).
\[ \text{Similarly,} \]

\[ (L^2 \otimes I^+)(E_2 \otimes NHT^2 \otimes I^+) \]