Reducer pattern

By using the reducer pattern, give a parallel solution for the following problems. When it is possible, use the predefined reducers of Cilk Plus. See https://www.cilkplus.org/tutorial-cilk-plus-reducers for a complete list of predefined reducers.

Minimum element

Use the reducer reducer_min of Cilk Plus to compute the index of the minimum element of an array of integers.

Now, compute the index of minimum element without using the reducer reducer_min. Compare the performance of both implementation. Which is faster? Which one has less cache misses and context switching?

Dot product

The *dot product* operation of two vectors $\mathbf{x}=[\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_n]$ and $\mathbf{y}=[\mathbf{y}_1,\mathbf{y}_2,...,\mathbf{y}_n]$ is defined as follows

$$\mathbf{x}\cdot\mathbf{y} = \mathbf{x}_1\mathbf{y}_1 + \mathbf{x}_2\mathbf{y}_2 + ... + \mathbf{x}_n\mathbf{y}_n$$

Design and implement an algorithm to compute the dot product operation. Give the work and span of your algorithm. Finally, show experimentally the scalability of your implementation.

DFS traversal

Given a binary tree, perform a DFS traversal in parallel using reducers. Test the scalability of your implementation

Printing in parallel

As we reviewed previously, printing in the standard output in parallel can generated a non-deterministic order in the printing. The reducer ostream of Cilk Plus allow us to print and preserve the order of the printings. Implement a thread-safe stream using the ostream reducer of Cilk Plus