PROBLEM STATEMENT

Problem

Your task in this assignment is to implement efficient C++/OpenMP parallel prefix function omp_scan . We make two assumptions: first, the number of input elements is much larger than the number of available processors, second, the number of available processors is relatively small (as in the case of e.g. compute nodes in CCR). To make your job easier, the main function and signature of omp_scan are provided such that you can focus on the main functionality only.

Instructions

1. Edit a0.hpp and implement omp_scan. You may include extra files and implement additional functionality as needed, however you are not allowed to change the signature of the omp_scan. Moreover, you should not edit a0.cpp as this is example of the engine that will be testing your code. Arguments of the omp_scan are as follows:

- T type of the elements on which prefix will be applied.
- Op type representing a binary associative operator compatible with T .
- n number of input elements.
- in pointer to the array of size n storing input items.
- out pointer to the output array of size n to store output prefix.
- op object representing associative operator of type Op . Just in case, invocation like out[1] = op(in[0], in[1]); applies the operator to
- items in[0] and in[1] and stores the result in out[1] .

When implementing omp_scan you may assume that all arguments are correct, and there is no need to test that. Moreover, you can assume that omp_scan is never invoked from within an OpenMP parallel region. Finally, the project backbone provides a simple Makefile . You can edit it as needed for your project. However, when invoked without arguments, Makefile must produce executable a0 from a0.cpp .