# Assignment 5: OpenMP - tasks

The purpose of this assignment is for you to learn more about

- the tasks construct of OpenMP,
- how parallel loops can be implemented with recursive tasks in OpenMP,
- how easy some parallel recursive algorithms are to write with OpenMP tasks.

As usual all time measurements are to be performed on the cluster.

Activate OpenMP in GCC by passing `-fopenmp` to the compiler and linker. (Note that if you omit this parameter, the code will probably still compile. But its execution will be sequential.)

You can control the number of threads in OpenMP in two ways: the environment variable `OMP_NUM_THREADS` or by calling the `omp_set_num_threads` function.

When recursively decomposing the workload, the granularity of the decomposition is set by the size of the smallest task that will not be further decomposed. Traditionally, a threshold is used under which all tasks are processed sequentially.

## 1 Reduce

**Question:** Implement computing the minimum of an array using the OpenMP task decomposition. Measure the time computing the minimum takes. Write the program so that you can control the length of the array, the number of thread and the scheduling policy.
**Question:** Plot a speedup chart for different numbers of thread (1, 2, 4, 8, 16), for arrays of sizes $10^8$, with granularity of 1000 and 100000. How does that compare to the speedup you obtained in assignment 4 ?

## 2 Find First

**Question:** Implement a parallel function using OpenMP tasks to find the first occurrence *pos* of an integer in an array
**Question:** Plot a speedup chart for different numbers of thread (1, 2, 4, 8, 16), for arrays of sizes $10^8$, Pick the granularity you feel is appropriate. How does that compare to the speedup you obtained in assignment 4 ?

## 3 Merge Sort

**Question:** Implement a parallel function using OpenMP tasks to perform merge sort on an array of integer.
**Question:** Plot a speedup chart for different numbers of thread (1, 2, 4, 8, 16), for arrays of sizes $10^9$. How does that compare to the speedup you obtained in assignment 4 ?

## 4 Strassen

**Question:** Parallelise the mock implementation of strassen from Assignment 3 using OpenMP tasks.
**Question:** Vary the operation intensity of both types of tasks. How does the speedup change with 16 computational threads? (That is to say keep a ratio of 1:5, but vary the absolute intensity.)