Project

N-coloring of a graph

Branea Alexandru

Calarasu Victor

Group : 932

1. Goal

Each project will have 2 implementations: one with "regular" threads or tasks/futures, and one distributed (possibly, but not required, using MPI). A third implementation, using OpenCL or CUDA, can be made for a bonus.

Problem 4 : Find an n-coloring of a graph.

The documentation will describe:

- the algorithms,
- the synchronization used in the parallelized variants,
- the performance measurements

2. Computer Specification

Processor: Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz 3.40GHz

RAM: 16GB

System Type: 64-bit operating system, x64-based processor

3. Description of the algorithm

Greedy Coloring Algorithm:

1. Color the first vertex with the first color.
2. Consider the currently picked vertex and color it with the lowest numbered color that has not been used on any previously colored vertices adjacent to it. If all previously used colors appear on vertices adjacent to v, assign a new color to it.
3. Do following for the remaining n-1 vertices.

The algorithm doesn't guarantee to use minimum colors, but it guarantees an upper bound on the number of colors.

Since d is maximum degree, a vertex cannot be attached to more than d vertices. When we color a vertex, at most d colors could have already been used by its adjacent.

4. Tests

Method	No of Threads/Tasks	Graph Size	Time
Sequential	-	1000	7ms
Parallel	2	1000	12ms
MPI	2	1000	147ms

Method	No of Threads/Tasks	Graph Size	Time
Sequential	-	2000	12ms
Parallel	5	2000	7ms
MPI	5	2000	434ms

Method	No of Threads/Tasks	Graph Size	Time
Sequential	-	5000	21ms
Parallel	10	5000	13ms
MPI	10	5000	1020ms

Method	No of Threads/Tasks	Graph Size	Time
Sequential	-	10000	30ms
Parallel	10	10000	19ms
MPI	10	10000	1207ms

5. Conclusion

The bigger the graph gets, the better the parallel method is. For smaller graphs, sequential is better. Because the greedy method needs the previous results, the MPI method is not effective in this case.