EUMaster4HPC Challenge: Graph Vertex Coloring

Giacomo Pauletti[1] Jan Marxen[2] Luca Lamperti[1] Mohamed Mandour[1]
[1]Politecnico di Milano, Italy [2]Sorbonne Université, France

Objective

Develop a parallel **branch-and-bound algorithm** to compute the **chromatic number** of graphs using the Vega supercomputer.
Vega architecture

CPU CLUSTER: 960 nodes, each with 2 AMD Rome 7H12 CPUs with 64 cores, for a total of 1,920 and 122,880 cores.

GPU CLUSTER: 60 nodes, each 4 Nvidia A100 GPUs and 2 AMD Rome7h12 Cpus240 NVIDIA, for a total of 240 A100 GPUs

Algorithms & Strategies

Zykov Algorithm

G graph with V vertices and E edges.
Given two **non-adjacent** vertices $x, y \in V$ two new graphs defined:

- G'_{xy} where x and y are **contracted** or *merged* into one single vertex xy.
- G''_{xy} where the **edge** $\{x, y\}$ has been added

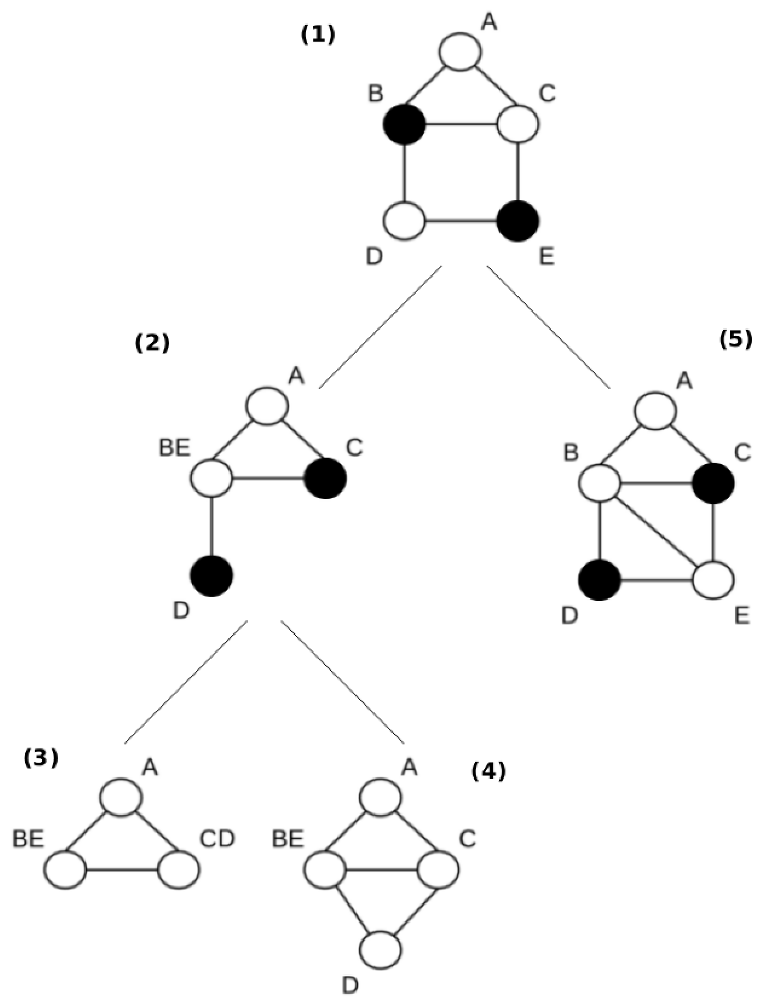
Recursive algorithm, called *Zykov's tree* (Figure 2), built upon following theorem:

Theorem 1. *The chromatic number of G is given by the recurrence*

$$\chi(G) = \min\{\chi(G'_{xy}), \chi(G''_{xy})\}$$

such that $x, y \in V(G)$ and $\{x, y\} \notin E(G)$

Leaf nodes are complete graphs. The chromatic number of the original graph G is the size of the **smallest leaf**.



Graph reduces size at every new branch $\implies$ Problem size decreases as depth increases $\implies$ complexity decreases.

Graph Representation Techniques

Optimize both memory usage and computational efficiency $\implies$ Efficient graph representation crucial.

Operation	Adjacency Matrix	Edge List	CSR	Adjacency List
Add Edge	$O(1)$	$O(1)$	$O(\|E\|)$	$O(1)$
Remove Edge	$O(1)$	$O(\|E\|)$	$O(\|E\|)$	$O(k)$
Add Vertex	$O(\|E\|)$	$O(1)$	$O(1)$	$O(1)$
Remove Vertex	$O(\|E\|)$	$O(\|E\|)$	$O(\|E\|)$	$O(\|E\|)$
Merge Vertices	$O(\|E\|)$	$O(\|E\|)$	$O(\|E\|)$	$O(\|E\|)$
Get Neighbors	$O(\|V\|)$	$O(\|E\|)$	$O(1)$	$O(1)$

Table 1: Comparison of Graph Representations

Adjacency list great performance in modifying the graph $\implies$ well suited for our application (Zykov).

Coloring Heuristics

Provide an **upper bound** for the chromatic number:

- **Greedy Coloring**: Assigns smallest available color to each vertex.
- **DSatur Algorithm**: Prioritizes vertices with most unique colored neighbors using priority queue.
- **Recoloring Algorithm**: Adjusts colors to minimize total number of colors used.

Clique Strategies

Estimate a lower bound by identifying large cliques in the graph:

- **FastWClq**: Quickly approximates maximum clique using heuristic search.
- **BMS Selection**: Selects vertices that maximize clique growth.
- **Graph Reduction**: Prunes vertices that cannot be part of a maximal clique, improving efficiency.

These heuristics improve pruning efficiency and speed up the search process.

OpenMP & MPI

Parallelization Strategies

Parallelize execution using **MPI** and **OpenMP**.
Work Distribution Models

- Simple Approach: Master orchestrates tasks.
- Scalable Approach 1 and 2: Each process explores its own search space, sharing solutions from time to time.

Multi-Threaded Processing

- Terminator: Monitors execution time and stops processes if needed.
- Gatherer: Collects solutions and updates global best results.
- Employer: Manages work-stealing (Idle workers take over unfinished tasks to balance the load).

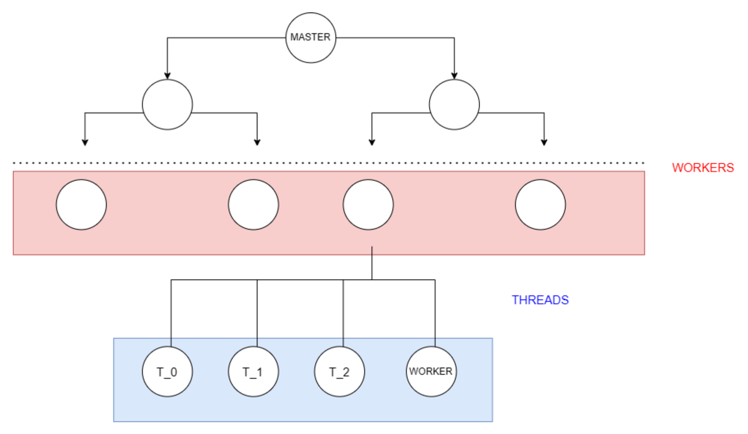


Figure 1: Simple Approach

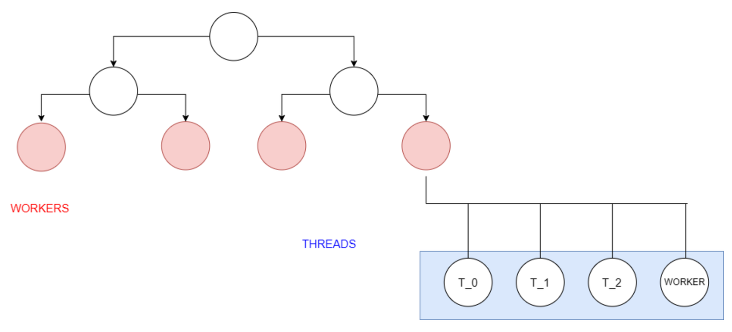


Figure 2: Scalable Approach 1, case 4 cores

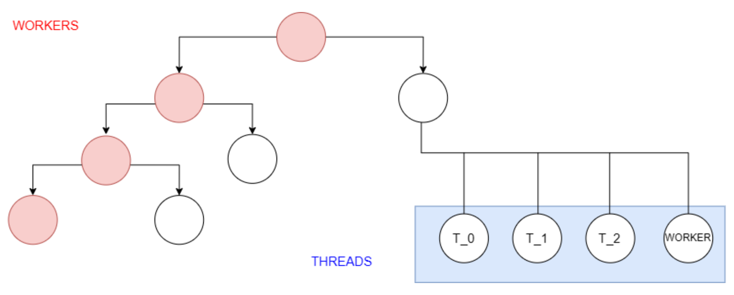


Figure 3: Scalable Approach 2, case 4 cores

Results

Table 2: Strong scaling results for solving the chromatic number of `queen7_7.col`.

Nodes	Time (s)	Total Cores	N processors
1	330.91	32	8
2	100.22	64	16
4	70.78	128	32
8	55.89	256	64
16	46.01	512	128
32	0.23	1024	256

Table 3: Strong Scaling Results (Set 1)

Figure 4: Execution time vs. number of nodes for solving `queen7_7.col`.

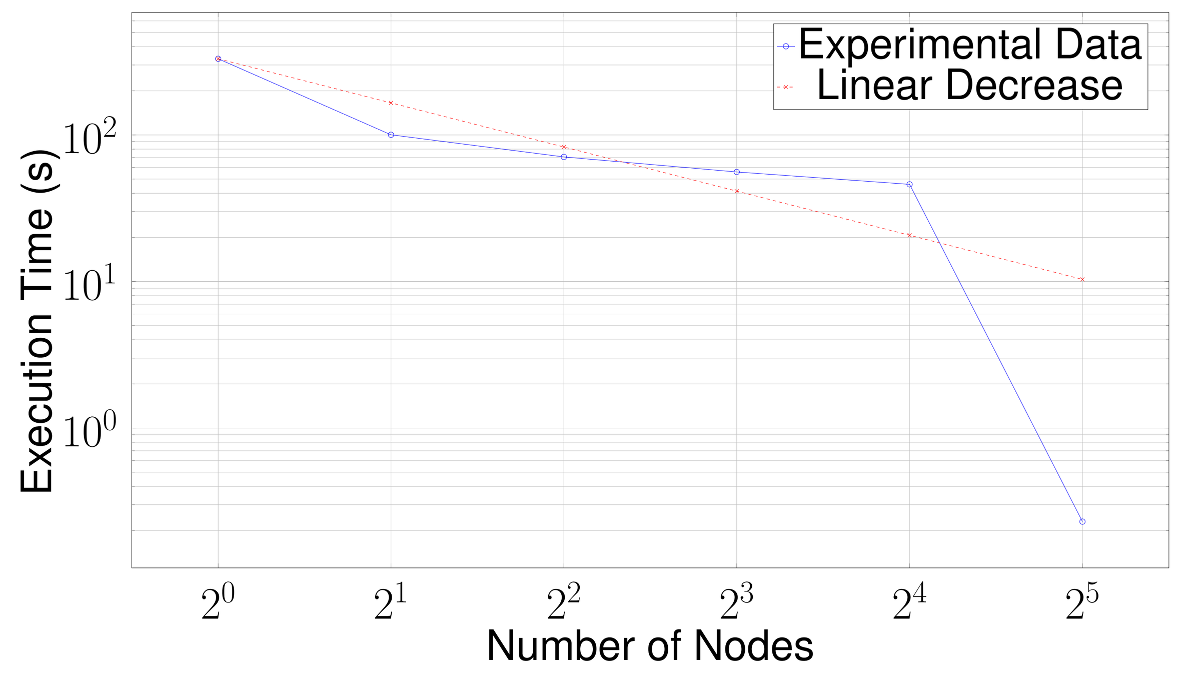
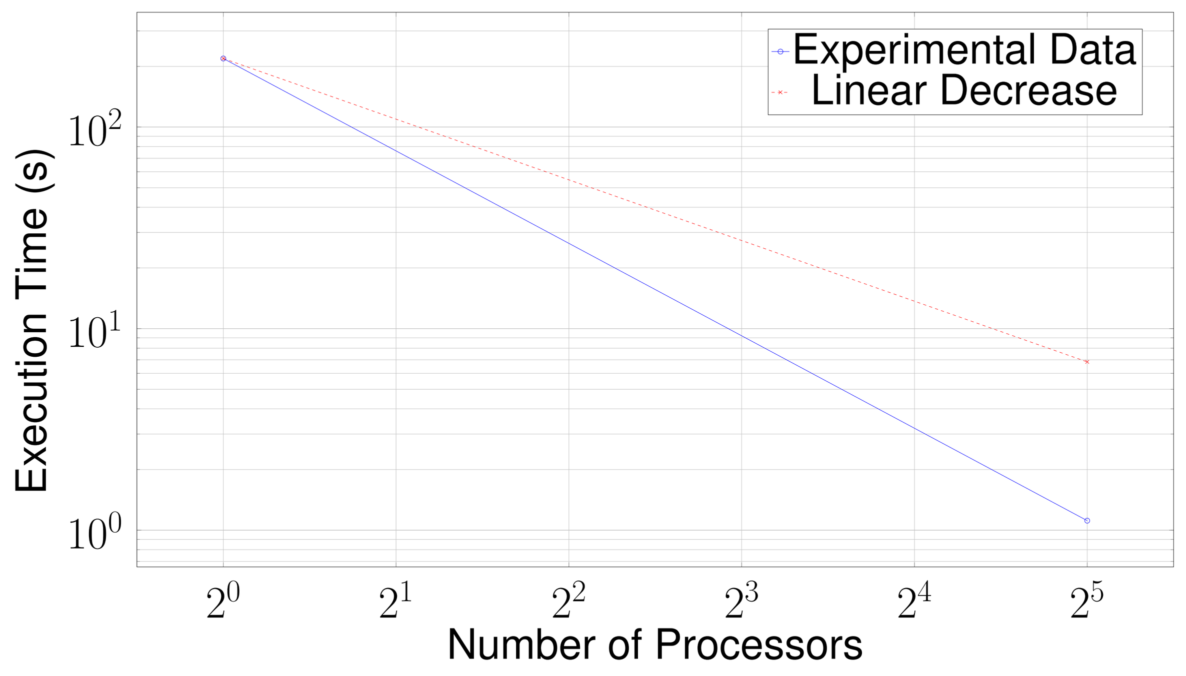


Table 4: Strong scaling results for solving the chromatic number of `queen9_9.col`.

Nodes	Time (s)	Total Cores	N processors
1	218.86	1	1
4	1.115	512	32

Table 5: Strong Scaling Results (Set 2)

Figure 5: Execution time vs. number of nodes for solving `queen9_9.col`.



Conclusion

The proposed parallel branch-and-bound algorithm demonstrates **strong scalability**. Further optimizations for sparse graphs remain an open direction.

Acknowledgements

The authors thank Dr. Arnaud Renard for his expert guidance and the Vega supercomputer team not only for their excellent support but also for the provision of the computing resources.

This project has received funding from the European High-Performance Computing Joint Undertaking under grant agreement No. 101051997.
EUROHPC SUMMIT 2025, Kraków — EUMaster4HPC - See more: `https://github.com/Rudolfovoorg/EuroHPC_Student_Challenge_2025_Team_2`
giacomo.pauletti@mail.polimi.it jan.marxen@gmail.com l.lamperti2002@gmail.com mohamedzeyad.mandour@mail.polimi.it