BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY
Excel@FIT 2025

GPU Acceleration of Dijkstra Algorithm

Author: Jan Fiala Supervisor: doc. RNDr. Milan Češka Ph.D.

INTRODUCTION

Single-Source Shortest Path (SSSP) problem finds the shortest paths from a given source node in a graph. The results are the most efficient (shortest) traversals from the source node to any destination node in the graph. The SSSP problem is solved primarily for real world problems in various fields such as logistics, robotics, and AI. Dijkstra's algorithm is optimal for SSSP, as it solves the problem with time complexity of $O(N \cdot \log N + E)$ for non-negative weighted graphs.

Why accelerate?

– The need to compute the shortest paths for graphs with millions of nodes repeatedly.
– Sequential solution is insufficient as it takes too long when dealing with dense graphs.
– In such cases, parallelization of the computation can bring us significant acceleration.
– While the sequential takes minutes or hours, the parallel solution takes a few seconds.

STATE OF THE ART

The main idea is to choose nodes with the **lowest tentative distance** across all unsettled nodes, marking them the "*Next to settle*" and processing them in parallel during the settlement phase. As the amount of such nodes is limited due to the sheer number of nodes with the lowest tentative distance in each iteration, they cannot identify multiple nodes fitting these conditions frequently. Therefore, being unable to take advantage of the GPU's resources.

Both of the existing GPU implementations have an issue with processing a wide weight range, as the wider the range, the less chance of having multiple nodes with the lowest tentative distance. Two GPU parallelizations – **Martín's** [1] and **Crauser's** [2] served as a baseline for evaluation. Both were re-implemented and compared to their original's run times to prove their comparability.

Dijkstra's settlement phase & relaxation process

Reached unsettled nodes (nodes whose tentative distance is known, less than ∞) are processed during the parallel settlement phase of the algorithm, relaxing edges of a node; thus, marking the node itself settled. Such node won't be re-settled ever again, as it was already once settled. During the relaxation phase of a node, its neighboring nodes tentative distance is updated if traversing an edge leading to the neighbor results in a smaller distance, hence a shorter path.

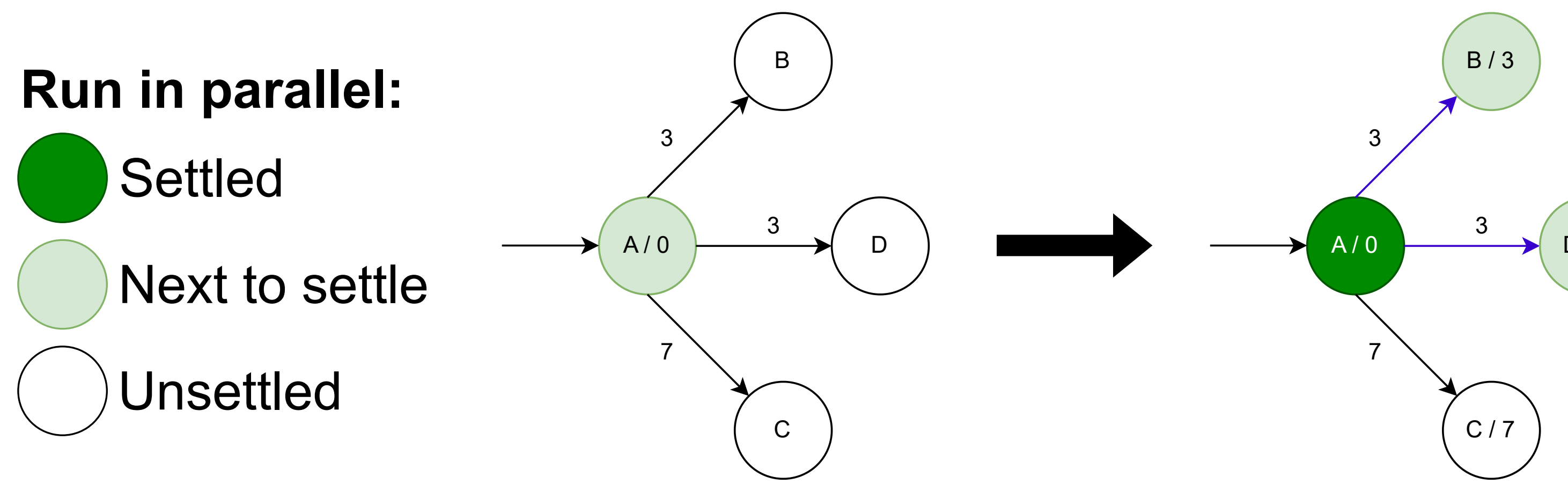

Figure 1: Relaxation process during the settlement phase in parallel

NOVEL APPROACH

KEY IDEA

We improve the usage of GPU's resources by increasing the number of nodes that are settled in the particular iterations (i.e. the nodes that can be processed in parallel). We achieve this by relaxing the condition of which nodes are settled: in contrast to the existing approaches, we also settle nodes that do not have the lowest tentative distance in the current iteration. This enables a more efficient parallelisation of the graph exploration, **mitigating the bottleneck** related to the low resource utilization by the State of The Art. The relaxed condition comes at the cost of re-settling some nodes repeatedly; thus introducing an overhead. In the experiments section, we demonstrate that despite the overhead, our approach significantly outperforms the sequential algorithm as well as the State of The Art approaches on a wide set of benchmarks.

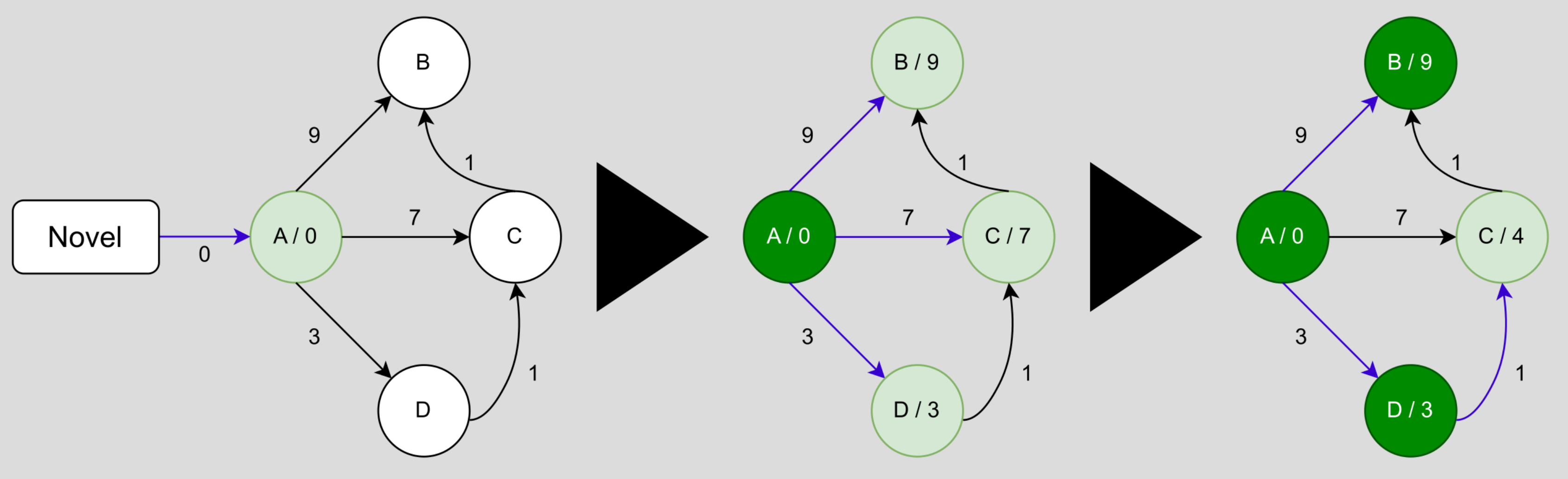

Figure 2: Re-settlement of a settled node C due to an update of the shortest path

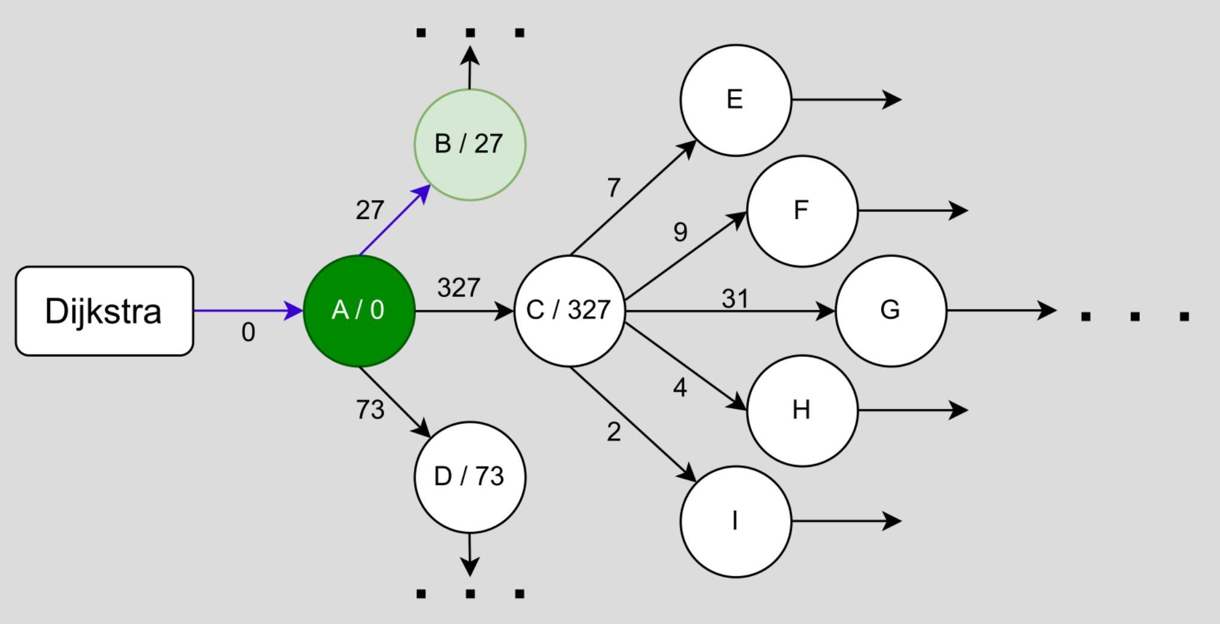

Figure 3: Parallel Martín/Crauser algorithm

As Dijkstra always identifies only the shortest paths during each iteration, it initially explores only the top part of the presented graph. As a result, there are no additional steps after a node is once settled, marking the shortest identified path to the node.

Note that the Novel algorithm manages to always identify at least a single path that is the shortest, which is the same path as the Dijkstra's algorithm would identify. Thanks to that, the approach is able to ensure that correct results of the SSSP are propagated each iteration.

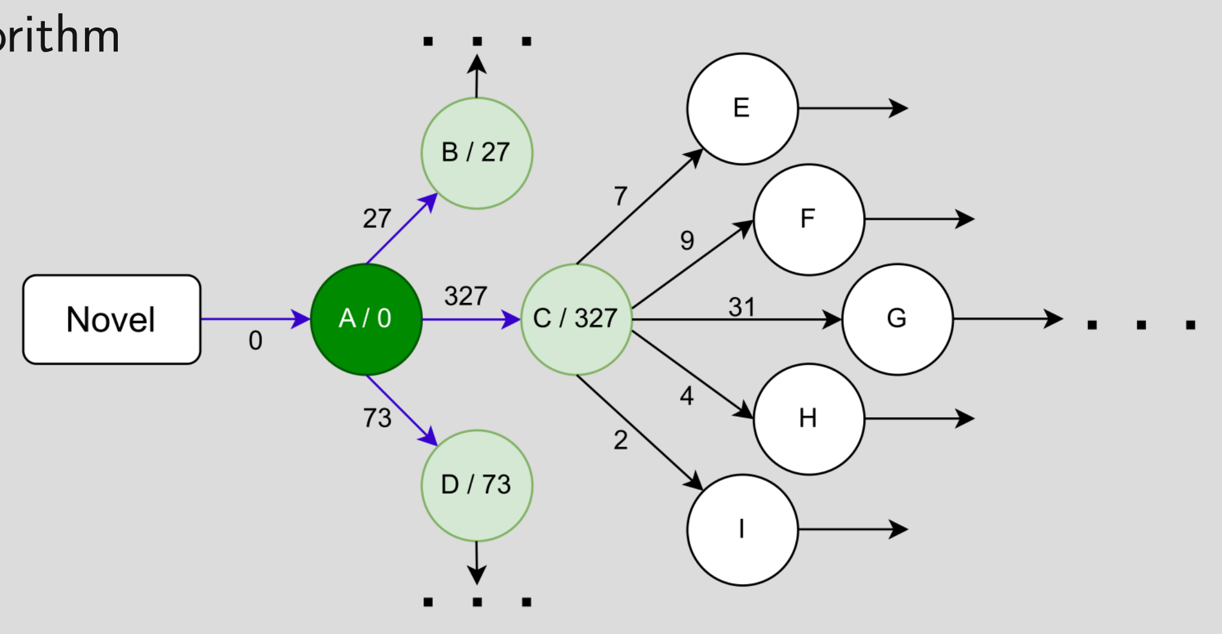

Figure 4: Parallel Novel approach

KEY OBSERVATIONS

The weight range has a **significant impact** on the computation time, especially for **State of The Art**, which is usually superior for a **narrow** weight range. As for wide weight range, **Novel Approach** comes out on top in comparison with the State of The Art. As can be seen in Figure 6, **Seq Boost** takes a few minutes, while the **Novel** takes barely a few seconds without fluctuations considering the **weight range**.

EXPERIMENTS

The evaluation was done for hundreds of graphs for not only synthetic, but also real world graphs of varied topologies and density. The main objective was to evaluate capabilities of the algorithms in a real world environment, as proficient synthetic results **do not always translate** into sufficient capabilities in **real world** scenarios, which is **Martín's** case. On the contrary, the Novel Approach enables massive parallelism, as it processes any **updated** node each iteration, with the trade-off of possibly performing useless computations and re-settling nodes repeatedly.

ALGORITHMS FOR THE EVALUATION

- **Novel** steadily outperforms State of The Art in majority of cases
- **Martin** is the best for a narrow weight range large dense graphs
- **Crauser** is the latest State of The Art parallelization of Dijkstra
- **Seq-Boost** is the BOOST Library sequential CPU Dijkstra

EXPERIMENTAL SETUP

GPU: GTX 1070 Pascal architecture
CPU: Intel(R) i5-9600KF CPU @ 3.7 GHz

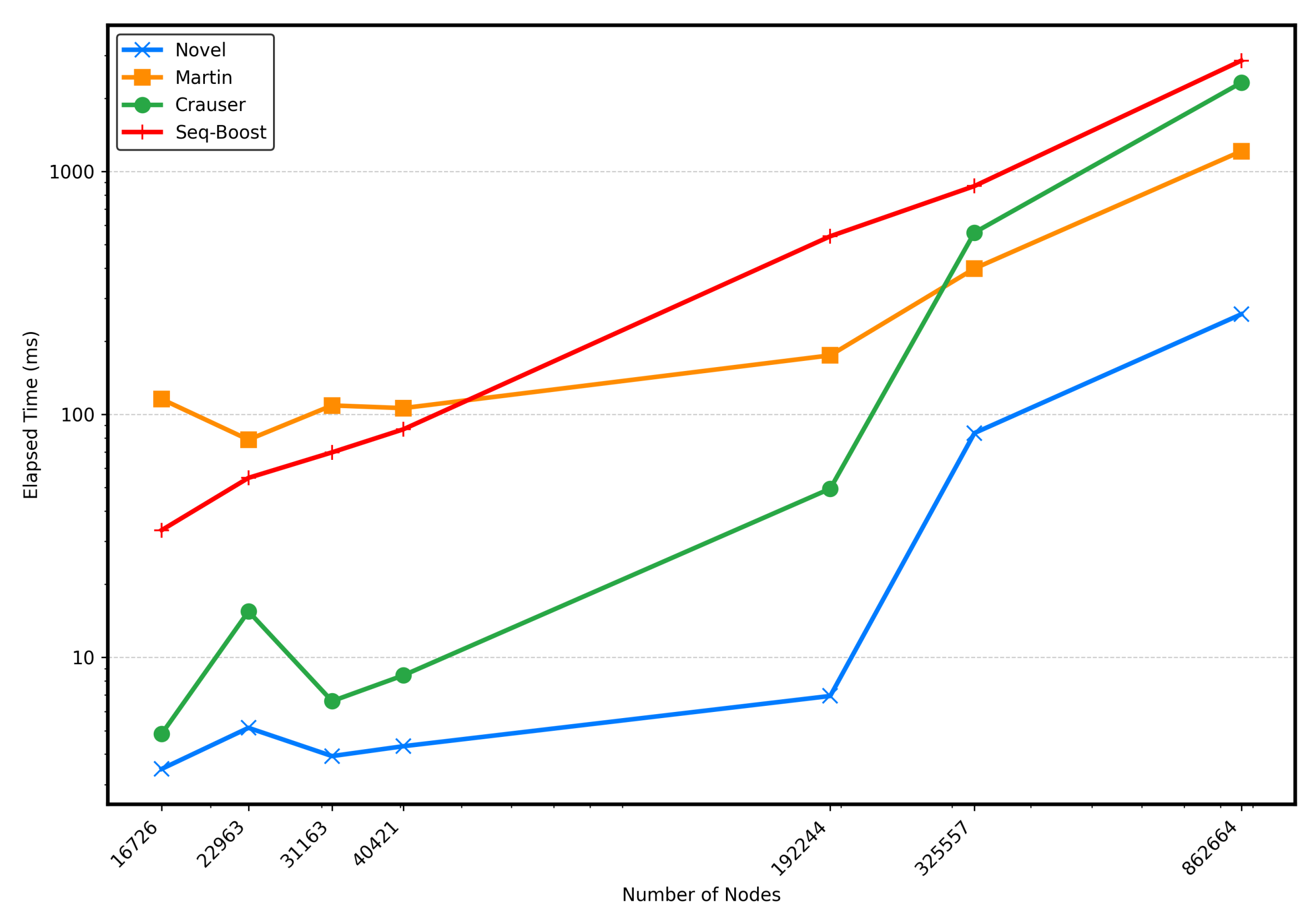

Figure 5: Medium density dataset (weight range 1–100)

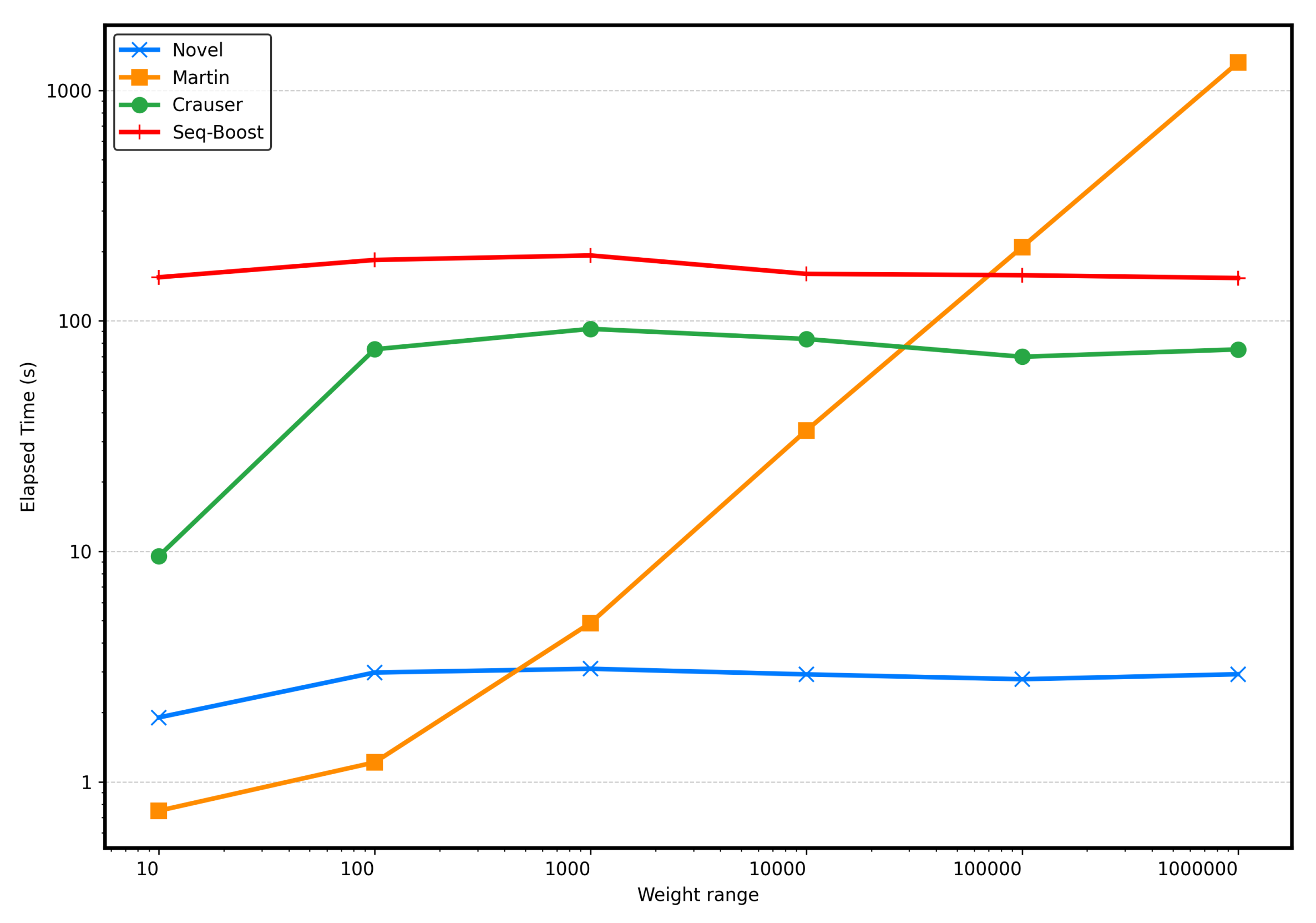

Figure 6: High density synthetic graphs (2^{24} Nodes, 45 Edges)

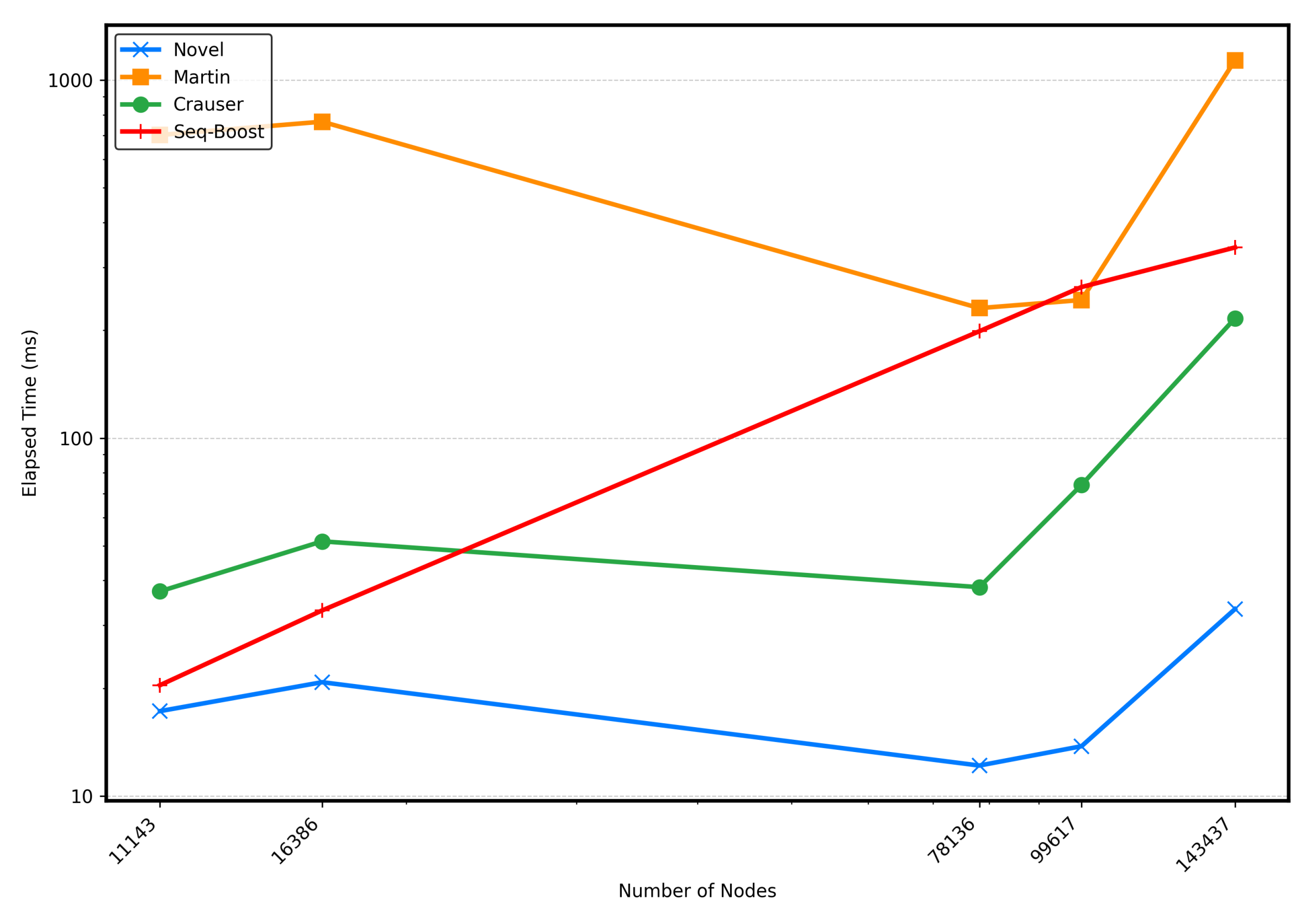

Figure 7: Low density dataset (weight range 1–100)

References:

[1] Hector Ortega-Arranz, Yuri Torres, Diego R Llanos, and Arturo Gonzalez-Escribano. A new gpu-based approach to the shortest path problem. pages 505–511. IEEE, 2013.

[2] Hector Ortega-Arranz, Yuri Torres, Arturo Gonzalez-Escribano, and Diego R Llanos. Comprehensive evaluation of a new gpu-based approach to the shortest path problem. *International Journal of Parallel Programming*, 43(5):918–938, 2015.