SOPMOA*: Unleashing Shared-Open Parallelism for High-Performance Multi-Objective Pathfinding

Long Viet Truong, Tien Minh Dam, Tuan Anh Nguyen, Linh Thuy Thi Nguyen, Duong Trung Dinh
Viettel High Technology Industries Corporation
{longtv28, tiendm9, tuanna63, linhntt6, duongdt2}@viettel.com.vn

About Us

Viettel Military Industry and Telecoms Group (Viettel Group) is Vietnam's largest telecommunications and technology conglomerate and operates under the Ministry of National Defense. Founded in 1989, Viettel has grown into a global digital services provider, currently serving more than 270 million customers across 11 countries in Asia, Africa, and Latin America, offering a wide range of services, including mobile and fixed telephone, broadband, cyber security, digital television, and defense systems. The company is also a pioneer in semiconductor and R&D efforts to advance Vietnam's national technological capabilities.
Viettel High Technology Industries Corporation (Viettel High Tech), established in 2011, serves as the primary R&D and manufacturing arm of Viettel Group in the defense and civilian electronics domain. Headquartered in Hanoi's Hi-Tech Park, Viettel High Tech specializes in developing encrypted radios, surveillance drones, radar systems, 5G / Open RAN telecommunications infrastructure and AI-driven solutions. It has delivered major international contracts—such as private 5G deployments in the Middle East with global partners—positioning itself as a key player in Vietnam's emerging global tech landscape .

1 Problem and Motivation

The multi-objective shortest path (MOSP) problem aims to find paths between two nodes in a graph that optimize multiple, often conflicting, objectives simultaneously. Unlike the traditional shortest path problem, which focuses on a single objective (e.g. distance), MOSP considers multiple attributes for each edge, such as cost, time, and economic cost. The goal is to find Pareto-optimal paths, meaning no path can improve one objective without worsening at least one other.
In recent years, the Multi-Objective Shortest Path (MOSP) problem has garnered growing interest within the field of network optimization, particularly following the introduction of the NAMOA* framework for efficient search. Subsequent advances—NAMOA*$_{dr}$ (2021), EMOA* (2022), LTMOA* (2023), and NWMOA* (2024)—have achieved substantial gains by employing specialized techniques, most notably **dimensionality reduction**. However, these approaches depend critically on the order in which search nodes are expanded, thereby limiting them to largely single-threaded implementations. A handful of parallel strategies have also been proposed: MDA and T-MDA (2023) exploit parallel dominance checks, while BOBA* parallelizes two search threads for bi-objective instances. Parallelizing the search process over graphs remains under-explored, yet is a promising frontier for MOSP.
In this work, we introduce the novel **SOPMOA***, one of the first MOSP algorithms to enable unrestricted parallel search across any number of objectives, yielding performance superior even to single-objective runs.

2 SOPMOA*: Shared-Open Parallelized Multi-Objective A*

The Shared-Open Parallelized Multi-Objective A* (SOPMOA*) framework begins by establishing a small number of shared data structures in a common memory space: a global priority queue (OPEN) that orders "labels" (partial paths) by their estimated cost, a collection of per-node Pareto-frontiers (G_{cl}) that record the best cost vectors seen so far at each vertex, a solution set (Sols) to accumulate completed paths to the target, and an array of worker flags to track which threads are still exploring. A single "start" label is created and pushed into OPEN, and then N parallel worker threads ("sub-searchers")

To ensure both correctness and performance under parallel execution, SOPMOA* employs two specialized procedures for managing the per-node frontiers. A "frontier check" operation takes a snapshot of a node's Pareto list under a shared lock, quickly ruling out any already-dominated vectors before performing a more precise dominance test on the remainder. A "frontier update" holds an exclusive lock, removing any weaker cost vectors and inserting the new one in proper lexicographic order. These lightweight locking strategies allow multiple workers to read the same frontier concurrently while ensuring that updates remain atomic and the frontier always represents a consistent, truncated Pareto set.
Because labels may be expanded out of strict global order, SOPMOA* can momentarily retain so-called "false Pareto" labels—paths that pass the local checks but would ultimately be dominated once all possibilities are explored. However, these extra labels are few, and they cannot displace truly optimal solutions because every label must first survive the frontier checks. In the final cleanup, any non-optimal labels remaining in the solution set are pruned, guaranteeing that the algorithm returns exactly the complete Pareto-optimal collection of paths to the target.

3 Experimental Results

Algorithms	*Solved*	*Runtime* (s)			
		Min	*Max*	*Mean*	*Median*
100 random NY instances (3 objectives) - *avg. solutions* ≈ 10220					
EMOA*	100/100	**0.209**	1896.45	357.84	58.36
LTMOA*	99/100	0.313	3600.00	823.95	158.38
LazyLTMOA*	100/100	0.273	3528.92	690.43	131.18
NWMOA*	100/100	0.234	3456.15	553.50	102.41
SOPMOA* - *4 workers*	100/100	0.311	3024.69	563.13	97.60
SOPMOA* - *20 workers*	100/100	0.257	**1730.34**	**248.58**	**51.83**
50 random NY instances (4 objectives) - *avg. solutions* ≈ 16866					
EMOA*	38/50	**0.507**	3600.00	1344.37	852.71
LazyLTMOA*	33/50	0.896	3600.00	1766.10	1804.97
NWMOA*	37/50	0.579	3600.00	1398.89	980.76
SOPMOA* - *20 workers*	**46/50**	0.768	3600.00	**1096.97**	**649.82**

We evaluated SOPMOA* the New York Road Map benchmarks from the 9th DIMACS Challenge with four objectives: distance, time, economic cost, random [1,100]. We compare our SOPMOA* with novel MOSP algorithms EMOA*, LTMOA*, LazyLTMOA*, NWMOA*, with a 3600s timeout.
Our first comparisons run on 100 three-objective New York instances. With 20 threads, SOPMOA* solved all instances with a mean runtime 1.5× faster than EMOA* and over 3× faster than LTMOA*, which also failed on one case. Even at 4 threads, SOPMOA* matched or slightly outperformed NWMOA*'s average time, demonstrating its efficiency across both high- and modest-parallel settings.
Extending to four objectives (50 instances), SOPMOA* (20 threads) solved 46/50—the highest success rate—while competitors solved 33–38 cases. Its mean runtime remained on par with the three-objective experiments, though the minimum time increased by 12s due to thread initialization and lock overhead. Single-threaded methods, by contrast, saw 30–50% runtime growth when adding the fourth objective. These results confirm SOPMOA*'s superior robustness and performance as dimensionality increases.

Conclusion: SOPMOA* consistently outperforms state-of-the-art MOSP algorithms in both speed and robustness, demonstrating the power of unrestricted parallelism.

4 Conclusion

SOPMOA* is a novel parallel MOSP algorithm that uses multiple sub-searchers on a shared priority queue to compute complete Pareto fronts via new expansion, dominance-checking, and frontier-update strategies. Early results demonstrate speedups over state-of-the-art methods. Future work will focus on reducing locking overhead and exploring bucket queues, lazier dominance