An Iterated Local Search Approach for One-Sided Crossing Minimization

Labeat Arbneshi[1] and Kadri Sylejmani[1⋆]

Faculty of Electrical and Computer Engineering,
University of Prishtina, Prishtina, Kosovo
{labeat.arbneshi, kadri.sylejmani}@uni-pr.edu

Abstract. The One-Sided Crossing Minimization (OSCM) problem seeks a vertex ordering of the free layer in a bipartite graph that minimises edge crossings—a core subroutine in the Sugiyama framework for layered graph drawing. We propose an Iterated Local Search (ILS) metaheuristic that combines six lightweight, *atomic* local search operators within an Adaptive Large Neighborhood Search (ALNS) selection scheme. Unlike conventional steepest-descent moves requiring quadratic time per invocation, each atomic operator evaluates a single candidate move in constant or near-linear time, enabling several thousand improvement steps per second. Experimental evaluation on 30 instances from the PACE 2024 Challenge demonstrates that our solver matches the best known results on 16 of 30 instances within 60 seconds—one-fifth of the competition budget—achieving a mean gap of 0.24 % across all thirty instances.

Keywords: Crossing minimization · Iterated local search · Graph drawing · ALNS · PACE 2024

1 Introduction

The OSCM problem is a fundamental task in layered graph drawing [1, 13]. Given a bipartite graph $G = (A \cup B, E)$ where layer A has a fixed order, the goal is to find a permutation π of layer B minimising edge crossings. Two edges (a_1, b_1) and (a_2, b_2) cross iff $\big(\mathrm{pos}_A(a_1) - \mathrm{pos}_A(a_2)\big) \cdot \big(\pi(b_1) - \pi(b_2)\big) < 0$. OSCM is NP-hard [2]; exact and parameterised algorithms [11, 12] exist but do not scale to the largest practical instances. The problem was the subject of the PACE 2024 Challenge [3, 14], which provided 100 instances of varying size (up to 91 000 edges) and structure, with a five-minute budget for the heuristic track.

Classical heuristics—*barycenter* and *median* [4]—sort B-vertices by the mean or median A-neighbour position. Local search refinements (adjacent-swap, insertion) close much of the gap but require $O(n^2)$ per pass. Our contribution is an ILS that replaces these steepest-descent operators with *atomic* (first-improvement, single-move) variants running in $O(n)$ or less, combined under an ALNS [5] weighting scheme.

⋆ Corresponding author.

2 Approach

2.1 Initial Solution and ILS Framework

The initial ordering is obtained by computing both the barycenter and median heuristics and retaining the permutation with fewer crossings. The entire time budget is then allocated to ILS. The search proceeds in *phases* of random duration drawn from $\{1, 2, 3, 5, 8\}$ s. Within each phase, an operator is chosen by roulette-wheel sampling over the ALNS weights, applied to the current ordering, and accepted if the crossing count decreases. At the end of each phase the incumbent solution is either updated or perturbed. After ten consecutive phases without improvement the search restarts from the global best [6].

2.2 Atomic Operators

All six operators share the same contract: attempt *one* move and return the (ordering, cost) pair. Below, $\bar{d}$ denotes the average B-degree.
Adjacent Swap. Scan from a random position, accept the first swap with negative pairwise delta. $O(n)$ worst case.
Vertex Insert. Pick a random vertex u, sweep all insertion positions incrementally. $O(n \cdot \bar{d})$.
Non-Adjacent Swap. Pick random position i, scan $j \in [i{+}1, i{+}w]$ (w=10), accept the first improving swap. $O(w^2 \cdot \bar{d})$.
Segment Reversal. Pick random start, extend incrementally, accept first improving reversal of length $\ell \in [2, L_{\max}]$. $O(L_{\max}^2 \cdot \bar{d})$.
Or-opt-k (k=2, 3). Remove a random block of k vertices, sweep insertion positions. $O(n \cdot k \cdot \bar{d})$.

2.3 ALNS Weights and Perturbation

Operator weights are initialised with a bias toward insert and non-adjacent swap. After each phase, successful operators receive reward +3.0 per improvement; all weights decay by 0.85 (floor 0.1).

Perturbation uses two mechanisms: a *random kick* (mix of swaps, reversals, insertions, and block moves) and a *double-bridge* move [7] that reconnects four ordering segments as $A{+}C{+}B{+}D$. The double-bridge probability increases from 50% to 90% as phases pass without improvement; when deeply stuck ($\geq$8 phases), two double-bridges are applied in sequence.

3 Computational Results

We selected 30 instances from the PACE 2024 heuristic track covering six structural categories. All experiments used a 60-second limit on a single-threaded consumer machine; each instance was run up to three times (best retained). Table 1 compares against three top heuristic-track solvers on the `optil.io` leaderboard: CIMAT_Team (1st, memetic) [8], LUNCH (2nd) [9], and Arcee (4th) [10].

Table 1. Results on 30 PACE 2024 instances (60 s budget, best of 3 runs). Superscripts on Best Known: * = all three teams, C = CIMAT, A = Arcee, L = LUNCH.

Grp	Inst.	Best Known	CIMAT	Arcee	LUNCH	Ours	Δ(%)
Tiny	65	72 910 *	72 910	72 910	72 910	72 910	–
	49	779 *	779	779	779	779	–
	45	1 019 861 *	1 019 861	1 019 861	1 019 861	1 019 877	<0.01
	92	59 032 *	59 032	59 032	59 032	59 038	0.01
	66	103 362 *	103 362	103 362	103 362	103 373	0.01
Dense	14	1 442 485 *	1 442 485	1 442 485	1 442 485	1 442 485	–
	15	10 852 981 *	10 852 981	10 852 981	10 852 981	10 852 981	–
	37	33 588 819 C	33 588 819	33 588 877	33 588 850	33 597 500	0.03
	38	30 602 519 C	30 602 519	30 602 616	30 602 551	30 623 828	0.07
	86	1 117 722 *	1 117 722	1 117 722	1 117 722	1 117 967	0.02
Regular	33	440 799 *	440 799	440 799	440 799	440 799	–
	42	447 480 *	447 480	447 480	447 480	447 480	–
	43	772 760 *	772 760	772 760	772 760	772 760	–
	41	1 333 254 *	1 333 254	1 333 254	1 333 254	1 333 254	–
	80	159 436 *	159 436	159 436	159 436	159 458	0.01
Hub	57	252 004 *	252 004	252 004	252 004	252 004	–
	5	7 536 *	7 536	7 536	7 536	7 536	–
	7	10 837 *	10 837	10 837	10 837	10 837	–
	1	12 432 *	12 432	12 432	12 432	12 432	–
	9	177 606 *	177 606	177 606	177 606	177 606	–
Medium	11	2 030 125 *	2 030 125	2 030 125	2 030 125	2 031 398	0.06
	89	365 618 *	365 618	365 618	365 618	365 885	0.07
	19	11 031 578 C	11 031 578	11 031 597	11 031 592	11 050 038	0.17
	25	265 109 *	265 109	265 109	265 109	268 760	1.38
	39	748 666 C	748 666	748 673	748 667	769 682	2.81
Unbal.	34	0 *	0	0	0	0	–
	35	0 *	0	0	0	0	–
	71	562 159 *	562 159	562 159	562 159	568 451	1.12
	72	829 116 A,L	829 123	829 116	829 116	840 075	1.32
	29	0 *	0	0	0	0	–
Matched						**16/30**	
Avg. gap							**0.24%**

Our solver matches the best known result on 16/30 instances within 60 s (one-fifth of the competition budget). Results are strongest on hub (5/5) and regular (4/5) instances. The largest gaps appear on Medium-group instances (up to 2.81%), which benefit from longer search time. The atomic-operator design enables scalability: instance 9.gr ($|B|$=45k) runs in 60 s, whereas steepest-descent warm-up alone took over four minutes.

4 Conclusions and Future Work

We have presented an ILS solver for the One-Sided Crossing Minimization problem whose *atomic* operators enable thousands of moves per second, achieving competitive results against the top PACE 2024 participants within one-fifth of the competition time budget. Future work includes: (1) a hybrid strategy combining steepest descent for small instances with atomic operators for large ones;

(2) population-based extensions with crossover operators [8]; and (3) evaluation on the full 100-instance set.

References

1. Sugiyama, K., Tagawa, S., Toda, M.: Methods for visual understanding of hierarchical system structures. IEEE Trans. Syst. Man Cybern. **11**(2), 109–125 (1981)
2. Garey, M.R., Johnson, D.S.: Crossing number is NP-complete. SIAM J. Algebr. Discret. Methods **4**(3), 312–316 (1983)
3. Kindermann, P., Klute, F., Terziadis, S.: The PACE 2024 Parameterized Algorithms and Computational Experiments Challenge: One-Sided Crossing Minimization. In: Bonnet, É., Rzążewski, P. (eds.) IPEC 2024. LIPIcs, vol. 321, pp. 26:1–26:20. Schloss Dagstuhl (2024)
4. Eades, P., Wormald, N.C.: Edge crossings in drawings of bipartite graphs. Algorithmica **11**(4), 379–403 (1994)
5. Ropke, S., Pisinger, D.: An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp. Sci. **40**(4), 455–472 (2006)
6. Lourenço, H.R., Martin, O.C., Stützle, T.: Iterated local search: Framework and applications. In: Handbook of Metaheuristics, 2nd edn., pp. 363–397. Springer (2010)
7. Martin, O., Otto, S.W., Felten, E.W.: Large-step Markov chains for the traveling salesman problem. Complex Syst. **5**(3), 299–326 (1991)
8. Segura, C., Lugo, L., Miranda, G., Serrano Cárdenas, E.D.: PACE Solver Description: CIMAT_Team. In: IPEC 2024. LIPIcs, vol. 321, pp. 31:1–31:4. Schloss Dagstuhl (2024)
9. Langedal, K., Bentert, M., Blanco, T., Drange, P.G.: PACE Solver Description: LUNCH – Linear Uncrossing Heuristics. In: IPEC 2024. LIPIcs, vol. 321, pp. 34:1–34:4. Schloss Dagstuhl (2024)
10. Boehmer, K., George, L.L., Hauser, F., Palarus, J.: PACE Solver Description: Arcee. In: IPEC 2024. LIPIcs, vol. 321, pp. 33:1–33:4. Schloss Dagstuhl (2024)
11. Nagamochi, H.: On the one-sided crossing minimization in a bipartite graph with large degrees. Theor. Comput. Sci. **332**(1–3), 417–446 (2005)
12. Dujmović, V., Fernau, H., Kaufmann, M.: Fixed parameter algorithms for one-sided crossing minimization revisited. J. Discret. Algorithms **6**(2), 313–323 (2008)
13. Tamassia, R. (ed.): Handbook of Graph Drawing and Visualization. CRC Press (2013)
14. PACE Challenge: One-Sided Crossing Minimization (2024). `https://pacechallenge.org/2024/`