Using Reinforcement Learning to Optimize the Global and Local Crossing Number

Timo Brand
Technical University of Munich, Germany

Henry Förster
Technical University of Munich, Germany

Stephen Kobourov
Technical University of Munich, Germany

Robin Schukrafft
Technical University of Munich, Germany

Markus Wallinger
Technical University of Munich, Germany

Johannes Zink
Technical University of Munich, Germany

Abstract

We present a novel approach to graph drawing based on reinforcement learning for minimizing the global and the local crossing number, that is, the total number of edge crossings and the maximum number of crossings on any edge, respectively. An agent learns how to move a vertex based on a given observation vector. The agent receives feedback in the form of local reward signals tied to crossing reduction. To generate an initial layout, we use a stress-based graph-drawing algorithm. We compare our method against force- and stress-based baseline algorithms as well as three established algorithms for global crossing minimization on a suite of benchmark graphs. The experiments show mixed results: our current algorithm is mainly competitive for the local crossing number.

2012 ACM Subject Classification Theory of computation → Reinforcement learning; Theory of computation → Computational geometry

Keywords and phrases Reinforcement Learning, Crossing Minimization, Local Crossing Number

Digital Object Identifier 10.4230/LIPIcs.GD.2025.56

Category Poster Abstract

Related Version *Full Version*: `https://arxiv.org/abs/2509.06108`

1 Introduction

In *reinforcement learning* (RL), an agent receives a description of its environment, based on which it chooses an action changing the environment The agent receives a *reward*, which is positive for a "good" change or negative otherwise. The agent tries to maximize its reward. RL programs like Alpha(Go)Zero [8, 9] were able to master games like Go, chess, and Shogi by just being taught the rules and objectives of the game. They were not given any strategy. We investigate the following question: *Can we use reinforcement learning to optimize graph drawings with respect to a given graph drawing aesthetics measure?* We present an RL framework for graph drawings to optimize local and global crossings.[1] We perform a quantitative analysis on two graph data sets comparing our models against other methods.

[1] With *local crossings* we refer to the maximum number of crossings an edge has in a straight-line graph drawing. With *global crossings* we refer to the total number of crossings in a straight-line graph drawing.

33rd International Symposium on Graph Drawing and Network Visualization (GD 2025).
Editors: Vida Dujmović and Fabrizio Montecchiani; Article No. 56; pp. 56:1–56:4
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

2 Description of the Approach

To obtain a size-invariant algorithm, we follow an idea of Safarli, Zhou, and Wang [7], who let an agent "sit" on a vertex and use as possible actions moving the vertex one step octagonally.

We provide the agent an observation vector describing the local neighborhood of a vertex v with respect to the eight octants defined by the eight octagonal rays around v. The first eight entries in the observation vector of v count the number of neighboring vertices in each octant, then eight entries count the total number of vertices. The next eight entries describe the distance to the closest neighbor, while the next eight entries describe the distance to the closest vertex. Finally, eight entries describe the amount of crossings that occur on edges incident to v in the respective octants and eight entries their local crossing number.

In each step, the agent has eight actions to choose from, which correspond to moving the vertex in one of the 16 main direction 22.5 degrees apart. The reward function is, for the global crossing number, the number of removed crossings. For the local crossing number, it is a combination of local and global crossing number. The choice of the vertex for the next step depends on a probability distribution weighted by crossings and vertex degree.

3 Experimental Evaluation

We compare our approach on the `rome` graphs [1] (relatively sparse; 10–100 vertices, 9–158 edges) and extended Barabási-Albert (`BA`) graphs [3] (random graphs; 50–150 vertices). An initial drawing used as input to the algorithms is obtained with the Kamada-Kawai algorithm [4]. We use 6,602 `rome` (1,000 `BA`) graphs for training and 1,650 (500) for testing. We record the global and local crossing number and the running time and use a time limit of 900 s; see Figure 1. The experiments ran on a server with a single CPU. The code is written in Python 3.12. The custom environment is implemented on top of the Gymnasium library, and we train an (artificial neural network) PPO agent using the stable-baselines3 library.

We compare our RL method trained for the global (RL(GC)) and local crossing number (RL(LC)) with a range of different methods: the *stress-minimization approach* Kamada-Kawai (KK) and the *spring embedder* Fruchterman-Reingold (FR), implemented in `networkx`, which do not directly optimize for (local) crossing number but generally achieve good results. We also compare with the machine-learning model SmartGD [10]; the authors released a model trained to minimize the crossing number, which we use. Regarding previous heuristic solutions, we compare to the *stochastic gradient descent* method (SGD)^2 [2], to two *geometric local-optimization* approaches presented by Radermacher et al. [6], namely, Vertex Movement (VM) and Edge Insertion (EI), and to the *probabilistic hill-climbing* method Tübingen-Bus (TB). Note that while the latter approach was originally designed for minimizing the number of crossings in upward drawings, we could easily adjust it to drop the upward constraint [5].

4 Conclusion and Future Work

We have presented a post-processing procedure to optimize quantifiable objectives like the global and local crossing number. For the global crossing number, some state-of-the-art approaches achieve better results. This may depend on our current RL implementation. We made some ad-hoc design choices for the observation vector and the action space based on octants; also we chose a specific vertex-selection procedure and reward function. For the local crossing number, our algorithm is among the best-performing methods. Note, however, that we are among the first to explicitly optimize the local crossing number. In the future, we will try to apply more and better-suited RL approaches to graph drawing problems.

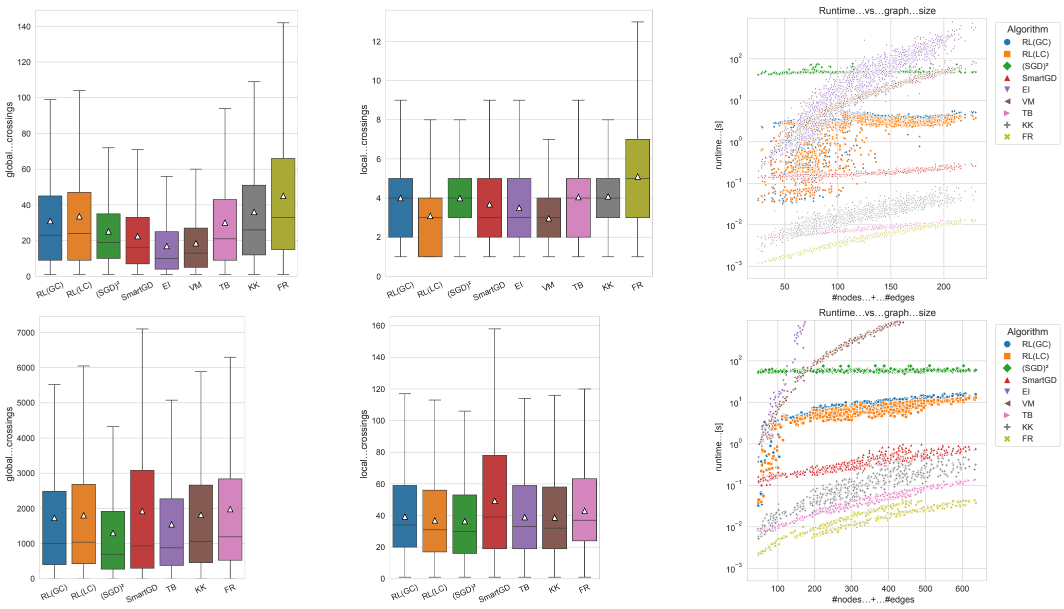


Figure 1 Boxplots for global (left) and local (middle) crossing number on the `rome` graphs (top) and `BA` graphs (bottom). On the right side, there are scatter plots for the running times.

References

1 Rome graphs. URL: `http://graphdrawing.org/data.html`.

2 Reyan Ahmed, Felice De Luca, Sabin Devkota, Stephen Kobourov, and Mingwei-ahmed2022multicriteria Li. Multicriteria scalable graph drawing via stochastic gradient descent, $(SGD)^2$. *IEEE Trans. Vis. Comput. Graph.*, 28(6):2388–2399, 2022. `doi:10.1109/TVCG.2022.3155564`.

3 Réka Albert and Albert-László Barabási. Topology of evolving networks: local events and universality. *Physical review letters*, 85(24):5234, 2000. `doi:10.1103/physrevlett.85.5234`.

4 Tomihisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989. `doi:10.1016/0020-0190(89)90102-6`.

5 Maximilian Pfister. Personal communication, 2025.

6 Marcel Radermacher, Klara Reichard, Ignaz Rutter, and Dorothea Wagner. Geometric heuristics for rectilinear crossing minimization. *J. Exp. Algorithmics*, 24:1–21, 2019. `doi:10.1145/3325861`.

7 Ilkin Safarli, Youjia Zhou, and Bei Wang. Interpreting graph drawing with multi-agent reinforcement learning. *CoRR*, abs/2011.00748, 2020. `arXiv:2011.00748`.

8 David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018. `doi:10.1126/science.aar6404`.

9 David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017. `doi:10.1038/NATURE24270`.

10 Xiaoqi Wang, Kevin Yen, Yifan Hu, and Han-Wei Shen. SmartGD: A GAN-based graph drawing framework for diverse aesthetic goals. *IEEE Trans. Vis. Comput. Graph.*, 30(8):5666–5678, 2023. `doi:10.1109/TVCG.2023.3306356`.

A Poster

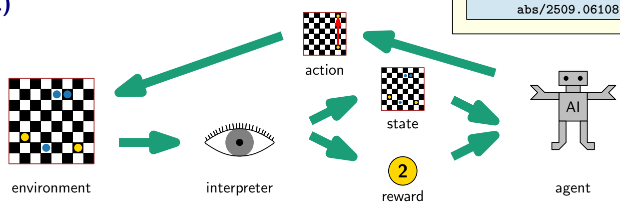
Using Reinforcement Learning to Optimize the Global and Local Crossing Number

Timo Brand, Henry Förster, Stephen Kobourov,
Robin Schukrafft, Markus Wallinger, and Johannes Zink

Full version: https://arxiv.org/abs/2509.06108

Reinforcement learning (RL)

- explore state space
- try to maximize reward
- learn from prior actions
- used in games like chess, go, etc. (e.g., AlphaGo, AlphaZero)
- used to tune large language models (LLMs) like ChatGPT
- powerful generic tool
- challenge: design states, rewards, possible actions, type of agent



Graph drawing as a game

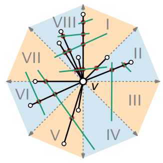
- choose measurable quality metric of a graph drawing like number of crossings or stress
- start with some initial drawing; quality is a number
- give the agent a state derived from the current drawing
- give the agent a set of possible actions
- for each choice of an action, update the graph drawing and re-compute the quality
- give the gain/loss of quality as reward to the agent

Experimental setup

- implemented in Python with the Gymnasium library
- PPO agent trained using stable-baselines3
- Trained on 6,602 `rome` and 1,000 `Barabasi-Albert` graphs, degree-1 vertices and planar graphs removed
- tested on one CPU with a time limit of 900 seconds
- methods do not start with random drawings but with a drawing computed by the Kamada–Kawai method
- 1,650 `rome` graphs and 500 `BA` graphs for testing

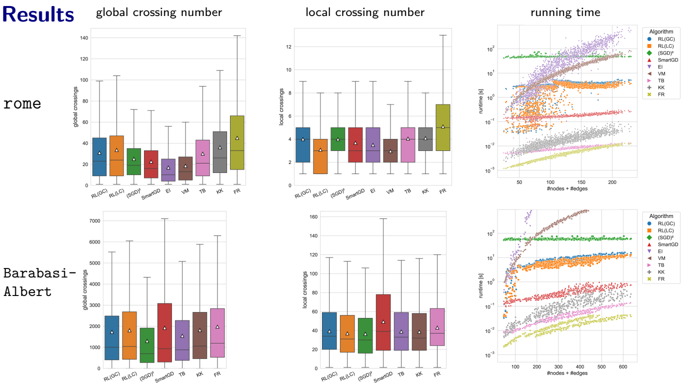
Minimizing the (rectilinear) local crossing number

- for size-invariance, we follow ideas of Safarli, Zhou, and Wang (2020)
- measurable quality metric: maximum number of crossings in which any edge is involved
- start with a random graph drawing
- iterate through the vertices
- give the agent the view of current vertex v:
  - state space: view of the 8 octants between the octagonal lines; for each octant number of neighbors, closest neighbor, closest non-neighbor, number of crossings of incident edges (all values normalized)
  - possible actions: move v along one of the 16 hexadecagonal lines
  - reward: decrease of the local crossing number plus (weighted less) decrease of the global crossing number; small negative reward if no progress is made



octant	I	II	III	IV	V	VI	VII	VIII
neighbors	.1	.2	0	0	.2	.1	0	.4
vertices	.14	.03	0	.11	.08	.2	.27	.17
closest ngb.	.32	.5	0	0	.25	.67	0	.21
closest vtx.	.32	.28	0	.94	.25	.67	.17	.21
crossings	.1	.2	0	0	.1	.3	0	1.0
local cr.	1	2	0	0	1	2	0	4

Results



- RL(GC): RL trained for global crossings
- RL(LC): RL trained for local crossings
- $(SGD)^2$: Stochastic Gradient Descent [1]
- SmartGD: GAN-based deep learning framework [2]
- EI: Edge Insertion algorithm from [3]
- VM: Vertex Movement algorithm from [3]
- TB: TübingenBus, probabilistic hill-climbing [4]
- KK: Kamada-Kawai method
- FR: Fruchterman-Reingold method

[1] R. Ahmed, F. De Luca, S. Devkota, S. Kobourov, and M. Li. Multicriteria scalable graph drawing via stochastic gradient descent, $(SGD)^2$. *IEEE Transactions on Visualization and Computer Graphics*, 2022.
[2] X. Wang, K. Yen, Y. Hu, and H.-W. Shen. SmartGD: A GAN-based graph drawing framework for diverse aesthetic goals. *IEEE Transactions on Visualization and Computer Graphics*, 2023.
[3] M. Radermacher, K. Reichard, I. Rutter, and D. Wagner. Geometric heuristics for rectilinear crossing minimization. *Journal of Experimental Algorithmics (JEA)*, 2019.
[4] Maximilian Pfister. Code for the Uwpard Crossing Minimization GD Contest 2019 and modified to drop the upward constraint. (Personal communication.)

Discussion

- Global crossing number: RL not among the best-performing approaches. This might be due to the design choices in our current RL implementation.
- Local crossing number: RL is among the best-performing methods. But, local crossing number has been hardly investigated and optimized. The other methods are generic or optimizing the global crossing number.
- Running time: large difference between algorithms and how they scale. RL is mid-table and can still handle the little larger `BA` graphs.

Conclusion and future work

- New post-processing procedure for graph drawing problems optimizing quantifiable objectives like global & local crossing number.
- Initial hope: plug in crossing minimization problem to some out-of-the-box RL solution. Turns out: not so obvious how, modeling seems to be crucial for learning success.
- Tune design of this RL approach. Apply more/different RL methods & frameworks to this and other graph drawing problems.