

Initial Placement for Fruchterman–Reingold Force Model with Coordinate Newton Direction

Hiroki Hamaguchi
5th lab
Supervisor: Prof. Akiko Takeda

2024/11/29

Introduction of Graph Drawing

Graph $G = (V, E)$ (vertices V / edges E)

Graph Visualization is an important task.

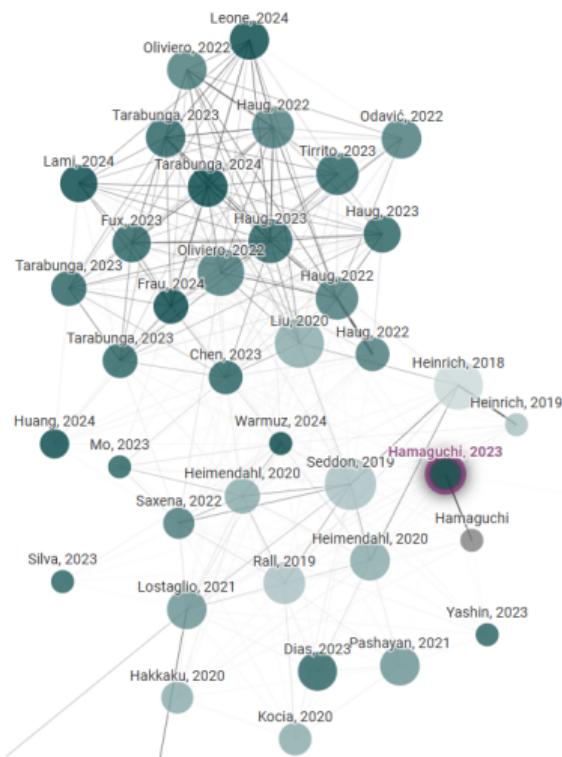
Force-directed graph drawing is a popular method.



Social Network Graph
Designed by [Freepik](#)



Railroad Graph
By [Bernese media](#),
CC BY-SA 3.0



By **CONNECTED PAPERS** ([Link](#))

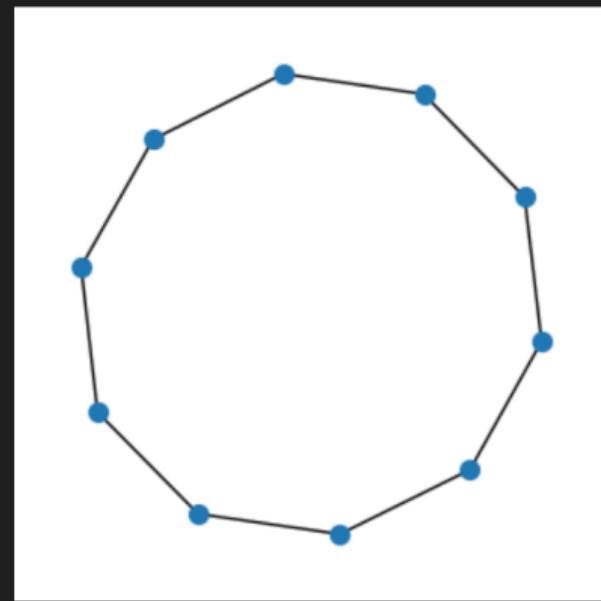
Graph Drawing by NetworkX

NetworkX  is a popular Python library.
nx.draw is the default function.

Fruchterman–Reingold algorithm is implemented.

$|V| = 10$: 0.2 sec / Well Visualized

```
import networkx as nx  
  
G = nx.cycle_graph(10)  
nx.draw(G, node_size=50)  
✓ 0.2s
```



Graph Drawing by NetworkX

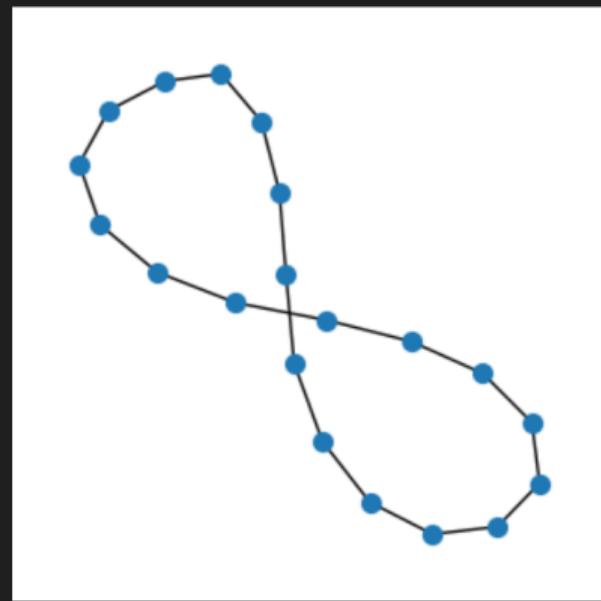
NetworkX  is a popular Python library.
nx.draw is the default function.

Fruchterman–Reingold algorithm is implemented.

$|V| = 10$: 0.2 sec / Well Visualized

$|V| = 20$: 0.2 sec / Tangled?

```
import networkx as nx  
  
G = nx.cycle_graph(20)  
nx.draw(G, node_size=50)  
✓ 0.2s
```



Graph Drawing by NetworkX

NetworkX  is a popular Python library.
nx.draw is the default function.

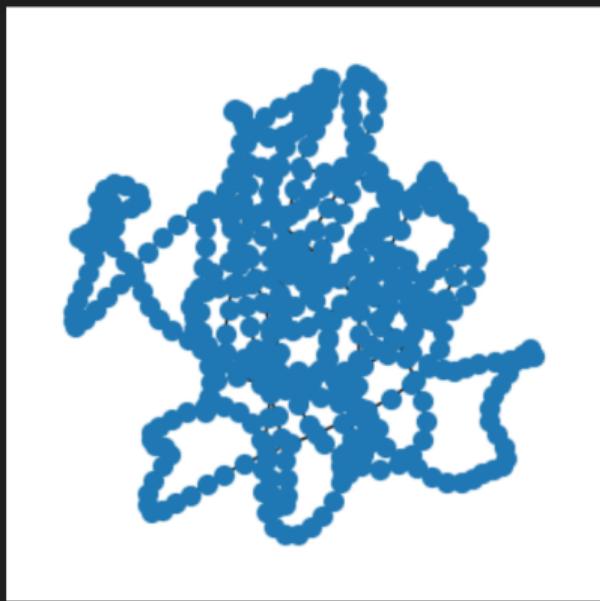
Fruchterman–Reingold algorithm is implemented.

$|V| = 10$: 0.2 sec / Well Visualized

$|V| = 20$: 0.2 sec / Tangled?

$|V| = 500$: 11.5 sec / **WHAT IS THIS???**

```
import networkx as nx  
  
G = nx.cycle_graph(500)  
nx.draw(G, node_size=50)  
✓ 11.5s
```



“Twist” Causes Stagnation

Twist: Unnecessary edge intersections or tangled structures.

Causing stagnation of the simulation process.

$\mathcal{O}(|V|^2)$ per iteration. Slow for large-scale graphs.

Previous Works

L-BFGS (Quasi-Newton Method)

Numerical optimization approach.

Effective for reducing stress in graph layouts.

Limitations

Treats just as a general optimization problem.

Ignored inherent graph structure.

Simulated Annealing (SA)

Stochastic optimization method.

Effective for addressing "twist" issues in graphs.

Improves visualization when combined with FR algorithm.

Limitations

Restricted to unweighted graphs.

Limited to circle placement.

Inefficient due to random swapping.

Ignored sparsity of graphs.

Our Contribution

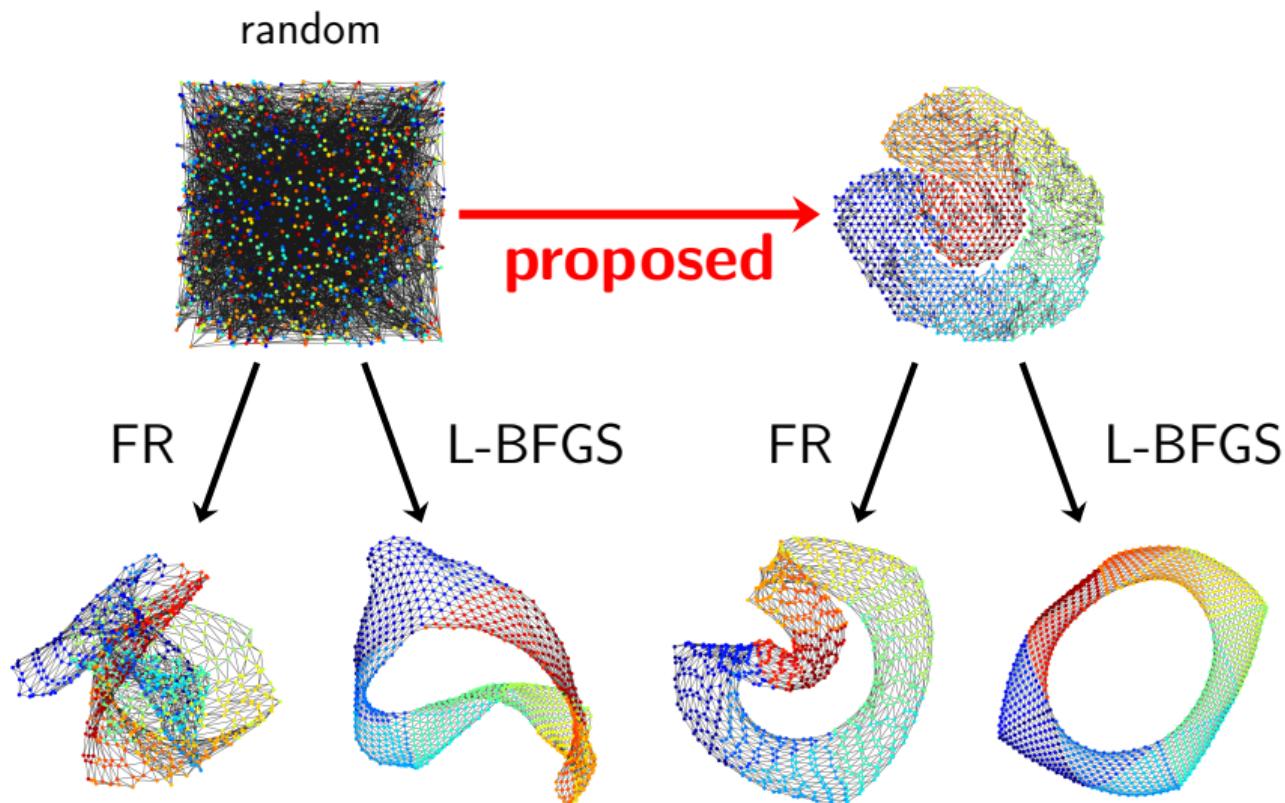


Figure: `jagmesh1` dataset after 50 iterations.

Coordinate Newton's Direction

$f: \mathbb{R}^n \rightarrow \mathbb{R}$: strictly convex function / The second order approximation at x_0 is

$$f(x) \approx f(x_0) + \nabla f(x_0)^\top (x - x_0) + \frac{1}{2}(x - x_0)^\top \nabla^2 f(x_0)(x - x_0).$$

$\operatorname{argmin} x^*$ satisfies

$$\begin{aligned}\nabla f(x_0) + \nabla^2 f(x_0)(x^* - x_0) &= 0 \\ \iff x^* &= x_0 - \nabla^2 f(x_0)^{-1} \nabla f(x_0).\end{aligned}$$

$d = -\nabla^2 f(x_0)^{-1} \nabla f(x_0)$: **the Newton direction (too expensive ...)**

coordinate Newton direction: $d_i = -\nabla^2 f_i(x_i)^{-1} \nabla f_i(x_i)$ **Computable!**

Definition of FR Force Model

Historically, FR force model seeks an **equilibrium of two forces**:

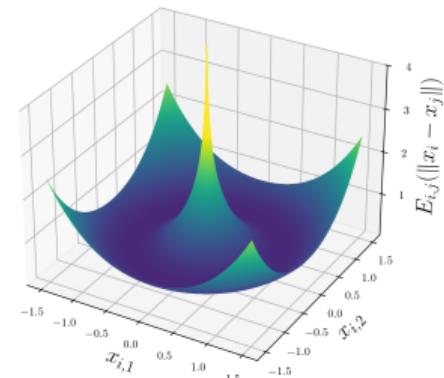
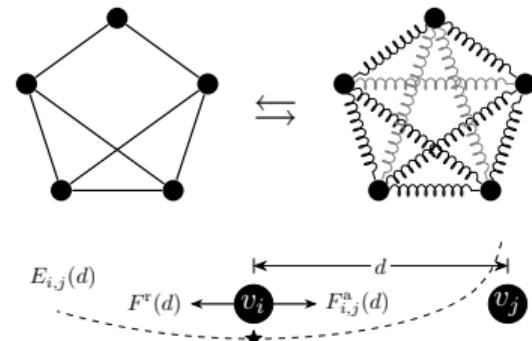
$$F_{i,j}^a(d) := \frac{w_{i,j}d^2}{k}, \quad F^r(d) := -\frac{k^2}{d}.$$

Its scalar potential, energy, is defined as

$$E_{i,j}^a(d) := \int_0^d F_{i,j}^a(r) dr = \frac{w_{i,j}d^3}{3k}, \quad E^r(d) := \int_\infty^d F^r(r) dr = -k^2 \log d,$$
$$E_{i,j}(d) := E_{i,j}^a(d) + E^r(d). \quad (1)$$

Seek equilibrium \Leftrightarrow **find local minimum of $f(X)$ (non-convex)**:

$$\underset{X \in \mathbb{R}^{2 \times n}}{\text{minimize}} \quad f(X) := \sum_{i < j} E_{i,j}(\|x_i - x_j\|). \quad (2)$$



Simplify the Problem

Graphs have sparsity : $|E| \ll |V|^2$ / the energy function $E_{i,j}$ is $-k^2 \log d$ for all $\{i,j\} \notin E$.

$$\underset{\substack{\text{minimize} \\ X \in \mathbb{R}^{2 \times n}}}{\sum_{\{i,j\} \in E}} \frac{w_{i,j} \|x_i - x_j\|^3}{3k} - \sum_{i < j} k^2 \log \|x_i - x_j\|. \quad (3)$$

Converting the second term into a constraint (obj: $\mathcal{O}(|E|)$ terms from $\mathcal{O}(|V|^2)$ terms)

$$\underset{\substack{\text{minimize} \\ X \in \mathbb{R}^{2 \times n}}}{f^a(X) := \sum_{\{i,j\} \in E}} \frac{w_{i,j} \|x_i - x_j\|^3}{3k} \quad (4)$$

subject to $\|x_i - x_j\| \geq \epsilon, \quad \forall \{i,j\} (i < j)$

where ϵ is a suitably chosen positive constant.

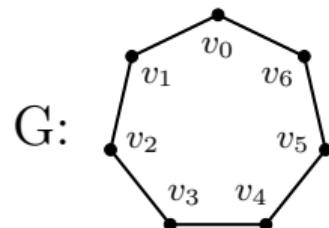
This conversion does not lose the essence of the problem too much because $E^r(d) = -k^2 \log d$ is monotonically decreasing.

Discrete Optimization

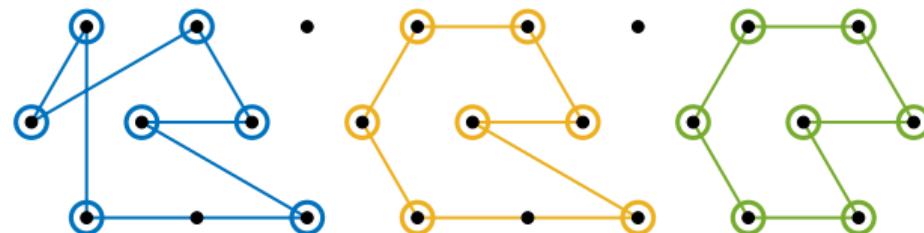
problem (4) still involves $\mathcal{O}(|V|^2)$ constraints. By **fixing** the possible point placement to Q (any two points apart ϵ), we can **skip the check of constraints** in the problem (4).

$$\begin{aligned} \text{minimize}_{\pi: V \rightarrow Q} \quad & \sum_{\{i,j\} \in E} \frac{w_{i,j} \|\pi(v_i) - \pi(v_j)\|^3}{3k}, \\ \text{subject to} \quad & \pi \text{ is injective.} \end{aligned} \tag{5}$$

Skipping the check of $\mathcal{O}(|V|^2)$ constraints
→ Reducing the complexity to $\mathcal{O}(|E|)$
→ **speedup**.



projections:
 π_1, π_2, π_3



Proposed Algorithm 1

We solve the problem (5) using the coordinate Newton direction.

Let the objective function $f_i^a(x_i)$ corresponding to a vertex v_i be

$$f_i^a(x_i) := \sum_{j \neq i} \frac{w_{i,j} \|x_i - x_j\|^3}{3k}.$$

Its gradient and Hessian matrix are

$$\nabla f_i^a(x_i) = \sum_{j \neq i} \frac{w_{i,j} \|x_i - x_j\|}{k} (x_i - x_j),$$

$$\nabla^2 f_i^a(x_i) = \sum_{j \neq i} \frac{w_{i,j} \|x_i - x_j\|}{k} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \sum_{j \neq i} \frac{w_{i,j}}{k \|x_i - x_j\|} (x_i - x_j)(x_i - x_j)^\top.$$

Since f_i^a is convex, we can use the coordinate Newton direction. This is a large difference from the functions $E_{i,j}(\|\cdot - x_j\|)$ in Eq. (1) and $f^a(\cdot)$ in Prob. (4), which are not convex.

proposed algorithm 2

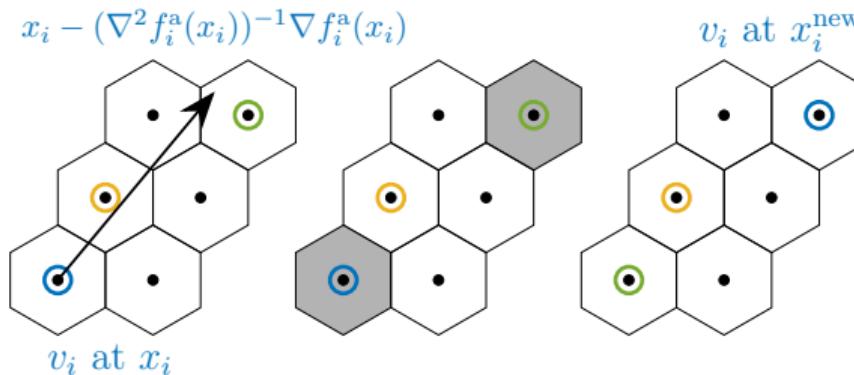
The ordinary updated rule with the coordinate Newton direction is

$$x_i - \nabla^2 f_i^a(x_i)^{-1} \nabla f_i^a(x_i).$$

x_i^{new} may not be in the hexagonal lattice Q^{hex} . Thus, we must round to the nearest point in Q^{hex} . We empirically found that adding a random vector to the Newton direction is effective.

$$x_i^{\text{new}} \leftarrow \text{round}\left(x_i - \nabla^2 f_i^a(x_i)^{-1} \nabla f_i^a(x_i) + t \cdot \text{rand}\right),$$

where $\text{round}(\hat{x})$ denotes the operation assigning \hat{x} to the nearest point in the hexagonal lattice Q^{hex} , rand is a random vector with a unit norm, and t is a parameter controlling the randomness.



optimal Scaling

We can find optimal scaling factor c^* . We scale $X = (x_1, \dots, x_n)$ as $x_i \leftarrow cx_i$ for all i . This problem is to minimize $\phi(c)$:

$$\phi(c) := \left(\sum_{\{i,j\} \in E} \frac{w_{i,j}(c\|x_i - x_j\|)^3}{3k} \right) - k^2 \sum_{i < j} \log(c\|x_i - x_j\|)$$

The function $\phi(c)$ is convex, and we can find the optimal scaling factor c^* by

$$c^* = \left(\frac{k^2 n(n-1)}{3 \sum_{\{i,j\} \in E} \frac{w_{i,j} \|x_i - x_j\|^3}{k}} \right)^{1/3}. \quad (6)$$

This value can be computed in $\mathcal{O}(|E|)$ complexity.

Notably, the optimal solution to Prob. (5) is invariant under scaling. Thus, we can select any ϵ to define the hexagonal lattice Q^{hex} as far as we scale the placement by c^* as post-processing.

Pseudo Code

Algorithm 1: Proposed algorithm as initial placement

Define parameters $N_{\text{iter}}^{\text{CN}}$, t , Δt and hexagonal lattice Q^{hex} ;

Set π as a injection from V to Q^{hex} randomly;

for $j \leftarrow 0$ **to** $N_{\text{iter}}^{\text{CN}}$ **do**

$v_i \leftarrow$ randomly selected vertex from V ;

$x_i \leftarrow \pi(v_i)$;

$x_i^{\text{new}} \leftarrow \text{round}(x_i - \nabla^2 f_i(x_i)^{-1} \nabla f_i(x_i) + t \cdot \text{rand})$;

if $\exists v_j$ s.t. $\pi(v_j) = x_i^{\text{new}}$ **then**

Swap v_i and v_j in π ;

else

$\pi(v_i) \leftarrow x_i^{\text{new}}$;

$x_i \leftarrow \pi(v_i)$ for all $v_i \in V$;

$c^* \leftarrow$ optimal scaling factor by Eq. (6);

$x_i \leftarrow c^* x_i$ for all $v_i \in V$;

return X

Experiments Result (overall)

As a dataset, we used matrices from Sparse Matrix Collection [1], in total 124 graphs.

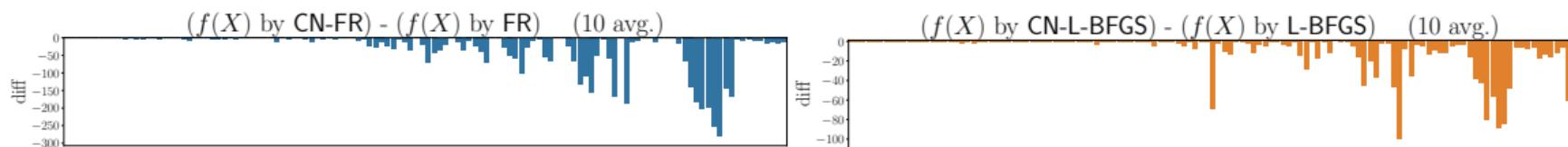


Figure: Comparison of the proposed initialization (CN) with random initialization (no prefix).

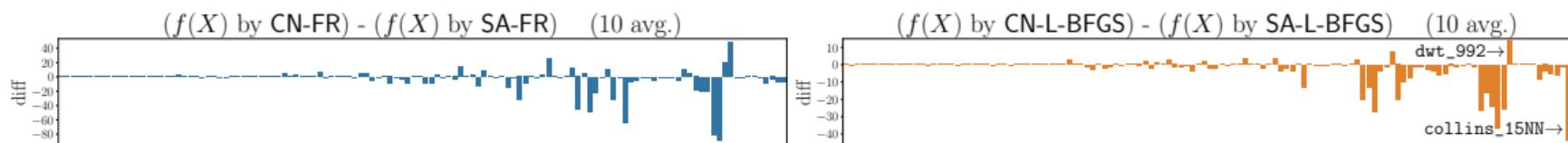
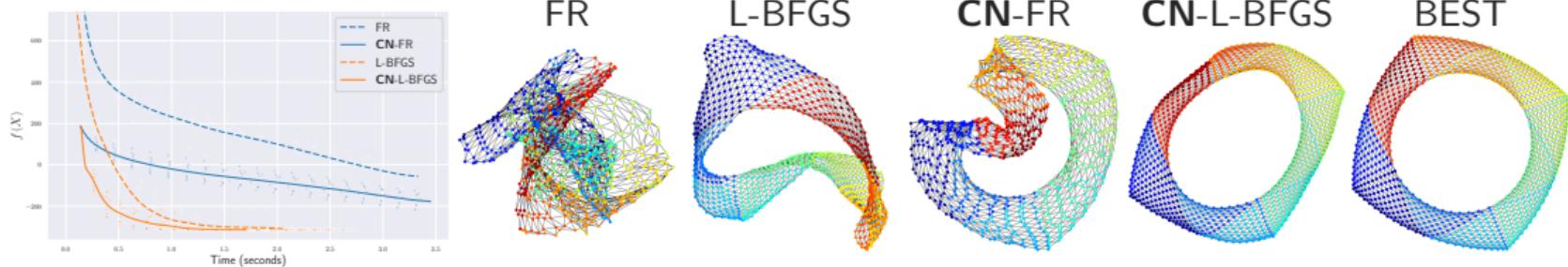


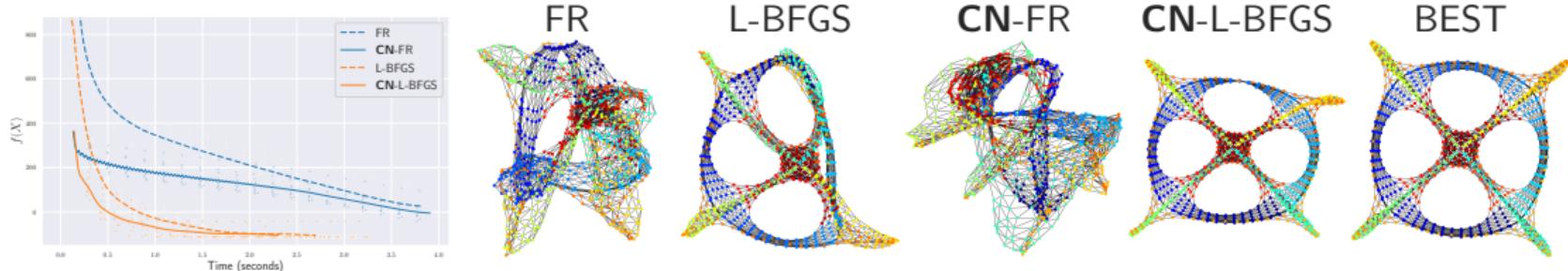
Figure: Comparison of the proposed initialization (CN) with circle initialization (SA) in Ref. [2].

Experiments Result (individual 1)

jagmesh1 ($|V| = 936$, $|E| = 2664$, sparsity = 0.609%) Figures are at 50 iterations.

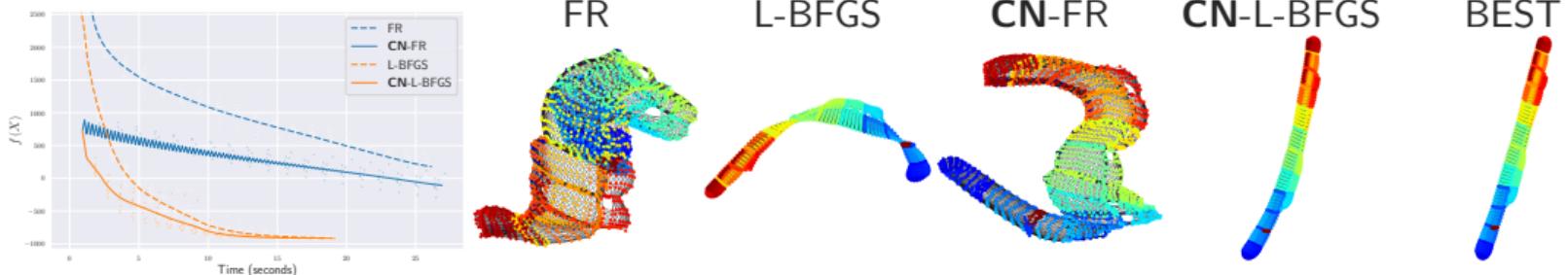


dwt_1005 ($|V| = 1005$, $|E| = 3808$, sparsity = 0.755%) Figures are at 100 iterations.

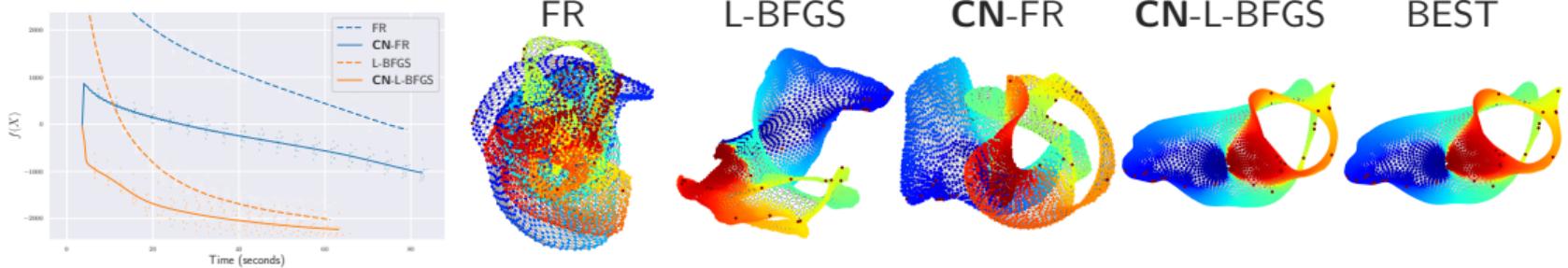


Experiments Result (individual 2)

dwt_2680 ($|V| = 2680$, $|E| = 11173$, sparsity = 0.311%) Figures are at 150 iterations.



3elt ($|V| = 4720$, $|E| = 13722$, sparsity = 0.123%) Figures are at 150 iterations.



Comparision with SA method

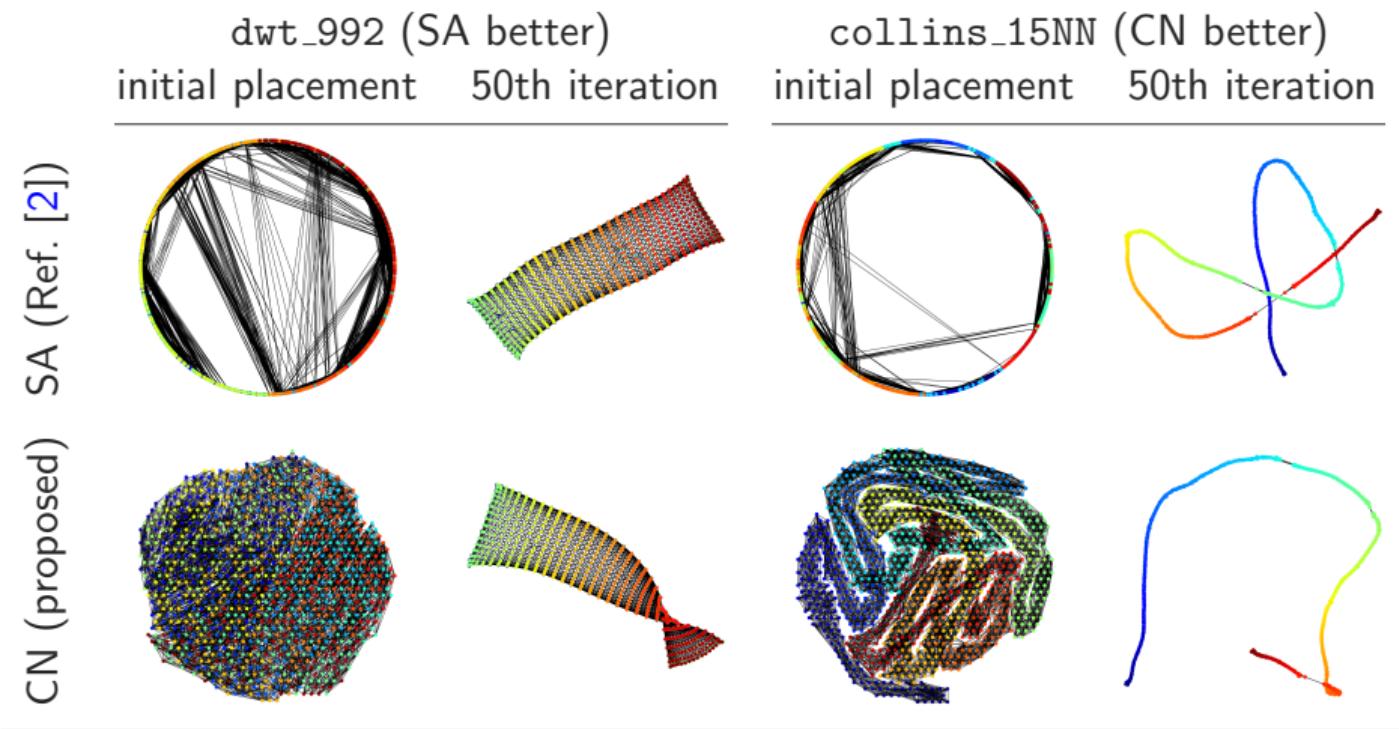


Figure: Visualization results showing initial and 50th iteration placements for dwt_992 and collins_15NN.

Rationale of the Proposed Method

Q: Why we cannot directly apply coordinate descent to the problem?

A: The ignorance of other vertex movements.

Discrete optimization problem

→ at least ϵ movement

→ more efficient than direct application.

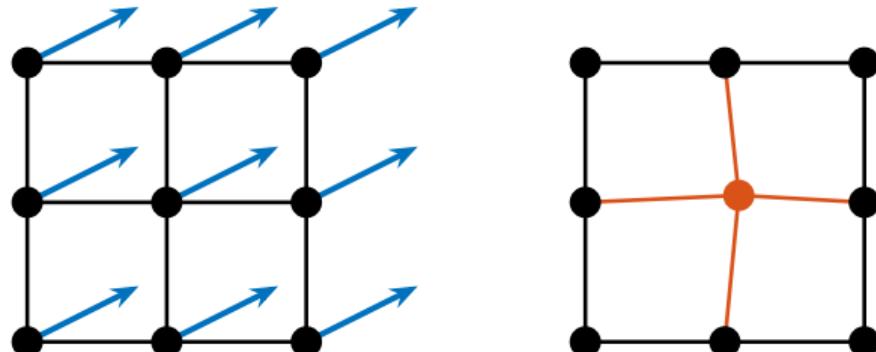


Figure: The ignorance of other vertex movements. Although the blue arrows show the forces in this situation, the red vertex barely moves by the coordinate Newton direction.

Future Work

sfdp in Graphviz

Scalable Force-Directed Placement

Multilevel approach

Barnes–Hut algorithm
(Q-tree)[3] [4]

These methods are compatible with the proposed method.

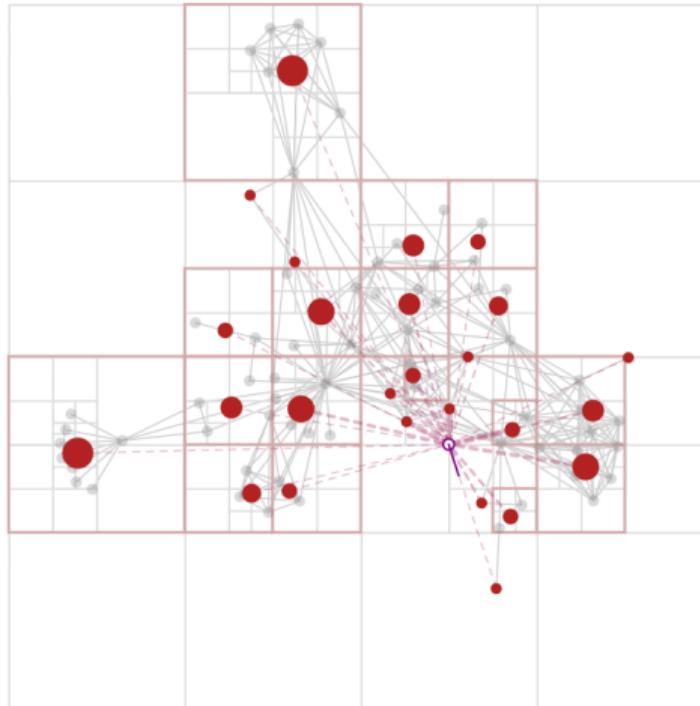


Figure: <https://jheer.github.io/barnes-hut/>

“objective functions arising from graphs” [5]

$$f(X) = \sum_{\{i,j\} \in E} f_{i,j}(x_i, x_j) + \lambda \sum_{i=1}^n \Omega_i(x_i)$$

General optimization problem arising from graphs.

stochastic coordinate descent [5] is a popular method.

Can we use coordinate Newton direction to these problems?

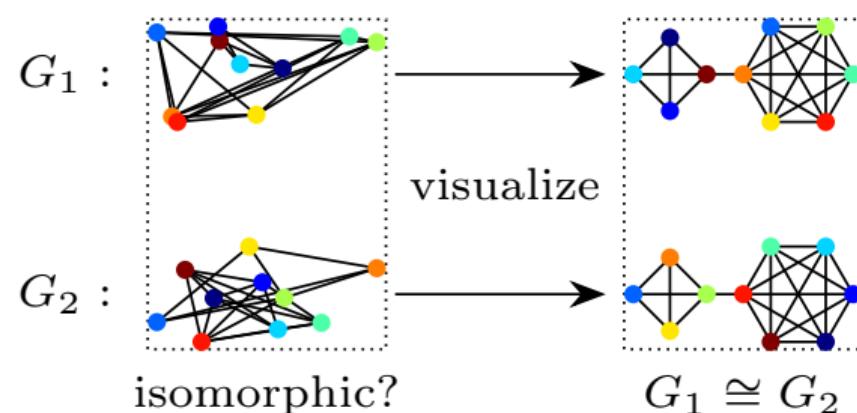
graph isomorphism problem

Drawing graph symmetry is at least as difficult as the graph isomorphism problem [6].

Continuous relaxation → optimization on

Riemannian manifolds [7, 7]

Can we utilize coordinate Newton direction to Riemannian optimization?



End of the Presentation

Summary

Fruchterman–Reingold layout is a tough problem

Hexagonal Lattice + coordinate Newton direction

initialization + L-BFGS gives the best result

Acknowledgement

I would like to express my sincere gratitude to **Pierre-Louis Poirion**, **Andi Han**, and **Naoki Marumo**.

Reference I

- [1] T. A. Davis and Y. Hu, "The University of Florida sparse matrix collection," *ACM Transactions on Mathematical Software (TOMS)*, vol. 38, no. 1, pp. 1–25, 2011.
- [2] F. Ghassemi Toosi, N. S. Nikolov, and M. Eaton, "Simulated Annealing as a Pre-Processing Step for Force-Directed Graph Drawing," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '16 Companion. Association for Computing Machinery, 2016, pp. 997–1000. [Online]. Available: <https://dl.acm.org/doi/10.1145/2908961.2931660>
- [3] Y. Hu, "Efficient, high-quality force-directed graph drawing," *The Mathematica journal*, vol. 10, pp. 37–71, 2006. [Online]. Available: <https://api.semanticscholar.org/CorpusID:14599587>
- [4] J. Barnes and P. Hut, "A hierarchical $O(N \log N)$ force-calculation algorithm," *Nature*, vol. 324, no. 6096, pp. 446–449, 1986. [Online]. Available: <https://www.nature.com/articles/324446a0>
- [5] B. Recht and S. J. Wright, "Optimization for modern data analysis," 2019. [Online]. Available: https://people.eecs.berkeley.edu/~brecht/opt4ml_book/
- [6] P. Eades, "A heuristic for graph drawing," *Congressus numerantium*, vol. 42, no. 11, pp. 149–160, 1984.
- [7] S. Klus and P. Gelß, "Continuous optimization methods for the graph isomorphism problem," 2023. [Online]. Available: <http://arxiv.org/abs/2311.16912>