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

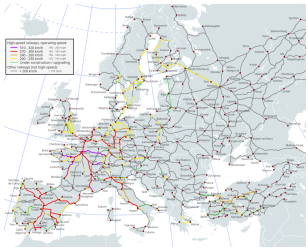
Hiroki Hamaguchi
5th lab
Supervisor: Prof. Akiko Takeda

2024/11/29

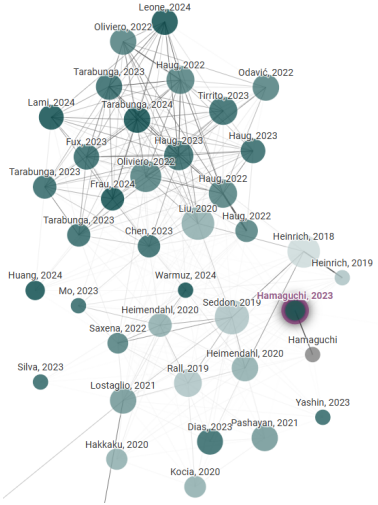
Introduction of Graph Drawing

Graph $G = (V, E)$ (vertices V / edges E)
Graph Drawing is an fundamental 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

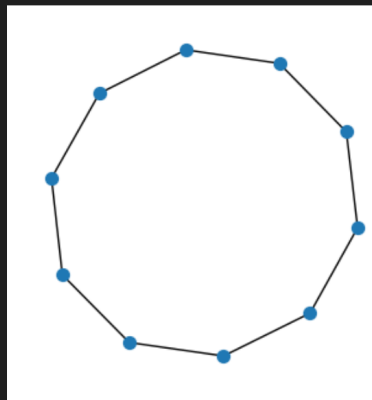
NetworkX [1] is a popular Python library.
`nx.draw` is the default function.
Fruchterman–Reingold (FR) algorithm works.
(with 50 iterations)

$|V| =$ 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

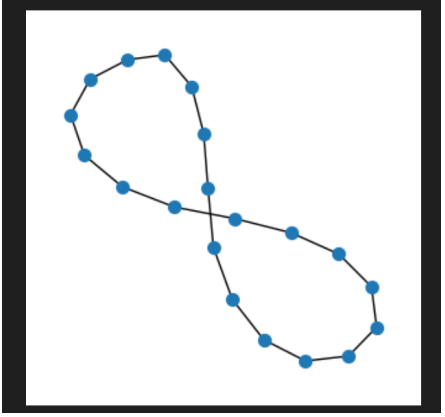
NetworkX [1] is a popular Python library.
`nx.draw` is the default function.
Fruchterman–Reingold (FR) algorithm works.
(with 50 iterations)

$|V| =$ 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

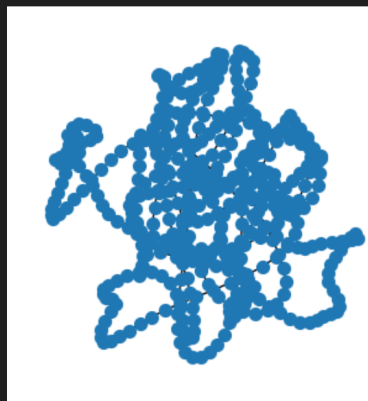
NetworkX [1] is a popular Python library.
`nx.draw` is the default function.
Fruchterman–Reingold (FR) algorithm works.
(with 50 iterations)

$|V| =$ 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

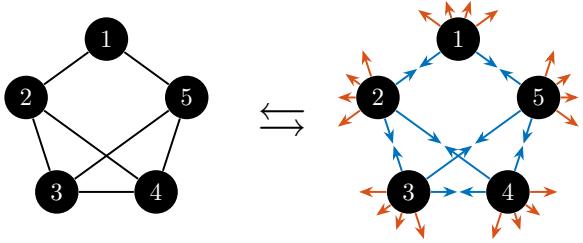


Fruchterman–Reingold Force Model

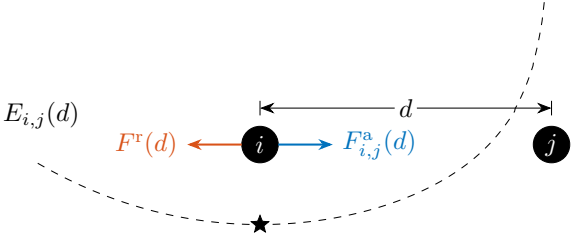
The **FR force model** uses a force model [2] with **a**ttractive force and **r**epulsive force:

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

The **FR algorithm** seeks an **equilibrium** of two kind forces:



$$w_{i,j} > 0 \iff \{i,j\} \in E$$



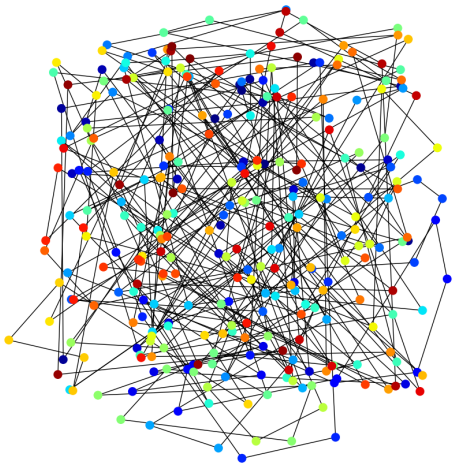
$$d = \|x_i - x_j\|$$

"Twist" Causes Stagnation

Twist: unnecessary folded and tangled structures [3, 4].

$\rightarrow$ Causing stagnation of the simulation process.

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



Previous Works

L-BFGS (Quasi-Newton Method) [5]

Numerical optimization approach.

Effective for reducing stress.

Limitations

Treats just as a general optimization problem.

Ignored inherent graph structure.

$$\min_{X \in \mathbb{R}^{2 \times n}} f(X) \rightarrow \min_{\overline{X} \in \mathbb{R}^{2n}} \overline{f}(\overline{X})$$

Our Aim

Provide an initial placement.

Accelerate the subsequent optimization.

Simulated Annealing (SA) [6]

Providing an initial placement

Effective for addressing "twist" issues.

Limitations

Restricted to unweighted graphs.

Limited to circle placement.

Inefficient due to random swapping.

Ignored sparsity of graphs.

Our Aim

Improve the strategy.

Extend the applicability.

Our Contribution

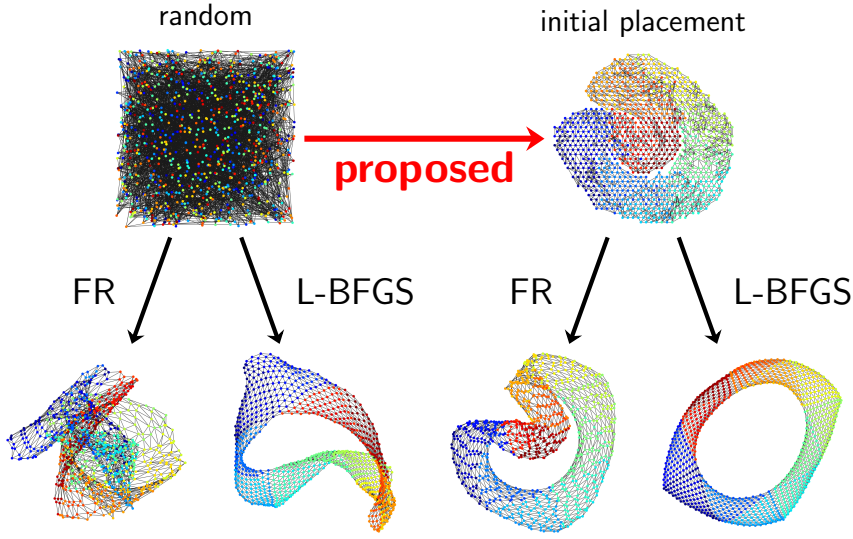
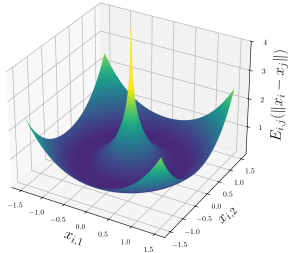


Figure: `jagmesh1` dataset after 50 iterations.

Formulation of the Problem



The two forces in the FR force model:

$$F^{\mathrm{a}}_{i,j}(d) \coloneqq \frac{w_{i,j} d^2}{k}, \quad F^{\mathrm{r}}(d) \coloneqq -\frac{k^2}{d}.$$

Its scalar potential, energy, is defined as

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

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

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

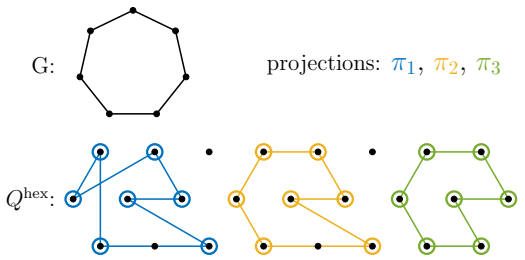
Simplify the Problem (1/2)

Instead of the problem (7), we solve:

$$\begin{aligned} &\underset{\pi\colon V \to Q^{\mathrm{hex}}}{\text{minimize}} && \sum_{\{i,j\}\in E} \frac{w_{i,j}\|\pi(i)-\pi(j)\|^3}{3k}, \\ &\text{subject to} && \pi \text{ is injective,} \end{aligned} \tag{3}$$

where

$$Q^{\mathrm{hex}} := \left\{ \left(q + \tfrac{1}{2}r, \tfrac{\sqrt{3}}{2}r \right) \;\middle|\; q \in \mathbb{Z}, r \in \mathbb{Z} \right\}.$$



We explain the reason. We simplify the problem (7).
Graphs have sparsity $|E| \ll |V|^2$. We first separate $f(X)$ into two terms.

$$\underset{X \in \mathbb{R}^{2\times n}}{\text{minimize}} \quad \sum_{\{i,j\}\in E} \frac{w_{i,j}\|x_i - x_j\|^3}{3k} - \sum_{i<j} k^2 \log \|x_i - x_j\|. \tag{4}$$

Simplify the Problem (2/2)

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

$$\begin{aligned} &\underset{X \in \mathbb{R}^{2\times n}}{\text{minimize}} && f^{\mathrm{a}}(X) \coloneqq \sum_{\{i,j\}\in E} \frac{w_{i,j}\|x_i - x_j\|^3}{3k} \\ &\text{subject to} && \|x_i - x_j\| \geq \epsilon, \quad \forall \{i,j\}\,(i<j) \end{aligned} \tag{5}$$

where ϵ is a constant. $-k^2 \log \|x_i - x_j\| \to \|x_i - x_j\| \geq \epsilon$ does not lose the problem essence.
Fix the point placement to Q (any two points apart ϵ) $\to$ **Skip the check of constraints**.

$$\begin{aligned} &\underset{\pi\colon V \to Q}{\text{minimize}} && \sum_{\{i,j\}\in E} \frac{w_{i,j}\|\pi(v_i) - \pi(v_j)\|^3}{3k} \\ &\text{subject to} && \pi \text{ is injective}. \end{aligned} \tag{6}$$

Only treat attractive forces. Thus, the points set Q should be as dense as possible.
$\to$ **closet packing** (hexagonal lattice Q^{hex}).

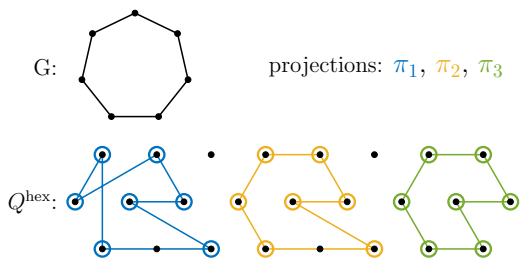
Summary of the First Half

The problem is

$$\underset{X \in \mathbb{R}^{2\times n}}{\text{minimize}} \quad f(X) \coloneqq \sum_{i<j} E_{i,j}(\|x_i - x_j\|) = \sum_{\{i,j\}\in E} \frac{w_{i,j}\|x_i - x_j\|^3}{3k} - \sum_{i<j} k^2 \log \|x_i - x_j\|. \tag{7}$$

We simplify the problem as

$$\begin{aligned} &\underset{\pi\colon V \to Q^{\text{hex}}}{\text{minimize}} && \sum_{\{i,j\}\in E} \frac{w_{i,j}\|\pi(i) - \pi(j)\|^3}{3k}, \\ &\text{subject to} && \pi \text{ is injective}, \end{aligned} \tag{8}$$



We will explain how to solve in the second half.

Base of Proposed Algorithm - Stochastic Coordinate Descent

Let $f : \mathbb{R}^n \to \mathbb{R}$ be strictly convex. The second order approximation at x_0 is

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

The argmin x^* satisfies

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

Although it is effective, computing the Newton direction is **too expensive**...

We use the **stochastic coordinate descent** with the **coordinate Newton direction**.
Let $f_i(x_i)$ be the limitation to the i-th coordinate (randomly selected).
The coordinate Newton direction: $d_i = -\nabla^2 f_i(x_i)^{-1} \nabla f_i(x_i)$. $\leftarrow$ **Cheap! Computable!**

Proposed Algorithm (1/3) - Coordinate Newton Direction

We solve the problem (8) using the coordinate Newton direction.
Let the objective function $f_i^{\mathrm{a}}(x_i)$ corresponding to a vertex v_i be

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

Its gradient and Hessian matrix are

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

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

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 $f^{\mathrm{a}}(\cdot)$ in Prob. (5), which are not convex.

Proposed Algorithm (2/3) - Update Rule

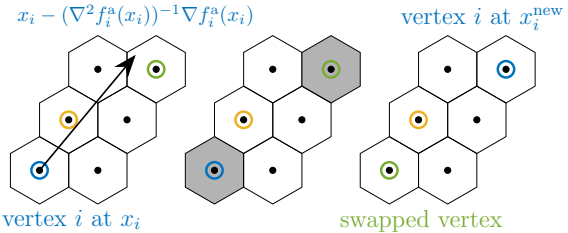
The ordinary updated rule with the coordinate Newton direction is

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

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.

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

where $\mathrm{round}(\hat{x})$ 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.



Proposed Algorithm (3/3) - Optimal Scaling

We can find optimal scaling factor c^*. We scale $X = (x_1, \ldots, x_n)$ 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

$$c^* = \left(\frac{k^2 n(n-1)}{3 \sum_{\{i,j\} \in E} \frac{w_{i,j} \|x_i - x_j\|^3}{k}} \right)^{1/3}. \tag{9}$$

This value can be computed in $\mathcal{O}(|E|)$ complexity.
Notably, the optimal solution to Prob. (8) 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 $N_{\text{iter}}^{\text{CN}}, t, \Delta t$ 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. (9);
$x_i \leftarrow c^* x_i$ for all $v_i \in V$;
return X

Experiments Result (overall)

As a dataset, we used matrices from Sparse Matrix Collection [7], in total 124 graphs.

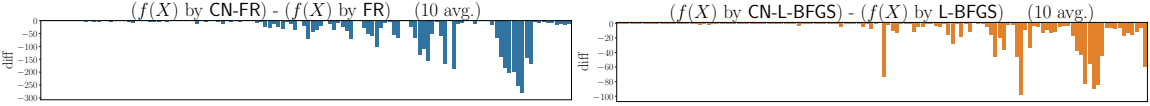


Figure: Comparison of the proposed initialization (CN) with random initialization (no prefix).

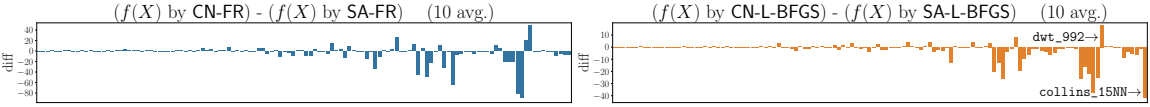
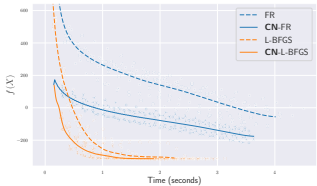


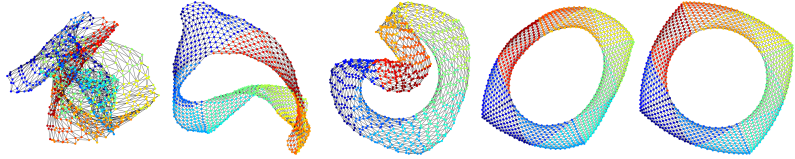
Figure: Comparison of the proposed initialization (CN) with circle initialization (SA) in Ref. [6].

Experiments Result (individual 1)

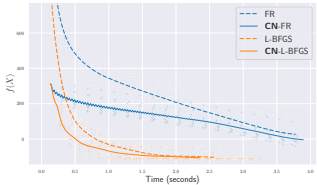
`jagmesh1` ($|V| = 936, |E| = 2664, \text{sparsity} = 0.609\%$) Figures are at 50 iterations.



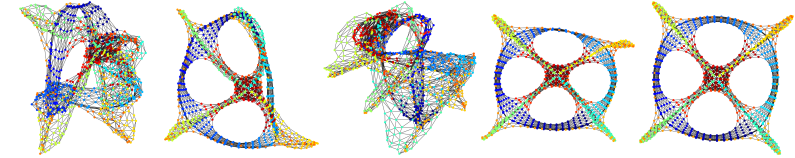
FR L-BFGS **CN**-FR **CN**-L-BFGS BEST



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

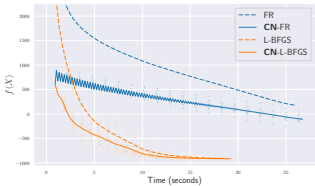


FR L-BFGS **CN**-FR **CN**-L-BFGS BEST

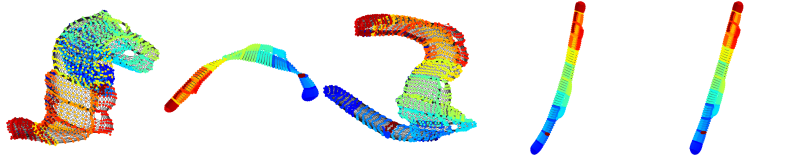


Experiments Result (individual 2)

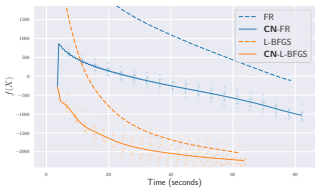
`dwt_2680` ($|V| = 2680, |E| = 11173, \text{sparsity} = 0.311\%$) Figures are at 150 iterations.



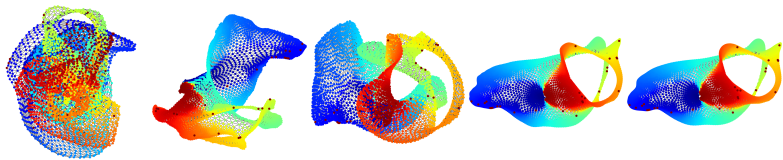
FR L-BFGS **CN**-FR **CN**-L-BFGS BEST



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



FR L-BFGS **CN**-FR **CN**-L-BFGS BEST



Comparision with SA method

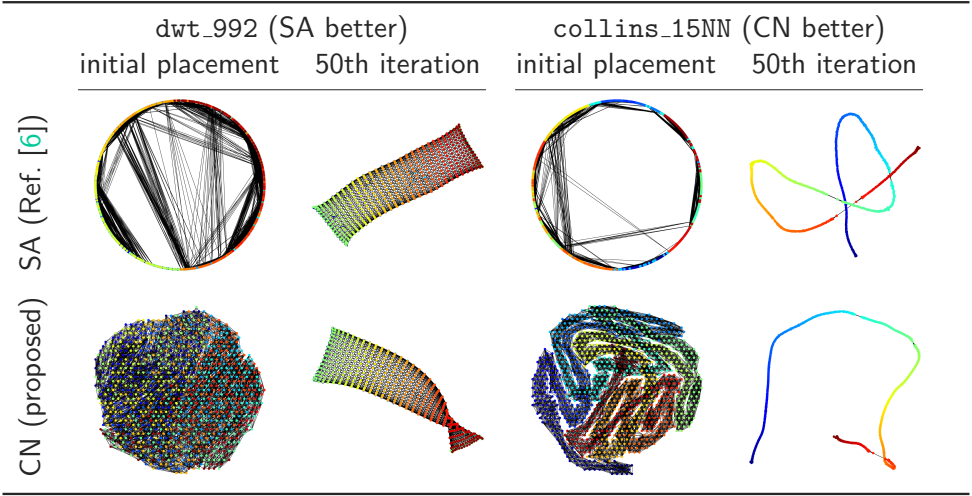


Figure: Visualization results showing initial and 50th iteration placements for `dwt_992` and `collins_15NN`.

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
$\rightarrow$ at least ϵ movement
$\rightarrow$ more efficient than direct application.

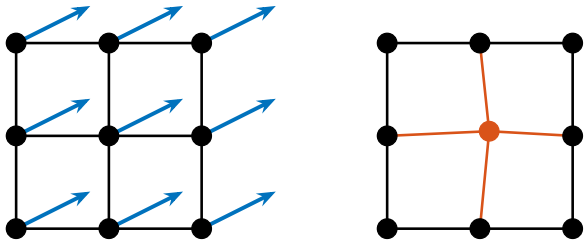


Figure: The ignorance of other vertex movements. Although the blue arrows show the forces in this situation, the red vertex barely moves by the coordinate Newton direction.

Future Work

`sfdp` **in Graphviz**

- Scalable Force-Directed Placement
- **Multilevel** approach
- Barnes–Hut algorithm (Q-tree)[8] [9]
- These methods are compatible with the proposed method.

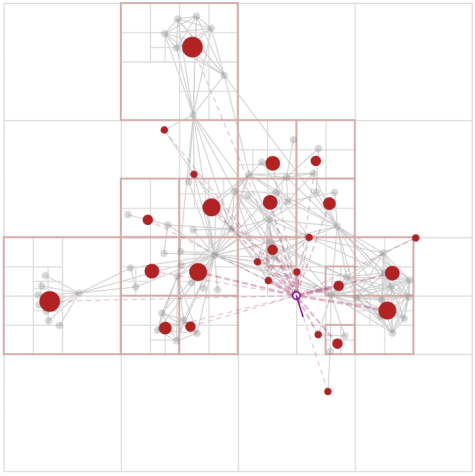


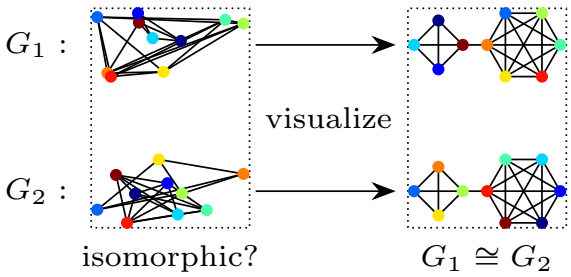
Figure: https://jheer.github.io/barnes-hut/

“objective functions arising from graphs” [10]

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

General optimization problem arising from graphs.
stochastic coordinate descent [10] 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 [11].
Continuous relaxation $\rightarrow$ optimization on Riemannian manifolds [12, 12]
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] A. Hagberg, P. J. Swart, and D. A. Schult, "Exploring network structure, dynamics, and function using NetworkX," Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

[2] T. M. J. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Software: Practice and Experience*, vol. 21, no. 11, pp. 1129–1164, 1991. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.4380211102

[3] T. L. Veldhuizen, "Dynamic Multilevel Graph Visualization," 2007. [Online]. Available: http://arxiv.org/abs/0712.1549

[4] S.-H. Cheong and Y.-W. Si, "Snapshot Visualization of Complex Graphs with Force-Directed Algorithms," in *2018 IEEE International Conference on Big Knowledge (ICBK)*, 2018, pp. 139–145. [Online]. Available: https://ieeexplore.ieee.org/document/8588785/?arnumber=8588785

[5] H. Hosobe, "Numerical optimization-based graph drawing revisited," in *2012 IEEE Pacific Visualization Symposium*, 2012, pp. 81–88.

[6] 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

Reference II

[7] 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.

[8] 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

[9] 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

[10] B. Recht and S. J. Wright, "Optimization for modern data analysis," 2019. [Online]. Available: https://people.eecs.berkeley.edu/~brecht/opt4ml_book/

[11] P. Eades, "A heuristic for graph drawing," *Congressus numerantium*, vol. 42, no. 11, pp. 149–160, 1984.

[12] S. Klus and P. Gelß, "Continuous optimization methods for the graph isomorphism problem," 2023. [Online]. Available: http://arxiv.org/abs/2311.16912