Vol.2011-HCI-144 No.16
2011/7/29

数値最適化による力学的グラフ可視化手法

細　部　博　史[†1]

グラフは事物の関係を表現する手段であり，その可視化はユーザインタフェースにおける有用な要素技術である．これまでに無向グラフの可視化のための力学的手法に関する多数の研究がなされており，これらは主にシミュレーションによるものと数値最適化によるものに分類できるが，前者に比べると後者は十分に研究されていない．本研究では数値最適化による新しい力学的グラフ可視化手法を構築し，1 万ノード規模のグラフに適用する．

A Force-Directed Graph Drawing Method Using Numerical Optimization

Hiroshi Hosobe[†1]

Graphs provide abstract relationships between objects, and graph drawing is useful for construction of user interfaces. Researchers have been studying force-directed methods for drawing undirected graphs. While most methods are based on simulation, others adopt numerical optimization. This technical report presents a numerical optimization-based method for force-directed graph drawing, and describes the result of its application to graphs with approximately ten thousand nodes.

1. はじめに

グラフは事物の関係を表現する手段である．グラフにおいて事物は**ノード**として表現され，2 つの事物の間の関係は，対応するノードを接続する**エッジ**として表現される．グラフはノードやエッジに関連する構造や属性に基づき，木，有向グラフ，**無向グラフ**などのクラスに分類される．グラフの**可視化**はその構造の直観的理解を支援するものであり，ユーザインタフェースにおける有用な要素技術である．グラフの可視化手法は通常，特定のクラスのグラフを対象として構成される．

無向グラフの可視化のため，これまでに**力学的手法**に関する多数の研究がなされている．力学的手法はノード同士が引力・斥力を及ぼし合う仮想的な物理系をグラフから構築し，その系の平衡状態を計算するもである．一般に力学的手法は**モデル**と**アルゴリズム**の 2 つの要素から構成される[2]．モデルは力の計算の仕方を定め，アルゴリズムは平衡状態の実際の計算方法を与える．

既存の力学的手法のほとんどは**シミュレーション**アルゴリズムを用いるものと見なすことができる．この種のアルゴリズムは疑似物理系の離散時間シミュレーションを行う．より具体的には，各離散時刻でノードに働く力を力学モデルに基づいて計算し，各ノードを力の方向に移動する[⋆1]．また，シミュレーションを高速化するために多数の研究がなされている．特にグラフの粗い近似を利用するマルチレベル法[1),6)] や，離れたノードの間の相互作用を無視することで力の計算を削減するツリーコード法[6),8)] がよく用いられる．

数値最適化[10)] も力学的グラフ可視化手法のために用いられる．この種の手法では Kamada-Kawai 法[7)] が有名である．この手法はノード間にバネを付与し，Newton 法の繰り返しによってバネの全エネルギーを最小化することで平衡状態を計算する．他にも Tunkelang[11),12)] によって，共役勾配法による数値最適化を行うグラフ可視化手法が提案されている．Tunkelang はまた，多くの力学的手法で用いられているシミュレーションアルゴリズムを，素朴な数値最適化手法である最急降下法として見なすことができることを指摘した．このことは数値最適化に基づく力学的手法の有望さを示唆しているとも言えるが，現状ではまだアルゴリズムの提案は多くない．

本研究では数値最適化による力学的グラフ可視化手法に注目し，以下の特徴を備えたシンプルな手法を提案する．

- シミュレーションまたは数値最適化に基づく従来手法で用いられている力学モデルに対応できる一般性を持つ．
- 効率的な数値最適化のために記憶制限付き Broyden-Fletcher-Goldfarb-Shanno (**L-BFGS**) 法[9)] を用いる．
- マルチレベル法やツリーコード法を用いずに一定水準の速度で動作する．

†1 国立情報学研究所
National Institute of Informatics, Japan

⋆1 Newton の第 2 法則 $ma = F$ ではなく，疑似物理法則 $mv = F$ が仮定されている．

本手法は**保存力**と**スカラーポテンシャル**の概念に基づき，Kamada-Kawai 法[7] や Fruchterman-Reingold 法[3] の力学モデルに適用可能である．数値最適化のために用いる L-BFGS 法は，2 階偏微分を必要としない準 Newton 法を効率化したもので，メモリ使用量を削減できる．本研究では本提案手法を 1 万ノード規模のグラフに適用し，特に Kamada-Kawai 法の力学モデルと組み合わせた場合に良好な結果が得られることを示す．

本稿は以下の構成からなる．まず 2 節で数値最適化によるグラフ可視化手法の関連研究を紹介する．次に 3 節で本研究で構築したグラフ可視化手法を提案し，4 節でその実装を述べる．5 節では本手法に関する実験結果を与える．最後に 6 節で本研究のまとめと今後の課題を述べる．

2. 関連研究

1 節で述べた通り，Kamada-Kawai (KK) 法[7] は数値最適化に基づく有名なグラフ可視化手法であり，Newton 法によって最適化問題を解く．KK と同じ力学モデルが stress majorization 法[4] で用いられている．この手法は KK モデルに特化された最適化アルゴリズムであり，KK 法よりも通常高速である．本稿の著者は以前に Chorus[5] と呼ぶ制約解消系を開発した．これは KK モデルに基づくグラフ配置制約を扱い，BFGS 法による局所探索と遺伝的アルゴリズムによる大域探索を採用している．Tunkelang[11),12)] は共役勾配法と Fruchterman-Reingold[3] に類似するモデルに基づくグラフ可視化手法を開発し，実験によって最急降下法よりも数倍程度高速であることを示した．

3. 提案手法

本節では数値最適化に基づく力学的グラフ可視化手法を提案する．必要な記法を導入した後，本手法の基礎と 2 つの基本的な力学モデルの扱い方を示し，最後に数値最適化アルゴリズムについて述べる．

$G = (V, E)$ を n 個のノードの集合 $V = \{1, \ldots, n\}$ と m 本のエッジの集合 $E \subseteq V \times V$ からなるグラフとする．本手法では無向グラフのみを仮定する．すなわち，任意の $(i, j) \in E$ に対して常に $(j, i) \in E$ が成立する．各ノード $i \in \{1, \ldots, n\}$ について $\mathbf{p}_i = (x_i, y_i)$ を i の位置とする．このとき，G の配置を $\mathbf{p} = (x_1, y_1, \ldots, x_n, y_n)$ で表すことができる．また，$\mathbf{p}_{ij} = \mathbf{p}_j - \mathbf{p}_i$ と定める．本稿では 2 次元グラフ可視化に焦点を合わせるが，本稿の結果は 3 次元以上にも容易に拡張可能である．

3.1 基　　礎

Kamada-Kawai 法[7] などの数値最適化に基づく手法では，最小化すべきエネルギーの導入が必要である場合が多い．また，Tunkelang[11),12)] によって示されたように，シミュレーションに基づくアルゴリズムのほとんどは最急降下法として見なすことができる．

数学的には保存力とスカラーポテンシャルの観点から数値最適化に基づく手法を説明できる．$\mathbf{F}_i = (F_i^x, F_i^y)$ を i に働く力とし，

$$\mathbf{F} = (F_1^x, F_1^y, \ldots, F_n^x, F_n^y)$$

と定める．力が保存力であれば，$\mathbf{F}$ との間に以下の関係を持つスカラーポテンシャル Φ を定めることができる．

$$\mathbf{F} = -\nabla\Phi \tag{1}$$

ここで $\nabla\Phi$ は Φ の勾配，すなわち

$$\nabla\Phi = \left(\frac{\partial\Phi}{\partial x_1}, \frac{\partial\Phi}{\partial y_1}, \ldots, \frac{\partial\Phi}{\partial x_n}, \frac{\partial\Phi}{\partial y_n} \right)$$

を示す．式 (1) は $\mathbf{F}$ が Φ の最急降下方向に向いていることを意味し，平衡状態 $\mathbf{F} = \mathbf{0}$ は Φ の極小値によって得られる．すなわち

$$\mathbf{F} = -\nabla\Phi = \mathbf{0} \tag{2}$$

が成立する．従って，Φ を目的関数とする最適化問題

$$\operatorname*{argmin}_{\mathbf{P}} \Phi \tag{3}$$

を解くことで G の配置を求めることができる．

言い換えれば，保存力が用いられる場合，力学的グラフ可視化手法は最適化問題として再定式化可能である．加えて，(3) が最急降下法を必須としないことにも注意する必要がある．最適化問題 (3) を解くために，より洗練された数値最適化手法を用いることが可能である．本研究はこのアプローチを採用する．

3.2 力学モデル

ここでは基本的な力学モデルである Kamada-Kawai モデルと Fruchterman-Reingold モデルで保存力が用いられ，対応するスカラーポテンシャルが存在することを示す．

3.2.1 Kamada-Kawai モデル

Kamada-Kawai (KK) 法[7] は数値最適化に基づく手法の中でも特に有名である．KK 法の力学モデル (KK モデル) は spring model とも呼ばれ，バネエネルギーによってスカラー

ポテンシャルを与えるため，本研究の手法に容易に導入可能である．

KK モデルは連結グラフを仮定し，すべてのノードの組 (隣接する場合と隣接しない場合の両方を含む) に対してバネを付与する．ノード i，j の間の各バネに対してその自然長として i，j 間のグラフ理論的距離 l_{ij} を設定する．さらに i，j 間のバネのバネ定数を $1/l_{ij}^2$ とする．このとき，各 i に働く力 $\mathbf{F}_i$ は以下の Hooke の法則によって計算される．

$$\mathbf{F}_i = -\sum_{j \neq i} \frac{|\mathbf{p}_{ij}| - l_{ij}}{l_{ij}^2} \frac{\mathbf{p}_{ij}}{|\mathbf{p}_{ij}|}$$

古典力学で知られている通り，スカラーポテンシャル Φ は以下の通り計算される．

$$\Phi = \sum_i \sum_{j>i} \frac{(|\mathbf{p}_{ij}| - l_{ij})^2}{2\, l_{ij}^2}$$

元の KK 法で必要とされる Φ の 2 階偏微分は本手法では不要である．この点については 3.3 節で再び述べる．

3.2.2 Fruchterman-Reingold モデル

Fruchterman-Reingold (FR) 法[3] のアルゴリズムはシミュレーションに基づき，その力学モデル (FR モデル) は他のシミュレーションに基づく手法 (例えば 6)，8)) で用いられることも多い．このモデルでは隣接するノードの組に引力が与えられ，任意のノードの組の間に斥力が与えられる．具体的には力 $\mathbf{F}_i$ は以下の通り定められる．

$$\mathbf{F}_i = -\sum_{(i,j)\in E} \frac{|\mathbf{p}_{ij}|}{k} \mathbf{p}_{ij} + \sum_{j \neq i} \frac{k^2}{|\mathbf{p}_{ij}| + \epsilon} \frac{\mathbf{p}_{ij}}{|\mathbf{p}_{ij}|}$$

ここで k は適当な定数 (エッジの理想の長さとして見なすことができる[3]) で，ϵ は $\mathbf{p}_{ij}$ が $\mathbf{0}$ に近い場合の数値的問題を回避するためのソフトニングパラメータ (微小な正の数) である．この力学モデルは保存力を用いており，以下のようにスカラーポテンシャル Φ を定めることができる．

$$\Phi = \sum_{(i,j)\in E \wedge j>i} \frac{|\mathbf{p}_{ij}|^3}{3k} - \sum_i \sum_{j>i} k^2 \log\left(1 + \frac{|\mathbf{p}_{ij}|}{\epsilon}\right)$$

3.3 L-BFGS 法

本手法では最適化問題 (3) を解くことで力学的グラフ可視化を行う．すでに示した通り，目的関数 Φ は通常，非線形なスカラーポテンシャルである．従来の力学的手法はこのような問題を最急降下法 (シミュレーションに基づくアルゴリズム) や，Newton 法 (Kamada-Kawai 法[7])，共役勾配法 (Tunkelang の手法[11),12)]) などで解いていた．数値最適化の観点から最急降下法が望ましくないことは言うまでもないが，素朴な Newton 法も適当であるとは言えない．なぜなら素朴な Newton 法は，計算コストが大きい Φ の 2 階偏微分 (Hesse 行列と呼ばれる) を必要とするからである[⋆1]．

準 Newton 法[10] はこの問題を解決できる．実際の Hesse 行列を用いる代わりに，更新式によって Hesse 行列を近似的に計算するためである．特に Broyden-Fletcher-Goldfarb-Shanno (BFGS) 法が有効であることが知られている．また，BFGS 法の変種である記憶制限付き BFGS (L-BFGS) 法[9] は，Hesse 行列の近似のために数個のベクトルのみを保存することで必要なメモリを削減でき，大規模数値最適化で広く利用されている．本研究では力学的グラフ可視化のために L-BFGS 法を用いる．

4. 実　　装

前節に述べた手法を用いてグラフ可視化システムを C++で実装した．プログラムは現時点で約 1,900 行からなる．L-BFGS 法の実行のため，libLBFGS ライブラリ[⋆2]を用いている．libLBFGS の有用な点は，計算の効率化のためにストリーミング SIMD 拡張命令 SSE および SSE2 を内部的にサポートしていることである．本研究の実装では libLBFGS の SSE2 サポートによって，数値最適化における倍精度浮動小数点数計算を効率化している．

5. 実　　験

前節に述べた実装を用いて提案手法に関する実験を行った．ベンチマークデータセットとして文献 1) と同様のもの[⋆3] を用いた．このデータセットは 34 個から 16840 個までのノードを持つ 43 個のグラフからなる．実験は 8 GB のメモリを備えた 4 コア[⋆4]の 2.8 GHz Core i7 プロセッサ上で行った．

最初に本手法で異なる力学モデルを用いた場合の性能比較を行った．表 1 は KK モデルと FR モデルを用いた場合の実行時間を各グラフについて示したものである．この結果から本手法は通常，KK モデルを用いる場合のほうが高速に動作することがわかる．

図 1 は異なる力学モデルによって生成されたグラフの可視化結果を示す．被験者実験は

⋆1 一般に素朴な Newton 法で数値最適化を行う場合，すでに 1 階偏微分 $\nabla\Phi$ を含んでいる問題 (2) を解くために，2 階偏微分がさらに必要となる．
⋆2 http://www.chokkan.org/software/liblbfgs/
⋆3 http://ls11-www.cs.tu-dortmund.de/staff/klein/gdmult10
⋆4 プロセッサは 4 つのコアを持っているが，提案手法は逐次に動作する．

行っていないが，一般的な傾向として本手法は KK モデルを用いた場合のほうが可視化結果において優れている印象を受けた．

以上により，本手法は KK モデルを用いた場合に実行時間と可視化の両面で良好な結果を示すと考えられる．図 2 は本手法で KK モデルを用いて計算した，より大規模なグラフの可視化である．

6. おわりに

本稿では数値最適化に基づく力学的グラフ可視化手法を提案した．本手法は異なる力学モデルに適用可能であり，L-BFGS 法による数値最適化を用いて実装されている．本手法では特に Kamada-Kawai モデルと組み合わせた場合に実行時間と可視化の両面で良好な結果が得られる．

今後の課題の 1 つは，エッジの方向や他種の制約[2] を扱う，より一般的な力学モデルを導入することである．別の課題としては，マルチレベル法やツリーコード法を導入してさらに性能を向上することがあげられる．

参考文献

1) Bartel, G., Gutwenger, C., Klein, K. and Mutzel, P.: An Experimental Evaluation of Multilevel Layout Methods, *Proc.GD*, LNCS, Vol.6502, pp.80–91 (2010).
2) Di Battista, G., Eades, P., Tamassia, R. and Tollis, I.G.: *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall (1999).
3) Fruchterman, T. M. J. and Reingold, E. M.: Graph Drawing by Force-directed Placement, *Softw.Pract.Exper.*, Vol.21, No.11, pp.1129–1164 (1991).
4) Gansner, E.R., Koren, Y. and North, S.C.: Graph Drawing by Stress Majorization, *Proc.GD*, LNCS, Vol.3383, pp.239–250 (2004).
5) Hosobe, H.: A Modular Geometric Constraint Solver for User Interface Applications, *Proc.ACM UIST*, pp.91–100 (2001).
6) Hu, Y.: Efficient, High Quality Force-Directed Graph Drawing, *Mathematica J.*, Vol.10, No.1, pp.37–71 (2005).
7) Kamada, T. and Kawai, S.: An Algorithm for Drawing General Undirected Graphs, *Inf.Process.Lett.*, Vol.31, No.1, pp.7–15 (1989).
8) Matsubayashi, T. and Yamada, T.: A Force-directed Graph Drawing based on the Hierarchical Individual Timestep Method, *Intl.J.Elect.Comput.Eng.*, Vol.2, No.10, pp.686–691 (2007).
9) Nocedal, J.: Updating Quasi-Newton Matrices with Limited Storage, *Math. Comput.*, Vol.35, No.151, pp.773–782 (1980).
10) Nocedal, J. and Wright, S. J.: *Numerical Optimization*, Springer, 2nd edition (2006).
11) Tunkelang, D.: JIGGLE: Java Interactive Graph Layout Environment, *Proc.GD*, LNCS, Vol.1547, pp.412–422 (1998).
12) Tunkelang, D.: A Numerical Optimization Approach to General Graph Drawing (Ph.D.Thesis), Tech.Rep. CMU-CS-98-189, Sch.Comput.Sci., Carnegie Mellon Univ. (1999).

Vol.2011-HCI-144 No.16
2011/7/29

表 1 提案手法の実行時間 (秒)
Table 1 Executions times of our methods in seconds.

グラフ	ノード数	エッジ数	KK モデル	FR モデル
karateclub	34	78	0.04	0.04
cylinder_rnd_010_010	97	178	0.05	0.07
snowflake_A	98	97	0.07	0.07
spider_A	100	220	0.05	0.07
sierpinski_04	123	243	0.07	0.08
flower_001	210	3057	0.78	0.79
tree_06_03	259	258	0.16	0.20
dg_617_part	341	797	0.35	0.41
rna	363	468	0.25	0.37
Grid_20_20_doublefolded	397	760	0.22	0.48
Grid_20_20_singlefolded	399	760	0.21	0.47
Grid_20_20	400	760	0.20	0.49
protein_part	417	597	0.38	0.49
516.graph	516	729	0.49	0.72
flower_005	930	13521	4.43	4.59
snowflake_B	971	970	1.65	2.30
cylinder_rnd_032_032	985	1866	0.95	2.27
grid_rnd_032	985	1834	0.84	2.28
spider_B	1000	2200	1.87	2.51
sierpinski_06	1095	2187	1.21	2.78
ug_380	1104	3231	2.81	3.30
tree_06_04	1555	1554	3.92	5.66
Grid_40_40_doublefolded	1597	3120	2.65	6.13
Grid_40_40_singlefolded	1599	3120	3.48	6.18
Grid_40_40	1600	3120	2.73	6.19
esslingen	2075	5530	8.48	10.44
add20	2395	7462	12.42	15.58
data	2851	15093	15.73	18.99
3elt	4720	13722	28.04	51.28
uk	4824	6837	40.17	52.77
add32	4960	9462	43.58	59.08
4970.graph	4970	7400	21.10	55.97
dg_1087	7602	7601	89.96	145.61
grid400_20	8000	15580	113.73	146.01
tree_06_05	9331	9330	149.57	197.01
cylinder_rnd_100_100	9497	17941	166.90	206.29
grid_rnd_100	9497	17849	108.68	206.24
snowflake_C	9701	9700	159.55	233.87
sierpinski_08	9843	19683	145.01	219.40
spider_C	10000	22000	181.74	241.99
crack	10240	30380	88.60	237.85
4elt	15606	45878	436.50	552.30
cti	16840	48232	364.64	656.99

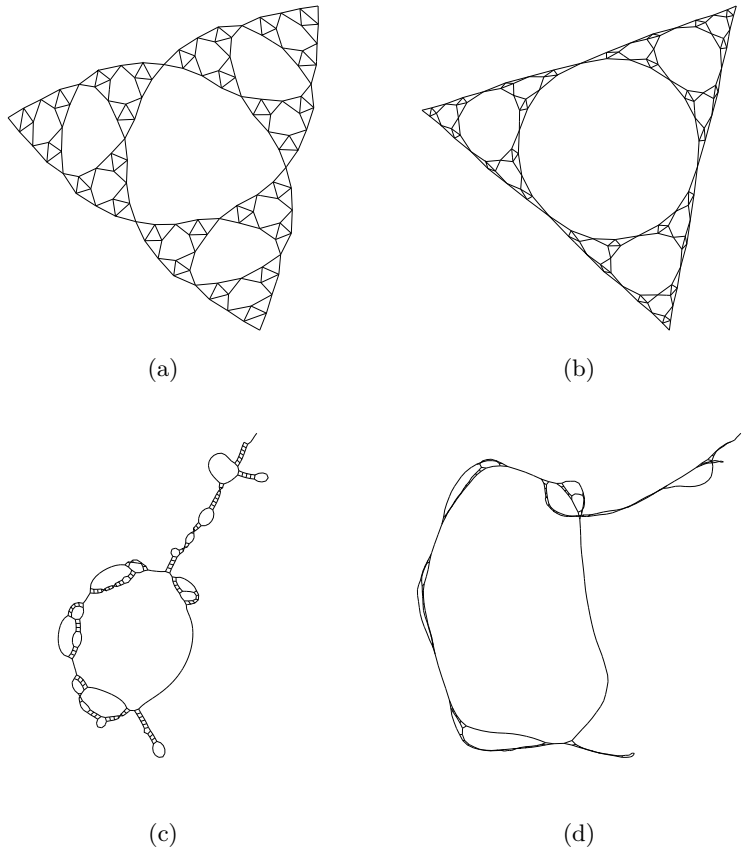

図 1 提案手法で (a) KK モデル，(b) FR モデルを用いた場合の sierpinski_04 グラフの可視化；提案手法で (c) KK モデル，(d) FR モデルを用いた場合の rna グラフの可視化
Fig. 1 The sierpinski_04 graphs drawn by our method using the (a) KK and (b) FR models; the rna graphs drawn by our method using the (c) KK and (d) FR models.

Vol.2011-HCI-144 No.16
2011/7/29

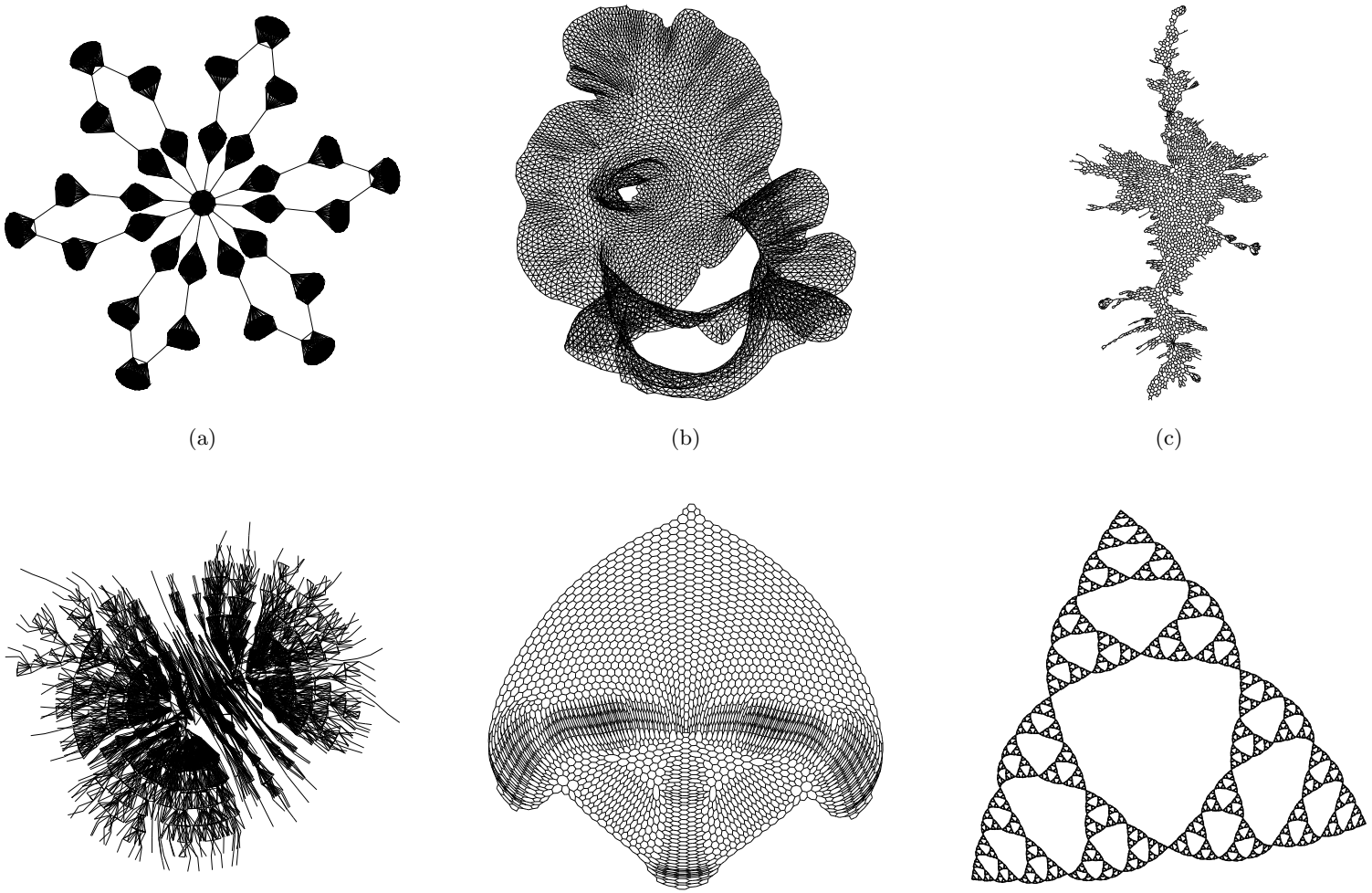

図 2 提案手法で KK モデルを用いた場合の (a) flower_005, (b) 3elt, (c) uk, (d) add32, (e) 4970.graph, (f) sierpinski_08 グラフの可視化
Fig. 2 Graphs drawn by our method using the KK model: (a) flower_005, (b) 3elt, (c) uk, (d) add32, (e) 4970.graph, and (f) sierpinski_08.