Vol.2022-HCI-197 No.45
2022/3/16

複合無向グラフの自動描画手法の開発

三末 和男[1,a)]　土井 周平[,b)]

概要：本論文が扱う複合無向グラフとは、2 種類のエッジを備えたグラフであり、一種類のエッジは、広く知られているグラフと同様にノードの無向隣接関係を表し、もう一種類のエッジはノード間の包含関係を表す。それらは概念的に隣接関係と包含関係を表すだけでなく、連結図によるノードの連結と領域図によるノード間の包含をモデル化したもので、視覚的表現と密接に結びついている。開発された自動描画手法は、無向グラフの自動描画手法として広く使用されている仮想力学モデルを拡張したものである。特徴は、複合無向グラフにおいて、他のノードを包含するグループノードが他のグループノードと要素を共有することを許したこと、グループノードを表す図形として凸多角形およびそれを基盤とする凸図形を使用することで、円や長方形だけに制限した場合に比べて空間効率を高めたことである。さらには、包含関係の入れ子形状を分りやすくするための工夫もなされている。

1. はじめに

本論文が扱う**複合グラフ**とは、2 種類のエッジを備えたグラフであり[*1]、一種類のエッジは、広く知られているグラフと同様にノードの隣接関係を表し、もう一種類のエッジはノード間の包含関係を表す。隣接関係を表すエッジを**隣接エッジ**とよび、包含関係を表すエッジを**包含エッジ**とよんで区別する。それらは概念的に隣接関係と包含関係を表すだけでなく、連結図によるノードの連結と領域図によるノード間の包含をモデル化したもので、視覚的表現と密接に結びついていることが特徴である。包含エッジは視覚的にもノードの包含関係を表すことから、有向エッジであり、ノードの集合と包含エッジの集合だけから構成される有向グラフには閉路がない。その一方で、隣接エッジは有向の場合と無向の場合、さらには有向と無向が混在した場合を考えることができる。隣接エッジが無向である複合グラフを**複合無向グラフ**とよぶ。

複合グラフのモデルは KJ 法 [1] で使用される図解である。KJ 法は「発想法」ともよばれ、情報の断片群を整理して統合することで、断片群に潜む知見を発見したり、さらにはそれらから新たな発想を生み出すことを支援する技法である。KJ 法で使用される図解は、情報の断片を記録したカード、それらを入れ子状にグループにした領域、カードや領域を連結する線から構成される。領域を表す図形や連結線に装飾が加えられるなど、図解の表す情報の印象を強める工夫も行われるが、抽象的にみると、その構造は複合グラフと言える。つまり、KJ 法の手順は複合グラフの視覚的な表現を操作する過程とみなすことができる[*2]。

筆者らの研究の目的は、複合グラフの自動描画を可能にすることである。ここでは、複合無向グラフの自動描画アルゴリズムの開発に焦点を合わせる。類似の技術はすでに存在するが、ノードの集合と包含エッジの集合だけから構成される有向グラフが根付き木に制限されている（二つ以上のグループがノードあるいはグループを共有しない）、あるいは、隣接エッジがそのような有向グラフのシンク[*3]間だけに制限されている（カード間にだけ関係線があり、グループには関係線が接続しない）、というように構造的な制限がある。また視覚的にも、ノードの視覚的表現が長方形や円などの幾何図形に限定されているものが多い。筆者らは、それらの制約を取り除くことを目指している。

元々の KJ 法はアナログ的な手法であり、計算機による支援は前提としていない。複合グラフの操作は模造紙などの上で人手で行われる。文房具などの小道具の活用による効率化は工夫されているが、それでも操作過程において人の記憶や想像に頼る部分が少なくない。たとえば、カード間の関係を意識しながら、空間的な配置を考える段階がある。空間的な配置が確定すれば、関係を表す線を模造紙に

[1] 筑波大学
Uniersity of Tsukuba
[a)] misue@cs.tsukuba.ac.jp
[b)] doi@vislab.cs.tsukuba.ac.jp

[*1] 棒グラフと折れ線グラフを重ねたようなグラフを**複合グラフ**とよぶことがあるが、それとは異なる概念である。

[*2] 書籍「グラフ自動描画法とその応用」[2] の図 6.3.1「KJ 法の図マトリクス表現」を見ると、KJ 法が複合グラフの操作過程であることが分る。

[*3] 出るエッジが接続していないノード。KJ 法の図解ではカードに相当する。

描くことができるが、配置が確定するまでは、関係の存在を記憶しつつ、関係線を想像しながら配置を検討することになる。もし記憶や想像に頼る部分を、頭の外で視覚的に表現できれば、人は配置の検討に専念できる。このことは思考過程の改善につながる可能性がある。

KJ 法に限らず、断片的なものを、(2 項) 関係付けとグループ化 (クラスタ化) によって整理し、組織化することは多くの場面で行われる。そのように整理された (あるいは整理されつつある) 構造を視覚的に表現するために、ゲシュタルトの法則 (連結の要因、包囲の要因) を活用することは、人の直感に合ったものであり、表現手法としても効果的と言える。ここでは、複合グラフを KJ 法で用いられる図のモデルとして説明したが、複合グラフはごく自然に行われる情報整理の構造を表現するものであり、視覚的に自然な形式での表現を前提としている。複合グラフの自動描画技術は、KJ 法のデジタル化を支援するとともに、情報の整理や体系化、あるいはその表現に貢献するものである。

2. 関連研究

三末・杉山は、KJ 法のような図を利用した思考過程の支援を目的として、複合有向グラフの階層的描画法を開発した [3], [4]。これは、Sugiyama らによって開発された有向グラフの階層描画手法 [5] を、複合グラフに拡張したものである。ここで言う「階層描画」とは、有向エッジが下に向くように、ノードを平行配置された水平線上に配置するものである。三末・杉山の複合有向グラフの階層的描画法は、ノードの集合と包含エッジの集合だけから構成される有向グラフが根付き木に制限されている。

Feng らは、クラスタグラフ (clustered graph) の描画法を開発した [6]。クラスタグラフは無向グラフのノード群がクラスタを構成したもので、複合グラフとして考えると、隣接エッジが、ノードの集合と包含エッジの集合だけから構成される有向グラフのシンク間だけに、制限されている。

Feng らの手法は、クラスタの交差は許さないため、ノードの集合と包含エッジの集合だけから構成される有向グラフが根付き木に制限されている。表および Omote & Sugiyama は、クラスタグラフの描画法に関して、クラスタの交差を許す手法を開発した [7], [8]。ただし、ノードの視覚的表現が長方形や円などの幾何図形に限定されている。

ところで、隣接エッジのない複合グラフの視覚的な表現はオイラー図とみなすことができる。オイラー図は集合間の関係の視覚的な表現手法であるが、どのような関係に対してもオイラー図が描けるわけではない。そこで、Simonetto らは描けなかった関係を描けるようにオイラー図を拡張した [9]。Simonetto らは、さらにオイラー図の自動的な描画方法を開発した [10]。彼らの描画手法では、交差する領域が境界線を共有するため、そのままでは領域の識別が難しい。そこで、Gossett ら [11] が示したような混色の工夫により領域を塗り分けることで領域の識別を助けている。

3. 技術的課題

3.1 描画対象: 複合無向グラフ

複合無向グラフを $G = (V, E_A, E_I)$ のように表す。このとき V はノードの集合である。また、E_A は隣接エッジの集合、E_I は包含エッジの集合である。ノード v と w をつなぐ無向エッジを $\{v, w\}$ のように集合で表すことにする。またノード v から w に向う有向エッジを (v, w) のように組で表すことにする。隣接エッジは無向エッジであり、$E_A \subseteq \{\{v, w\} | v, w \in V \text{ かつ } v \neq w\}$ である。包含エッジは有向エッジの集合であり、$E_I \subset V \times V$ である。二つのノードが相互に包含することがないように、V と E_I で構成される有向グラフ $G_I = (V, E_I)$ には閉路はないとする。

ノード $v \in V$ に着目したとき、$(v, w) \in E_I$ のような包含エッジが存在するなら v を**グループノード**とよび、存在しないなら**単位ノード**とよぶことにする。グループノードは他のノードを包含するが、単位ノードは他のノードを包含しない。

3.2 描画対象の制約条件

描画対象の制約条件として、任意の二つのノード $v, w \in V$ に関して、G_I において v から w へ向う有向パスがある場合、$\{v, w\} \notin E_A$ とする。つまり v が直接間接を問わず w を包含する場合、v と w の間に隣接エッジはないとする。

3.3 描画規約

複合無向グラフの描画規約を以下のように設定する。

C1 ノードを点とみなせる図形あるいは閉曲線 (ジョルダン曲線) で表す。

C2 隣接エッジ $\{v, w\} \in E_A$ を、ノード v の表現とノード w の表現をつなぐ線分で表す。線分には折れ線や曲線を含むとする。

C3 包含エッジ $(v, w) \in E_I$ を、ノード v を表現する閉曲線がノード w を表現する図形を完全に囲むことで表す。

3.4 美的基準

複合無向グラフの美的基準を以下のように設定する。

- 隣接エッジでつながれた二つのノードを近くに配置する。つまり、$\{v, w\} \in E_A$ のときノード v の表現とノード w の表現を近くに配置する。
- 閉曲線で表すノードの占める領域を小さくする。つまり、始点 $v \in V$ を共有する包含エッジの終点のノード群 $\{w \in V | (v, w) \in E_I\}$ を近くに配置する。

3.5 アルゴリズムの入出力

アルゴリズムの入出力は以下の通りとする。

入力

- 複合無向グラフ $G = (V, E_A, E_I)$
- 各単位ノードのサイズ（横と縦の長さ）

出力

- 各ノード $v \in V$ を表す多角形あるいは閉曲線
- 各隣接エッジ $e \in E_A$ を表す線分

4. 描画手法の概要

4.1 記法

ノード $v \in V$ の位置を $\boldsymbol{p}(v)$ で表すことにする。またノード v を表す多角形の重心を $\boldsymbol{b}(v)$ で表すことにする。ノード v が単位ノードのときは $\boldsymbol{p}(v)$ を中心とする長方形として描かれるため、$\boldsymbol{p}(v) = \boldsymbol{b}(v)$ であるが、グループノードのときには必ずしも $\boldsymbol{p}(v)$ を中心とする多角形として描かれるとは限らない。

4.2 レイアウトの求め方の手順

描画規約 C2 および C3 を満すために、下のようなステップでレイアウトを求めることにする。ステップ S2 では描画規約 C3 を満すこと、ステップ S3 では描画規約 C2 を満すことを目指す。

S1 単位ノードの配置を決める

S2 単位ノードの位置と形状を使用してグループノードの位置と形状を決める

S3 ノードの位置と形状を使用してエッジの配線を求める

ステップ S1 で決めた単位ノードの配置が複合グラフ全体の描画をおおよそ決めることになる。そのため、ステップ S1 では単位ノードを含むグループの形状や配置、そのグループに接続する隣接エッジの配線状態などにも注意する必要がある。そのため、ステップ S1–S3 は 1 パスの処理ではなく、繰り返しの処理により漸次的に改善する。その際、描画の処理の全体は、以下の通り二つのフェーズに分かれる。

P1 粗い配置を求める

P2 配置の微調整を行う

このように二つのフェーズを用いるが、P1 から P2 へとある瞬間に完全に切り替えるのではなく、フェーズ間の移行も漸次的に行うことにした。

4.3 描画に使用する 3 種類の多角形

描画処理において、各ノードに対して以下のような 3 種類の凸多角形を使用する。

表示多角形 ノードの形状として表示される、あるいは表示形状の基盤として使用される多角形

境界多角形 表示多角形よりもひと回り大きい多角形。重なり検出に使用する。ひと回り大きくしておくことで、ノード間に一定の最小距離を確保する。

輪郭多角形 包含される際に、ノードの輪郭形状として使用する多角形。表示多角形から開始して、包含されるたびに徐々に拡大する。

5. ノードの形状の求め方

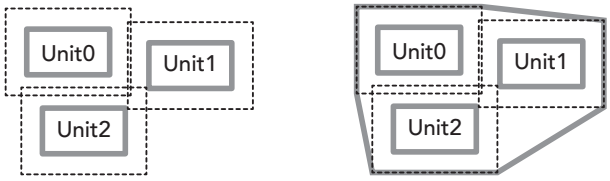
単位ノードの形状は与えられているものとし、グループノードの形状はそれが包含する単位ノードの形状と配置を利用して求められる。

5.1 単位ノードの表示多角形

単位ノードの形状は長方形を基本とする。そのサイズ（横と縦の長さ）は事前に与えられているとする。なお、ノードの位置は長方形の中心とする。

5.2 グループノードの表示多角形

描画規約 C3（包含エッジに関する規約）を満足するために、グループノードはそれが含むノードを表す図形を完全に囲む必要がある。そこで、$v \in V$ がグループノードのとき、$\{w \in V | (v, w) \in E_I\}$ の要素に対応する輪郭多角形のそれぞれを少しずつ拡大した図形を構成しておき（図 1(a) 参照）、それら全体を含む（拡大した輪郭多角形を構成する全頂点の集合の）凸包を v の表示多角形とする（図 1(b) 参照）。



(a) 輪郭多角形（実線）をそれぞれ少し拡大した様子　(b) 拡大したすべての図形を包含する凸包を求める

図 1 グループノードの輪郭形状の求め方

5.3 表示多角形を求める順序

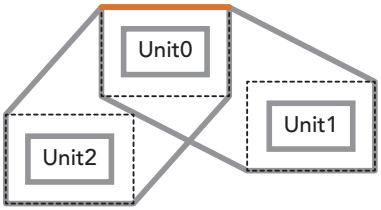
上に示した手順で表示多角形を求めるためには、ノード v の表示多角形を求める際に、v に包含されるすべてのノードの輪郭多角形が決っていなければならない。そのため、包含エッジの集合 E_I を利用して、V の要素をトポロジカルソートしておく。$G_I = (V, E_I)$ には閉路はないため、トポロジカルソートは常に成功する。そして、ソートされた順（包含エッジの終点側を初めとする順）に輪郭形状を求めれば、包含されるすべてのノードの輪郭形状がすでに決まっていることを保証できる。

5.4 交差するグループが輪郭線を共有しない工夫

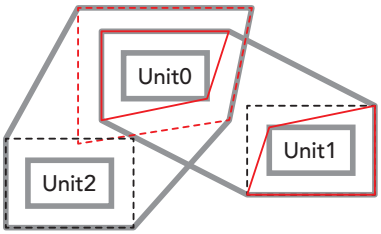
上に示した手順では、二つの異なるグループノードが同じノードを包含する場合に図 2(a) に示すように、輪郭線の

一部を共有することになる。

グループノード $v \in V$ の表示多角形を求める際に、$\{w \in V | (v,w) \in E_I\}$ の要素に対応する輪郭多角形のそれぞれを少しずつ拡大した図形を構成しておき、それら全体を含む凸包を求めたが、それに先立って、輪郭多角形を、図 2(b) 内の赤色の実線のように、凸包に採用された頂点の位置に拡大した状態に更新しておく。こうすることで、次にそのノードを囲むグループノードの表示多角形を求める場合には、少し外側の輪郭線が形成されることになる。



(a) 輪郭線を共有する例（橙色が共有部分）



(b) 拡大した輪郭多角形（赤色の実線）に基いて凸包を求めた様子

図 2 交差するグループノードの輪郭形状の求め方

5.5 グループノードのラベル領域の確保

グループのラベルを表示する際には、図 3 に示すように、グループの表示多角形の上部に少し重ねてラベルを配置することにした。グループのラベルを表示しても、グループノードの表示多角形は変らない。ただし、境界多角形と輪郭多角形はラベル領域も含む凸包として求めなおす。

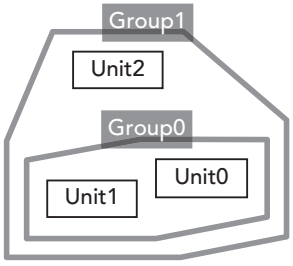


図 3 グループノードのラベル領域

6. 隣接エッジの配線の決め方

隣接エッジは直線分として描くことにした。接続する二つのノードの重心をつなぐ直線分のうち、どちらのノードの表示多角形にも含まれない部分を隣接エッジとして描く。

7. 単位ノードの配置の決め方

7.1 無向グラフの構成

複合無向グラフ $G = (V, E_A, E_I)$ から、無向グラフ $G_U = (V, E_A \cup E'_I)$ を構成する。ここで、E'_I は、包含エッジを無向エッジに変換したもので、$E'_I = \{\{v,w\} | (v,w) \in E_I\}$ とする。なお、描画対象の制約条件 (3.2) より $E_A \cap E'_I = \emptyset$ である。

7.2 仮想力学モデル

この段階では、各ノードは大きさのない点として扱う。Eades のスプリングモデル [12] にならい、エッジに起因する力と隣接しないノード間に働く力から構成される仮想力学モデルを採用した。ノード v に対して、それに接続する隣接エッジ $\{v,w\}$ によって働く力が $\boldsymbol{f}_a(v,w)$ のように表されるとすると、ノード v に接続するすべての隣接エッジから受ける力 $\boldsymbol{f}^*_a(v)$ は式 (1) のように表される。同様に、接続するすべての包含エッジから受ける力 $\boldsymbol{f}^*_i(v)$ は式 (2) のように表される。

$$\boldsymbol{f}^*_a(v) = \sum_{\{v,w\} \in \{e \in E_A | v \in e\}} \boldsymbol{f}_a(v,w) \qquad (1)$$

$$\boldsymbol{f}^*_i(v) = \sum_{\{v,w\} \in \{e \in E'_I | v \in e\}} \boldsymbol{f}_i(v,w) \qquad (2)$$

ノード v に対して、それに隣接していないノード w によって働く力が $\boldsymbol{f}_r(v,w)$ のように表されるとすると、ノード v が、隣接していない他のノードから受ける力 $\boldsymbol{f}^*_r(v)$ は式 (3) で表される。ここで、W_v はノード v が隣接していないノードの集合、すなわち $W_v = V \backslash (\{w \in V | \{v,w\} \in E_A \cup E'_I\} \cup \{v\})$ である。

$$\boldsymbol{f}^*_r(v) = \sum_{w \in W_v} \boldsymbol{f}_r(v,w) \qquad (3)$$

7.3 ノード間の不適切な重なりの排除

二つのノード $v, w \in V$ に関して、グループノード v が論理的に w を包含する、あるいは逆にグループノード w が論理的に v を包含する場合、およびグループノード v と w がそれぞれ他の同一ノードを論理的に包含する場合には、必然的に v と w を表す領域は重なる。ノード v と w がこれら以外の関係にある場合に、それらの境界多角形が重なることは不適切である。このような場合には、v と w の境界多角形が重なることがないようにノードの位置を補正する必要がある。

ノード v と w が不適切に重なる状態を述語 $P_{nn}(v,w)$ で表し、ノード v に対してノード w から遠ざかる向きに働く力を $\boldsymbol{f}_{nn}(v,w)$ のように表すと、ノード v がすべての不適

切な重なりを排除するために受ける力 $\boldsymbol{f}^*_{nn}(v)$ は式 (4) で表される。

$$\boldsymbol{f}^*_{nn}(v) = \sum_{w \in \{w \in V | P_{nn}(v,w)\}} \boldsymbol{f}_{nn}(v,w) \qquad (4)$$

7.4 単位ノードと隣接エッジの不適切な重なりの排除

単位ノード u をそれに接続しない隣接エッジ a が横切ることはグラフの視認性を低下させる。このような場合には、単位ノード u と隣接エッジ a が重ならないように単位ノードと隣接エッジの位置を補正する必要がある。

単位ノード u をそれに接続しない隣接エッジ a が横切っている状態を述語 $P_{ua}(u,a)$ で表し、ノード u に対して隣接エッジ a から遠ざかる向きに働く力を $\boldsymbol{f}_{ua}(u,a)$ のように表すと、ノード u がすべての隣接エッジと不適切な重なりを排除するために受ける力 $\boldsymbol{f}^*_{ua}(u)$ は式 (5) で表される。また、隣接エッジ a を単位ノード u から遠ざけるために、a に接続するノード v に働く力を $\boldsymbol{f}_{au}(a,u)$ のように表すと、ノード v が、それが接続する隣接エッジが単位ノードと不適切な重なりを排除するために受ける力 $\boldsymbol{f}^*_{au}(v)$ は式 (6) で表される。

$$\boldsymbol{f}^*_{ua}(u) = \sum_{a \in \{a \in E_A | P_{ua}(u,a)\}} \boldsymbol{f}_{ua}(u,a) \qquad (5)$$

$$\boldsymbol{f}^*_{au}(v) = \sum_{a \in \{a \in E_A | v \in a\}} \sum_{u \in \{u \in V | P_{ua}(u,a)\}} \boldsymbol{f}_{au}(a,u) \qquad (6)$$

7.5 隣接エッジの長さの補正

仮想力学モデルにより、隣接エッジが自然長に近付くようにノードの位置を決めているが、実際にはノードは点ではなく領域を占めるため、隣接エッジが見掛け上短かくなることがある。特に、グループノードのような大きなノードに接続する隣接エッジは、極端に短かくなることがある。このような場合には、隣接する二つのノードの位置を補正する必要がある。

ノード v と w をつなぐ線分の見掛け上の長さが隣接エッジの理想長よりも短かい状態を述語 $P_{al}(v,w)$ で表し、ノード v に対してノード w から遠ざかる向きに働く力を $\boldsymbol{f}_{al}(v,w)$ のように表すと、ノード v が、それに接続するすべての隣接エッジの長さを補正するために受ける力 $\boldsymbol{f}^*_{al}(u)$ は式 (7) で表される。

$$\boldsymbol{f}^*_{al}(v) = \sum_{w \in \{w \in V | P_{al}(v,w)\}} \boldsymbol{f}_{al}(v,w) \qquad (7)$$

7.6 グループノードに働く力の扱い

式 (4)、(6)、(7) で表される力が働くノード v がグループノードの場合には、実際には、v に含まれるすべての単位ノードに、単位ノードの数で割った大きさの力が働くとする。

7.7 重み調整のスケジューリング

これまでに導入した仮想的な力をすべて同時に働かせるのではなく、Omote & Sugiyam の手法 [8] と同様に、グループ分けしてタイミングを変えて働かせる。基本的には、下記のように粗い配置を決めるためグループ F_c と、配置の微調整を行うためのグループ F_f の二つに分けることにした。

- F_c: $\boldsymbol{f}^*_a$、$\boldsymbol{f}^*_r$
- F_f: $\boldsymbol{f}^*_{nn}$、$\boldsymbol{f}^*_{ua}$、$\boldsymbol{f}^*_{au}$、$\boldsymbol{f}^*_{al}$

このとき、力の影響度を制御するためのパラメータ t を導入する。つまり、$F_c(t)$ の計算では、すべてのノード $v \in V$ に対して $t \cdot \boldsymbol{f}^*_a(v)$ と $t \cdot \boldsymbol{f}^*_r(v)$ を求める。また、$F_f(t)$ の計算では、すべてのノード $v \in V$ に対して $t \cdot \boldsymbol{f}^*_{nn}(v)$、$t \cdot \boldsymbol{f}^*_{ua}(v)$、$t \cdot \boldsymbol{f}^*_{au}(v)$、$t \cdot \boldsymbol{f}^*_{al}(v)$ を求める。F_c の計算と F_f の計算により、各ノードに働く合力を求め、各ノードを合力の方向にその大きさに応じた距離だけ移動させる。この処理を手続き的に示すと Algorithm 1 のようになる。

Algorithm 1 Compound Undirected Graph Layout

Input: $G = (V, E_A, E_I)$ – 複合無向グラフ
Input: 単位ノードの長方形のサイズ
Output: 各ノード $v \in V$ を表す閉曲線
Output: 各隣接エッジ $e \in E_A$ を表す直線分

```
function CompoundUndirectedGraphLayout(G)
    全ノード v ∈ V の初期配置を求める
    i ← 0
    while i ≤ n do
        t ← s_n(i)
        F_c(t) を求める
        F_f(1 − t) を求める
        ノードの位置を更新する
        隣接エッジの配線を行う
        i ← i + 1
    end while
    while ¬ 安定状態 do
        F_f(1) を求める
        ノードの位置を更新する
        隣接エッジの配線を行う
    end while
end function
```

Algorithm 1 に含まれる関数 $s_n : \{0, 1, \ldots, n\} \to [0, 1]$ は、繰り返しの各ステップで使用する重みを与える。関数 s_n の基本的な例を式 (8) に示す。この関数を使用すると、まず、F_c に含まれる力のグループが重み 1 から徐々に弱くなるのに対して、F_f に含まれる力のグループは重み 0 から徐々に強くなる。つまり、粗い配置を決めるために F_c のグループを働かせておき、配置の微調整を行うための F_f へと比重を移すというように、力のスケジューリングができる。

$$s_n(i) = 1 - i/n \tag{8}$$

8. 実装

我々は以下に示すような具体化を行なった。ただし、力の定義およびそこで使用する定数に関しては、様々な調整の余地がある。

8.1 力の式の具体化

第 7 節で使用した力は具体的には以下のように定義した。ここで、$\boldsymbol{d}_{v,w} = \boldsymbol{p}(w) - \boldsymbol{p}(v)$ とし、$\hat{\boldsymbol{d}}$ は $\boldsymbol{d}$ と同じ向きの単位ベクトル、すなわち $\hat{\boldsymbol{d}} = \boldsymbol{d}/|\boldsymbol{d}|$ とする。

$$\boldsymbol{f}_a(v,w) = c_s \log(|\boldsymbol{d}_{v,w}|/l_a) \cdot \hat{\boldsymbol{d}}_{v,w} \tag{9}$$
$$\boldsymbol{f}_i(v,w) = c_s \log(|\boldsymbol{d}_{v,w}|/l_i) \cdot \hat{\boldsymbol{d}}_{v,w} \tag{10}$$
$$\boldsymbol{f}_r(v,w) = (c_r/|\boldsymbol{d}_{v,w}|^2) \cdot \hat{\boldsymbol{d}}_{w,v} \tag{11}$$
$$\boldsymbol{f}_{nn}(v,w) = c_{nn} \cdot \hat{\boldsymbol{d}}_{w,v} \tag{12}$$
$$\boldsymbol{f}_{ua}(u,a) = c_{ua} \cdot \hat{\boldsymbol{d}}_{p_a,u} \tag{13}$$
$$\boldsymbol{f}_{au}(a,u) = c_{au} \cdot \hat{\boldsymbol{d}}_{u,p_a} \tag{14}$$
$$\boldsymbol{f}_{al}(v,w) = c_{al} \cdot \hat{\boldsymbol{d}}_{w,v} \tag{15}$$

式 (9) と (10) はノード v が隣接するノード w から受ける力を表している。式 (11) は隣接しないノードから受ける力を表している。これらに関しては、Eades のスプリングモデル [12] をベースにした。

式 (12) と (15) はどちらもノード v をノード w から遠ざけるための力を表している。ただし、式 (4) と (7) に示したように、そららが働く状況は異なる。

式 (13) と (14) に出現する p_a は、ノード u の中心から隣接エッジを表す直線 a に下した垂線の足を表す。$\boldsymbol{f}_{ua}$ と $\boldsymbol{f}_{au}$ は逆向きの力である。

8.2 定数の設定

定数 l_a と l_i は、スプリングの自然長（エッジの理想長）に相当するパラメータであり、式 (9) と (10) の違いはこれらの定数だけである。具体的には、$l_a = 30$、$l_i = 5$ とした。E_A の要素はノード間の隣接関係を表す線分として描かれるのに対して、E'_I の要素は同じグループに含まれるノード群を近くに集めるために使われる。そのため、l_a に比べて、l_i は小さくした。

その他の定数は、以下のように設定した。$c_s = 1.0$、$c_r = 1.0$、$c_{nn} = 1.0$、$c_{ua} = 1.0$、$c_{au} = 1.0$、$c_{al} = 1.0$。

8.3 初期配置

仮想力学モデルによるグラフレイアウトに結果は初期配置の影響を受けやすい。我々の実装では、初期配置において、すべてのノードを円周上に配置することにした。その際、無用な領域の交差ができるだけ生じないように、配置の順序は、包含グラフ $G_I = (V, E_I)$ において深さ優先探索で訪れる順とした。ただし、包含グラフは必ずしも木ではないため、根ではなく、ソース[*4]を兄弟とみなし、そのなかで包含する単位ノードが最も多いものから探索を開始することにした。そして、各ノードには 2 回以上は訪問しない、兄弟間では包含する単位ノードが多いものを優先する、という規則で探索を行った。

8.4 重みのスケジューリング

重みのスケジューリング関数としては、基本的なものとして式 (16) に示すように式 (8) において $n = 100$ とした。

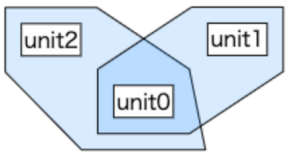
$$s_{100}(i) = 1 - i/100 \tag{16}$$

8.5 安定状態の判定

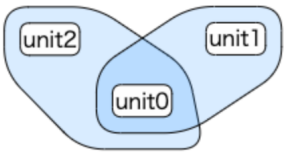
全ノードの移動が一定値 ϵ よりも小さくなった場合に、安定状態に到達したとみなすことにした。具体的には $\epsilon = 0.1$ とした。

8.6 ノードの形状のバリエーション

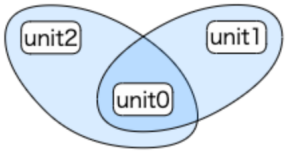
ノードの形状として 3 種類のバリエーションを用意した。それらは、図 4 に示すように、表示多角形をそのまま表示するもの、表示多角形を基本として角を丸めたもの、表示多角形の頂点を制御点とする 2 次ベジェ曲線により自由曲線風にしたものである。



(a) 多角形



(b) 角丸多角形



(c) 自由曲線風

図 4 ノードの形状の 3 種類のバリエーション

[*4] 入るエッジが接続していないノード。すなわち、他のノードに包含されていないノード。

8.7 彩色

セクション 5.3 に示した表示多角形を求める順序とは逆の順序で、ノードの塗り潰しを行う。グループノードは半透明色を使用することで重なりが濃く表示されるようにする。単位ノードは背景色を白で描く。ラベル領域は単位ノードと同様の長方形で、見掛け上は半透明の黒背景に白文字とした。

8.8 プログラミング言語

実装には TypeScript を使用した。描画には Canvas API を使用した。

9. 描画例

二つの描画例を紹介する。

9.1 オイラー図の例

図 5 は、Simonetto らの論文 [10] の Figure 1 に示されたオイラー図と同じ集合を表したものである。Simonetto らの手法では集合 Monuments と集合 Itary、および集合 Monument と集合 France がそれぞれ境界線をなめらかに共有しているが、本手法では、図 5 に示したように、境界線は交差するだけであるため、それぞれ別の集合であることが分りやすい。

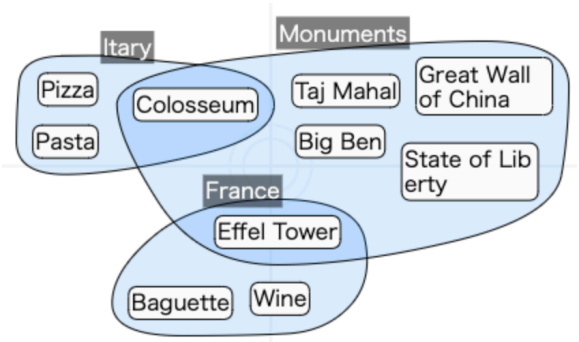


図 5 Simonetto らと同じオイラー図 [10] を本手法で描いた例

9.2 KJ 法で用いられる図解

図 6 は、KJ 法でつくられた図解（文献 [13] の図 3-11-1-3）から抽出した複合無向グラフを本手法で描いたものである。元の図解に描かれた隣接エッジは有向であったが、本手法への入力データとしては無向エッジとしている。元の図解とは配置が異なっている部分はあるが、複合グラフという観点での構造は正しく表現されている。

10. 計算時間

我々が実装したプログラムでは、Algorithm 1 に含まれる二つの while ループのそれぞれにおいて毎回その時点のレイアウトを表示している。そのままではレイアウト計算に要した時間の計測が容易でないため、ここで提示するデータは計算時間に関する参考情報である。

実行環境は、MacBook Pro (13-inch, 2017 Four Thunderbolt 3 Ports), プロセッサ 3.5GHz デュアルコア Intel Core i7、メモリ 16GB 2133MHz LPDDR3、OS macOS Big Sur 11.6.1 において、ローカルに Web サーバーを起動し、Chrome バージョン: 96.0.4664.55（Official Build）（x86_64）である。

重みのスケジューリングには、式 (16) を使用し、図 6 に関して、Algorithm 1 における、4 行目から 16 行目までの所要時間を計測した。処理の進行は目視で、計測はストップウォッチで行なった。11 回計測し、中央値を求めたところ、2.6 秒であった。

11. 残された課題

まだ数多くの課題が残っているが、そのなかでも以下の二つが重要だと考えている。

11.1 グループ領域と隣接エッジの重なり除去

式 (5) および式 (6) に示した力を導入することで、単位ノードと隣接エッジの交差はある程度排除できる。しかしながら、グループノードと隣接エッジの重なりについては現時点では対処していない。単位ノードと隣接エッジの関係と同様に力を加えて排除する方法も考えられるが、単にそれらを引きはなす向きに力を加えるたけでは、空間効率が悪化することが問題である。空間効率を低下させずに、重なりを除去するためには、隣接エッジの直線描画をあきらめ、多くの KJ 法図解がそうしているように、曲線で隣接エッジを描くことを考える必要がありそうである。

11.2 有向エッジの一方向描画

図 6 においては隣接エッジをすべて無向エッジとして扱ったが、元の図解 [13] には、単なる線分、片方に矢じりのついた線分、両端に矢じりのついた線分が描かれている。片方に矢じりのついた線分はその線分が表す関係に向きがあると考えられる。このような隣接エッジの向きを揃えて描くことで、図解全体が表す「流れ」のようなものが把握しやすくなると期待できる。力学モデルにおいてエッジの向きを制御するためには磁場の導入が考えられる [14]。

12. まとめ

無向の隣接関係を表す隣接エッジと、包含関係を表す包含エッジを備えた、複合無向グラフの描画法を開発した。関連する従来手法の多くが、複合グラフの特殊形であるクラスタグラフに焦点を対象としているのに対して、本手法が対象とする複合グラフは、グループノードにも隣接エッジがつながり得ることが特徴である。本手法の特徴は、複合グラフにおいて、グループノードが要素を共有すること

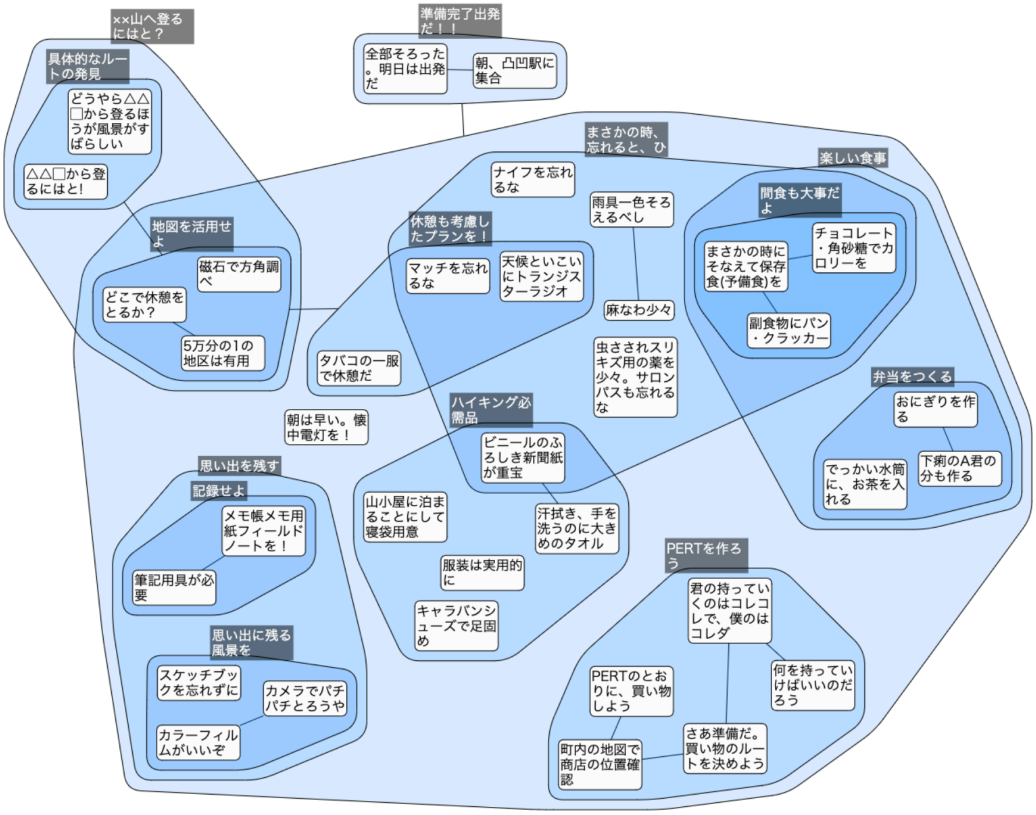


図 6 KJ 法でつくられた図解 [13] を本手法で描いた例

を許したこと、グループノードを表す図形として凸図形を使用することで、円や長方形だけに比べて空間効率を高めたことである。

参考文献

[1] 川喜田二郎. KJ 法 — 渾沌をして語らしめる. 中央公論社, 1986.
[2] 杉山公造. グラフ自動描画法とその応用. 計測自動制御学会, 1993.
[3] 三末和男, 杉山公造. 図的思考支援を目的とした複合グラフの階層的描画法について. 情報処理学会論文誌, Vol. 30, No. 10, pp. 1324–1334, 1989.
[4] Kozo Sugiyama and Kazuo Misue. Visualization of structural information: automatic drawing of compound digraphs. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 21, No. 4, pp. 876–892, Jul 1991.
[5] Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 11, No. 2, pp. 109–125, Feb 1981.
[6] Qing-Wen Feng, Robert F Cohen, and Peter Eades. How to draw a planar clustered graph. In *International Computing and Combinatorics Conference*, pp. 21–30. Springer, 1995.
[7] 表寛樹. インターセクションの扱いが可能なクラスターグラフの力指向による自動描画法の研究. Master's thesis, 北陸先端科学技術大学院大学 知識科学研究科知識システム基礎学専攻, 2000.
[8] Hiroki Omote and Kozo Sugiyama. Method for visualizing complicated structures based on unified simplification strategy. *IEICE Transactions on Information and Systems*, Vol. E90-D, No. 10, pp. 1649–1656, 2007.
[9] Paolo Simonetto and David Auber. Visualise undrawable euler diagrams. *12th International Conference on Information Visualisation (IV 2008)*, Vol. 00, pp. 594–599, 2008.
[10] Paolo Simonetto, David Auber, and Daniel Archambault. Fully automatic visualisation of overlapping sets. *Computer Graphics Forum*, Vol. 28, No. 3, pp. 967–974, 2009.
[11] Nathan Gossett and Baoquan Chen. Paint inspired color mixing and compositing for visualization. In *IEEE Symposium on Information Visualization*, pp. 113–118, 2004.
[12] Peter Eades. A heuristic for graph drawing. *Congressus Numerantium*, Vol. 42, pp. 149–160, 1984.
[13] 川喜田二郎, 牧島信一. 問題解決学: KJ 法ワークショップ. 講談社, 1970.
[14] Kozo Sugiyama and Kazuo Misue. Graph drawing by the magnetic spring model. *Journal of Visual Languages & Computing*, Vol. 6, No. 3, pp. 217–231, 1995.