階層グラフの可視化

尾上　洋介

杉山フレームワークによる階層グラフ描画は，時間的前後関係や依存関係などの階層構造をもったグラフを理解するときに有効なネットワーク可視化の手法である．杉山フレームワークのアルゴリズムには，オペレーションズ・リサーチ分野で登場するさまざまな最適化の技術が活用されている．本稿では，階層グラフ描画の理論およびアルゴリズムについて概説する．また，読者がそれらを活用できるように，杉山フレームワークを実装した主要なソフトウェアとライブラリについても紹介する．

キーワード：階層グラフ描画，杉山フレームワーク，ネットワーク可視化，情報可視化，組合せ最適化

1. はじめに

ものごとのつながりをネットワーク，あるいはグラフ理論におけるグラフとして捉えることで，世の中に存在するさまざまな現象の解析が可能になる．それらのネットワークの理解を促進するために可視化は重要な役割を果たしてきた．本稿では，それらの可視化方法の中から，有向グラフ可視化の一手法である階層グラフ描画について紹介する．グラフを入力として，それに含まれる頂点や辺のレイアウトを決定する手法をグラフ描画と呼ぶ．さまざまなグラフ描画の方法が提案されており，入力グラフの特性や可視化・分析の目的に応じてそれらは使い分けられる．階層グラフ描画は，特に，閉路が少ないようなグラフを効果的に可視化できる．階層グラフ描画のアルゴリズムは，杉山フレームワーク (Sugiyama Framework) [1] の名前で国際的に広く知られている．家系図やソフトウェアモジュールの依存関係，論文の引用関係は階層グラフ描画の典型的な応用例である．

はじめに階層グラフ描画の二つの例を示す．図 1 は，非常に有名なコマンドラインベースのネットワーク可視化ソフトウェアである GraphViz (http://www.graphviz.org/) を用いた描画の例である．階層グラフ描画では，このように上から下などの一方向に向かって，頂点を階層的に配置する．図 2 は，TensorBoard という可視化ツールにおける階層グラフ描画である．TensorBoard は昨今注目を集めている深層学習向けのライブラリである TensorFlow (https://www.tensorflow.org/) に組み込まれた可視化ツールである．TensorBoard では，深層学習における演算の依存関係を表した計算グラフを階層描画を用いて可視化している．階層グラフ描画という言葉は聞いたことがなくても，これらの可視化図を目にしたことがある読者は少なくないのではないだろうか．

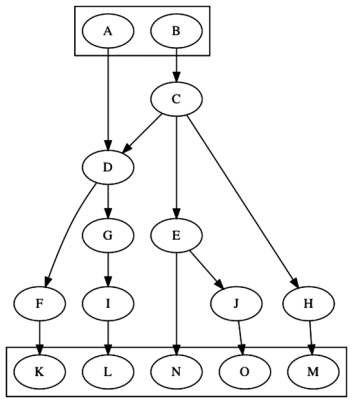

図 1　GraphViz による階層グラフ描画

本稿では，階層グラフ描画の基礎的な理論とそれが実装されたソフトウェアについて紹介する．階層グラフ描画を実現するためのアルゴリズムには数多くの最適化問題が登場する．杉山フレームワークでは，美的基準を定量化し，その定量的な指標を最適化することでレイアウトを生成するというオペレーションズ・リサーチ (OR) 的な側面を垣間見ることができる．また，先に示した GraphViz をはじめとして，階層グラフ描画を実装したソフトウェアやライブラリは多数存在する．読者が今後階層グラフ描画を行う際の参考になるように，主要なソフトウェアやライブラリについて紹介する．

おのうえ　ようすけ
京都大学学際融合教育研究推進センター
政策のための科学ユニット
〒 606–8501　京都府京都市左京区吉田本町
onoue@viz.media.kyoto-u.ac.jp

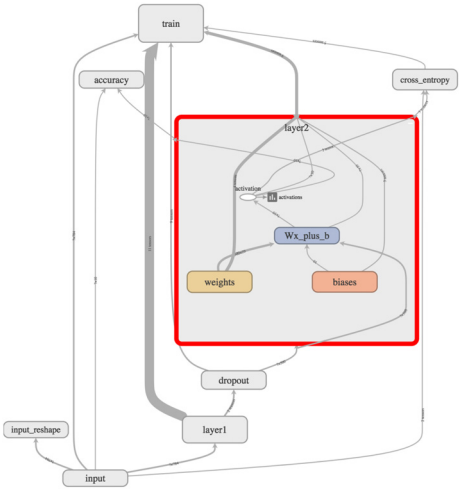

図 2　TensorBoard における計算グラフの階層グラフ描画

2.　杉山フレームワークの手順

本節では，階層グラフ描画を実現するための杉山フレームワークの手順について概説する．詳細については，本稿で示す参考文献や Healy and Nikolov の総説 [2] を読んでいただきたい．

杉山フレームワークでは，いくつかの美的基準に基づいて可読性の高いグラフレイアウトを生成する．一般的には以下の五つの基準が用いられる [2].

1. すべての辺をできるだけ一方向に向ける
2. 辺の長さをできるだけ短くする
3. すべての階層に頂点を一様に配置する
4. 辺の交差をできるだけ少なくする
5. 辺をなるべく直線に保つ

杉山フレームワークでは (1) 閉路除去，(2) 階層割当，(3) 交差削減，(4) 座標割当の 4 ステップの手順によって，それらの美的基準を満たしたレイアウトを生成する．図 3 は，(2) から (4) までの手順を表している．

現実的には，すべての美的基準を同時に満たすことができず，トレードオフが生じる場合がある．また，それらの美的基準を満たしたレイアウトの生成の多くは，NP 困難な最適化問題として定式化されるため，実用的な時間内で可読性の高いレイアウトを生成するためには，近似解法を用いてそれらの問題を解く場合が多い．そのため，計算時間や美的基準同士のトレードオフを考慮して，上述のそれぞれのステップでさまざまな手法が提案されている．一般的には，用途に応じてそれらの手法を自由に交換して組み合わせることができる．また，目的によっては，美的基準の追加や修正を行い，それに対応したアルゴリズムをいずれかのステップで導入する場合もある．

以降は入力グラフを $G = (V, A)$ と記載する．ここで，V と $A \subseteq V \times V$ はそれぞれ頂点と有向辺の集合である．$\mathrm{succ}(u) = \{v \mid (u, v) \in A\}$ と $\mathrm{pred}(u) = \{v \mid (v, u) \in A\}$ をそれぞれ u の出頂点（あるいは後続点）集合と入頂点（あるいは先行点）集合とする．さらに，$d^+(u) = |\mathrm{succ}(u)|$ と $d^-(u) = |\mathrm{pred}(u)|$ をそれぞれ u の出次数と入次数と呼ぶ．$\mathrm{pred}(u) = \emptyset$, $\mathrm{succ}(v) = \emptyset$ となる u, v をそれぞれソースとシンクと呼ぶ．

2.1　閉路除去

階層グラフ描画は，入力グラフが無閉路有向グラフ (DAG; Directed Acyclic Graph) に近い構造をもっているときに有効に働くレイアウト手法である．階層割当以降のステップでは入力グラフが DAG であることを仮定するため，本ステップにおいて入力グラフの一部の辺を反転することで閉路の除去を行う．$F \subseteq A$, $A' = (A \setminus F) \cup \{(v, u) \mid (u, v) \in F\}$ としたとき，$G' = (V, A')$ が閉路をもたないとき F を帰還辺集合 (FAS; Feedback Arc Set) と呼ぶ．杉山フレームワークにおける閉路除去は，$|F|$ が最小となる帰還辺集合 F を求める問題と考えることができる．この最小 FAS 問題は NP 困難な問題として知られている [3].

杉山フレームワークにおいて，最小 FAS 問題を効率よく解くためのヒューリスティック解法が数多く提案されている．Eades et al. による貪欲閉路除去法は杉山フレームワークにおける最小 FAS を解くための主要な方法の一つである [4]．貪欲閉路除去法では，閉路に含まれないソースとシンクをすべて取り除き，残った閉路に対して $d^+(u) - d^-(u)$ が最も大きい頂点 u を取り除く処理を繰り返す．閉路除去ステップでは，辺に重み付けを行うこともできる．この場合，重みの総和が最小となる FAS が解となる．ここで反転した辺の情報は保持しておき，最終的な描画時に矢印の向きの反転などへ反映される．

2.2　階層割当

階層割当では，図 3(2) のように，DAG を入力として階層 $V_1, V_2, \ldots, V_H$ を出力する．ここで，H は階層の高さである．$V_1, V_2, \ldots, V_H$ は，$\cup_{i=1}^{H} V_i = V$ かつ $V_i \cap V_j = \emptyset$ $(1 \leq i < j \leq H)$ である．また，$(u, v) \in A, u \in V_i, v \in V_j$ について $i < j$ である．階層割当を行った DAG を階層グラフと呼ぶ．

階層割当を行う最も簡単なアルゴリズムの一つは最長パス法である [2]．最長パス法は，入力グラフのソースを階層 1 として，その出頂点に対して再帰的に次の階層を割り当てていく．最長パス法は $\mathrm{O}(|A|)$ の計算量で

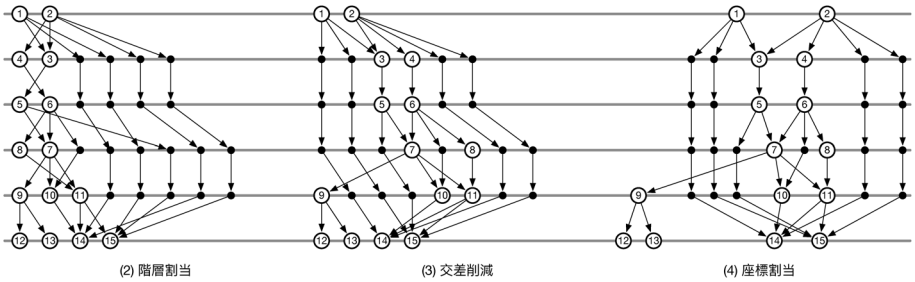

図 3　杉山フレームワークの手順

階層割当を行うことができる．最長パス法はリストスケジューリングアルゴリズムの一種である．同様にリストスケジューリングによる階層割当としては Coffman–Graham のアルゴリズムもよく知られている．

最長パス法は実行可能な階層を容易に得られるが，美的基準の観点では改善の余地が多い．階層割当において，いくつかの美的基準を明示的に達成するために，線形整数計画 (ILP; Integer Linear Programming) 問題を用いたアプローチが行われている．頂点 u の階層を y_u とするとき，$y_v - y_u$ を辺 (u,v) のスパンと呼ぶ．Gansner et al. は，頂点の前後関係を保ちつつ，スパンの総和を最小化する問題を以下のように定式化した [5].

$$\text{min.} \quad \sum_{(u,v)\in A} w_{uv}(y_v - y_u) \tag{1}$$

$$\text{s.t.} \quad y_v - y_u \geq \lambda_{uv}, \quad (u,v) \in A, \tag{2}$$

$$y_v \in \mathbb{Z}^+, \quad v \in V. \tag{3}$$

ここで，w_{uv} と λ_{uv} はそれぞれ，辺 (u,v) の重みと最小スパンである．これらを定数として与えることで，レイアウトの調整を行うことができる．この問題は ILP であるが，LP 緩和問題の解が必ず整数解となるため，効率よく解くことができる．

階層割当を ILP として考えることで，美的基準に沿ったさまざまな制約を加えることが容易となる．Healy and Nikolov は，各階層に含まれる頂点数の上限 W を持った階層割当問題を ILP として以下のように定式化した [6].

$$\text{min.} \quad \sum_{(u,v)\in A} w_{uv}(y_v - y_u) \tag{4}$$

$$\text{s.t.} \quad y_v - y_u \geq \lambda_{uv}, \quad (u,v) \in A, \tag{5}$$

$$y_v \in \mathbb{Z}^+, \quad v \in V, \tag{6}$$

$$\sum_{u\in V} a_{uh} + \sum_{(u,v)\in A} \left(\sum_{i=h+1}^{H} a_{vi} - \sum_{i=h}^{H} a_{ui} \right) \leq W,$$

$$1 \leq h \leq H, \tag{7}$$

$$y_u = \sum_{h=1}^{H} h a_{uh}, \quad u \in V, \tag{8}$$

$$\sum_{h=1}^{H} a_{uh} = 1, \quad u \in V, \tag{9}$$

$$a_{uh} \in \{0,1\}, \quad u \in V, \quad 1 \leq h \leq H. \tag{10}$$

a_{uh} は，頂点 u を階層 h に割り当てるときに 1 となる 0-1 変数である．式 (7) が各階層の頂点数の上限を表している．ここでは，階層をまたがる辺も 1 頂点として数えられる．式 (8) から式 (10) は，頂点 u の階層を 0-1 変数 a_{uh} を用いて $y_u = \sum_{h=1}^{H} h a_{uh}$ と記述するための条件である．Healy and Nikolov は，この問題が分枝カット法で解けることを示した [6].

尾上らは，グラフのソースとシンクを同一階層に並べる制約を加えた階層割当の提案を行った [7, 8]．ソースとシンクの集合 V_1 と V_H について，階層をそれぞれ 1 と H に固定する制約が追加される．ここで，H は最長パス法などを用いることで容易に得ることができる．また，そのときに頂点配置の一様性を満たすため，目的関数はスパンの重み付き二乗和に修正されている．この問題は以下のように定式化される．

$$\text{min.} \quad \sum_{(u,v)\in A} w_{uv}(y_v - y_u)^2 \tag{11}$$

$$\text{s.t.} \quad y_v - y_u \geq \lambda_{uv}, \quad (u,v) \in A, \tag{12}$$

$$y_v = 1, \quad v \in V_1, \tag{13}$$

$$y_v = H, \quad v \in V_H, \tag{14}$$

$$y_v \in \mathbb{Z}^+, \quad v \in V. \tag{15}$$

これは二次整数計画 (QIP; Quadratic Integer Programming) 問題による定式化である．この問題は等価な ILP に変換可能であり，汎用の ILP ソルバで解くことができる [8].

なお，本ステップにおいては，アルゴリズムによる割り当てを行わず，ユーザーがあらかじめ定義した階層を使用する場合もある．

2.3 交差削減

階層グラフ描画において，辺交差が発生するかどうかは，頂点の座標ではなく各階層の頂点の順序によって決まる．交差削減ステップでは，図 3(3) のように，辺交差が最小となるような頂点の順序決定を行う．交差削減は，2 階層ごとの交差削減を順次行う方法と，全階層を同時に交差削減を行う方法の二つに分類される．一般的に，後者のほうが全体の交差数を削減することができるが，計算量は膨大になる．

スパンが 1 より大きい辺の扱いが複雑となるため，交差削減の前にすべての辺のスパンが 1 となるように正規化を行う．階層グラフの正規化では，スパンが 2 以上の辺に対してダミー頂点を挿入する．ダミー頂点を挿入し，すべての辺のスパンが 1 となった階層グラフを正規階層グラフと呼ぶ．図 3 の番号なしの頂点はダミー頂点を表している．

はじめに，二部グラフにおける交差削減について考える．これは，正規階層グラフから隣り合った 2 階層を取り出した形である．さらに，最も単純な交差削減問題として，二部グラフの片側の順序を固定して，もう片側の順序のみを変更することを考える．この問題を OLCM (One-layered Crossing Minimization) と呼ぶ．

階層 h の頂点集合を $V_h = \{v_1^h, v_2^h, \ldots, v_{|V_h|}^h\}$ とし，頂点 v_i^h の順序を o_i^h とする．なお，ここでは読みやすさのため階層の添字は右肩に表示している．δ_{ij}^h を，$o_i^h < o_j^h$ のときに 1，そうでないときに 0 となるような変数とする．ある 2 階層 V_h，V_{h+1} 上の辺 $(v_k^h, v_i^{h+1}), (v_l^h, v_j^{h+1}) \in A$ について，交差が発生するとき $\delta_{kl}^h \delta_{ji}^{h+1} + \delta_{lk}^h \delta_{ij}^{h+1}$ が 1，発生しないとき 0 である．片側 V_h の順序を固定すると，$c_{ij}^h = \sum_{k \in \mathrm{pred}(v_i^{h+1})} \sum_{l \in \mathrm{pred}(v_j^{h+1})} \delta_{lk}^h$ とおくことができる．OLCM は以下の線型順序付け問題として定式化される [9].

$$\text{min.} \quad \sum_{i=1}^{|V_{h+1}|-1} \sum_{j=i+1}^{|V_{h+1}|} (c_{ij}^h - c_{ji}^h) \delta_{ij}^{h+1} \tag{16}$$

$$\text{s.t.} \quad 0 \le \delta_{ij}^{h+1} + \delta_{jk}^{h+1} - \delta_{ik}^{h+1} \le 1, \quad 1 \le i < j < k \le |V_{h+1}|, \tag{17}$$

$$\delta_{ij}^{h+1} \in \{0,1\}, \quad 1 \le i < j \le |V_{h+1}|. \tag{18}$$

これを厳密に解くことで OLCM の最適解が得られる．

OLCM のヒューリスティック解法としては重心法がよく用いられる [1]．重心法では，頂点 $v \in V_{h+1}$ の順序を $\sum_{u \in \mathrm{pred}(v)} o_u^h / d^-(v)$ に従ってソートする．計算量は $O(\min(|A_h|, |V_{h+1}| \log |V_{h+1}|))$ であり，高速に OLCM の近似解を得ることができる．ここで，$A_h = A \cap (V_h \times V_{h+1})$ とする．重心法は，実装が容易でありながら，実用上十分な交差削減が得られるので広く用いられる．重心法以外のヒューリスティック解法として，貪欲挿入法や中央値法が存在する．中央値法では，重心法における重心計算の代わりに中央値が用いられている．

正規階層グラフ全体の交差削減を行うためには，OLCM を 2 階層ずつ順番に適用する．$h = 1, 2, \ldots, H-1$ について，V_h を固定して V_{h+1} の並び替えを行う．$h = H-1$ に達したら，次は逆の $h = H, H-1, \ldots, 2$ の順で，V_h を固定して V_{h-1} の並び替えを行う．交差数が削減されなくなるか，十分な回数繰り返しを行うまで上述の手順を繰り返す．なお，OLCM を厳密に解いても，全階層に適用した場合の最適性は保証されない．実装の容易さや実用上十分な結果が得られることから，重心法を用いた OLCM の順次適用が行われることが多い．また，ここまでは，片側を固定した OLCM について考えたが，片側の固定を行わない TLCM (Two-layered Crossing Minimization) でも同様のアルゴリズムを適用することができる．

OLCM の順次適用は，十分に実用的な方法であるが，ヒューリスティックな方法であるため辺交差を十分に削減できない場合がある．Jünger et al. は全階層の交差削減を同時に行う MLCM (Multi-layered Crossing Minimization) について検討した [10]．c_{ijkl}^h を辺 $(v_k^h, v_i^{h+1}), (v_l^h, v_j^{h+1}) \in A_h$ が交差するときに 1，そうでないときに 0 となる変数とする．MLCM は以下の ILP として定式化される．

$$\text{min.} \quad \sum_{h=1}^{H-1} \sum_{(v_k^h, v_i^{h+1}), (v_l^h, v_j^{h+1}) \in A_h} c_{ijkl}^h \tag{19}$$

$$\text{s.t.} \quad -c_{ijkl}^h \le \delta_{ij}^{h+1} - \delta_{kl}^h \le c_{ijkl}^h, \quad (v_k^h, v_i^{h+1}), (v_l^h, v_j^{h+1}) \in A_h, \tag{20}$$

$$0 \le \delta_{ij}^2 + \delta_{jk}^2 - \delta_{ik}^2 \le 1, \quad 1 \le h \le H, \quad 1 \le i < j < k \le |V_h|, \tag{21}$$

$$c_{ijkl}^h \in \{0,1\}, \quad 1 \le h \le H, \quad (v_k^h, v_i^{h+1}), (v_l^h, v_j^{h+1}) \in A_h, \tag{22}$$

$$\delta_{ij}^h \in \{0,1\}, \quad 1 \le h \le H, \quad 1 \le i, j \le |V_h|. \tag{23}$$

ILP に対するヒューリスティック解法を用いれば，

MLCM の近似解を得られる．Chimani et al. は，Jünger et al. の ILP に対して半正定値計画 (SDP; Semidefinite Programming) 緩和を用いた MLCM の解法を提案している [11]．組合せ最適化問題における SDP 緩和は，LP 緩和と比較してより強い下界を得られることが知られている．Jünger et al. は，SDP 緩和を用いた MLCM のヒューリスティック解法によって，効率的に最適解に近い解が得られることを示した．

2.4 座標割当

最後のステップとして，図 3(4) のように，階層グラフおよび各階層の頂点順序を入力として各階層軸上の各頂点の描画座標 $x_u (u \in V)$ を決定する．ここでは，階層を上から下へと縦向きに描画する場合，それと垂直な横軸の座標決定を行う．説明上，横軸座標と表記するが，階層を横向きに描画する場合でも同様のアルゴリズムが適用できる．なお，各階層の頂点順序は維持されたまま座標決定が行われる．

はじめに，杉山らによる二次計画 (QP; Quadratic Programming) 問題を用いた定式化について述べる [1]．ここでは，辺の間を最小距離以上離すこと，また，複数階層にわたる辺を直線で描画することを制約として，隣り合う階層の隣接頂点を近くに配置する，および，各頂点を隣接頂点の重心に配置するように最適化を行う．隣り合う階層の隣接頂点を近くに配置することは以下の式で表される．

$$f_1 = \sum_{h=1}^{H-1} \sum_{(u,v)\in A_h} (x_u - x_v)^2 \tag{24}$$

また，各頂点を隣接頂点の重心に配置することは以下の式で表される．

$$f_2 = \sum_{h=1}^{H-1} \sum_{u\in V_h} \left(x_u - \frac{\sum_{v\in \mathrm{succ}(u)} x_v}{|\mathrm{succ}(u)|} \right)^2 + \sum_{h=2}^{H} \sum_{u\in V_h} \left(x_u - \frac{\sum_{v\in \mathrm{pred}(u)} x_v}{|\mathrm{pred}(u)|} \right)^2 \tag{25}$$

f_1 と f_2 はいずれも二次の目的関数である．重み定数 $w (0 \le w \le 1)$ を導入することで二つの目的関数の単一化を行う．そして，制約条件を加えた座標割当問題は最終的に以下のように定式化される．

$$\text{min.} \quad wf_1 + (1-w)f_2 \tag{26}$$

$$\text{s.t.} \quad x_v - x_u \ge D, \quad 1 \le h \le H, \quad u, v \in V_h, \quad o_v^h - o_u^h = 1, \tag{27}$$

$$x_u = x_v, \quad 1 \le h \le H, \quad (u,v) \in A_h^{\mathrm{d}}. \tag{28}$$

ここで，D は辺の間の最小距離であり，A_k^{d} は $(u,v) \in A_k$ のうち u と v 両方がダミー頂点である部分集合である．なお，o_u^h は 2.3 節と同様に階層 h における頂点 u の順序である．

QP による定式化において，x_u は整数変数ではなく連続変数であるため，比較的容易に解が得られる．しかしながら，より高速に解を得るためのヒューリスティックアルゴリズムもいくつか提案されている．Sugiyama et al. によって提案された優先度法では，交差削減における重心法の順次適用と同様に，2 階層ごとに座標割当を順次適用する [1]．次数の高い頂点から優先して座標を決定することから優先度法と呼ばれている．また，Brandes and Köpf は，$O(|V|)$ の計算量のヒューリスティックアルゴリズムを提案している [12]．Brandes and Köpf のアルゴリズムでは，左上寄せ，右上寄せ，左下寄せ，右下寄せの四つのレイアウトを生成し，それらを平均化することで最終的に可読性の高い座標割当を行う．Brandes and Köpf のアルゴリズムは，可読性の高いレイアウトを高速に生成できる点で実用性が高い．

横軸座標を決定した後，縦軸座標の決定を行う．一般的に，同一階層の縦軸座標は同一にする．これらは，頂点の高さや階層間の余白を与えることで容易に計算できる．これにより，杉山フレームワークの一連のステップが完了し，これらの座標情報に基づいてグラフの描画を行う．

3. 杉山フレームワークの改良

前節では，杉山フレームワークの基本的なアルゴリズムの紹介をした．杉山フレームワークは，フレームワークという名前のとおり，柔軟性と拡張性を備えている．そのため，標準的なステップ以外にもさまざまなオプションステップや拡張が存在する．本節では，それらの主要なものについて三つ紹介する．

3.1 辺集中化

階層グラフ描画のある 2 階層において，バイクリーク（完全二部部分グラフ）が含まれていると交差削減でどのように頂点の順番を入れ替えても辺交差を削減することができない．そのため，特に密なグラフにおいては描画されたグラフの可読性が下がる場合がある．そのような場合には，交差削減前のオプションステップとして辺集中化 (Edge Concentration) を適用することで可読性の向上ができる [13, 14]．辺集中化では，入力グラフに含まれるバイクリークを，集中化頂点を挿入することで置き換える．辺集中化の例を図 4 に表す．ここでは，二つのバイクリーク $K_{2,3}$ を置き換え

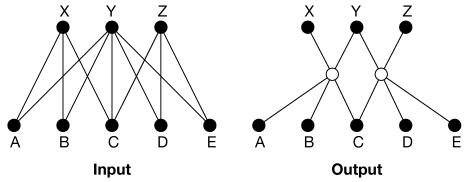

図 4　辺集中化の例

ている．

辺集中化問題は，適用後の辺数が最小となるようなバイクリーク被覆を見つける組合せ最適化問題である．辺数を少なくすることで，辺交差がより生じにくくなる．辺集中化問題に類似する問題として，最小バイクリーク被覆問題が存在する．最小バイクリーク被覆問題では，二部グラフのすべての辺を被覆できる最小のバイクリーク集合を見つけるが，その最適解は必ずしも辺集中化問題の最適解ではない．

3.2　円状レイアウト

一般的な杉山フレームワークでは，上から下あるいは左から右などの一方向に階層を配置する．そのような一方向の階層配置ではなく，階層の軸を放射状あるいは同心円状に配置する杉山フレームワークの拡張も提案されている．階層の軸を放射状に描画する場合，一方向レイアウトとの大きな違いは，最終階層の頂点から最初の階層への辺の存在が許されるため，閉路の描画が可能となることである．図 5 左における階層 6 から階層 1 への辺はその例である．Bachmaier et al. は，巡回を許した階層割当問題を定式化し，それに対するヒューリスティックアルゴリズムを提案している [15]．杉山フレームワークの同心円状への拡張では，図 5 右のように内側から外側（あるいはその逆）へと階層が配置される．Bachmaier は，同心円状の階層配置に適するように杉山フレームワークの各ステップを修正した効果的なアルゴリズムを提案している [16]．これにより，通常の一方向レイアウトと比較して，辺交差を 30％程度削減できるといった効果が報告されている．また同心円状配置では，階層が浅く各階層の頂点数が多い場合には，より小さい面積でグラフ構造を可視化することができる．

3.3　頂点の入れ子構造の可視化

頂点集合に対して，複数の辺集合が定義されているグラフを複合グラフと呼ぶ．本稿の冒頭で示した図 1 と図 2 では，一部の頂点が長方形領域にグループ化されていることが見て取れる．このような構造は，通常の有向辺の集合に加えて，頂点間の包含関係を定義した包含木をもった複合グラフとして定義できる．三末らは，杉山フレームワークの各ステップを複合グラフへと拡張した一連のアルゴリズムを提案した [17, 18]．これらのアルゴリズムには再帰的処理が用いられており，多段階の入れ子構造へも対応可能となっている．特に頂点数の多い大規模なグラフでは，全体を俯瞰する場合には画面に表示する情報を絞り込み，拡大するにつれて情報を増やすといった詳細度制御が必要となる．頂点間の包含関係を可視化することは，階層グラフ描画における詳細度制御にも有効である．また，対話的なネットワーク可視化システムでは，部分グラフの展開・折り畳みといったユーザインタラクションが導入されることが多い．

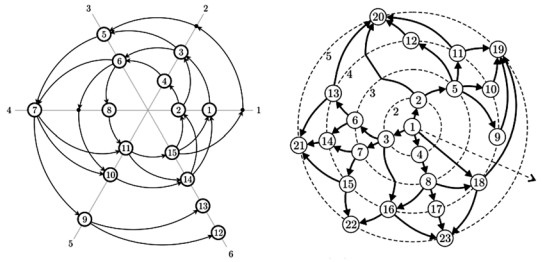

図 5　階層軸の放射状配置（左）と同心円状配置（右）の例（[2] より引用）

4.　階層グラフ描画の利用方法

前節まででは杉山フレームワークの理論的背景について概説した．本節では，それらが実装されたフリーソフトウェアおよびライブラリの紹介を行う．これらを利用するうえでは，アルゴリズムなどを理解している必要はないが，理解しているほうがより希望通りの出力が得られやすいであろう．

プログラミングが不要で杉山フレームワークを利用できるソフトウェアとしては GraphViz (http://www.graphviz.org/) や Tulip (http://tulip.labri.fr/) などがあげられる．GraphViz は有名なコマンドラインベースのネットワーク可視化ソフトウェアであり，階層グラフ描画の実装としてもよく知られている．GraphViz では，dot という形式でグラフの構造および見た目を定義する．GraphViz は，パラメータ設定不要で質の高いレイアウトを生成することができる．ラスタ画像の PNG，JPEG などや，ベクタ画像の SVG，PDF などといった豊富な出力形式を備えている．Tulip は，GraphViz とは対極的に，GUI ベースのネットワーク可視化ソフトウェアである．Tulip では，グラフ描画エンジンとして後述の OGDF を組み込んでおり，対話的な GUI 環境で杉山フレームワークの利用が可能

である．

いくつかの主要なプログラミング言語においては，杉山フレームワークを実装したオープンソースのライブラリが公開されている．OGDF (http://www.ogdf.net/) は，ドイツやオーストラリアのネットワーク可視化研究者が中心となり開発している C++ 用ライブラリで，最先端の研究成果を取り入れたグラフ描画手法が多数実装されている．igraph (http://igraph.org/) は，C 言語で実装されたネットワーク分析ライブラリであり，グラフ描画機能も数多く備えている．igraph は，R や Python といったデータサイエンス向けのプログラミング言語で利用することを念頭において開発されている．近年では Web ブラウザ上での情報可視化も盛んになっており，数多くの JavaScript 向け可視化ライブラリが開発されている．dagre (https://github.com/cpettitt/dagre) は，JavaScript による杉山フレームワークの有名な実装の一つである．これらのライブラリでは，レイアウトに必要なパラメータを細かく設定できる場合が多く，望んだ出力結果を得るためのチューニングが可能である．

5. おわりに

本稿では，ネットワーク可視化の一手法である階層グラフ描画の基礎と利用方法を紹介した．階層グラフ描画は有向グラフの可視化に有効な方法である．本稿で紹介した以外にもフリーソフトウェアやプログラミング言語用ライブラリとして多数の実装が存在しているので，ぜひ活用いただきたい．また，階層グラフ描画にはまだまだ応用や発展の余地が残されている．階層グラフ描画を用いたよりよい可視化を実現するためには OR 手法が必要不可欠であるため，それらについても本誌の読者に興味をもっていただけたなら幸いである．

謝辞　本特集オーガナイザーの三末和男先生ならびに編集委員の石井儀光先生には，本稿に多数の有益なコメントをいただきました．深く感謝申し上げます．

参考文献

[1] K. Sugiyama, S. Tagawa and M. Toda, “Methods for visual understanding of hierarchical system structures,” *IEEE Transactions on Systems, Man, and Cybernetics*, **11**, pp. 109–125, 1981.

[2] P. Healy and N. S. Nikolov, “Hierarchical drawing algorithms,” *Handbook of Graph Drawing and Visualization*, CRC Press, pp. 409–454, 2013.

[3] M. R. Garey and D. S. Johnson, “Computers and intractability: A guide to the theory of NP-completeness,” W. H. Freeman, 1979.

[4] P. Eades, X. Lin and W. F. Smyth, “A fast and effective heuristic for the feedback arc set problem,” *Information Processing Letters*, **47**, pp. 319–323, 1993.

[5] E. R. Gansner, E. Koutsofios, S. C. North and K. P. Vo, “A technique for drawing directed graphs,” *IEEE Transactions on Software Engineering*, **19**, pp. 214–230, 1993.

[6] P. Healy and N. S. Nikolov, “A branch-and-cut approach to the directed acyclic graph layering problem,” In *Proceedings of the 10th International Conference on Graph Drawing*, pp. 98–109, 2002.

[7] Y. Onoue, N. Kukimoto, N. Sakamoto and K. Koyamada, “E-Grid: A visual analytics system for evaluation structures,” *Journal of Visualization*, **19**, pp. 753–768, 2016.

[8] Y. Onoue, N. Kukimoto, N. Sakamoto, K. Misue and K. Koyamada, “Layered graph drawing for visualizing evaluation structures,” *IEEE Computer Graphics and Applications*, **37**, pp. 20–30, 2017.

[9] M. Jünger and P. Mutzel, “2-layer straightline crossing minimization: Performance of exact and heuristic algorithms,” *Journal of Groph Algorithms and Applications*, **1**, pp. 3–27, 1997.

[10] M. Jünger, E. K. Lee, P. Mutzel and T. Odenthal, “A polyhedral approach to the multi-layer crossing minimization problem,” *International Symposium on Graph Drawing*, pp. 13–24, 1997.

[11] M. Chimani, P. Hungerländer, M. Jünger and P. Mutzel, “An SDP approach to multi-level crossing minimization,” *Journal of Experimental Algorithmics*, **17**, article No. 3.3, 2012.

[12] U. Brandes and B. Köpf, “Fast and simple horizontal coordinate assignment,” In *Proceedings of the 10th International Conference on Graph Drawing*, pp. 31–44, 2002.

[13] Y. Onoue, N. Kukimoto, N. Sakamoto and K. Koyamada, “Minimizing the number of edges via edge concentration in dense layered graphs,” *IEEE Transactions on Visualization and Computer Graphics*, **22**, pp. 1652–1661, 2016.

[14] F. J. Newbery, “Edge concentration: A method for clustering directed graphs,” *ACM SIGSOFT Software Engineering Notes*, **14**, pp. 76–85, 1989.

[15] C. Bachmaier, F. J. Brandenburg, W. Brunner and G. Lovász, “Cyclic leveling of directed graphs,” In *International Symposium on Graph Drawing*, pp. 348–359, 2009.

[16] C. Bachmaier, “A radial adaptation of the sugiyama framework for visualizing hierarchical information,” *IEEE Transactions on Visualization and Computer Graphics*, **13**, pp. 583–594, 2007.

[17] 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，**30**, pp. 1324–1334, 1989.

[18] K. Sugiyama and K. Misue, “Visualization of structural information: Automatic drawing of compound digraphs,” *IEEE Transactions on Systems, Man, and Cybernetics*, **21**, pp. 876–892, 1991.