インタラクティブシステムのためのグラフ描画アルゴリズム

南雲 淳
筑波大学大学院理工学研究科
nagumo@softlab.is.tsukuba.ac.jp

田中 二郎
筑波大学電子・情報工学系
jiro@softlab.is.tsukuba.ac.jp

Graph drawing algorithm for the interactive system

Jun Nagumo
Master's program in Science and Engineering
Univ. of Tsukuba

Jiro Tanaka
Institute of Information Sciences and Electronics
Univ. of Tsukuba

概要

本研究では，グラフ自動描画アルゴリズムの改良として，配置対象のグラフの構造が動的に変化する場合に注目し，その変化を見る側に分りやすく提示し，インタラクション性を向上するための手法を示す．また，多視点遠近画法を導入することにより，辺や節が多い複雑なグラフの一覧性と詳細性を両立させて，可読性を向上するためのグラフ自動描画アルゴリズムの改良を行う．

また，このアルゴリズムの改良の応用として並列論理型言語をベースとしたビジュアルプログラミングシステムを作成し，このアルゴリズムが実際のアプリケーションに適用可能であることを示したので，これについても述べる．

Abstruct

In this research, we propose an improvement of the graph drawing algorithm based on a force-directed placement approach. Paying attention to the case where the graph structure dynamically changes, the improved approach shows us the change in more human-friendly manner and it results to increase interactivity.

We also propose to introduce "Fisheye-view" into the graph drawing algorithm. By introducing "Fisheye-view," the readability of the graph will be increased while preserving the totality and the detaility.

Moreover, we have implemented a visual programming system, based on a parallel logic language, in order to confirm this improvement of the graph drawing algorithm.

1 はじめに

情報を表現する手段の一つとしてグラフ構造がある．グラフ構造は，ビジュアルプログラミングシステムにおいてプログラムやデータの構造を表示したり，発想支援システム・検索システムなどで大規模データを視覚的に表示するのに用いられる．

グラフ構造の辺や節を「見やすさ」や「美的基準」を元に画面上に配置するためのアルゴリズムとしてグラフ描画アルゴリズムの研究がある [1, 2].

ビジュアルプログラミングシステムや発想支援システム・検索システムなどにグラフ描画アルゴリズムを用いる研究も現われている [3, 4]

これまでのグラフ描画アルゴリズムの研究はグラフの論理的構造のみから配置を求める静的配置の研究であり，表示されているグラフの構造・配置が動的に変化するような動的配置の研究はあまりなされていない．

実際にグラフ構造を表示し，それをユーザが操作するようなアプリケーションを創る際には，途中でユーザの入力によってグラフ構造が動的に変化する可能性がある．また，グラフ構造の変化の様子をスムースに見せる必要がある．グラフ構造の表示を用いたアプリケーションのインタラクティブ性を高めるためには，グラフ描画アルゴリズムもこのような要求に対応しなくてはならない．したがって，これらの要求を満たすグラフ変形のアニメーション的表現について研究を行う必要がある．

また，画面上の図が複雑になったときの見易さの向上のために，ユーザが注目したい部分の周辺だけを拡大するような画法が多視点遠近画法として研究されている [6, 7, 8]．これをグラフ描画アルゴリズムに取り入れることによってグラフ構造の表示を用いたアプリケーションの使い勝手の向上が図れる．

これら 2 つのことに注目し，本論文ではグラフ変形のアニメーション的表現と多視点遠近画法の自動描画アルゴリズムへの組み込みについて述べる．

2 アニメーション表現向け動的配置

ユーザに提示されるグラフの構造が動的に変化する場合，その変化の様子はスムースな表示によって表現されなくてはならない．特に，変化前と変化後で節や辺の位置が大幅に変わる場合には，ユーザの混乱を招かないためにも「どの節がどこに移動したのか」をアニメーションなどの手法によって表現する必要がある．

アニメーション表現の実現には，配置前と配置後の状態の間を「コマ割り」して，それを順次表示する手法がまず考えられるが，これだと，まず「配置」して，それから「コマ割り」して，それを順次表示するというプロセスを踏む必要があり，そのような手法で創られたアプリケーションは「再配置計算→アニメーション」というモードの流れが生じるために，システムに充分なインタラクション性を持たせることが難しい．

一方，反復法を用いてグラフ再配置を行うアルゴリズムを使い，その各計算ステップの状態をそのまま表示することによってアニメーションを実現すれば，「コマ割り」のプロセスを踏む必要はない．

また，例えばグラフの表示されるフィールドに対してバックグラウンドで再配置アルゴリズムを動かしておき，ユーザのグラフに対する操作を並列に受けつけるような処理の実装もしやすくなり，アプリケーションのインタラクション性が向上すると考えられる．

この考え方を元に，反復法を用いるグラフ描画アルゴリズムを改良することでアニメーション的表現向けのアルゴリズムを提案する．以下で，改良のベースとした Eades スプリングモデルとその改良について述べる．

2.1 Eades スプリングモデル

一般にグラフ描画アルゴリズムは対象とするグラフの種類によって様々な描画アルゴリズムが提案されている．ここで対象とするデータ構造はグラフ構造であり，エッジの方向を抽象化すれば無向グラフの描画アルゴリズムが使用できる．無向グラフの描画アルゴリズムとして最も基本的なものは Eades スプリングモデルである．

そこで本研究では，改良のベースとして Eades スプリングモデル [5] を用いることにした．

Eades スプリングモデルは，各節同士の間に引力や斥力を定義した力学系を考え，それの釣り合いを求めることによってグラフの再配置を行うものである．辺により直接接続された辺同士の間には引力 (ただし，節間の距離がごく小さいときには引力)，直接接続されない節同士の間には斥力が働く．

直接接続された節の間に働く力を f_s，そうでない節の間に働く力を f_r とすると，

$$f_s(d) = C_s \log \frac{d}{d_0} \tag{1}$$
$$f_r(d) = C_r \frac{1}{d^2}$$

のように定義される．

ここで，C_s, C_r は，引力と斥力のバランスをとるための定数である．また，再配置の際の理想的な辺の長さを (基本的にはアプリケーション開発側が) 定数 d_0 として与える．d は節の間の実際の距離である．

実際に再配置を行う際は，まず，ある 1 つの節に注目して，それ以外の全ての節から受ける引力・斥力を上の式で計算し，その合計を求めて (ベクトルとして求まる)，その大きさに比例した距離だけその節を動かし．これを全ての節に対して行う．このことを何度も反復させることによって各節を適当な位置に収束させていく．

初期配置などによっては，目標地点付近で振動することなどがあるが，1 ステップあたりの移動距離が微小となったところで収束したとみなして計算を終了すれば良い．

このように，反復法を用いるアルゴリズムであるので，この計算過程を順次表示することによってアニメーション表現が実現できれば，アニメーション作成のためのコマ割りなどは一切不要になる．

2.2 Eades スプリングモデルの改良

しかし従来の Eades のアルゴリズムでは，反復の過程で節がどのような経路を通って目的位置に動くかということが考慮されていないので，計算過程を順次表示することによってアニメーション表現を生成することができない．

我々は，従来の Eades のアルゴリズムの計算過程をそのまま順次表示すると，特に斥力の働く節同士の距離が短い場合に斥力が大きくなりすぎ，節が「跳ねる」ような動きを見せるため，アニメーション表現としては不適切であることを見つけた．

グラフ構造の動的な変形を考える場合，ユーザによる節の追加や削除だけでなく，自動的に起こる節の分割なども考える必要がある．

たとえば，ビジュアルプログラミングシステムにおいてはある節がリダクションなどにより複数の節や辺を持つグラフ構造に動的に変化する場合がある．

この場合，節同士が接近した状態に置かれるわけであるが，オリジナルの Eades のモデルでは節間距離が短い節同士の斥力が強いために，これをそのまま表示することによってアニメーション生成するとこれらの節の動きが不適切なものになってしまう．

これを解消し，反復計算の計算過程を順次表示することによって自然なアニメーションを実現するために，斥力を計算する関数を以下の式のようにする [3]．

$$f_r(d) = \begin{cases} C_{r'} & (d < d_0) \\ C_r \frac{1}{d^2} & (\text{Otherwise}) \end{cases} \tag{2}$$

すなわち，節の間の実際の距離が d_0 より小さいときには節の間に働く斥力 f_r を一定の値とするのである．

これにより，直接接続されない節同士のうちの特に接近している節の斥力を抑える．

この変更による効果を図 **1**に示す．図の左側を初期配置としたときの改良前のアルゴリズムによる移動軌跡が中央，改良後のアルゴリズムによる移動軌跡が右側である．

また，アニメーション的表現の見やすさにとって，各ステップ毎の各節の移動量が平均化されていることが重要であるが，この改良ではそれを実現できる．実際に，あるグラフに対して改良前と改良後の

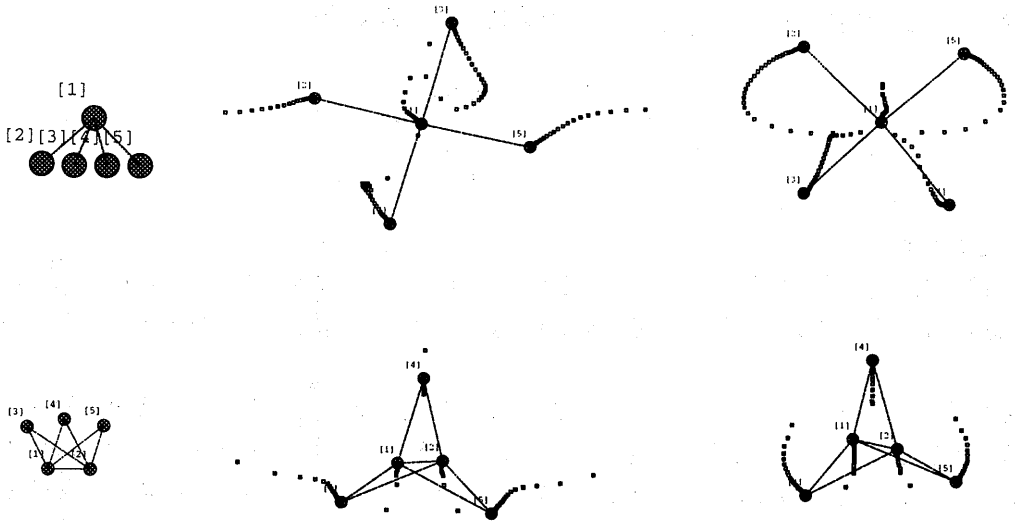

図 1: 改良前と改良後の比較 (左が初期位置，中央が改良前，右が改良後)

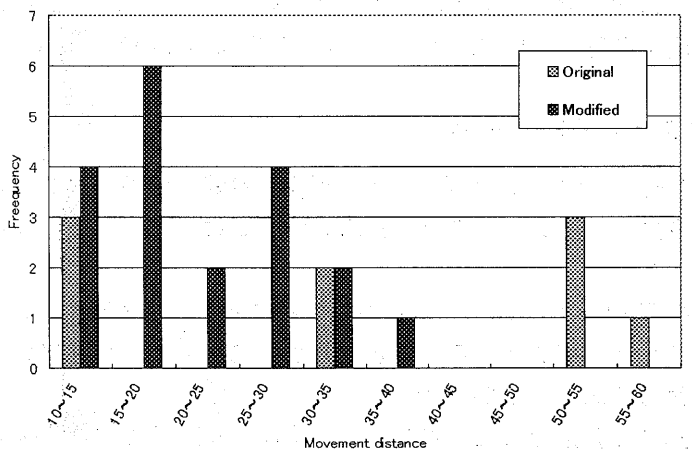

図 2: 節移動量の比較

アルゴリズムで各ステップ毎の節の移動量を計測した結果を図 **2**に示す．1 ステップ当りの移動距離が 10 ピクセルを越えるものについて集計してある．

図の横軸は 1 ステップ毎の節の移動距離の平均であり，縦軸は度数である．

この図から，改良前のアルゴリズム (図中では “Original”) では 1 ステップで大きく動くことが多いのに比べ，改良後のアルゴリズム (図中では “Modified”) ではそのようなことがなくなっていることがわかる．

3 多視点遠近画法の取りこみ

複雑なグラフ構造を表示してユーザに提示する場合，ユーザが注目したい部分を拡大により詳細に表示し，そうでない部分を縮小などの方法によって簡略な表示をすることで見やすさの向上を図る手法がある．

グラフ構造の魚眼表示については，Sarkar らによる研究 [6]，三末らによる研究 [7] や Storey によるもの [8] などがあるが，これらはいずれも節の位置を元にした多視点遠近画法の実現である．

こうした節の位置に基く多視点遠近画法を Eades スプリングモデルと組み合わせる場合，まず Eades

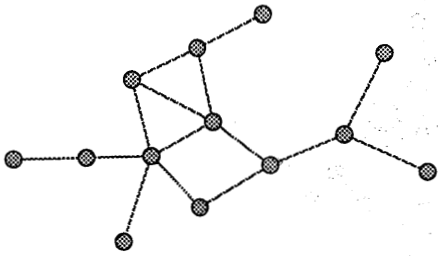

図 3: 従来の Eades での再配置

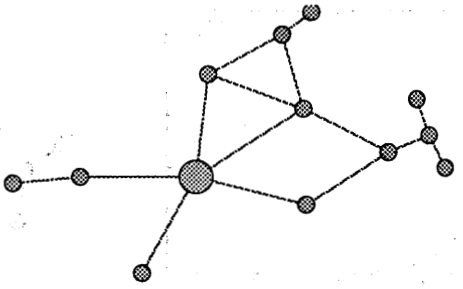

図 4: 注目点を定めた例

スプリングモデルに基き節や辺の配置を計算し，引き続き多視点遠近画法により節の位置を変換する必要がある．MVC (Model-View-Controller) モデルで言えば，Eades スプリングモデルは Model を変形し，多視点遠近画法では View を指定するといった二重のプロセスを踏む必要がある．

このような枠組を用いた場合，アプリケーションに動的な操作を行う場合，それが Model に対するものか View に対するものかを指定しなくてはならず，操作の枠組が複雑となる．

本論文では，節の位置を元にした多視点遠近画法ではなく，注目点の周辺の辺の長さを長くし，逆に注目点から離れた部分の辺を短くすることによって注目点の周りに空間を作り，節を大きく表示できるようにするような辺の長さに基づく多視点遠近画法について述べる．

この手法は，節の位置の情報を用いないため，Eades のアルゴリズムとの親和性が良く，Eades のアルゴリズムに基く節や辺の位置の計算と多視点遠近画法の計算を一度に行うことができる．

また，先に挙げたアニメーション的表現向けの改良をそのまま適用することが可能であり，グラフ構造の表示を用いたアプリケーションのインタラクション性の向上に対する効果も大きい．

3.1 辺の長さに基く多視点遠近画法

今注目する節と再配置のために移動させようとしている節の間の距離 (2 つの節の間にある辺の数の最小値) を l，注目する節に接続する辺の理想的な長さを d_0 とする，

さらに a と b を定数として，再配置の際の「視点つき理想的辺長」を $d_{0'}$ を

$$d_{0'} = d_0 \left\{ \frac{1}{1+e^{a(l-b)}} \right\} \tag{3}$$

と定義する．この $d_{0'}$ を 式 **(1)** および式 **(2)** の d_0 のかわりに適用することによって，視点付近の辺は長くなり，視点から遠いところの辺が短かくなる．これによって，視点付近の拡大と視点から遠い部分の縮小が擬似的に表現可能になる．この関数の形は 図 **5**のグラフのようになる．グラフの横軸は注目点からの距離に相当する値であり，縦軸はそのときの辺の長さを表す値である．

従来の Eades のアルゴリズムでの配置を図 **3**に，多視点遠近画法を組み込んだアルゴリズムでの配置を図 **4**に示す．図 4で，他の節より少し大きく描かれている節が注目点である．

上の式の中の定数 a および b は，グラフの見え方を変えるためのパラメータとして使うことができる．

a を変化させると，図 5のグラフの「値が減少している部分」の傾きが変化する．たとえば，a を大きくすれば，注目点から遠くなるにつれて「急激に」辺が短かくなる．

b を変化させると，図 5のグラフ全体が左右に移動する．b を大きくすれば右に移動するので，注目点を中心とした拡大表示をされる範囲が広くなることになる．小さくした場合は逆に拡大表示される範囲が狭くなる．

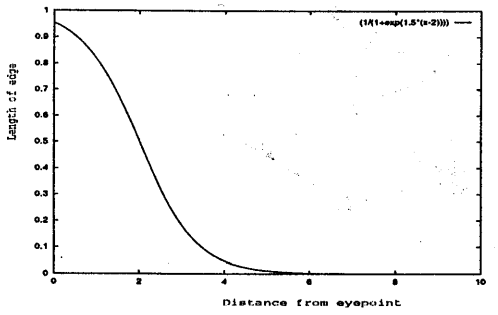

図 5: 関数のグラフ

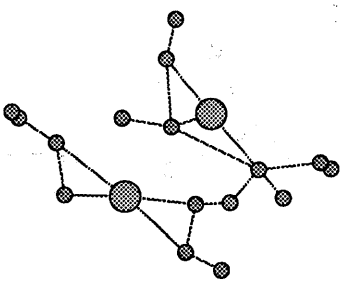

図 6: 複数視点の例

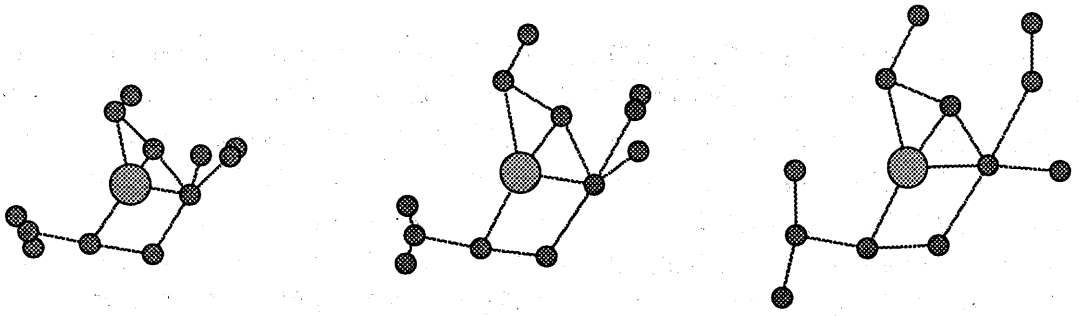

図 7: パラメータを変える例：左から $b = 1.0, 2.0, 3.0$ と変化させている

パラメータ b を変化させたときのグラフの見え方の変化を図7に示す．左からそれぞれ $b = 1.0, b = 2.0, b = 3.0$ としたものである．辺の長さが長くなる範囲が，注目点を中心として変化しているのがわかる．

このように，改良後のアルゴリズムではパラメータ変更による見え方の操作が容易である．

複雑なグラフ構造をユーザが見ている場合，その中で注目したい部分というのは 1 つであるとは限らない．そこで，このアルゴリズムを複数の視点に対応する必要がある．

ある辺に注目した場合，複数の視点が存在することにより，式 3 によって計算される辺の長さも複数 (視点の数だけ) 存在することになる．このような場合，その複数の辺の長さの中での最大値を用いることにすれば，適正な表示が得られる．

このことは，注目点 $1, \cdots, n$ のそれぞれについて式 **(3)** で計算した $d_{0'}$ の値を d_n とすれば，その辺に対して使うべき $d_{0'}$ は

$$d_{0'} \quad = \quad \max\{d_1, \cdots, d_n\}$$

と表現することができる．

複数視点の例を図6に示す．各視点から近い部分の辺は長くなっており，遠い部分の辺は短くなっている．

各節の大きさについては，それに接続する各辺の長さの最小値の半分を考え，その長さを半径とする円の中に収めれば，視点からの距離に従った適当な拡大・縮小が可能である．

辺の長さのパラメータを変化させることによってグラフの表示を変化させようとするものには，他に Wang らによるもの [9] がある．これは，Eades のアルゴリズムを改良した Fruchterman のアルゴリズム [10] に適用するものである．このアルゴリズムでは，引力と斥力の計算に含まれる長さのパラメータを別々に変更しており，本研究のやり方とは多少異っている．

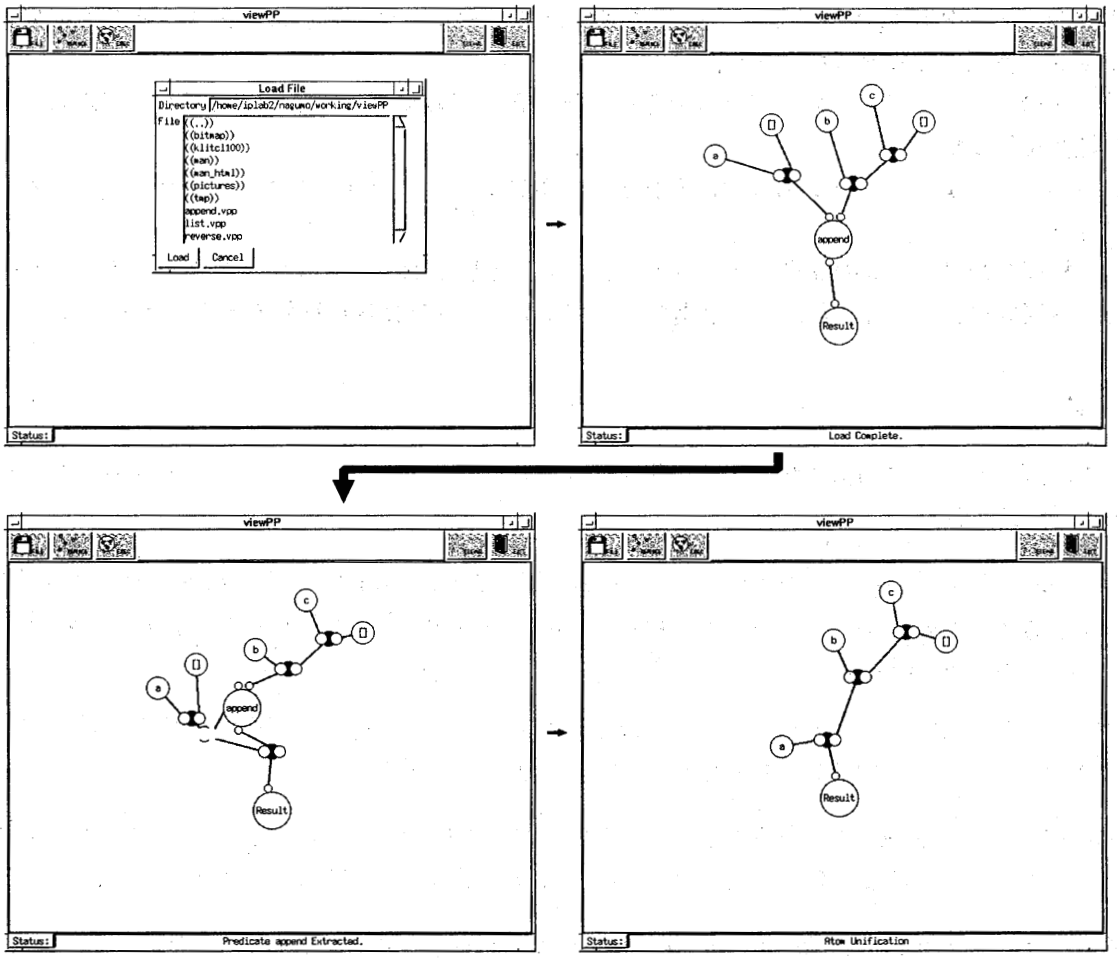

図 8: viewPP でのプログラムの実行の流れ

4 アプリケーション

我々は，本論文で述べてた描画アルゴリズムの改良を実際に用いて並列論理型言語のプログラムの実行を視覚化するシステムである “viewPP” を開発した (図 8).

並列論理型言語の視覚化にはグラフ構造が有効であり，我々は以前から無向グラフを用いて並列論理型言語ベースのビジュアルプログラミングシステムを開発してきた [11]. 今回，プログラム実行の視覚化のためにグラフ描画アルゴリズムとグラフ変形のアニメーション的表現を取り入れ，並列論理型言語の実行の様子がわかりやすく提示できるようになった.

このビジュアルプログラミングシステムでは，並列論理型言語プログラムを無向グラフ構造として抽象化し，その構造の変形をスムースなアニメーションとして表示することによってプログラムの実行過程を視覚化しているが，その際にアニメーション表示のためのコマ割りといった処理はしておらず，グラフの節の再配置の過程を順次表示することによってスムースな表示を実現している. また，節の移動中でもユーザによる節の移動などの直接操作が可能になっている.

5 まとめ

これまで考えられてきたグラフ描画アルゴリズムは「見やすい配置を求める」ことが主であり，表示されたグラフをユーザが操作することまで考慮されたものはほとんどなかった. しかし，ここでは表示されたグラフに対するインタラクティブ性まで考慮した上での描画アルゴリズムの必要性を述べると共に，改良した描画アルゴリズムを提案した.

無向グラフの動的描画について，Eadesスプリングモデルをベースとしたグラフ構造の変形のアニメーション的表現向けの改良を行った．

また，このアルゴリズムへの多視点遠近画法の組み込みが容易であることに着目し，実際にアルゴリズムの拡張を行った．

これらの改良により，グラフ構造の画面表示によるアプリケーションのインタラクティブ性の向上が期待できる．

また，描画アルゴリズムの改良のアプリケーション例としてビジュアルプログラミングシステム "PP" を挙げ，本研究によって行われた改良が実際のアプリケーションに適用可能であることを示した．

参考文献

[1] 杉山公造. グラフ自動描画法とその応用 — ビジュアル ヒューマンインタフェース —. 計測自動制御学会, 1995.

[2] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis Tollis. Annotated bibliography on graph drawing algorithms. *Computational Geometry: Theory and Applications*, Vol. 4, pp. 235–282, 1994.

[3] 南雲淳, 田中二郎. "viewpp": グラフ構造とアニメーション表現に基づくプログラム実行の視覚化. 日本ソフトウェア科学会第 14 回大会論文集, pp. 17–20, 1997.

[4] 舘村純一. Docspace: 文献空間のインタラクティブ視覚化. 田中二郎（編）, インタラクティブシステムとソフトウェア IV 日本ソフトウェア科学会 WISS'96, pp. 11–20. 近代科学社, 1996.

[5] P. Eades. A heuristics for graph drawing. *Congressus Numeranitium*, Vol. 42, pp. 149–160, 1984.

[6] Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. In *Proc. CHI '92*, pp. 83–91, 1992.

[7] 三末和男, 杉山公造. 図的思考支援を目的とした図の多視点遠近画法について. 情報処理学会論文誌, Vol. 32, pp. 997–1005, 1991.

[8] Margaret-Anne D. Storey and Hausi A. Müller. Graph layout adjustment strategies. In *Graph Drawing, Symposium on Graph Drawing, GD'95, LNCS 1027*, pp. 487–499, 1995.

[9] Xiaobo Wang and Isao Miyano. Generating customized layouts. In *Graph Drawing, Symposium on Graph Drawing, GD'95, LNCS 1027*, pp. 504–515, 1995.

[10] Thomas M.J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software-Practice and Experience 21-11*, pp. 1129–1164, 1991.

[11] Jiro Tanaka. Visual programming system for parallel logic languages. In *Workshop on Parallel Logic Programming and its Program Environments, the University of Oregon*, pp. 175–186, 1994.