芸術科学会論文誌 Vol.3 No.4 pp250-263

# 力学モデルを用いた階層型グラフデータ画面配置手法の改良手法とウェブサイト視覚化への応用

土井 淳　　伊藤 貴之*

日本アイ・ビー・エム(株)　東京基礎研究所　(* 京都大学情報学研究科と兼任)

# An Improvement of Force-directed Hierarchical Graph Layout And Its Application to Web Site Visualization

Jun DOI　　Takayuki ITOH

IBM Research, Tokyo Research Laboratory

doichan@jp.ibm.com, itot@computer.org

グラフデータの視覚化技術は，近年活発に研究が進められており，金融・交通・通信・社会組織・科学・計算機システム・インターネットなど，非常に幅広い分野のデータ分析およびデータ監視の目的での実用が報告されている．グラフデータの視覚化における最も大きな問題は，「グラフを誤読させない適切なノードの画面配置を，自動的に実現する」という問題である．この問題を解決するために，ノードに分子間力モデル，アークにバネモデルを適用して，運動方程式によって良質なノード配置結果を得る手法が提案されている．

本論文では，上記のような「力学モデルを用いたグラフデータの画面配置手法」の改良手法および階層型グラフデータへの拡張手法を提案する．本手法は，ノードを１個ずつ配置するインクリメンタルなアルゴリズムにより，配置結果を改善するとともに，計算時間の増加を抑えることに成功している．

また本論文では，上記手法を用いたウェブサイトの視覚化結果を提示する．本手法では，ウェブサイトを構成するウェブページをノード，ウェブページ間のハイパーリンクをアークとして，またウェブページのディレクトリ階層を参照してウェブページを階層型データに格納することにより，ウェブサイトを階層型グラフデータとして表現する．この階層型グラフデータを上記手法により画面配置し，個々のウェブページをサムネイル画像で表示することにより，ウェブサイトの全体像を表現する．

## 1. はじめに

グラフデータのグラフィックス表示技術は，非常に幅広い分野のデータ分析およびデータ整理に有用である．最近では，

- ウェブサイトのリンク構造の表示．
- 金融・通信・交通・社会組織などの各種ネットワークの表示．
- 化学・生物などのサンプリングデータの傾向理解のために，データを関連性で連結したグラフの表示．
- テキストデータや画像データの整理のために，データを関連性で連結したグラフの表示．
- 例えば並列計算機のプロセスなどのように，関連あるモジュールの集合で構成されるシステムの振舞いを表したグラフの表示．

などの分野での実用例が報告されている．また，グラフデータ視覚化のための基礎技術については，近年になって有用な解説書やサーベイ論文が出版されている[1] [2].

グラフデータの視覚化における最も大きな問題は，「グラフを誤読させない適切なノード画面配置を，自動的に実現する」という問題である．この問題を解決するために例えば，

**[条件 1]** 近隣ノードが一定以上の距離を保つことによ

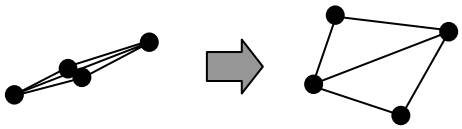
り，ノードどうしの画面上の重なりを避ける．

**[条件 2]** アークの長さの総和をできるだけ短くして，アークで連結されたノードをお互いに近い位置に配置する．

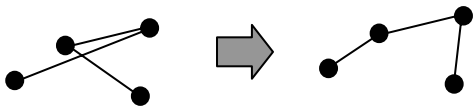
**[条件 3]** アークどうしが交差しないようにする．

**[条件 4]** アークが端点以外の場所で別のノードと重ならないようにする．

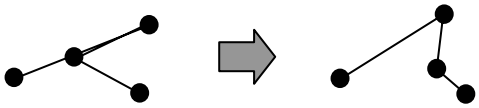
などの条件（図 1 参照）を最大限満たすような配置結果を算出する手法が多く報告されている．



［条件 1］ノード同士の重なりを避ける



［条件 2］アークの長さの総和を小さくする
［条件 3］アーク同士の交差を避ける



［条件 4］アークとノードの重なりを避ける

図 1 グラフデータ画面配置の典型的な条件

上記の問題に対して，多くの視覚化アプリケーションが求めている要件は，「最適でなくてもいいから，誤読を防ぐことのできる最低限以上の配置結果を，高速に求める」ことである．この「最低限以上の結果」を「高速に求める」というトレードオフ的な条件に対して，比較的バランスのとれた手法として，グラフのノードに分子間力モデル，グラフのアークにバネモデルを用いて，運動方程式を解くことによって各々のノードの適切な位置を算出する手法（図 2 参照）が知られている．しかし，この手法にしても，2 章にて後述するとおり，いくつかの課題を残している．

本論文では，分子間力モデルやバネモデルなどの力学モデルを用いたグラフ配置に関する改良手法を提案する． 本論文が提案する改良手法は，処理開始時からすべてのノードを配置する従来手法と異なり，1 個ずつインクリメンタルにノードを配置する．このアルゴリズムにより本手法は，「ノードの配置結果が初期配置に大きく依存する」という問題を改善するとともに，力学モデルの計算を局所化することで，従来手法よりも高速なノード配置を実現する．

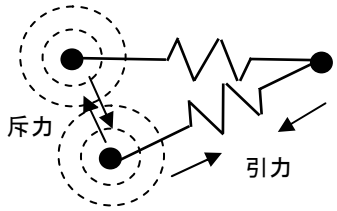



図 2　力学モデルのグラフデータ配置への適用

続いて本論文では，上記手法を階層型グラフデータの画面配置に応用するアルゴリズムも提案する．ここで階層型グラフデータとは，グラフを構成するノードに階層構造を設けた上で，ノード間を連結するアークを設けたグラフデータを指す．

さらに本論文では，上記手法を用いたウェブサイトの視覚化結果を提示する．本手法では，ウェブサイトを構成するウェブページをノード，ウェブページ間のリンクをアークとして，またウェブページのディレクトリ階層を参照してウェブページを階層型データに格納することにより，ウェブサイトを階層型グラフデータとして表現する．この階層型グラフデータを上記手法により画面配置し，個々のウェブページをサムネイル画像で表示することにより，ウェブサイトの全体像を表現する．

## 2. 関連研究

### 2.1. グラフデータの視覚化手法

グラフデータの視覚化手法は 1980 年代から広く研究が進んでおり，その歴史と構成はいくつかのサーベイ文献[1][2]に要約されている．1 章にて，分子間力やバネモデルなどの力学モデルを適用したグラフデータ視覚化手法について述べたが，これも[3]に代表されるように，1980 年代から研究が進んでいる．この手法は 1 章であげた条件のうち，主に[条件 1][条件 2]を満たすための視覚化手法であるといえる．また一般的に，[条件 2]を満たすことができれば，同時に[条件 3]を満たす可能性が高い．

この力学モデルを用いた視覚化手法には，以下のような問題点が指摘されている．

- 対話的に処理ができるほど高速ではない．特に大規模なグラフデータにおいて，非常に大きな計算時間を要する．
- ノードの配置結果が初期配置に大きく依存するにもかかわらず，適切な初期配置を決める方法がない．
- アークとノードの重なりを小さな計算時間で解消する有効な方法がない．つまり 1 章で示した[条件 4]を満たすことが難しい．

これらの問題点を解消するために，近年になって力学モデルを用いたグラフデータ視覚化手法には多くの改良手法が提案されている．例えば文献[4]では，10 万個以上のノードをもつ巨大なグラフデータへの適用が試みられている．また文献[5]では，アークどうしの交差を減らすことに主眼をおきながら，同時に適切なノードの配置を実現している．また文献[6]では，ボロノイ図を参照しながら力学モデルを適用することで，ノードの配置密度を改善している．

本論文の 3 章で提案しているインクリメンタルなノード配置手法に類似する手法として，文献[7][8]などの従来手法がすでに報告されている．しかしこれらの手法は，すでに部分的に配置されているグラフデータに対して，局所的な追加・削除・修正を行うことで，グラフデータのナビゲーションを実現するものであり，グラフデータ全体の画面配置の高速化を目的とするものではない．

本論文の 4 章で提案している階層型グラフデータ視覚化手法の一環として，階層にあわせてグラフデータを立体的に視覚化する手法や，その一部をズーム操作する手法が提案されている[9][10]．しかしこれらの手法も，階層型グラフデータ全体を 2 次元平面に高速配置する本論文の提案手法とは目的が異なる．

### 2.2. ウェブサイトの視覚化手法

ウェブサイトの視覚化手法の多くは，ウェブサイトを構成するウェブページ群を一覧表示する．最も多く見られるウェブサイト視覚化手法は，ウェブページをディレクトリ階層またはトップページからのリンク構造で階層化して表現する手法である．ウェブサイトの階層構造を表現するサイトマップの典型的な例として，Inxight 社が公開しているユーザーインタフェース[11] がある．この手法は，Hyperbolic Tree [12] という木構造データ視覚化手法を応用して，トップページを根にしたウェブサイトの階層構造を表現しており，ユーザーの対話操作に連動した局所ズームによる詳細表示を実現している．また，Durand らが提案した MAPA [13] も，ウェブページを階層構造にしたがって縦横に並べる表現を用いており，ユーザーの対話操作にしたがって段階的にかつ選択的にウェブページ群を表示できる．また山口らは，長方形の入れ子構造を適用した階層型データ視覚化手法を用いて，ウェブサイトのアクセス統計を視覚化する手法を提案している[14].

一方，ウェブページ群をノード，ウェブページ間のハイパーリンクをアーク，に変換することで構築されるグラフを視覚化する手法も多く提案されている．一例として，力学モデルを用いてグラフ構造を 3 次元空間に自動配置する手法を，ウェブサイトのリンク関係の視覚化に適用した手法が報告されている [15]. 一方，自動配置ではなく対話的操作によってグラフ構造を理解させる視覚化手法も多く知られている．塩澤らは，あらかじめ平面上に配置されたノードを持ち上げる操作によって，特定のウェブページのリンク関係を対話的に視覚化する手法を提案している [16]．これらの手法はいずれも 3 次元空間上にウェブページ群を配置しており, 2 次元平面上にウェブページ群を配置する本論文の提案手法とは大きく異なる．

## 3. 力学モデルを用いたインクリメンタルなノード配置の提案

### 3.1. 本手法で適用する力学モデル

本手法では，グラフのアークにバネ力，ノードに分子間力を想定し，運動方程式を用いてアークの安定長およびノード間の安定距離を求める．以下の説明では，対象となる 2 個のノードを分子中心点として，「分子としての半径の総和」に対する「ノード中心点間の距離(またはアークの長さ)」の比を $d$ とする．またすべての分子は等しい半径をもつものとする．

本手法では，以下の式を用いて，アークで連結された 2 個のノード間の斥力（または引力）$F_a$ を求める（図 3(左)参照）．

$$F_a = \begin{cases} -k(d-1.0) & ...(d < D) \\ -k(D-1.0) & ...(d \geq D) \end{cases}$$

ここで $k, D$ は定数であり，$D > 1.0$ であるとする．この式はフックの法則によるバネモデルに類似しているが，本手法では計算の発散を防ぐために，アーク長が十分長い $(d \geq D)$ ときに力の大きさが一定であるという条件を加えている．

また本手法では，以下の式を用いて，アークで連結されていない 2 個のノード間の斥力 $F_b$ を求める（図 3(右)参照）．

$$F_b = \begin{cases} k(\frac{5}{4}d^3 - \frac{19}{8}d^2 + \frac{9}{8}) & ...(0.0 \leq d < 1.0) \\ 0 & ...(d \geq 1.0) \end{cases}$$

ここで $k$ は定数であるとする．この式は，分子間力モデルの一つであるファン・デル・ワールス力を 3 次関数で近似したもの [17] であるが，本手法では引力は不要なので $d \geq 1.0$ のときに斥力をゼロとしている．

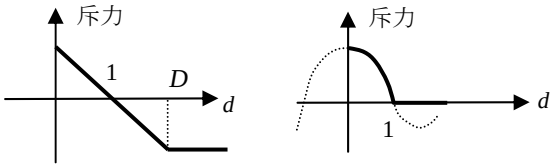



図 3　本手法で用いる引力・斥力の算出式

本手法では，ノードを 1 個ずつ画面空間に配置し，すでに配置されているノード間の引力および斥力を算出する．ここで，$i$ 番目のノードにかかる力の合計を $F_i$，位置を $x_i$，ノードの質量を $m$，ダンピング係数を $c$ とすると，下記の運動方程式

$$mx_i'' + cx_i' = F_i$$

の反復演算によって，ノードの安定な位置を求めることができる．

## 3.2. インクリメンタルなノード配置による力学計算の局所化

すでにいくつかのノードが配置されている図 4(a)のような状態から，1 個のノード $N_0$ を追加することを考える．本手法では，以下のような方法で力学計算を局所化し，計算時間の増加を抑える．まず $N_0$ だけに「変位中」のフラグを立てる（図 4(b)参照）．続いて，

- 変位中ノードとアークで連結されたノード
- 変位中ノードとアークで連結されていないが，十分近い距離にあるノード

のみに対して力を算出し，その力の総和を運動方程式に代入して，ノードの新しい位置を計算する．移動量が大きいノードについては，「変位中」のフラグを立てなおし（図 4(c)参照），「変位中」のノードに対して力を算出する（図 4(d)参照）．続いて，この力の総和によって再びノードの新しい位置を計算し（図 4(e)参照），同様に力を算出する（図 4(f)参照）．この処理の反復により，本手法では力学モデルの計算を局所化し，計算時間の増加を抑える．

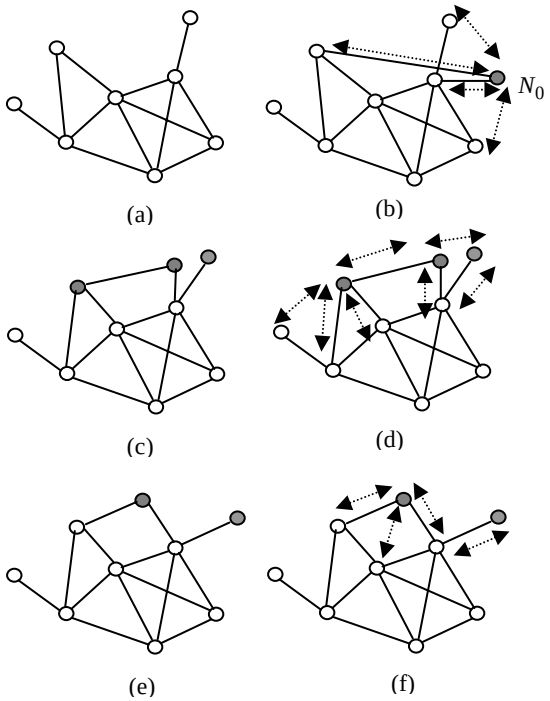



図 4　力学計算の局所化．灰色のノードに「変位中」のフラグがたてられている．力学計算は矢印で結ばれたノード間のみに行われ，それ以外のノード間の計算はすべて省略されている．

筆者らの実装では，力学計算の反復の各ステップにて，ノードの分子半径の 0.5 倍以上の移動量を有するノードに対してフラグを立てる．そしてフラグの立っているノードが全くなくなったときに，力学モデルの反復計算が収束した，と判断する．この「分子半径の 0.5 倍」という数字には理論的根拠はなく，筆者自身による観察から経験的に決めている．筆者らの実装では，大半の場合において 5～30 回程度の反復計算で，1 個のノード追加配置に対する反復計算を終了する．しか

し残念ながら，まれにノードの運動が振動して収束を妨げる事例もある．そこで筆者らは経験的に，1 個のノード配置に対する最大反復回数を 50 回とし，それまでに収束しなかった場合は 50 回目の反復計算時点での配置結果を用いる．

なお力学計算が収束しないという問題は，インクリメンタルなノード配置アルゴリズムに固有の問題ではなく，力学モデルを用いた配置問題の多くに共通する問題である．現実問題として，商用のグラフ描画ソフトウェア[18]に採用されている力学モデルにも，反復計算が収束しない現象は観察できる．またグラフデータ視覚化以外の問題，例えば三角メッシュ生成のための力学モデル[17]においても，反復計算が収束しない現象は観察されている．

### 3.3. ノード配置順の自動算出

本手法ではノードを 1 個ずつインクリメンタルに配置することを前提としているが，その配置順はユーザーが定義することもできるし，また自動算出することもできる．ユーザーが定義する重要度順にノードを配置すれば，ユーザーの関心の高い順にプログレッシブにグラフデータを表示することができる．逆に，良好な配置結果を得やすいような配置順を自動算出できれば，それが望ましい場合もある．

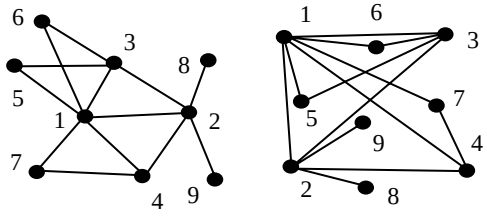



図 5　(左)隣接アーク数の多いノードが中央部に位置するような配置結果．(右)隣接アーク数の多いノードが周辺部に位置するような配置結果．

本手法ではノードの配置順を自動算出する一手法として，ノードを隣接アーク数でソートし，隣接アーク数の多いノードから順に配置する，という配置順を適用する．ここで本手法では，先に配置するノードが中央部に，後から配置するノードが周辺部に配置されるようにノード位置を決定する．なぜなら図 5 に示す通り，隣接アーク数の多いノードが中央部に位置する配置結果のほうが，アークの交差が少ない場合が多いと予想されるからである．この予想にしたがって本手法では，すでに k 個のノードが配置されているときに，(k+1)番目のノードの初期位置 $\mathbf{p}$ を以下の通り決定する．

$$\mathbf{p} = \mathbf{p}_0 + t(\mathbf{p}_0 - \mathbf{O})$$

ここで，$\mathbf{p}_0$ は(k+1)番目のノードにアークで連結されたノードの位置，$\mathbf{O}$ はすでに配置されているノード群の中心点，$t$ は正の実数である．この算出式によって初期位置を決定してから，力学計算を反復することにより，(k+1)番目のノードは隣接ノードよりも外側に配置される．この反復により，図 5(左)のような配置を実現する．

### 3.4. 本手法のアルゴリズムおよび特徴

本手法によるインクリメンタルなノード配置のアルゴリズムを，図 6 に示す．

1. ノードを所定の順で FIFO に登録する．
2. FIFO からノードを抽出し，「変位中」のフラグをたてる．
3. 抽出されたノードの初期位置を算出する．
4. (4a)～(4d)を，最低 1 個のノードに「変位中」のフラグがたっている限り反復する．
(4a) 変位中ノードについて，アークで結ばれたノードとの力を，バネモデルで算出する．
(4b) 変位中ノードについて，アークで結ばれていない近隣ノードとの力を，分子間力モデルで算出する．
(4c) 力を算出したノードの新しい位置を求める．
(4d) 変位の大きいノードに「変位中」のフラグを立てる．
5. FIFO が空になるまで，2.～4. を反復する．

図 6　インクリメンタルなノード配置手法のアルゴリズム．

すべてのノードを最初から画面配置する非インクリメンタルな手法と比較して，本手法には以下のような特徴がある．

一つ目の特徴として，本手法は非インクリメンタルな手法と比較して，計算時間を大きく短縮できる点が

あげられる．力学モデルを用いた反復計算の計算量は，ノード数を $n$ としたときに，力学計算がまったく局所化されていない状態で $O(n^2)$ であり，力学計算を局所化するにしたがって $O(n)$ に近づく．このことから，力学モデルの局所化に貢献している本手法は，計算時間の短縮に貢献できることがわかる．5 章に示す実験結果からも，計算時間の大幅な短縮が実証できている．

二つ目の特徴として，1 章で示した[条件 1][条件 2]の観点において，非インクリメンタルな手法と同等，あるいはそれ以上に良好な配置結果を得ることができる点があげられる．この特徴も，5 章に示す実験結果から実証できている．

三つ目の特徴として，3.3 節で示したノード配置順により，アークの交差を減らす，つまり 1 章で示した[条件 3]の観点で改善を達成している点である．この特徴も，5 章に示す実験結果から実証できている．

一方で，以下の点が課題として残っている．本手法は 1 章で示した[条件 4]を改善するものではない．したがって依然として，ノードとアークの干渉は見られることがある．この問題に対して筆者らは，アークを曲線化する手法によって，非階層型グラフデータの視覚化に対して改善を達成している[19]が，これを階層型グラフデータに適用するには問題を残している．よって 6 章で示すウェブサイトの視覚化には，アークの曲線化手法は用いていない．階層型グラフデータ視覚化およびウェブサイト視覚化のためのアーク曲線化手法を，7 章にて今後の課題として位置づけている．

## 4. 階層型グラフデータへの応用

### 4.1. 階層型グラフデータ

グラフデータは，ノード間の関係を，アークで結ぶことで表現されたデータであるが，これとは別に同じ属性のノードの集合をグループとしてとらえることで，階層型のグラフデータを表現することが可能である．さらに，グループ自体をノードとしてとらえることにより，グループ間の関係をアークで結ぶことで，新たなグラフデータを表現することができる．このようにして，ノード間の横のつながりとツリー構造のように階層的な縦のつながりを持ったようなグラフデータのことを，階層型グラフデータと呼ぶ．

図 7 に階層型グラフデータの例を示す．図 7(i)は階層化する前の通常のグラフデータである．ここで a～h のグループにノードを振り分けると，図 7(ii)のような新たなグラフが生成される．同じように図 7(iii)のようにさらに上位の階層のグラフが生成される．このとき，図 7(iii)のノード A に対して，図 7(ii)のノード a~c で構成されるグラフのことを A の子グラフ，逆にノード a~c から見てノード A を親ノード，ノード ABC で構成されるグラフのことを親グラフと呼ぶ．この例では，階層型グラフの階層の深さは同一であるが，図 7(i)のノードにさらに子グラフがあるような，深さの異なるグラフも考えられる．

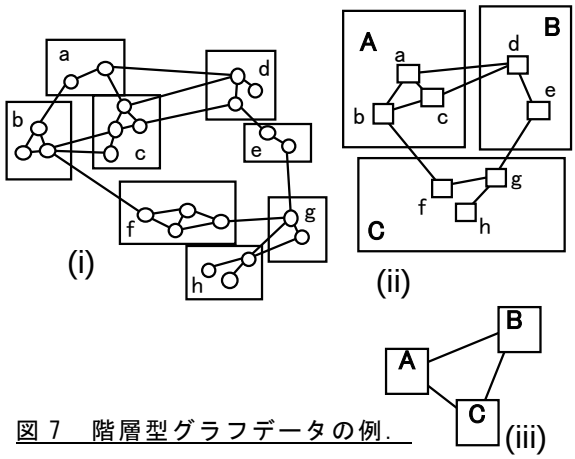



図 7　階層型グラフデータの例．

このような階層型のグラフデータは，例えば，地域毎の通信や輸送や電力等のネットワークの可視化，社内組織の関係やビジネスプロセスの可視化，等に有効である．

階層型のグラフデータを用いることによって，次のような効果を得ることができると考えられる．

・　大規模グラフの大まかな理解

上位の階層のグラフを表示することによってグラフ全体像の理解を助ける．

・　注目箇所の効果的な可視化

注目箇所の子グラフを任意に表示することにより，全体的な理解を損なわずに注目箇所の詳細なデータを得ることができる．

・　可視化処理の効率化

上位のノードから適応的にノードを配置することで，可視化処理を高速化することができる．この手法については後述する．

## 4.2. 階層型グラフデータのノード配置

階層型グラフデータも任意の階層でグループやノードをアークで結ばれたグラフデータであると考えられる．そこで，任意の階層のグラフデータのグループやノードを配置するには3章で述べた，力学的手法を用いたノード配置手法を用いることができる．本研究では，大規模グラフデータのノード配置処理の効率化と，インタラクティブな可視化を実現するために，最上位階層から再帰的に子グラフを配置していく手法を採用した．この手法のアルゴリズムを図8に示し，詳細について述べる．

1. グラフGを構成するノードを配置する
2. グラフGに属するノードNi(i=0,1,…,n)について処理3〜6を行う
3. ノードNiが子グラフGiを持つとき，子グラフGiについて別の空間を用意し，1からの処理を再帰的に呼び出す
4. 処理3で子グラフGiの占有領域にあわせてノードNiの大きさを拡大する
5. ノードNiのサイズ変更に伴いグラフGを局所的に再配置する
6. ノードNiに子グラフGiをマッピングする
7. 親ノードにグラフGの占有領域を返し親グラフの処理4に反映する

図 8　階層型グラフデータの再帰的なノード配置手法のアルゴリズム．

図8に示したアルゴリズムは，階層型グラフの任意の階層の部分グラフのノード配置処理に適用することができ，この処理を再帰的に呼び出すことによって，グラフ全体をノードの重なりなしに配置することができる．図8のそれぞれの処理について図9を用いて説明する．

処理1は，図9(i)のように着目している階層のグラフABCのノード配置を行う．続いて処理2のように各ノードABCについて処理3〜6を行う．

処理3では，図9(ii)のように，ノードCについて子グラフの配置を行う．図9(ii)の円内のように，親グラフとは別の空間内でノード配置処理を行う．この処理は，図8のアルゴリズムを再帰的に呼び出すことで実現する．

処理4では，図9(iii)のように，別空間に配置された子グラフのバウンディングボックスを親ノードCの大きさに反映させ，親ノードCを拡大する．ノードの拡大に伴い，親ノードが他のノードと重なってしまうことがある．このような場合，処理5において，重なりがなくなるように，親グラフのノードを再配置する必要がある．この再配置処理は，下の階層のグラフが配置されるたびに頻繁に呼び出されるため，効率よく処理するために3.2章で述べた局所的なノード配置処理を適用する．拡大された親ノードに「変位中」のフラグを立てて，図4(b)〜(f)と同様な局所的な力学計算を反復することにより，高速な再配置処理を実現する．なお，処理4によって拡大された親ノードが既に配置されているアークと交差してしまう可能性がある．この問題は，3.4節でも述べられているように，本手法においては，根本的にノードとアークの交差を除去することを解決してはいない．今後，このような場合についても，交差を取り除くような処理の導入を検討していきたい．

処理6では，図9(iv)のように，拡大された親ノードの位置に，別の空間で配置された子グラフをマッピングすることで，既に配置されているノードと重なりなしに子グラフのノードを配置することができる．

このような処理により，任意の階層のグラフを適応的に可視化することが可能となる．ユーザーはグラフの全体像をつかみつつ，注目したい部分の詳細なグラフを見るといった操作が可能になる．また，階層型グラフデータの配置処理を用いることで近い性質のノードを近い位置に配置することができ，グラフの理解を助ける効果もある．

階層型グラフデータの可視化の例をデジタルデータとして用意した．図8に示した処理の過程の様に，親グラフの配置結果から，親ノードを選択していくことにより，子グラフが配置されていく様子が分かるようになっている．

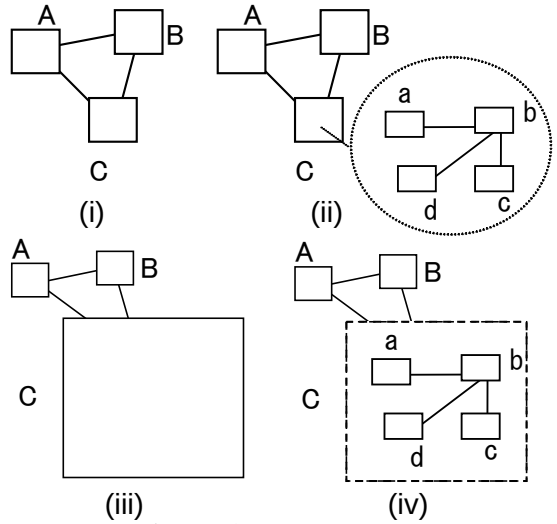



図 9 階層型グラフデータのノード配置の例.

図 9(iv)において, 親ノードに挿入された子グラフのノードから, 実際にはノード A および B に属する子グラフのノードへ, アークが接続されていることになるが, ここでは, 親ノード C と親ノード A および B とを結ぶアークで代表して表示することにしている. 本手法では, 正確なノード間の繋がりを表示することよりも, 階層構造と, その間の関係を見やすくすることに重点をおいて可視化を行う方向で考え, 親ノード間のアークによる接続で子ノード間の複雑な接続関係を代表して表示することで表示が複雑になるのを防いでいる. なお, 子ノード間のアークの情報は保持されているので, 特定のノードを選択して, 元のアークを再現することも可能であるが, ここでは, そのアークが他のノードやアークと交差する問題については解決していない. アークを再現した後, 局所的な再配置処理を子ノードに対して実行することで, 交差を取り除くことは可能であるが, これによってグループ化されたノードが遠くに配置されるといった問題や, 処理の効率を落とす問題がある.

また, 大規模なグラフデータにおいては, 同時にすべてのノードの配置処理を行おうとすると膨大な計算時間が必要となる. 本手法による力学計算によるノード配置の計算量は, $n$ 個のノードの配置を行う場合, 力学計算は実際には局所的に計算されるが 1 個のノードに対して最大で $n-1$ 組の計算を行うため, $O(n^2)$である.

しかし, 階層型グラフデータを用いることで各部分グラフでは少ないノード数で配置処理を行うことができ, 上位グラフの再配置処理も局所的な処理で済むのでグラフ全体の配置処理の計算時間を比較的短くすることができると考えられる. 仮に, $n$ 個のノードのグラフについて, $k$ 個ずつのノードを $m$ 個のグループにわけ親グラフを作り, さらに $k$ 個ずつの親ノードをグループにまとめて・・・という処理を反復して階層型グラフを作ったとする. このときの階層の深さは平均 $\log m$ となり, 部分グラフの総数は平均 $m \log m$ と見積もることができる. このとき, 図 8 に示した再帰的なノード再配置処理の回数は $m(\log m)^2$ と見積もることができる. また, 階層型グラフデータのノード配置処理では, 力学計算以外の計算量も多くなるが, その計算量は力学計算の計算量と比較して無視できるくらい小さい. よって, 階層型グラフデータの配置処理における力学計算の計算量は, $m(\log m)^2 O(k^2)$となり, $O(n^2)$よりも少ない計算量となる. 実際にノード配置処理の処理時間を比較してみると, 図 16 のようになる.

## 5. グラフデータ配置手法の実行例

本章では, 3 章および 4 章で提案したグラフデータ配置手法を実装して実行した結果を示す. 筆者らは, 本手法を Microsoft Visual C++で実装し, IBM IntelliStation M Pro (Intel Pentium III 3.06 GHz RAM 1.0GB)および, Microsoft Windows XP 上にて実行した ※.

図 10 は, 3 章で提案した手法によるインクリメンタルなノード配置処理のスナップショットを示したものである. なおこのグラフデータは, 表 1 に示されている図 11 のグラフデータと同一のグラフデータである.

図 11 および図 12 は, 3 章で提案した手法によるインクリメンタルなノード配置結果と, 初期段階からすべてのノードを配置する非インクリメンタルな手法によるノード配置結果を比較した

※ Visual C++および WindowsXP は Microsoft 社の登録商標. IntelliStation は IBM 社の登録商標.

ものである．ここで非インクリメンタル手法では，すべてのノードを画面空間の原点近傍に，座標値を乱数として発生させることで初期配置した．また非インクリメンタル手法と提案手法とでは，力学計算の係数や収束条件などを同一にしている．

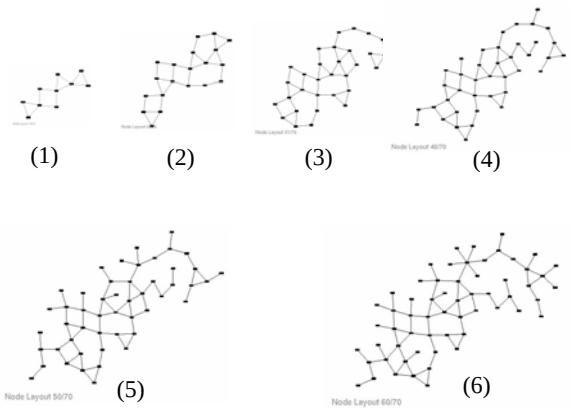



図 10 本手法によるインクリメンタルなノード配置処理のスナップショット．

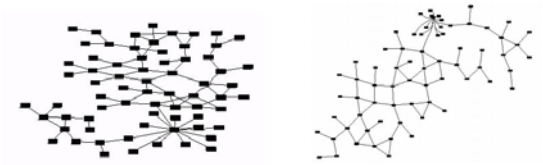


図 11 ノード配置結果の比較(1)．左は非インクリメンタル手法，右は本手法．

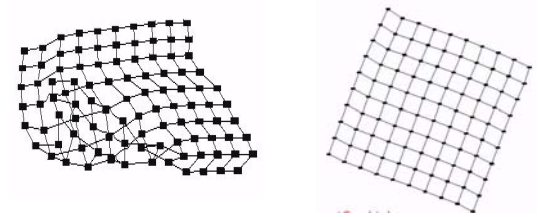


図 12 ノード配置結果の比較(2)．左は非インクリメンタル，右は本手法．

表 1 は，図 11 および図 12 の対象となるグラフデータのノード数およびアーク数を示したものである．

表 2 は，図 11 および図 12 の結果に対する処理時間を実測したものである．表 2 および表 3 は，図 11 および図 12 の結果に対して，アークの長さの平均値，アークの交差数，を算出することで，配置結果を数値評価したものである．ここでアークの長さは，理想値（力学モデル上の分子半径）を 1 とした相対値で表現しており，その平均値が 1 に近いほど良好な配置であるといえる．またアークの交差数は少ないほど良好な配置であるといえる．

表 1　図 11,12 の対象となるグラフデータ

| | 図 11 | 図 12 |
|---|---|---|
| ノード数 | 70 | 100 |
| アーク数 | 81 | 180 |

表 2　処理時間の測定結果(1) (単位ミリ秒)

| | 図 11 | 図 12 |
|---|---|---|
| 非インクリメンタル手法 | 9542.5 | 19477.1 |
| 本手法 | 2483.7 | 4052.3 |

表 3　配置結果の数値評価 (アークの長さの平均)

| | 図 11 | 図 12 |
|---|---|---|
| 非インクリメンタル手法 | 1.14 | 1.24 |
| 本手法 | 0.94 | 0.99 |

表 4　配置結果の数値評価 (アークの交差数)

| | 図 11 | 図 12 |
|---|---|---|
| 非インクリメンタル手法 | 8 | 31 |
| 本手法 | 0 | 0 |

図 11 については，表 3 よりアークの長さについては両手法とも好ましい結果となっており，また表 4 よりアークの交差数についても両手法とも少なくなっている．しかし表 2 からわかるように，提案手法のほうが計算時間は大幅に小さい．

図 12 については，表 4 からわかるように，非インクリメンタル手法ではアークの交差数が非常に多いことから，提案手法のほうが明らかに良好な配置結果を実現している．また表 2 からわかるように，提案手法のほうが計算時間は大幅に小さい．

図 13 は，本手法による階層型グラフデータのノード配置処理のスナップショットを示したものである．図 13(1)～(4)に矢印で示したノードを選択しその子グラフを展開することでノード配置を行う様子を示した．図 13(6)が最終的なノード配置結果である．

図 14 および図 15 は，それぞれ同一のグラフに対して，階層化したものと，そのままの状態のものとで，ノード配置処理をした結果を示したものある．この結果から，階層型グラフデータのノード配置処理では，ノードのグループが明確に表現され，グラフの全体的な構造を把握しやすいものとなっている．

なお，図 14，図 15，図 16 で使用したグラフデータは，*n* 個（図 14 では *n=50*，図 15 では *n=100*）のノードをランダムにアークで結び，階層型グラフは，同じグラフに対して， アークで隣接する 5～10 個のノードが入るようにグループを作っていき，グループからできるグラフを反復的に階層化して作られたものである．

表 5 は，図 14，図 15 それぞれの配置処理の処理時間を比較したものである．また図 16 は，階層型グラフと非階層型グラフのノード配置処理の処理時間を，グラフのノード数で比較した測定結果である．階層の深さや各グループに属するノード数等の違いで結果が変化するが，階層型のグラフデータのノード配置処理によって，効率よくグラフの配置処理が行われていることがわかる．

表 5　処理時間の測定結果(2) (単位ミリ秒)

| | 図 14 | 図 15 |
|---|---|---|
| 通常のグラフ | 510.2 | 1993.1 |
| 階層型グラフ | 66.9 | 156.6 |

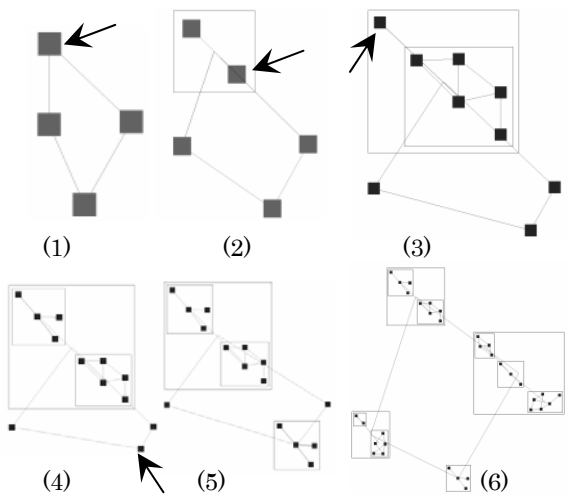



図 13 本手法による階層型グラフデータのノード配置処理のスナップショット．

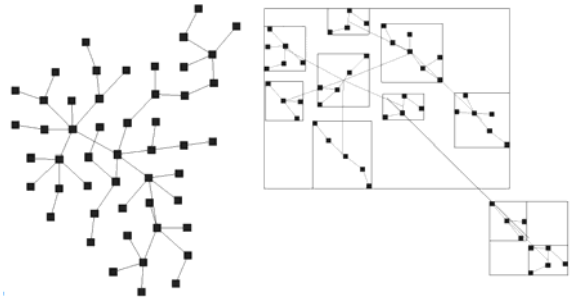


図 14 ノード配置結果の比較(3)．同一グラフに対して右は階層化しノード配置を行った．

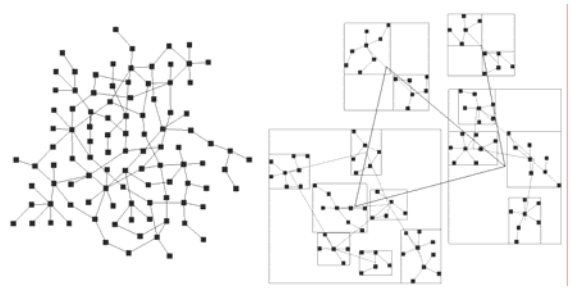


図 15 ノード配置結果の比較(4)．同一グラフに対して右は階層化しノード配置を行った．

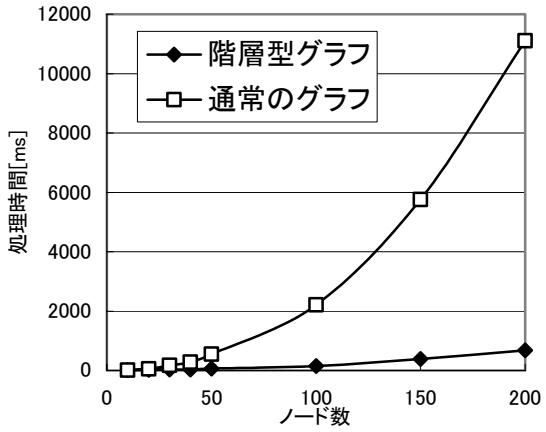



図 16 階層型と通常のグラフデータのノード配置処理時間の比較.

## 6. ウェブサイト視覚化への応用例

階層型グラフデータの可視化例として，ここではウェブサイト可視化を例にあげる．ウェブサイトは HTML 言語で記述されたウェブページから構成され，そのウェブページは，http://で始まるホスト名と，そのあとに続くディレクトリ名とファイル名で識別される．また，ウェブページはHTMLのハイパーリンクによって他のウェブページへ接続される．ここではウェブサイトの構造を，ウェブページをノード，ハイパーリンクをアークとし，ホスト名とディレクトリ階層でグループ化した階層型グラフデータとして可視化を行う．

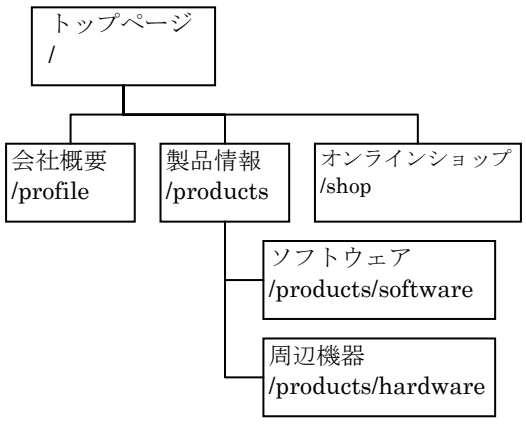



図 17 ウェブサイトのディレクトリ階層と構成の例.

一般的なウェブサイトの構築方法として，図 17 のように，共通する内容のページを同一ディレクトリにまとめて公開する方法がよく使われる．ディレクトリ階層を用いて階層化されたグラフデータを可視化することにより，共通の内容のページをまとめて表示でき，ウェブサイトの構成を理解しやすくなる．

階層型のグラフ構造を用いることで，階層構造を表現するとともに，木構造を用いたウェブサイト可視化手法[11][13][14]では表現しきれないウェブページ間のリンク関係をも表現できる利点がある．また，三次元空間にウェブページを配置するウェブサイト可視化手法[15][16]とは異なり，視点の回転操作等を必要とせずに一画面にウェブサイトの全体像を可視化できる利点がある．また階層構造を用いない，グラフデータによる可視化の場合，ウェブページ同士のリンクの数が多く，とても煩雑な可視化結果になってしまうことがあるが，階層化しグループ化することによって，グループ間のリンクで，ページ間のリンクを代表する手法を用いることで，リンクの混雑具合を調整することができる利点がある．特に，ディレクト構造を用いて整理されたウェブサイトであれば，他のディレクトリへのリンクよりも同一ディレクトリ内でのリンク構造の方が重要であることが多く，階層化グラフデータの可視化によって，同一ディレクトリ内のページのリンク構造を見やすく提供することができる．

ウェブサイトの階層型グラフを構築するには，トップページからリンクをたどることによってノードを収集していく．このとき，ウェブサイト外部のページ（ホスト名の違うページやトップページの属するディレクトリ階層の下に属さないもの）は同じグループにまとめておき，それ以上先のリンクはたどらない．また，上の階層へのリンクや既にたどったページへのリンクは，それ以上先をたどらない．ここでは，無向グラフを用いるため，元のページへ戻るようなリンクは考慮しない．ウェブサイト内のページはディレクトリ階層ごとにグループ化し，階層型グラフを構築する．

図 18 はウェブサイトを階層型グラフ化しノー

ド配置を行い，ノード部分にそのページのサムネイルを表示した可視化例である．このような可視化の結果の画像を用いて，ページの繋がりを理解しながらウェブサイト内をナビゲートすることが可能である．

図 19，図 20 はウェブサイトの情報を可視化した例である．図 19 ではウェブページの更新情報をノードの色で示した．赤が最近更新されたページで，青が最後に更新されてからの月日の長く経過したページであることを示している．また図 20 は図 19 に加えて，ページの一日のアクセス数をノードの大きさで示した結果である．このように，アクセス数や更新頻度等の情報を示すことで，ウェブサイトの管理者向けの情報を，ウェブサイトの階層構造やリンク構造と共に可視化することができる．このような可視化結果を用いることで，例えば，人気のあるページとその関連するページを見つけたり，更新が頻繁に行われているにも関らずアクセスが少ないページを見つけ，その原因はそのページにたどり着くまでの階層の深さが問題であるといったことを発見するために使用したり，といった使用法が考えられる．

なお，図 18，図 19，図 20 に示すウェブサイトの可視化例では，4.2 節で述べたように，すべてのページ同士のリンクを正確に表示することは避け，下位の階層のページ同士のリンクは，親ノード同士のリンクで代表してページ間の関係を表すようにすることで，リンク表示による混雑の度合いを調整するようにしている．しかしながら，この手法ではあるページが，どのページから参照されているという正確な情報を知ることができない．例えば，あるページを選択して，そのページに繋がるリンクのみを正確に表示するといった可視化の手法を取り入れていくべきであると考えている．また，管理者向けの情報可視化の例に挙げた，頻繁に更新されているにもかかわらずアクセス数の少ないページを発見する際，例えばトップページからそのページへ至るリンクによる経路のみを表示するといった可視化手法も実用的であると考えられる．今後は，このような適応的に可視化を行う手法についても検討していきたい．

ウェブページの可視化の例として，デジタルデータを用意した．1 つ目は，実際のウェブページの可視化結果であり，サムネイルをクリックすると実際のページへ行くことができる．2 つ目は，図 19, 図 20 に対応する実際の可視化結果である．

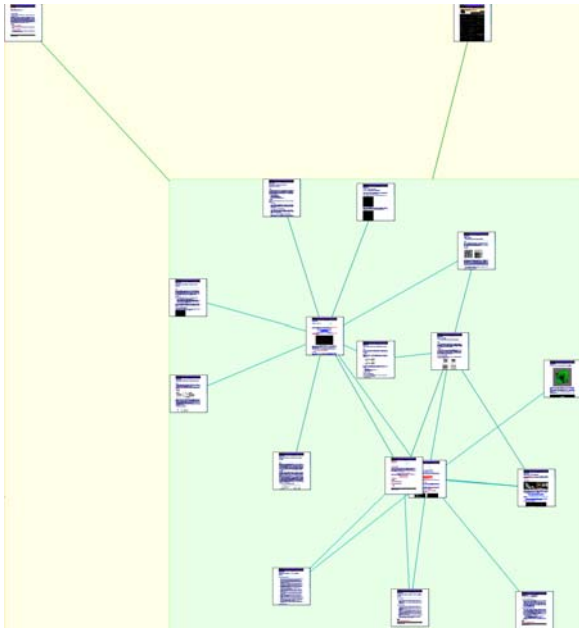


図 18 ウェブサイトの可視化の例．

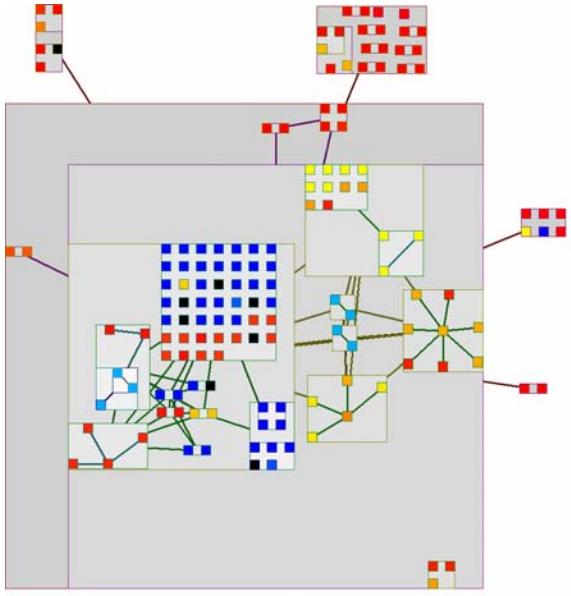


図 19 ウェブページの更新状況の可視化例．

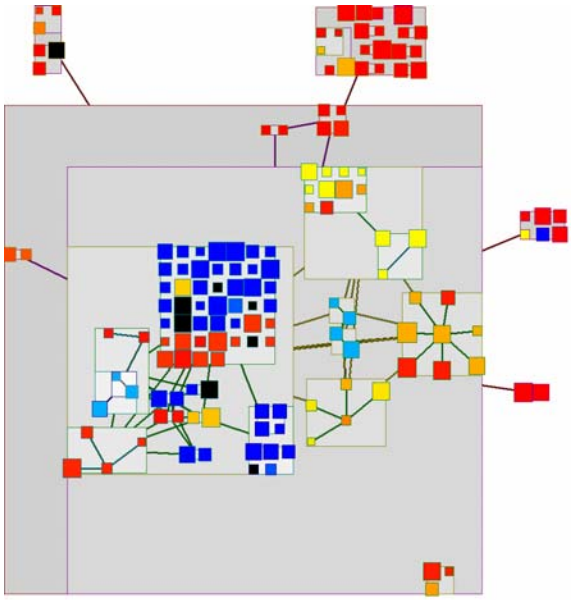


図 20 ウェブサイトの情報の可視化例.

## 7. むすび

本論文では，力学モデルを用いたグラフデータ視覚化手法に対して，ノードをインクリメンタルに 1 個ずつ画面配置する，という考え方により配置結果および処理時間を改善する手法を提案した．また，この手法を階層型グラフデータの視覚化に応用する手法を提案した．また，これらの手法の従来手法に比べる優位性を実験結果で示すとともに，ウェブサイトの視覚化に適用した例を示した．

今後の課題として，ウェブサイト以外の階層型データに対して提案手法を適用し，効果的な視覚化を実証することがあげられる．

また筆者らは，グラフデータ視覚化に関する関連手法として，

- アークの曲線化により，アークとノードの重なりを避ける手法[19]
- 階層型グラフデータを効果的に透視投影する手法[20]

も提案しているが，これらの手法は本論文ではウェブサイトの視覚化に適用していない．これらの手法がどのように効果的に実用化できるか，についても実証したい．

また，階層的なグラフ構造を可視化する上で，ユーザーの操作によっていかに効果的に可視化を行うかという点について，例えば，効果的なアークの表示手法や，ユーザーの注目点の効果的な表示手法，等についても検討していきたい．

## 参考文献

[1] Battista G. D., Eades P., Tamassia R., Tollis I. G., Graph Drawing – Algorithms for the Visualization of Graphs, Prentice Hall, ISBN0-13-301615-3, 1999.

[2] Herman I., Melancon G., Marshall M. S., Graph Visualization and Navigation in Information Visualization: A Survey, IEEE Transactions on Visualization and Computer Graphics, Vol. 6, No. 1, pp. 24-43, 2000.

[3] Eades P., A Heuristic for Graph Drawing, Congressus Numerantium, Vol. 42, pp. 149-160, 1984.

[4] Quigley A., Eades P., FADE: Graph Drawing, Clustering and Visual Abstraction, Symp. Graph Drawing 2000, 2000.

[5] Bertault F., A Force-Directed Algorithm that Preserves Edge Crossing Properties, Graph Drawing ’99, pp. 351-358, 1999.

[6] Gansner E., et al., Improved Force-Directed Layouts, Graph Drawing '98, LNCS 1547, pp. 364-373, 1998.

[7] Huang M. L., Eades P., Wang J., On-line Animated Visualization of Huge Graphs Using a Modified Spring Algorithm, Journal of Visual Languages and Computing, Vol. 9, pp. 623-645, 1998.

[8] North S., Incremental Layout in DynaDAG, Graph Drawing ’95, pp. 409-418, 1995.

[9] P. Eades, et al., Multilevel Visualization of Clustered Graphs, Graph Drawing ’96, pp. 101-112, 1996.

[10] D. Schaffer, et al., Navigating Hierarchically Clustered Networks through Fisheye and Full-Zoom Methods, ACM Trans. Computer-Human Interaction, Vol. 3, No. 2, pp. 162-188, 1996.

[11] Inxight Star Tree (TM) SDKs, http://www.inxight.com/products/sp/ht/sdk/index.html

[12] Lamping J., Rao R., The Hyperbolic Browser: A Focus+context Technique for Visualizing Large Hierarchies, Journal of Visual Languages and Computing, Vol. 7, No. 1,

pp. 33-55, 1996.
[13] Durand D., et al., MAPA: A System for Inducing and Visualizing Hierarchy in Web sites, 9th ACM Conference on Hypertext and Hypermedia, pp. 66-76, 1998.
[14] 山口, 伊藤, 池端, 梶永, 階層型データ視覚化手法「データ宝石箱」とウェブサイトの視覚化, 画像電子学会論文誌 Visul Computing 特集号, Vol. 32, No. 4, pp. 407-417, 2003.
[15] Hendley R. J. , Drew N. S., Wood A., Beale R., Narcissus: Visualizing Information, IEEE Information Visualization '95, pp. 90-96, 1995.
[16] 塩澤, 西山, 松下, 「納豆ビュー」の対話的な情報視覚化における位置付け, 情報処理学会論文誌, Vol. 38, No. 11, pp. 2331-2342, 1997.
[17] 嶋田, 物理モデルによる自動メッシュ分割, シミュレーション, 12, 1, 11-20, 1993.
[18] LEDA, http://www.algorithmic-solutions.com/enleda.htm
[19] 伊藤，井上，土井，梶永，池端，力学モデルを用いたグラフデータの視覚化手法の改良，情報処理学会グラフィクス&CAD 研究会，2001-CG-103, pp. 7-12, 2001.
[20] 土井，伊藤，梶永，池端，階層型グラフデータのための可視化手法，グラフィクスと CAD シンポジウム 2001 予稿集，pp.47-50,2001.