<原稿種別>

論文

<和文題名>

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

＜英語タイトル＞

Web Site Visualization by Hierarchical Data Visualization Technique “Data Jewelry Box”

＜あらまし＞

計算機のファイルシステム、大会社の人事組織、ウェブサイト。身のまわりには、階層構造で整理されたデータは非常に多く存在する。われわれはこのような階層型データの新しい視覚化手法「データ宝石箱」を提案している。この手法はデータ全体を一画面に展開して表示するので、階層型データの全貌を一目で眺観することができる。

本論文では、「データ宝石箱」の概要を示し、続いて「データ宝石箱」を用いたウェブサイトの視覚化手法を提案する。本手法では、ウェブサーバーに蓄積されるアクセスログファイルを入力すると、ウェブページの構成図である「サイトマップ」と、アクセス統計を表現する棒グラフを生成する。さらに本手法では、この２つを連携させたユーザーインタフェースを実装している。ここでは、このユーザーインタフェースによって興味深いアクセス傾向が発見できた事例も示す。

<Summary >

There are many hierarchical data in daily lives, such as file systems, human resources of companies, and web sites. We have proposed new visualization method for hierarchical data, “Data Jewelry Box” algorithm, which displays overviews of huge hierarchical data in one window.

This paper presents the overview of “Data Jewelry Box” algorithm, and proposes a web site visualization tool using the algorithm. Our tool constructs directory-based hierarchical data of a web site, and represents its sitemap by “Data Jewelry Box” algorithm. The tool represents web pages as icons, and directories as nested rectangles. Importing web access log file, our tool automatically builds access statistics bar charts in addition to the web sitemap. Our tool provides a user interface which displays number of accesses onto a sitemap when user clicks a part of bar chats. This paper shows interesting access trends discovered by using our tool.

<和文キーワード>

情報可視化、ユーザーインタフェース、ウェブサイト、階層型データ、ドロネー三角メッシュ

<英文キーワード>

Information visualization, user interface, web site, hierarchical data, Delaunay triangular mesh

1. はじめに

本論文では、大規模階層型データの新しい視覚化手法「データ宝石箱」を提案する[1]。「データ宝石箱」では、階層型データを構成する下位階層データをすべて画面空間に配置することで、データの全貌を一画面に表示する。

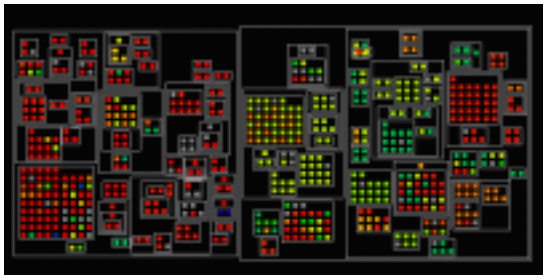

図1 「データ宝石箱」による階層型データの表現例。
Fig.1 Example of layout hierarchical data using "Data Jewelry Box"

図1に、本手法によるデータ表示例を示す。本手法では、葉ノードを色のついたアイコンで、枝ノードを長方形の枠で表現する。また、長方形を入れ子構造にすることで、階層の深さを表現する。宝石店のショーケースが、宝石やアクセサリーを種類ごとに分類して、商品を一望できるように陳列しているのと同じように、本手法は階層で分類されたデータを画面上で一望できるような視覚化を実現している。なお本手法の名前「データ宝石箱」は、あくまでも視覚化の方針を示した名前に過ぎない。また本手法は、宝石箱そのもののリアルな CG 表現を目的にしているわけではない。

「データ宝石箱」では、以下の要求を満たすように長方形を高速配置するアルゴリズムを用いている。

[要求1] 階層型データを構成するすべての葉ノードおよび枝ノードを、重なりなく、しかもできるだけ小さいディスプレイ領域に表示する。

[要求2] 階層型データの葉ノードおよび枝ノードを構成する長方形を、できるだけ好ましい形状で表現する。最上位階層の枝ノードを表す長方形は、ディスプレイやウィンドウの縦横比に近い形状で表現する。それ以外の葉ノードおよび枝ノードは、正方形に近い形状で表現する。

また本論文では、「データ宝石箱」を用いたウェブサイトのアクセス傾向の視覚化手法を提案する。本手法は主に、エンドユーザーよりも、ウェブサイトを運営するサイト管理者や設計者が利用することを想定している。ウェブサイトの管理や設計において、ウェブサーバーに蓄積されたアクセスログの解析は重要である。管理者は、サイト内で発生している通信状況を把握するために、設計者は、サイト閲覧者が関心のある情報に確実にアクセスできるようなサイトを作るために、それぞれアクセスログを解析している。このように、ウェブサイトのアクセスログ解析には多種な目的があるにも関わらず、アクセスログ解析の既存ツールは棒グラフ、円グラフ、アクセス数ランキング表、といった簡単な表現だけでアクセス統計を表示するものが多い。このような抽象化された表示は、ウェブサイト全体の概略的なアクセス傾向を表現することには向いているが、その中に潜む局所的な興味深いアクセス現象を発見するには非常に手間がかかることが多い。

本論文が提案する手法では、数千、数万、といった大規模なウェブサイトのアクセス傾向の全貌を一画面に表示するために、「データ宝石箱」を適用する。本論文では、ウェブページをディレクトリ階層にしたがって階層型データに格納し、ウェブページをアイコンで、ディレクトリ階層を長方形の枠で表現する。このようにして表現されたウェブページの集合を、本論文ではサイトマップと呼ぶ。本手法では、サイトマップ上のアイコンに高さを与えることでアクセス数を表して、ウェブサイトを構成するすべてのウェブページのアクセス数を1画面で表示する。

本手法ではサイトマップだけでなく、ウェブサイト全体のアクセス統計の棒グラフも表示し、これを連携操作するユーザーインタフェースを提供する。この機能により、ウェブサイト全体のアクセス統計を表示し、その統計に見られる特徴的なアクセス分布をサイトマップ上で表現することができる。この2つの表示結果から、ユーザーは、サイトマップから特徴的なアクセスをもつウェブページを探すことができる。さらにこのユーザーインタフェースでは、個々のウェブページのアクセス傾向も視覚化することができる。本論文では、この連携操作によってウェブサイトの興味深いアクセス傾向を発見した事例を紹介する。

本論文の構成は以下の通りである。まず2章にて、階層型データの視覚化手法およびウェブサイトの視覚化手法に関する関連研究を紹介する。続いて3章にて「データ宝石箱」の概要を紹介し、4章にてそのキーアルゴリズムとなる長方形高速配置アルゴリズムを紹介する。続いて5章にて本手法によるウェブサイトの視覚化事例を紹介し、6章で本論文のまとめを述べる。

2. 関連手法

2.1 階層型データの視覚化手法

以下に、「データ宝石箱」の関連研究となる階層型データの視覚化手法をいくつか紹介する。

2.1.1 木構造表示型の視覚化手法

Windows Explorer などの普及ソフトに代表されるように、階層型データの視覚化手法で最もポピュラーな手法は、木構造を表示するものである。大規模階層型データの対話的な探索に向いた手法として、Hyperbolic Tree [2], Cone Tree [3], Fractal Views [4] などの各手法が提案されている。近年では Cone Tree を DAG に拡張した手法 [5] も提案されている。これらの手法は、最初からデータ全体を表示するよりも、まずデータの上位階層を表示し、続いてユーザーの対話的操作によって注視部を選択的にズームアップする、というような用途を想定していることが多い。そのため本論文が示す[要求1]とは目的が異なることが多い。

2.1.2 空間分割型の視覚化手法

帯グラフを入れ子状にして、階層型データを構成するシェアを表示する TreeMaps [6] が有名である。この手法では、下位階層が非常に細長くなって視覚的に認識できなくなるケースが多いこと、また下位階層データをアイコンで表現することが難しいことなどが問題となっていた。最近ではこれらの問題点を解決する改善手法 [7-10]が発表されている。しかしいずれの手法も、[要求2]を満たすことは難しい。

2.1.3 3次元の入れ子構造を用いた表現

半透明な直方体を入れ子状に配置することで 3 次元的に階

層構造を表現する Information Cube [11] が知られている。半透明表示のできるグラフィックス環境が必要であること、ユーザー側に 3 次元 CG の操作スキルを要すること、階層構造が深いときに透明度の調節が難しいなどの制限がある。また、データ配置の計算時間の測定結果などが論文中に見当たらないので、対話性に関して不明な点が残る。

2.2 ウェブサイトの視覚化手法

以下に、5 章で提案するウェブサイトの視覚化手法の関連研究をいくつか紹介する。

2.2.1 ウェブサイトの階層構造の視覚化手法

ウェブサイトのサイトマップを表現する手法の多くは、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。すでに実用化されているウェブサイト視覚化手法の多くは、木構造表示型の視覚化手法をサイトマップに適用している。

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 [12] がある。この手法は、Hyperbolic Tree [2] を用いて、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上位階層から下位階層へ段階的に表示する。

2.2.2 ウェブサイトのリンク構造の視覚化

ウェブページ間のリンク関係を視覚化する手法は過去にも多く報告されているが、実用化に到達している例は少ない。ウェブページ間のリンク構造は一種のグラフ構造であると考えられるので、グラフ構造の視覚化手法を適用することでウェブページ間のリンク関係を視覚化することができる。一例として、力学モデルを用いたグラフ構造の自動画面配置手法を、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 [14]。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報告されている。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 [15]。

2.2.3 ウェブアクセス履歴の視覚化手法

ウェブのアクセス情報を視覚化する手法はすでに多く報告されているが、ウェブサイト全体のアクセス傾向を視覚化する手法の報告は非常に少なく、むしろ個人のアクセス履歴を表現する手法が多く報告されている。代表例として、個人がアクセスしたウェブページ群をアイコン化して、それをアクセス順に線で結んでグラフ表示する手法[16,17,18] が多く報告されている。

3.「データ宝石箱」による階層型データの視覚化

1 章で述べた通り、筆者らが提案している階層型データ視覚化手法「データ宝石箱」は、葉ノードをアイコンで表現し、枝ノードを入れ子状の長方形の枠で表現している。

ここで「データ宝石箱」が対象としている一般的な階層型データは、座標情報を持たない。よって図1のようなデータ視覚化を実現するためには、データを構成するノードに画面空間上の座標値を与え、データを画面空間に配置するアルゴリズムが必要である。

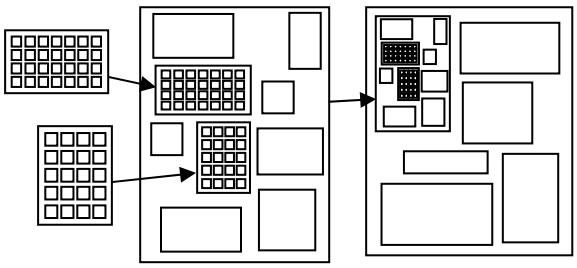

図 2　階層型データの画面配置順。まず最下位階層の葉ノードを配置し、続いて下位階層から上位階層に向かって配置処理を反復する。
Fig.2 Order of placing hierarchical data.

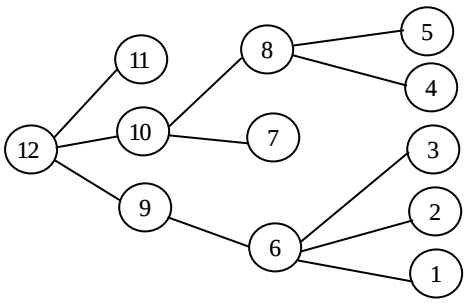

図 3　データの配置順。この木表現のノード 1 個が、表示画面上の長方形の枠 1 個に相当する。
Fig.3 Definition of order of placing hierarchical data.

図 2 に、「データ宝石箱」による階層型データの画面配置アルゴリズムを示す。本手法では、まず最下位階層に属する葉ノードに対応するアイコン（図 2 の場合は正方形）を隙間無く配置する。続いて、この上位階層に属する枝ノードを表現するために、アイコンを包括する長方形を生成する。さらに、上位階層の枝ノードを表現する長方形群を隙間無く配置し、同様にこれを包括する長方形を生成する。以上の処理を、最下位階層から最上位階層に向けて反復することで、データ全体の配置を決定する。

本手法における階層の配置順を、木構造で図示したものが図3である。配置順を決定するために本手法では、最上位階層から幅優先アルゴリズムで下位階層を探索し、探索順を記録する。すべての階層を探索し終えたら、探索順と逆の順序で長方形の配置処理を行う。

本手法では、1 個の枝ノードを表現する長方形の枠の内部に、複数の葉ノード(アイコン)や枝ノード(長方形の枠)が配置される。これらのノードを長方形の集合であるとすると、本手法を実現するためには、1階層を構成する長方形の集合を、以下の条件を満たすように画面空間に配置する技術が必要である。

[条件 1] 長方形どうしが重なってしまうと、データの視覚的理解を妨げるので、隣接長方形と重ならないように長方形を配置する。

[条件 2a] 配置結果の占有面積が大きくなると、それだけ大きなディスプレイ領域を要するので、占有面積を拡大しないように長方形を配置する。
[条件 2b] やむを得ず配置結果の占有面積を拡大するときは、できるだけ占有面積の拡大量が小さい位置に長方形を配置する。また、できるだけ好ましい縦横比の占有領域を構成するように長方形を配置する。

このような条件を満たすように形状データを配置する問題は、「占有面積の最小化問題」として、VLSI 回路の基板配置、板金や服飾型紙への部品配置、などの用途で知られている [19]。これらの用途では、遺伝子アルゴリズムなどの最適化手法を用いて部品の配置を実現している例が多い。しかし最適化手法には、数分～数時間の計算時間を要する事例が多く、対話的操作を要する視覚化の分野には向かない。「データ宝石箱」では、占有面積が最小でなくてもいいから、ある程度良好な結果を短時間に算出する配置手法を用いる。次章ではその配置手法の一例として、長方形群を隙間なく配置する高速な新しいアルゴリズムを示す。

なお、[条件 2b]における「好ましい縦横比」とは、最上位階層においてはディスプレイやウィンドウの縦横比、それ以外の階層においては縦横=1:1 であるとする。

4. 「データ宝石箱」に用いている長方形群の配置アルゴリズム

本章で提案する長方形配置アルゴリズムでは、与えられた長方形を 1 個ずつ順番に配置するインクリメンタルな処理手順を用いる。本手法では、すでに数個の長方形が配置されているときに、長方形の配置候補となるサンプリング位置を求め、その中から前章で述べた[条件 1][条件 2a または 2b]を満たす位置に長方形を配置する。適切にかつ高速にサンプリング位置を算出するアルゴリズムはいくつか考えられるが、本論文はその一例として、すでに配置されている長方形の中心点を連結する三角メッシュを用いるアルゴリズムを提案する。このアルゴリズムでは、三角メッシュを参照しながら、隙間がある可能性が高いと推測される部位から順にサンプリング位置を算出する。そして、各々のサンプリング位置が[条件 1][条件 2a または 2b]を満たすか判定し、長方形の配置を決定する。

4.1 概要

n個の長方形群を与えられたとき、これをできるだけ小さい占有面積で xy 平面上に配置することを考える。ただし、長方形の辺はすべて x 軸または y 軸に平行であるとする。このとき本手法では、以下の手順で長方形を配置する。また、図 4 に長方形群の配置例を示す。

1. 長方形を面積順にソートする。
2. 大きい長方形から順に、

(2a) [条件 1][条件 2a]の両方を満たすサンプリング位置が見つかったら、その位置に長方形を配置する。
(2b) [条件 1][条件 2a]の両方を満たすサンプリング位置が見つからなかったら、[条件 1][条件 2b]を満たす位置に長方形を配置する。

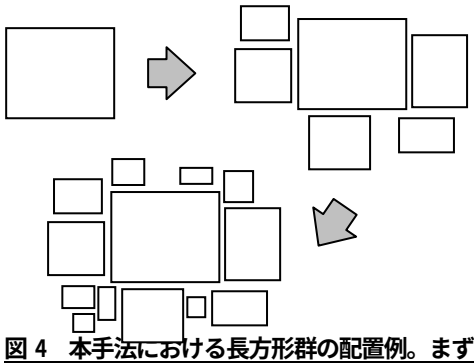

図 4 本手法における長方形群の配置例。まず面積が最大である長方形を配置し、続いて隙間を探しながら、互いに隣接するように大きい順に長方形を配置する。
Fig.4 Example of the placement of rectangles using our algorithm.

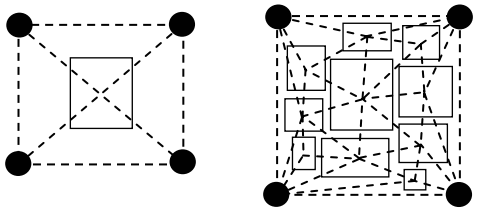

図 5 (左) 三角メッシュ(点線)の初期状態。面積が最大である長方形だけが配置された状態である。黒丸で示す 4 点は占有領域の頂点を表す。(右) すでに配置された長方形の中心点群、占有領域の 4 頂点の中心点、を連結する三角メッシュ。
Fig.5 (left) Initial configuration of triangular mesh. (right) Triangular mesh connecting centers of rectangles and four corners.

ここで本手法では、すでに配置された長方形の中心点を連結する三角メッシュを生成する。以下、三角メッシュの生成処理について説明する。本手法ではまず、面積が最大である長方形を画面空間の原点に配置する。続いて、その長方形よりやや大きく、長方形を完全に包括する長方形領域（以下、占有領域と呼ぶ）を生成し、その 4 頂点と長方形の中心点を連結する三角メッシュを生成する（図 5(左)参照)。以後、長方形の配置にあわせて、三角メッシュを更新する。大きい順に k 番目までの長方形を配置したときに、三角メッシュは k 個の長方形の中心点、およびそれを囲む占有領域の 4 頂点、の合計 $(k+4)$ 個の頂点によって構築される(図 5(右)参照)。

4.2 長方形を配置する隙間の検出方法

本手法では、長方形を適切に配置できる位置を高速検出するために、以下のような方針に基づいた手順により、三角メッシュを参照しながら長方形の配置候補となるサンプリング位置を決定する。

[方針 1] 長方形を干渉することなく配置できる隙間がある、

と推定される部位を検出し、その部位から優先的に長方形の配置を試みる。

[方針2] まず占有領域の内側の部位から長方形を配置することを試みる。なぜなら内側に長方形を配置したほうが、その長方形が占有領域からはみ出して占有領域を拡大する、という可能性が低いからである。

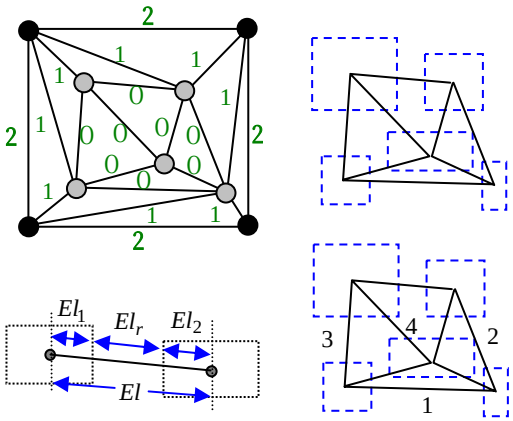

図 6　(左上)三角メッシュを構成する辺を、占有領域の 4 頂点との隣接数 n_Bで分類した結果。黒丸は占有領域の頂点を、灰色丸は長方形の中心を表す。(右上) n_B =0である三角メッシュ辺と、それに連結された長方形の集合。(左下)隙間のある可能性を判断するための指標。(右下) El_r の大きさで三角メッシュ辺に順位をつけた結果。

Fig.6 (left upper) Edges categorized by n_B.(right upper) Edges of triangles with n_B=0. (left lower) Measurement of gaps.(right lower) Priority of edges with n_B = 0.

本手法ではまず、各々の三角メッシュ辺について、占有領域の4頂点と重複する端点の数 $n_B\{n_B = 0,1,2\}$ を特定する。続いて、三角メッシュ辺を n_B 値で 0～2 の 3 種類に分類する。図 6(左上)に n_B 値の例を示す。この例からもわかる通り、 n_B 値が小さい三角形要素ほど占有領域の内側にあることがわかる。

続いて、$n_B = 0$ である三角メッシュ辺(図 6(右上)参照)について、辺の長さを El ,辺の両端で長方形と重なる領域の長さを El_1 および El_2 としたときに、$El_r = El - (El_1 + El_2)$ の値を算出する（図 6(左下)参照)。さらに、El_r の大きさでメッシュ辺に順位をつける(図 6(右下)参照)。この順位が高いほど、その三角メッシュ辺の上に隙間がある可能性が高いと仮定する。そして、この順位にしたがって三角メッシュ辺の上にサンプリング位置を設定し、長方形の配置を試みる。[条件 1][条件 2a]両方を満たすサンプリング位置が見つかったら、長方形の位置をそこに決定する。2 条件の両方を満たさない場合は、他の三角メッシュ辺に対して同様な判定処理を行う。

[条件 1]は満たすものの[条件 2a]は満たさない、というサンプリング位置が見つかったときには、その位置が[条件 2b]を満たすかどうか判定する。そしてその位置が、いままでで最も[条件 2b]を満たすようであれば、その位置を記録しておく。本論文では一例として、長方形の配置後の占有面積を A_a , 長方形の配置後の占有領域の縦横比と好ましい縦横比の比を A_r としたときに、$aA_a + rA_r$ の値が最小である位置を、[条件 2b]を満たす位置とみなす。ここで a および r はユーザーが指定する定数値とする。

もし $n_B = 0$ である三角メッシュ辺の上に、[条件 1][条件 2a]の両方を満たすサンプリング位置を発見できなかった場合には、 $n_B = 1$, $n_B = 2$ の順に三角メッシュ辺を探索し、同様な判定処理を行う。ただし、 $n_B > 0$ の場合には、三角メッシュ辺の端点は占有領域の 4 頂点のいずれかに重なる。このような場合には El_1 または El_2 の値を 0 として El_r を算出する。

もし[条件 1][条件 2a]の両方を満たすサンプリング位置をまったく発見できない場合には、[条件 1][条件 2b]を満たす位置に長方形を配置する。このとき、配置した長方形は占有領域からはみ出しているので、占有領域の 4 頂点のうちいくつかを移動して、すべての長方形を内包する占有領域を再構成する(図 7 参照)。

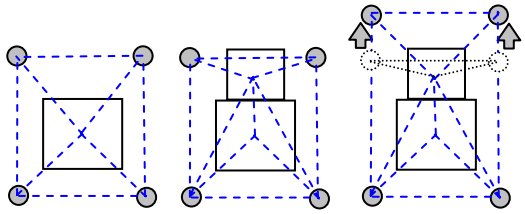

図 7　(左)長方形の配置前の状態。(中)新しく配置した長方形が占有領域をはみ出した状態。(右)新しく配置した長方形を包括する占有領域を形成するために、4 頂点のうち 2 頂点を矢印の向きに従って移動した状態。

Fig.7 (left) Before placing new rectangle.(middle) Side drop of rectangle from the layout region. (right) Enlargement of the layout region.

4.3 長方形の配置にともなう三角メッシュの局所修正

本手法では、長方形を 1 個配置するごとに、その長方形の中心点を三角メッシュに追加して、三角メッシュを更新する。このとき、極端に長い三角メッシュ辺を多く含むような三角形要素が生成されると、サンプリング位置の検出において不利である。そこで本手法では、長い三角メッシュ辺の生成を低減するために、Delaunay 三角メッシュを適用する。

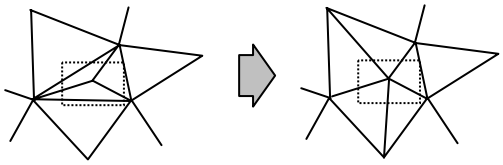

図 8 (左)長方形の中心点を三角形要素の 3 頂点に連結した状態。(右) 三角形要素の辺を局所修正して、三角メッシュを更新した状態。

Fig.8 (left) Triangular mesh whose new vertex is connected to three vertices of a triangle. (right) Locally modified triangular mesh under Delaunay condition.

この処理ではまず、長方形の中心点を内包する三角形要素の 3 頂点と、いま配置した長方形の中心点を連結する（図 8(左)参照)。続いて、三角メッシュが Delaunay の条件を満たし続けるように、三角形要素の辺を局所修正する（図 8(右)

参照)。なお三角メッシュにおける Delaunay の条件とは、「任意の三角形要素に対して生成された外接円の内部に、その三角形要素の 3 頂点以外の頂点を包括しない」という条件を、全ての三角形要素が満たすことである。

以上の処理は、Incremental Delaunay 三角メッシュ生成法 [20] という手法に類似している。

4.4 アルゴリズム

本章のまとめとして、Delaunay 三角メッシュを用いた長方形群の配置アルゴリズムの擬似コードを図 12 に示す。

5. 「データ宝石箱」を用いたウェブサイトの視覚化

5.1 概要

図 9 に、本論文で提案するウェブサイト視覚化手法の処理手順を示す。本手法は、ウェブサーバーに蓄積されるアクセスログファイルを入力データとする。このとき本手法は、ユーザーが指定した属性に基づいてアクセス数を集計し、その結果を「統計グラフ」という棒グラフで表示する。それと同時に、アクセスログから、アクセスされたページの URL を抽出し、階層型データを構築し、「データ宝石箱」によりサイトマップを自動生成して表示する。さらに本手法では、サイトマップと統計グラフとの間に以下の 2 種類の連携操作機能を提供する。

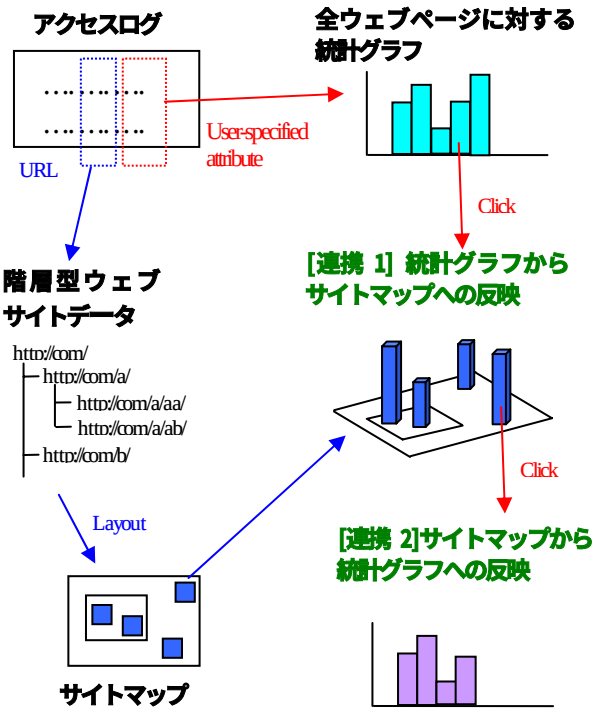

図 9　本手法によるウェブサイト視覚化の処理手順
Fig.9　Processing flow of our web site visualization tool.

[連携 1] 統計グラフからサイトマップへの反映

本手法は統計グラフをクリック可能な状態で表示するものとする。ユーザーが統計グラフの 1 箇所をクリックすると、本手法はクリックされた箇所に該当するアクセス数をウェブページごとに集計する。そして、個々のウェブページのアクセス数に応じて、アイコンに高さを与える。このようにして、アクセスログに記録されたアクセスの中から、特定の属性値をもつアクセスの分布を、サイトマップ上で視覚化することができる。

[連携 2] サイトマップから統計グラフへの反映

本手法はサイトマップをクリック可能な状態で表示するものとする。ユーザーがサイトマップ上で関心のあるウェブページのアイコンをクリックすると、そのウェブページへのアクセスを集計した統計グラフが表示される。これによりユーザーは、サイト全体のアクセス傾向だけでなく、関心のある特定のウェブページに対するアクセス傾向も知ることができる。

サイトマップと統計グラフの双方向の連携操作によって、多数のサイト閲覧者によるウェブサイトのアクセス傾向を、大局的にも局所的にも表現できる、という点が本手法の特徴であり、2.2.3 節で紹介した従来のウェブアクセス視覚化手法と異なっている。

5.2 アクセスの統計グラフ

本手法では、ウェブサイト全体のアクセス傾向を視覚化するために、ユーザーの指定した属性を基準にしてアクセスログからアクセス数を集計した棒グラフを作成する (図 10 参照)。本論文では、この棒グラフを統計グラフと呼ぶ。この統計グラフは、属性に基づいた分類を横軸に、アクセス数を縦軸にとる。

アクセスログは、ウェブページをはじめとするリソースにサイト閲覧者がアクセスしたときの、アクセス状況を 1 行ずつ記録したファイルである。1 行のアクセスログには、原則として以下の属性が記述されている。

1) サイト閲覧者の IP アドレス
2) 認証を行った際のサイト閲覧者名
3) アクセス日時
4) リクエスト内容
5) ステータス
6) 転送バイト数
7) アクセスされたページの URL
8) 7) のリンク元の URL

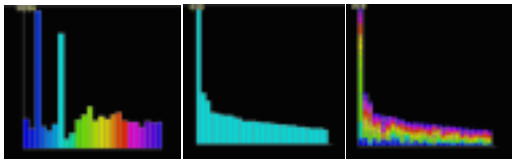

図 10 アクセスの統計グラフ。(左) 時間帯で分類。(中) アクセスされた URL で分類。(右) アクセスされた URL で分類した後、さらに時間帯で分類。
Fig.10 Access statistics bar charts. (left) Each bar denotes the number of accesses in one hour.(middle) Each bar denotes the number of accesses of one URL. (right) Each bar denotes the number of accesses of one URL, and the bars are divided into 24 parts by hours.

図 10(左)は、アクセスされた時刻を 1 時間単位で分類集計し、0 時台から 23 時台まで 24 本の棒グラフを時間順に並べて表示したものである。図 10 (中)は、アクセスされた URL ごとにアクセスを分類集計し、アクセス数の多い順に左から 30 ウェブページ

のアクセス数を表示したものである。

また、アクセスを 1 つの属性で分類した集計結果について、さらに別の属性で分類して集計することも可能である。図 10(右)は、図 10(中)と同じ URL 別の集計結果を表す棒グラフを、さらに URL ごとの集計結果を一日の時間帯で分類集計して 24 分割し、図 10(左)と同じ 24 色で表示したものである。この表示結果から、アクセス数の高いページがどの時間帯にアクセスが多いか、ということを知ることができる。

5.3 「データ宝石箱」によるサイトマップ表現

本論文で提案するウェブサイトの視覚化手法では、ディレクトリ階層を参照してウェブページ群の階層型データを構成し、それを「データ宝石箱」によって表現する。本手法では、サイトマップのディレクトリ構造を入れ子状の長方形の枠で表現し、その内部にウェブページを表すアイコンを配置することで、サイトマップの階層構造全体を画面空間に配置する。

ここで、「データ宝石箱」による画面配置結果を数値評価するために、5 種類の異なるウェブサイトから構築した階層型データを用いて、計算時間(秒)、アスペクト比、空領域比を測定した。ここでアスペクト比には、枝ノードを表現する各長方形領域について、y 方向の長さと x 方向の長さの比（1 未満のときはその逆数）を算出し、その平均値を用いた。空領域比には、各長方形領域について、その面積と、そこから内部に包括するアイコンと長方形領域の面積総計を差し引いた値との比を算出し、その平均値の逆数を用いた。これは、Bederson がアイコン群を画面空間に隙間なく配置させるアルゴリズム[7]を、定量的に検証する際に使用した項目である。本論文でも同様に実験した。この結果を表 1 に示す。

表 1 「データ宝石箱」による画面配置結果の数値評価

Table.1 Numerical evaluation

アイコン数	ディレクトリ数	計算時間(秒)	アスペクト比	空領域比
676	160	0.047	1.307	0.499
1193	264	0.413	1.179	0.569
1704	260	0.297	1.208	0.515
4319	625	0.625	1.251	0.511
4793	892	0.562	1.270	0.490

表 1 から以下のような結果がわかった。計算時間に関しては、数千ウェブページをもつウェブサイトの階層型データに対して、いずれも 1 秒未満で画面配置を実現していることがわかった。このことから「データ宝石箱」は、ウェブサイトのような静的な階層型データに限らず、秒単位で刻々と変化するような階層型データのリアルタイム表示や、ユーザーの対話的操作による階層型データの探索などの目的にも実用化できることがわかる。アスペクト比と空領域比に関しては、Bederson の実験結果[7]と比較するとアスペクト比は「データ宝石箱」の方がよい結果が得られたが、空領域比は Bederson が提案したアルゴリズムを用いた方がよい。この点においては、改良の余地があるといえる。

5.4 統計グラフとサイトマップの連携

筆者らの実装では、5.2 節で提案した統計グラフを右画面に、5.3 節で説明したサイトマップを左画面に表示し、これらの連携操作機能を提供している。

ここで、本手法を用いて、実在するウェブサイトの 1 週間のアクセスログを視覚化した実験例を示す。ウェブページに対するアクセスだけを視覚化の対象にするために、本実験ではまず前処理として、アクセスログから URL が画像ファイルやスタイルファイルなどを示している行を削除した。このアクセスログを日付で分類集計して 7 本の棒グラフを作成し、さらに各項目を 1 時間単位で分割することで棒グラフを 24 色に色分けした。以上の分類にしたがって、横軸が日付、縦軸がアクセス数を示す統計グラフを作成した(図 11(左上))。これを見ると、最終日のアクセス数が他の日に比べて極端に高いことがわかった。

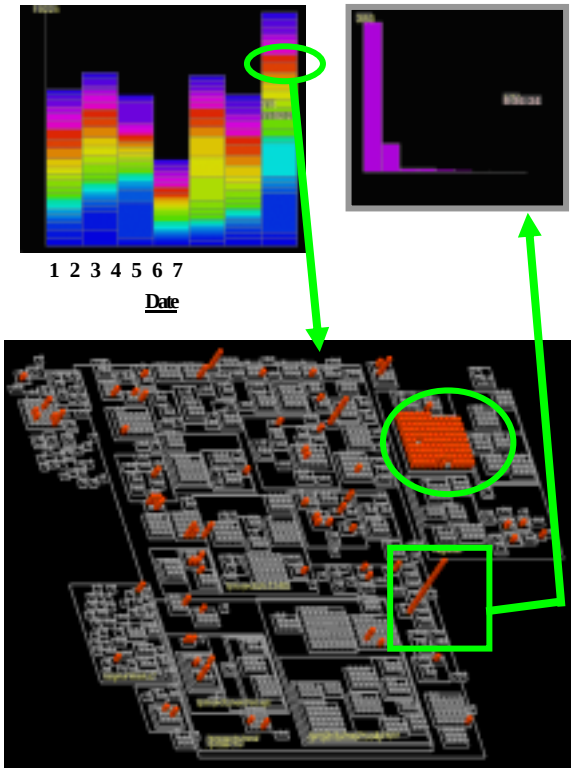

図 11 (左上)実在するウェブサイトの 1 週間のアクセスの統計グラフ。(下)[連携 1]の操作例。(左上)の統計グラフから、ある 1 時間をクリックして、その 1 時間のアクセス分布をサイトマップ上で表現したもの。(右上)[連携 2]の操作例(下)のサイトマップ中の四角で囲んだアイコンをクリックして、そのウェブページに対するアクセスの統計グラフを表示したもの。

Fig.11 (left upper) Access statistics bar chart grouped by day. (bottom) Example of [operation1]. (right upper) Example of [operation 2].

ここでまず[連携 1]を用いて、図 11(左上)の統計グラフ上で、最終日のある 1 時間をクリックし、サイトマップ上でその 1 時間のアクセス分布を表示した。図 11(下)にそのアクセス分布を示す。

このとき、図 11(下)中の右下部にある、四角で囲んだアイコンが表すページが、午前中に突出して多くのアクセス数をもつことがわかった。[連携 2]を用いて、そのページへのアクセスを対

象としてリンク元のURLで分類した統計グラフを表示した。すると、ある新聞会社のオンラインニュースのURLから訪れているサイト閲覧者が多いことがわかった（図11(右上)）。リンク元であるオンラインニュースにアクセスしてみると、そこにアクセス数の多かったページが取り上げられていたことがわかった。以上の結果から、午前中に新聞サイトを見て、そのリンクをたどってこのページに来た人が多かったことを推測した。

また、図 11(下)中の丸く囲んだ長方形が表すディレクトリ中のほとんどのページが、1時間以内にアクセスされていたことがわかった。続いて[連携2]を用いて、これらのページへのアクセスをサイト閲覧者の IP アドレスで分類して表示すると、すべてのページに同一 IP アドレスからのアクセスがあることがわかった。これらの結果から、あるディレクトリのファイルをすべて見ている熱心なサイト閲覧者が存在していたことがわかった。

このように、本ツールの[連携 1]を用いて、ウェブサイト全体のアクセス分布を視覚化することで、特徴的なアクセス傾向をもつウェブページを発見することができた。さらに、本ツールの[連携2]を用いて、そのアクセス傾向の背景を知ることができた。

6. むすび

本論文では、大規模階層型データの全体を一画面に配置する新しい視覚化手法「データ宝石箱」を提案した。また、「データ宝石箱」によって自動生成したサイトマップと統計グラフを連携操作することによって、アクセスの大局的な傾向から局所的な傾向までを一画面で視覚化できる、アクセスログの視覚化手法を提案した。

今後の課題として、以下のようなことを検討中である。

[ユーザーの意図を反映したデータ配置] 本論文で提案した「データ宝石箱」では、データの位置は何の意図も表現していない。現在筆者らは、ユーザーのデザインを反映するデータ配置を実現するように、また座標軸に意味をもたせたデータ配置を実現するように、「データ宝石箱」を拡張中である。この拡張手法を適用することで、例えば「新しいウェブページを左上に」というようなユーザーの意図にしたがったサイトマップの自動生成が実現できると考えられる。

[リンクの表示] 「データ宝石箱」は階層型データを対象としているが、データの葉ノードどうしがリンクを持つ「階層型グラフ」の表現 [21] に拡張することができれば、さらに応用例が広がると考えられる。ウェブサイトの視覚化においては、サイト閲覧者がサイトに訪問して別のサイトに移動するまでの、リンクをたどる動きの傾向を表示することで、より詳細なアクセス傾向を表現できると考えられる。

[高速表示を必要とするアプリケーションへの適用] 5.3 節で示した実行例からもわかる通り、「データ宝石箱」は大規模な階層型データの高速な画面配置を実現している。よって「データ宝石箱」は、ウェブサイトのような静的なデータだけでなく、秒単位で刻々と変化する階層型データのリアルタイム表示にも適用できると考えられる。一例として筆者らは、並列計算環境のプロセス分布のリアルタイム監視に「データ宝石箱」を適用する実験をしている。

参考文献

[1] 伊藤貴之, 梶永泰正, 池端裕子: “ データ宝石箱:大規模階層型データのグラフィックスショーケース ”, 情報処理学会グラフィックスとCAD研究報告, 2001-CG-104, 2001.

[2] Lamping J., Rao R.,: ”The Hyperbolic Browser: A Focus+context Technique for Visualizing Large Hierarchies,” *J. Visual Languages and Computing*, Vol. 7, No. 1, pp. 33-55, 1996.

[3] Carriere J., et al.: “ Research Report: Interacting with Huge Hierarchies Beyond Cone Trees,” *IEEE Information Visualization '95*, pp. 74-81, 1995.

[4] Koike H., Fractal Views: “A Fractal-Based Method for Controlling Information Display,” *ACM Trans. on Information Systems*, Vol. 13, No. 3, pp. 305-323, 1995.

[5] 山下由美, 藤代一成, 高橋成雄, 堀井秀之, 拡張 ConeTrees 技法による DAG情報の可視化, Visual Computing グラフィクスとCAD合同シンポジウム 2002, pp. 1-6, 2002.

[6] Johnson B., et al., “Tree-Maps: A Space Filling Approach to the Visualization of Hierarchical Information Space”, *IEEE Visualization '91*, pp. 275-282, 1991.

[7] Bederson B.: “PhotoMesa: a zoomable image browser using quantum treemaps and bubblemaps,” *UIST 2001*, pp. 71-80, 2001.

[8] Bruls D.M., et al.: “Squarified Treemaps,” *Data Visualization 2000 (joint Eurographics and IEEE TCVG Symposium on Visualization)*, pp. 33-42, 2000.

[9] Shneiderman B., et al..: “Ordered treemap layouts, “*IEEE Information Visualization Symposium 2001*, pp. 73-78, 2001.

[10]

[11] Rekimoto J. ,Green M..: “The Information Cube: Using Transparency in 3D Information Visualization,“ *Third Annual Workshop on Information Technologies & Systems*, pp. 125-132, 1993.

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

[13] Durand D. G., Kahn P.: “MAPA: A System for Inducing and Visualizing Hierarchy in Websites, “*Proceedings of ACM Hypertext '98*, pp. 66-78, 1998.

[14] Hendley R. J., Drew N. S., Wood A., Beale R., Narcissus: “Visualizing Information,” *Proceedings of the 1995 Information Visualization Symposium*, pp. 90-96, 1995.

[15] 塩澤秀和, 西山晴彦, 松下温: ”「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.

[16] Ayers E., Stasko J.: “Using Graphic History in Browsing the World Wide Web,” *4th International World Wide Web Conference*, 1995.

[17] Frecon E.: “Webpath – A Three Dimensional Web History,” *IEEE Information Visualization '98*, pp. 3-10, 1998.

[18] Bederson, B.B., Hollan, J.D., Stewart, J., Rogers, D., Vick, D., Ring, L.T., Grose, E., Forsythe, C.: “A Zooming Web Browser, “*Human Factors in Web Development*, pp. 255-266, 1998.

[19] Murata H., Fujiyoshi K., Shigetoshi N., Kajitani Y.: “VLSI Module Placement Based on Rectangle-Packing by the Sequence-Pair,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol. 15, No. 12, pp. 1518-1524, 1996.

[20] Sloan S.W.: “A Fast Algorithm for Constructing Delaunay Triangulation in the Plane,” *Advances in Engineering Software*, 9, pp. 34-55, 1987.

[21] 土井淳, 伊藤貴之, 梶永泰正, 池端裕子: “ 階層型グラフデータのための可視化手法 ”, Visual Computing / 情報処理学会グラフィクスとCAD合同シンポジウム, pp. 47-50, 2001.

```
長方形群の配置(){

 /*初期化(図5(左)参照)*/
  長方形群を面積でソートする;
  占有領域の初期状態をつくる;
  三角メッシュを初期化する;

 /*長方形ごとのループ(図4参照)*/
 面積の大きな長方形から順に{

   /*図6参照*/
   三角メッシュ辺を n_B = 0,1,2 の順に {
     未処理三角メッシュ辺のうち、
     El_r が最大であるものを選択する;
     選択した三角メッシュ辺上で
       サンプリング位置を決定する;
     隣接長方形と干渉するなら {
       戻って他の三角メッシュ辺を選択;
     }
   占有領域の拡大が必要なら{
     aA_a + rA_r 値が過去最小であれば
           このサンプリング位置を記録する;
     戻って他の三角メッシュ辺を選択;
   }

   /*図8参照*/
   長方形を配置し、メッシュを更新する;
   次の長方形へ;

 }/*end for each 三角メッシュ辺*/

 もし占有領域を拡大せずに長方形を配置
 できないなら{
   /*図7,8参照*/
   記録してあったサンプリング位置に
   長方形を配置し、メッシュを更新する;
   次の長方形へ;
 }

}/*end for each 長方形*/
}
```

図12　本論文4章で提案する長方形配置アルゴリズムの擬似コード。
Fig.12 procedure of placing rectangle algorithm.