平成 24 年度

筑波大学情報学群情報科学類

卒業研究論文

題目

大規模な多次元データの活用を促進するための可視化ツールの開発

主専攻　　情報システム主専攻

著者　　萬成 亮太

指導教員　三末和男、志築文太郎、高橋伸、田中二郎

要　旨

情報技術の発達に伴い、日々大量のデータが生成されている。これらのデータを分析すると、データに隠された有益な情報を得ることが出来る。一方で分析のために大規模なデータを可視化する場合、大規模データを想定せずに設計された可視化手法では、データの数が画面のピクセル数を超過してデータをすべて表示できなかったり、折れ線や点などが重なりあってグラフが見づらくなるなどの問題が起こる。本研究では、このような大規模データのうち、1 つのレコードが複数の値を持つ多次元データに的を絞り、数十億規模のデータを表現する手法を開発した。レコードが持つ変数のうちの一つをカテゴリとみなし、カテゴリ毎にまとめることで、表現すべき情報の削減を試みた。描画の着色方法を工夫することで、描画内容が重なりあう問題を解決する。ユースケースとして、開発したツールを用いて数十億規模のデータを分析できることを示す。

目 次

図 目 次

第1章　序論

1.1　大規模な多次元データ

情報技術の発達に伴い、大量のデータが日々生成されている。これらのデータは、ソーシャルメディアのアクティビティやインターネットの検索ログ、オンラインショップの購入履歴やGPS の位置情報など多岐に渡る。大量のデータを分析することで、データに隠された有益な情報を得ることが出来る。

このようなデータは、常に同じ形式で保存されていく。例えばソーシャルメディアなら、アクティビティ[1]が更新された時に更新時刻、ユーザー名、更新内容といった情報をレコードとして保存する。一件のレコードが複数の値を持てるように設けられた属性を次元という（図1.1）。

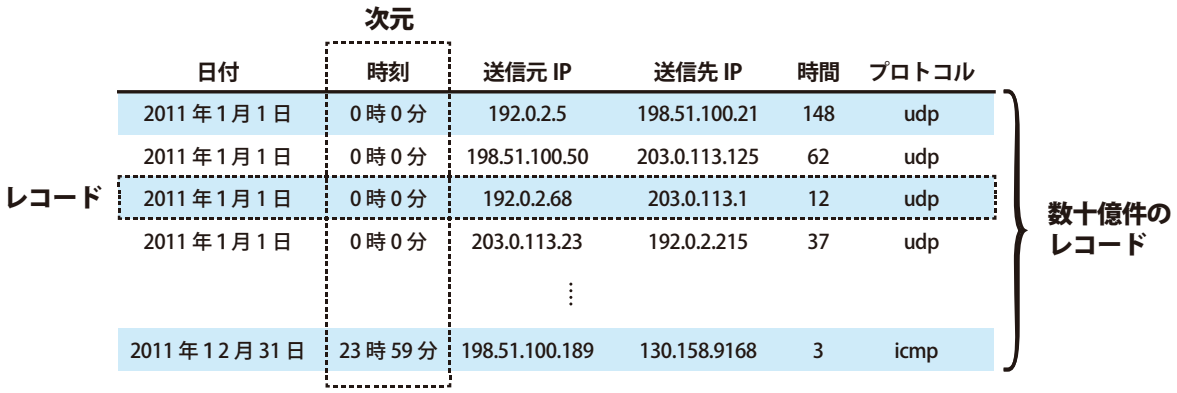

図 1.1: 大規模な多次元データの例

スマートフォンやソーシャルメディアの普及により、大規模データを分析するニーズが高まっている。本研究では、一般的なパーソナルコンピュータで大規模データの活用を促進するツールを目指す。

[1]ユーザーの近況やコメントのこと

1.2 大規模データを可視化する上での問題点

大規模な多次元データの分析には、情報の可視化というアプローチがしばしば利用される。しかし、大規模データを可視化する際、コンピュータのスペックの限界と Overplotting という 2 つの問題に直面する。

1.2.1 コンピュータの限界

可視化手法によっては、はじめにすべてのデータをメモリに読み込んでから加工を行うものがある。しかし、数十億規模のデータすべてをメモリに読み込むことは現実的ではない。ファイルから読み込んだデータをメモリに長く保存することは難しく、一時的にメモリに読み込んだデータのみで加工をしたり、膨大なデータをメモリに収まる容量で加工をする制限が課せられる。

1.2.2 Overplotting

可視化ではしばしば点や線を重ねた時に出来るシルエットからデータの傾向を見る手法が取られる。このような手法では、レコードの数が多いと描画される点や線が重なり合い、塊となって表示されてしまう。その結果、データの傾向が見えなくなってしまうという問題が生じる。これを Overplotting と呼ぶ。特に今日、研究対象としている数十億規模のデータでは、点や線も数十億個描画されるため、Overplotting は容易に発生する。

1.3 研究の目的とアプローチ

本研究では、メモリ不足に起因するコンピュータの限界と、可視化における Overplotting の問題を克服する。そのためのアプローチとして、大規模データの活用を促進するための手法を考案する。パーソナルコンピュータ上での使用を前提とした大規模な多次元データを扱うことが出来る可視化ツールを開発する。

1.4 貢献

節 1.2 の問題を解決するために、以下の 2 つを提案する。

1. コンピュータの限界について 大規模な多次元データを値の傾向を残して分別することで、大規模データをメモリに乗る容量に圧縮する。

2.Overplotting について 線や点の描画にデータの傾向による半透明なグラデーションを基準とした色付けを行うことで、見たい傾向のデータほど画面に強く現れるフィルタリングを提案する。

1. について、着目したい値によって大規模データを分別することで、一般的なパーソナルコンピュータで扱える大きさまでデータを縮小するとともに、大量のデータを分別によって整理し、データの特徴や傾向を発見するのに役立てられる。

2. について、1. で分別したデータからさらに見たいデータのみをフィルタリングするため、データの傾向を透明度に割り当てた描画による可視化手法を提案する。色を使った従来のフィルタリングは、データを非表示にする二値的な手段がほとんどであった。本研究ではデータの傾向を半透明なグラデーションに割り当てることで、描画時における二値的な絞り込みを越えたフィルタリング手法を提案する。

第2章　関連研究

本章では、大規模な多次元データの可視化手法を、多次元である点と大規模である点の 2 つの側面から紹介する。また、本研究で扱うデータをそれらの手法で可視化すると、どのような問題が生じるかを述べる。

2.1　多次元量的データの可視化

本研究では多次元データを扱う。多次元データの可視化手法は、これまで様々に研究されてきた。ここでは、多次元データを扱うための代表的な手法と、大規模故の問題点を述べる。

2.1.1　Parallel Coordinates

多次元のデータの表現手法として Parallel Coordinates[1, 2] がある。Parallel Coordinates は、各次元の値を表すための数直線を等間隔に並列に配置し、あるレコードに対して、各次元の数直線上での値の位置を折れ線グラフのように結んで表現する可視化手法である。1 つのレコードを描画した場合、折れ線が数直線と交差している位置からデータの値を読み取ることが出来る。これをレコードの数だけ描くことで、データ全体の傾向が幾重にも重なった線のシルエットから読み取ることが出来る。この手法を基に、Angular Histograms[3] や Parallel Coordinates Matrix[4] といった様々な可視化手法が提案されている。

2.1.2　Scatterplot Matrix

Scatterplot Matrix[5] は、散布図を発展させた可視化手法である。散布図とは 2 次平面の 2 つの軸にそれぞれ次元を表すための尺度を設け、レコードを点として表すものである。2 つの軸を使って次元を表すため、この手法では二次元データのみを扱える。レコードを点として大量に打った時に出来るシルエットから、データの傾向や外れ値を見ることが出来る。この手法を拡張し、二次元ではなく多次元を表現できるようにしたものが Scatterplot Matrix である。Scatterplot Matrix は、試合の対戦表のように次元を縦横に配置して碁盤目を作り、異なる次元同士が交差する枠に、その次元を軸とする散布図を表示するものである。すべての二次元同士の組み合わせを表示することで、多次元のデータに対応する。

可視化における問題

Parallel Coordinates や Scatterplot Matrix には Overplotting の問題がある。Parallel Coordinates ではレコードを線、Scatterplot Matrix では点として、レコード数に比例した数だけ描画する。そのため、データの数が増えると描画する点や線が塊となってしまい、データの傾向が見えなくなってしまう。

2.2 多次元カテゴリデータの可視化

カテゴリデータとは、値が定量的なデータではなく、性別や血液型といった名前や、背番号や電話番号といった値によって分類付けされているデータのことである。この形式のデータは分類付けを表したものであるため、順序という概念が存在しない。Parallel Coordinates のように各レコードの値を表現することよりも、データ全体に対して各カテゴリが有する割合を示すことが重要である。

2.2.1 Parallel sets

多次元のカテゴリデータを表現する最も基本的な表現として Parallel sets[6] が挙げられる。この手法は Parallel Coordinates のように次元軸を並べ、軸をその次元のカテゴリ毎の割合で区切る。隣り合う次元軸のカテゴリ毎の区間を、それぞれの次元における 2 カテゴリ間の出現頻度でさらに区切る。区切った 2 つの領域を上底と下底とする台形を描画することで、隣り合う次元のカテゴリ毎の相関を表現する。Parallel sets は軸をカテゴリの数だけ区切るため、各次元に含まれるカテゴリの総数が多い場合、矩形が分割され過ぎてデータが見えなくなってしまう。

2.2.2 Mosaic Plot

Mosaic Plot[7] は、はじめに四角形をある次元のカテゴリの割合で並行に分割する。次に分割した四角形を別の次元のカテゴリの割合で更に分割する。このとき、2 回目の分割は 1 回目の分割と垂直方向に分割にする。分割する方向を交互に変えながら次元の数だけ分割すると、四角形の面積によってカテゴリ間の割合が見える。次元数が増えるにつれて、四角形の分割が増え、視覚的に繁雑となってしまう。

2.3 大規模データの可視化

ClockMaps[8] は時刻情報に着目し、24 時間を一周とする円形表現を用いている。時刻ごとのイベントの発生頻度を、時計を模した円の背景色によって表現する。この手法では、大量のデータをイベント毎にグループ化することでレコード数を削減している。さらに、幾つかの

イベントを一つのイベントとして統合する、という作業を繰り返すことで、限られたディスプレイ解像度で大量のデータを表現する。この手法の場合、大量のデータに対して階層構造によるグループ化を行うことで、大規模データの可視化における問題を解決している。しかし、円内をイベントの出現頻度によって塗り分けている手法であるため、多次元データを表現すると、各色の領域面積が小さくなってしまい、有意な差を表現しきれない可能性がある。

Angular Histograms[3] は、Parallel Coordinates で線で表現しているものを、角度付きのヒストグラムに置き換えたものである。この手法の場合、大量の線を各次元毎に近い値同士をグループ化することで、Parallel Coordinates で大量のデータを可視化した場合に起きる Overplotting の問題を解決している。大量の線の方向を 1 つの角度付きヒストグラムとしてまとめるため、元のデータの詳細は見えなくなってしまう。

CloudLines[9] は、時間軸上でイベントが発生した時刻に円を描画することで、イベントの発生頻度を表現するものである。時系列順に円を大量に描画することで、頻度に応じて太さの変わる線のように表現する。複数のイベントを時間軸を揃えて描画することで、イベント事の関連性を見つけることが出来る。この手法も ClockMaps 同様、イベントの種類毎に線を描画するため、多次元を表現することは出来ない。

ChronoView[10] は、大量の時刻情報付きのイベントデータを可視化する手法である。24 時間を一周とする時計を模した、イベントが発生した時刻にイベントをプロットして行く。それらのプロットの重心をそのイベントの位置として円内にプロットする。データをイベント毎に集計することで、大量のデータを表現することが出来る。一方で、この手法で表せる次元は時刻とイベントのセットのみであり、多次元を表現することは出来ない。

RadViz[11, 12] は、多次元データの内、次元数の特に多い高次元データのための可視化手法である。円周上に等間隔に次元を配置し、円内にレコードを配置する。レコードの位置は、レコードから円周上の各次元に伸ばしたバネモデルによって決定する。この手法は高次元データのための手法であるため、本研究で扱うようなレコード数の多いデータには向かない。

第3章　問題分析

3.1　大規模データの問題

本研究では数十億規模のデータを扱う。このようなデータを一般的な可視化手法、例えば表計算ソフトのグラフツールで扱うことを考える。表計算ソフトではファイルを読み込み、可視化したいデータを選択し、可視化手法を選ぶことで可視化が出来る。しかし、この方法には問題がある。表計算ソフトは、大規模データの分析を行うために作られておらず、数十億規模のデータを読み込むことは困難である。さらに、全データを読み込めたとしても、Overplotting 問題やデータ加工に時間がかかりすぎるといった問題が起きる。従って、このような方法で数十億規模のデータの可視化は現実的ではない。

大規模データを、一般的な可視化手法として Parallel Coordinates や Scatterplot Matrix で可視化することを考える。レコード数の少ないデータであればこの手法は有用であるが、数十億規模のデータをこの手法で表すと、大量の情報がひとつの画面に凝縮され、視覚的に繁雑となる。また、これらの手法は点や線でレコードを表現するため、レコード数が多いと重なり合いが増えすぎて埋もれてしまう。このようなデータですべてのレコードを描画するのは現実的ではなく、何らかの方法で集計を行い、レコード数を削減する必要がある。

3.2　多次元データの問題

本研究で扱う多次元データは、1 つのレコードが複数の値を持つ。このようなデータの可視化手法は、多くの場合次元毎に表現が切り分けられる。この切り分けられた表現の間で、レコードが同一であることを表現する必要がある。例えば Parallel Coordinate では、1 件のレコードを描画すると、軸と折れ線との交点から各次元の値を読み取ることが出来る。しかし大量のレコードを描画すると、描画するグラフの数が比例して重なりあうため、レコードの次元毎のデータが読めなくなってしまう。また、Scatterplot Marix では散布図から 2 次元の関係を見ることが出来る。3 次元以上の関係を見る場合は、2 つ以上の散布図を同時に見る必要がある。しかし、散布図間でレコードがどのように分布しているかを見ることは困難である。

第4章　表現手法の設計

4.1　手法の概要

4.1.1　表現手法の方針

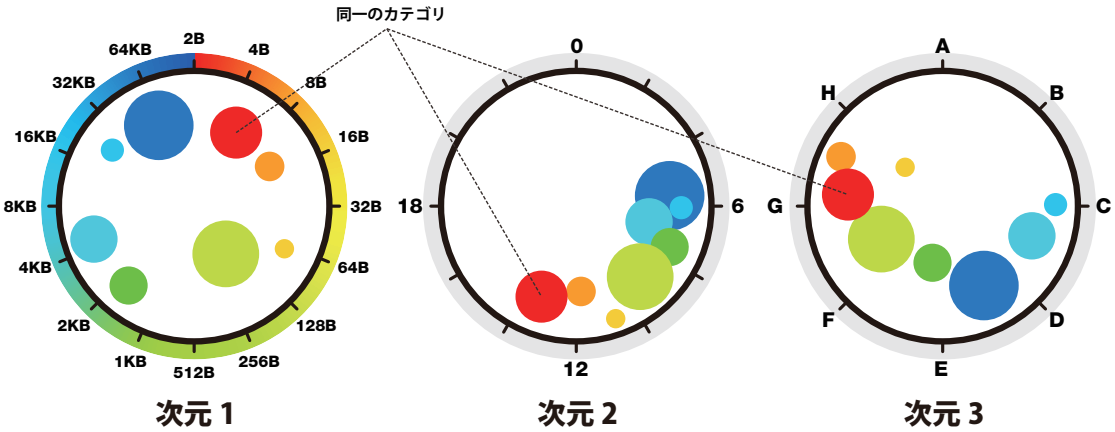

図 4.1: 手法の全体像

本研究では、大規模な多次元データの表現手法として、レコードをカテゴリに分別し、カテゴリの次元の値を円の内部にプロットして表現する手法を提案する（図 4.1）。プロットの位置決定には、次元を表す円周上に敷かれた目盛りを用いる。あるカテゴリに属するレコードの次元の値を、次元を表すための円の目盛り上に当てはめ、それらの位置の重心にカテゴリをプロットする。同じようにして他のすべてのカテゴリや次元も描画する。同一のカテゴリは、別々の次元を表す円で常に同じ色で塗る。こうすると、次元間の色の散らばり方から、次元間のデータの相関を見ることが出来る。

4.1.2　可視化のプロセス

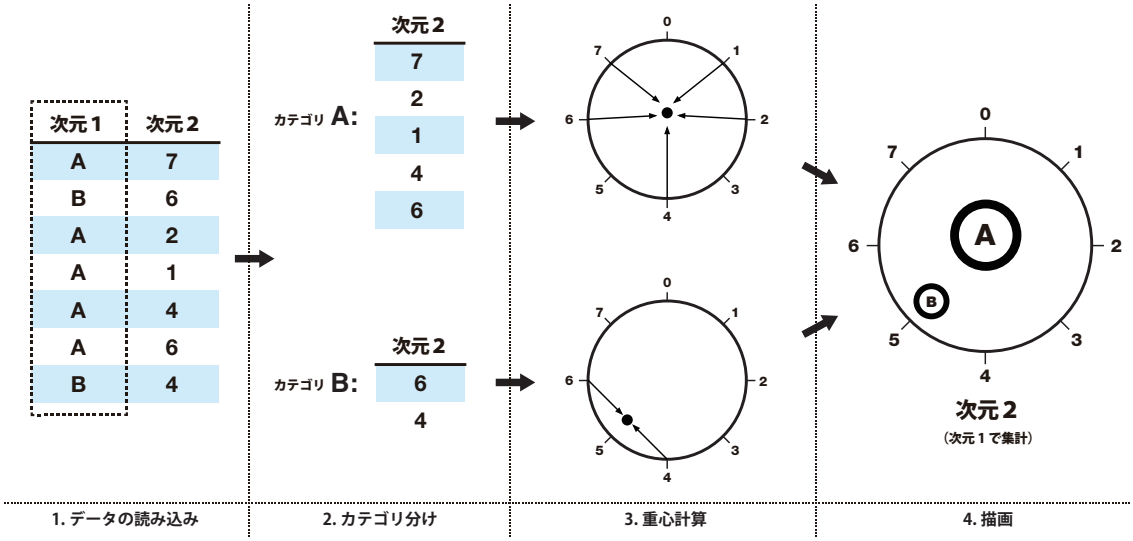

図 4.2: データの加工から描画までの流れ

提案する可視化手法の処理を図 4.2 に示す。

分析したいデータから 1 行のレコードを読み込むと、はじめにそのレコードのカテゴリ名を決定する。カテゴリ名は、読み込みを行う前に予め指定しておいた次元の値を使って決定する。図 4.2 の場合、カテゴリ分けのための次元は「次元 1」とし、1 行目のレコードのカテゴリ名は「A」となる。

カテゴリ名毎に分別が終わると、次にカテゴリ毎の重心を求める。重心の元となる位置は、円周上の目盛り位置である。例えば図 4.2 の場合、カテゴリ A の値は、円周の目盛り上での位置がまんべんなく出現しているため、カテゴリ A の重心位置は円のほぼ中央に寄っている。逆にカテゴリ B は、目盛り上の位置が偏っているため、重心位置は円の縁の近くとなる。このようにカテゴリ毎の値を重心としてまとめた後でも、おおよその値の分布が分かる。

カテゴリ毎の重心が求まると、最後に次元を表す円の中にすべてのカテゴリを小さな円として描いていく。円の面積はカテゴリ内の出現数に比例する。図 4.2 の場合、カテゴリ A の出現数は 5、カテゴリ B の出現数は 2 であるため、この 2 つの円は面積の比が 5 : 2 となるように描画される。図 4.2 の例では「次元 2」という 1 つの次元のみの描画だが、複数の次元の値も同様にして処理することで、多次元データを表現することが出来る。

4.2　カテゴリデータへの変換

ファイルから読み込んだレコードに対し、レコードのカテゴリを決定する。ここでは 3 種類のデータ形式別のカテゴリの決定方法を説明する。

4.2.1 カテゴリデータ

カテゴリデータとは、定量的な値ではない、一意なラベル付けのデータのことである。例えば通信ログの場合は IP アドレス、ユーザーデータの場合は氏名、血液型、アカウント名、といったようなデータである。この形式のデータは、すでにカテゴリデータになっているため変換の必要はない。カテゴリデータの変換用の次元としてカテゴリデータが含まれる次元が選択されている場合は、値をそのまま利用してカテゴリ分けをする。例えば、血液型の次元であれば血液型毎に集計する、といった具合である。

4.2.2 数値データ

数値データとは、定量的な値のデータのことである。例えばユーザーデータの場合は年齢、通信ログの場合は送信したバイト数、といったようなデータである。この形式のデータをそのままカテゴリとして分別に使うことは出来ない。なぜならカテゴリデータへの変換はレコード数の削減の目的があり、数値データをカテゴリとして集計しても、レコード数の削減には繋がらないためである。

数値データをカテゴリデータに変換するために、値を区切ることを行う。区切り方には比例で区切るものと、対数で区切るものがある。

値の範囲が限定されているものは、比例で区切る。例えば年齢の場合、比例で区切ると 10 歳未満、10 代、20 代、30 代というカテゴリ分けになる。また、年度の場合は、1980 年代、1990 年代、2000 年代といったカテゴリ分けになる。

値の範囲が広範囲の場合は、対数で区切る。例えばバイト数の場合、対数で区切ると 2B、4B、8B、16B というカテゴリ分けになる。この区切り方は対数グラフのような効果が得られるため、値の範囲が広い場合でも、カテゴリ分けを少なくすることが出来る。

4.2.3 日時データ

日時データは、先に述べたカテゴリデータや数値データとは異なり、周期の見方によって年、月、週、日、時、分、秒、というようにカテゴリデータへの変換の仕方が複数存在する。例えば分で集計した場合は全 60 カテゴリ、日で集計した場合は全 31 カテゴリとなる。

4.3 軸目盛りの設計

重心を計算するために、目盛り上におけるレコードの位置を知る必要がある。この位置は必ず円周上に存在するため、角度 $\{\theta\colon 0 \leq \theta < 2\pi\}$ によって表現することが出来る。ここでは 3 種類のデータ形式のそれぞれの取得方法を説明する。

4.3.1 カテゴリデータ

カテゴリデータの場合、ファイルの中にどのようなカテゴリデータが含まれているか、読み込まなければ分からない。そこで、データを読み込みながらカテゴリの円周上の位置を決定する手法を取る。読み込んだレコードのカテゴリが初めて出現したものである場合、円周上に新たにカテゴリを配置する（図 4.3）。配置した円周上の位置を、重心計算に用いる。

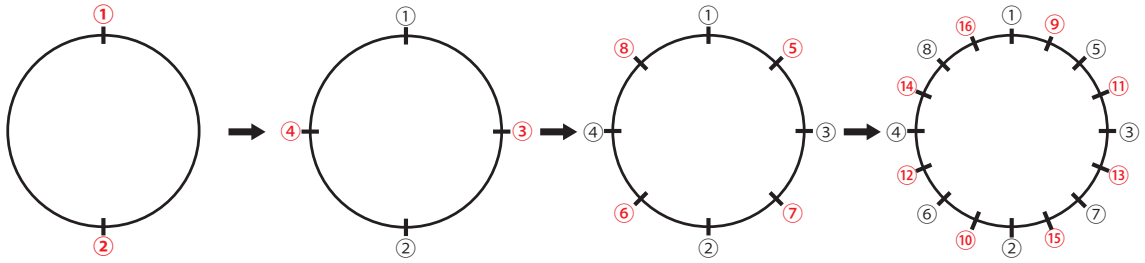

図 4.3: カテゴリの配置の変遷

4.3.2 数値データ

数値データは値の最大値と最小値、比例か対数かを予め指定しておくことで、円周上の目盛りの位置を決めることができ、角度を求めることが出来る（図 4.4, 図 4.5, 図 4.6, 図 4.7）。本研究では、これらの情報はデータの内容からおおよその見当を付けて設定しておく。

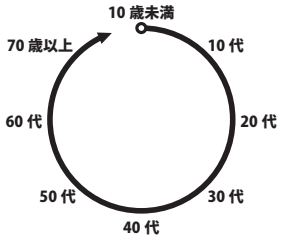

図 4.4: 年齢の目盛り

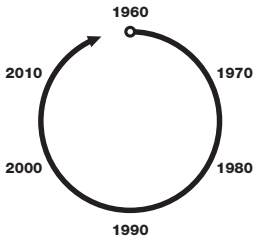

図 4.5: 年代の目盛り

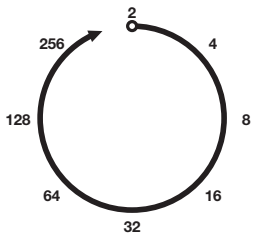

図 4.6: 対数区切りの目盛り

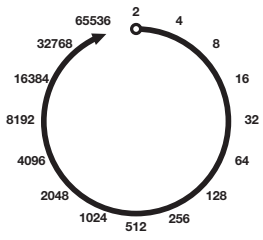

図 4.7: より広範囲な対数区切りの目盛り

4.3.3 日時データ

日時データは、円の 1 周分の長さを時、日、週、月、年と変えた場合の、日時データの位置から角度を求める（図 4.8, 図 4.9, 図 4.10, 図 4.11, 図 4.12）。

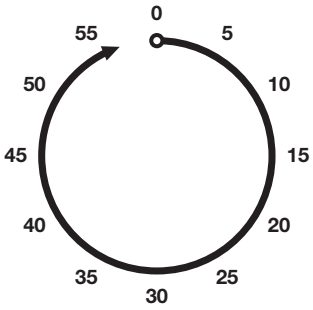

図 4.8: 時周期の場合の目盛り

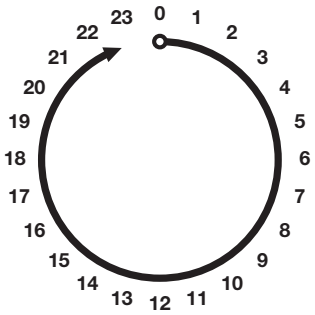

図 4.9: 日周期の場合の目盛り

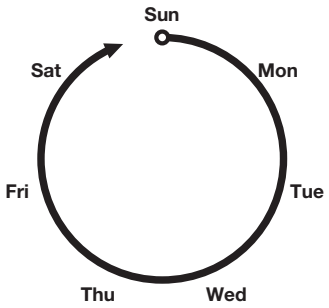

図 4.10: 週周期の場合の目盛り

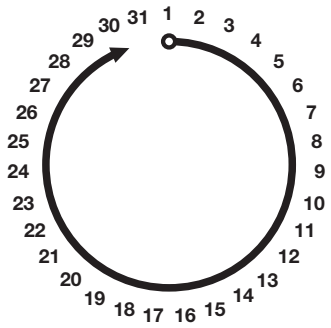

図 4.11: 月周期の場合の目盛り

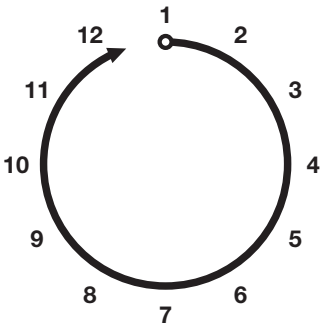

図 4.12: 年周期の場合の目盛り

4.4 重心座標の決定

角度が求まったら、次元におけるカテゴリの重心、つまり平均を求める。一般的には集合 S の要素の平均は $\frac{1}{|S|}\sum_{s\in S} s$ によって求められる。しかし、本手法はメモリ上に集合 S のような配列を持つことが出来ない。そこで、平均を求めたい数列を $N =< n_0, n_1, n_2, ... >$、$n_0$ から n_i まで順番に計算した際の平均を μ_i とした時、常に最新の平均を算出する式 4.1 を用いる。

$$\mu_i = \begin{cases} n_0 & (i = 0) \\ \dfrac{\mu_{i-1} \cdot i + n_i}{i+1} & (i > 0) \end{cases} \tag{4.1}$$

円の半径を r、4.3 節でカテゴリ毎に求めた角度の列を $T =< t_0, t_1, t_2, ... >$ とすると、t_0 から t_i まで順番に計算した際の重心座標 p_i は、式 4.2 を用いて式 4.3 で表わされる。

$$f_0(\theta) = (r\sin\theta, r\cos\theta) \tag{4.2}$$

$$p_i = \begin{cases} f_0(t_0) & (i = 0) \\ \dfrac{p_{i-1} \cdot i + f_0(t_i)}{i+1} & (i > 0) \end{cases} \tag{4.3}$$

4.5 プロット

重心が求まったら、最後にカテゴリを表すプロットを描画する。カテゴリを、重心座標を中心とする小さな円として次元を表すための円の中に描く。プロットのサイズはすべて均一に描画する（図 4.13)、カテゴリに属するレコード数の出現数に比例させる（図 4.14, 図 4.15）などの方法がある。

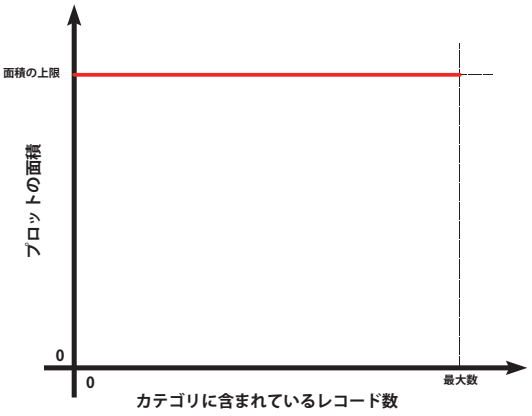

図 4.13: 比率を考慮しないプロットサイズ

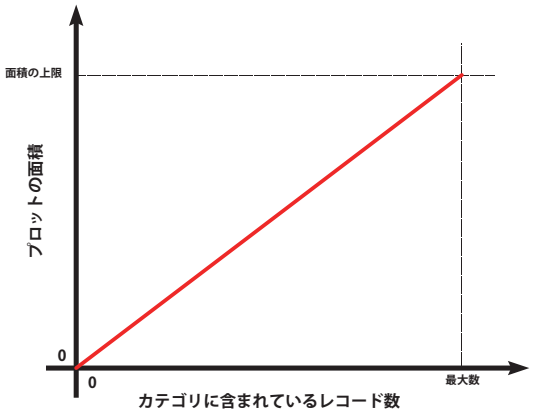

図 4.14: 最大レコード数を基準としたプロットサイズ

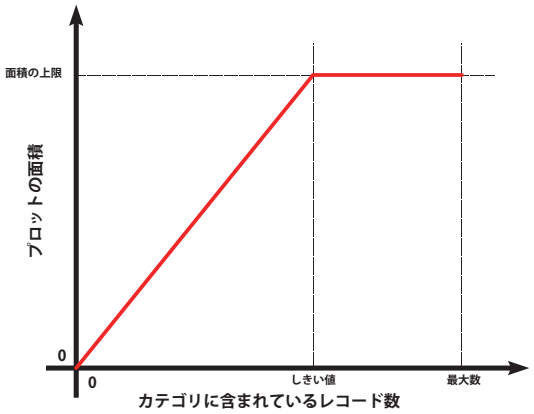

図 4.15: しきい値を基準としたプロットサイズ

4.6　アニメーション表現

大規模データは、情報をリアルタイムに記録するため、ファイルの上から時系列順にレコードが格納されている事が多い。本提案手法は、描画に必要な情報を逐次更新していく。そのため、データを読み込みながら定期的に再描画することで、最新の分析情報を表示することが出来る。また、このような閲覧方法は、レコードの瞬間的な大量発生や周期的な発生といった、静的な可視化手法では見ることの出来ない“ 動き”の発見につながる。

第5章　分析ツールの開発

5.1　ツールの概要

第 4 章で説明した手法を用い、大規模データを分析するためのツールを開発した。図 5.1 は、開発したツールの概観である。中央の領域が分析ビュー、その右にあるのがツールボックスである。分析ビューの中には、多次元データの次元毎の値を表現するための次元ビューがある。分析のための操作は、ツールボックスの中にある各種コントローラーから行う。

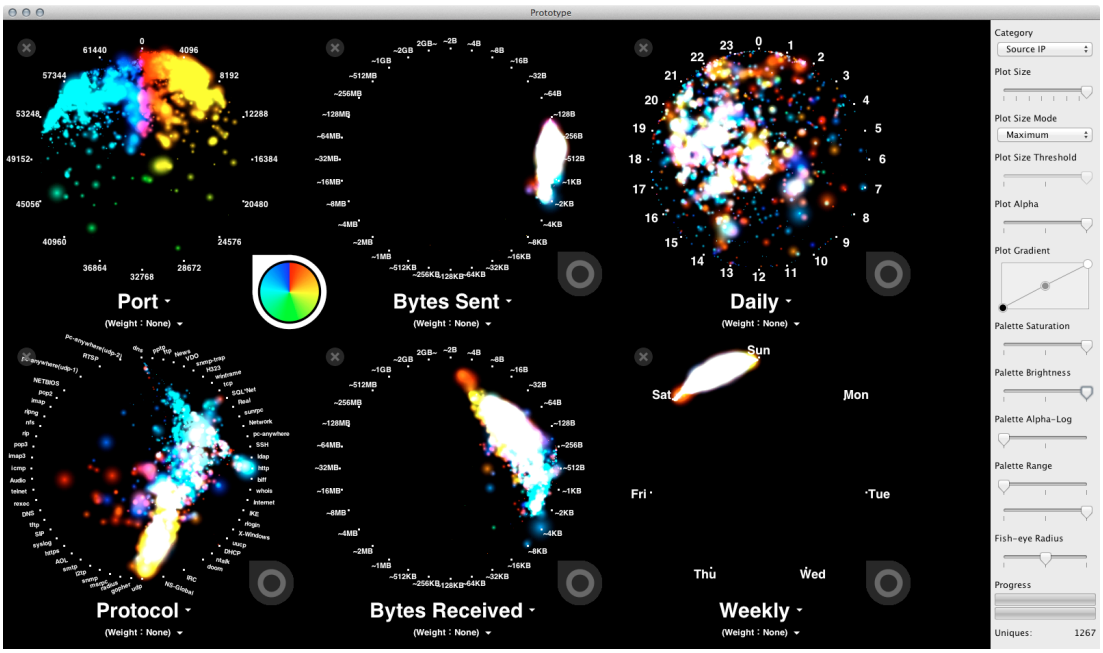

図 5.1: 分析ツールの概観

5.2　システムの実装

本ツールは大規模データの活用を促進するツールを目指すため、処理速度が速く、かつ異なったオペレーティングシステムでも動作するように、Java を使用して開発を行った。本ツールは Java Runtime Environment 上で動作する。GUI や可視化の基本的な部分の描画には Java の swing を用いているが、プロットを打つ部分には OpenGL を利用している。多次元データのフォーマットには tsv 形式を用いている。

5.3　次元ビュー

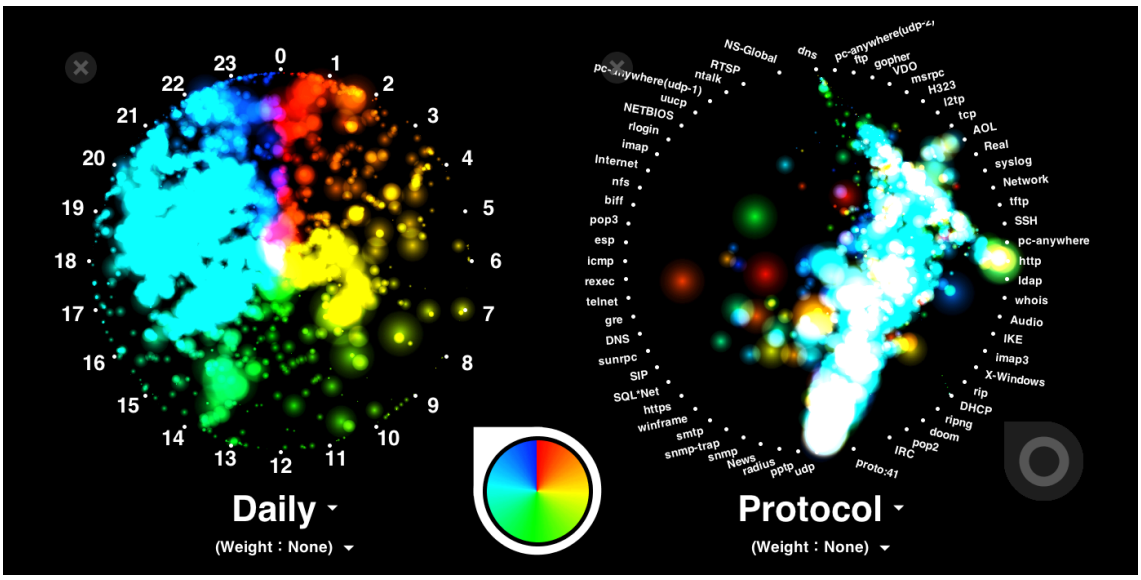

図 5.2: 次元ビュー

次元ビューは多次元データのある次元を表示するための目盛り付きの円を表示する（図 5.2）。目盛りは、その次元が持つデータの形式によって変わる。データを表すための円の下には、その次元ビューで表示される次元の名前と、集計の際の重みづけの次元の名前が表示される。データの読み込みを開始すると、この円内にカテゴリがそれぞれの重心位置にプロットとして描画される。それぞれの次元ビューの右下には、色付け用に設定された次元にはパレットがある。また、それ以外の次元にはこの次元を色付け用の次元に設定するためのボタンが表示される。

5.3.1　目盛り

次元ビューの目盛りは、3 種類あるデータ形式でそれぞれ異なる。3 種類とも、時計の 12 時を原点として、時計回りに配置する。

カテゴリデータの目盛り

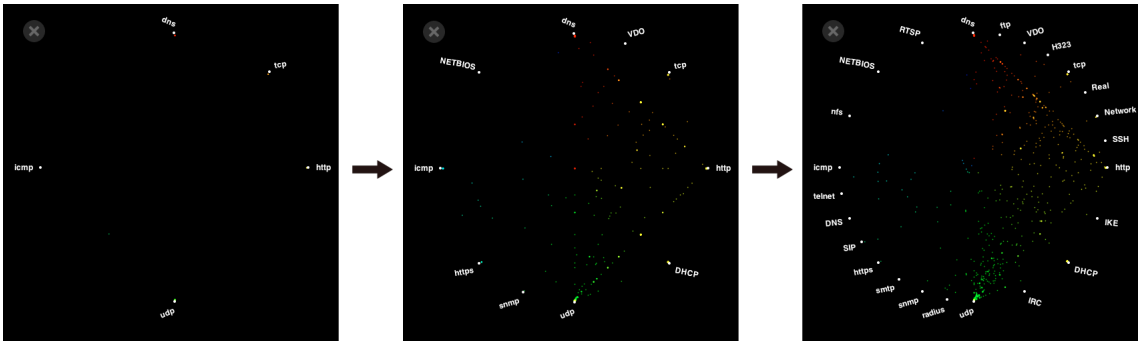

図 5.3: カテゴリデータビューの目盛りの変遷

カテゴリデータの場合、データを読み込みながら円周上にラベルを配置していく。図 5.3 は、実際にデータを読みながら結果を描画される間に、目盛りが配置されていく様子である。カテゴリデータの目盛りのラベルは、重なりを防ぐために少しずつ角度を変えている。

数値データの目盛り

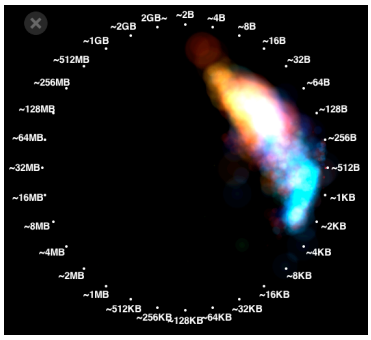

図 5.4: バイト数の目盛り

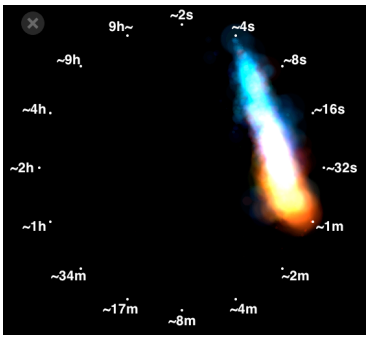

図 5.5: 所要時間の目盛り

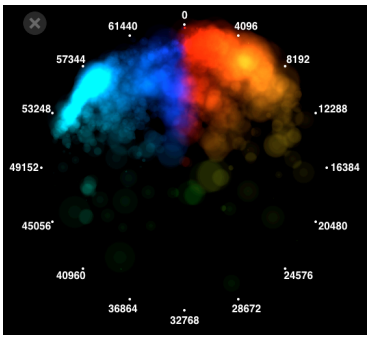

図 5.6: ポート番号の目盛り

数値データは、予め値の最小値と最大値を設け、データを読み込む前に目盛りの位置を決定しておく。目盛りの値は 4.3.2 節で述べたように比例または対数を用いる。単位のある形式には、目盛りに単位を表示する。この時、ラベル表記の混雑度を軽減するため、ラベルの文字数ができるだけ少なくなるように、可能な場合は単位の変換を行う。例えばバイト数の場合は 1024B ではなく 1KB、所要時間の場合は 3600 秒ではなく 1 時間、といった具合である（図 5.4, 図 5.5）。単位のない数値データの場合は数値のみを表示する（図 5.6）。

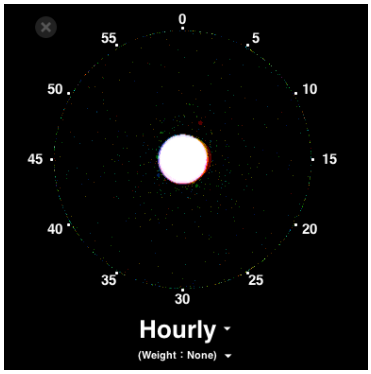

図 5.7: 時周期の場合の目盛り

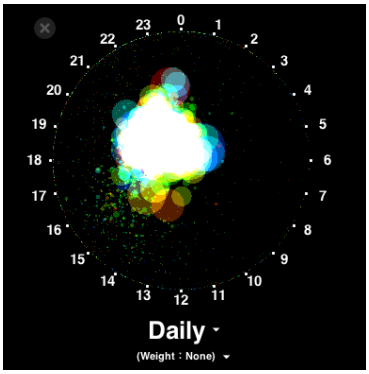

図 5.8: 日周期の場合の目盛り

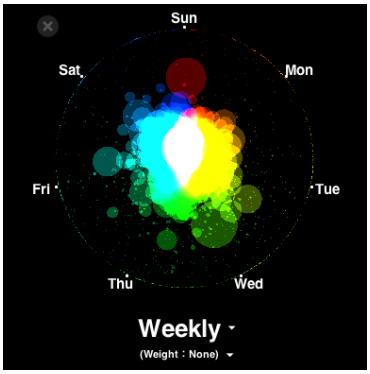

図 5.9: 週周期の場合の目盛り

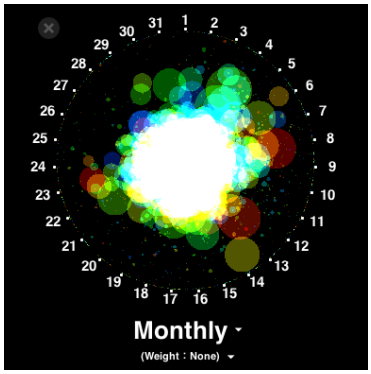

図 5.10: 月周期の場合の目盛り

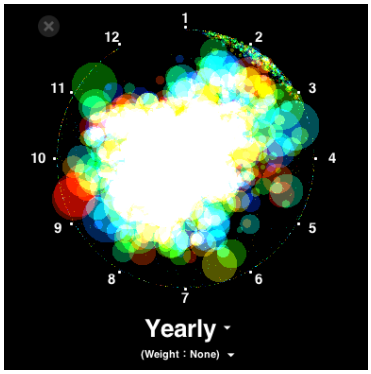

図 5.11: 年周期の場合の目盛り

日時データの目盛り

日時データは周期の長さを変えることで様々な視点で見ることが出来る。本手法では時、日、週、月、年を用意した（図 5.7, 図 5.8, 図 5.9, 図 5.10, 図 5.11）。

5.4 パレット

プロットを着色するためのパレットは、分析内容に応じて切り替えることが出来る。本研究では 4 種類のパレットを提案する。

5.4.1 通常のパレット

通常のパレットは、円周方向に色相環を割り当てたパレットである（図 5.12, 図 5.13）。色相環は本来は途切れなくつなげることができるが、このパレットでは時計の 12 時の位置で途切れるようにしている。これは順序尺度の次元は円周上の目盛りが 12 時の位置で途切れているためである。順序尺度の場合、目盛りの最も低い値の位置と最も高い値の位置が隣り合っている。そのため、パレットの 12 時の位置で色が途切れていないと、12 時付近にあるプロットの値の大きさが区別出来ない。

5.4.2 中央が見えるパレット

このパレットは、通常のパレットを、円周に近くなるほど透明にしたパレットである（図 5.14, 図 5.15）。本手法ではパレットの位置は円周上の値の位置から求めた重心で決まる。つまり、このパレットはカテゴリの重心が円の中央に来るもの、すなわちカテゴリ内の値が分散しているカテゴリほど見えやすく表示されるパレットである。

5.4.3 縁が見えるパレット

このパレットは、中央が見えるパレットとは逆に、円の中心ほど透明になっている（図 5.16, 図 5.17）。つまり、このパレットではカテゴリ内の値がある値に集中しているカテゴリほど見えやすく表示されるパレットである。

5.4.4 偏りを色に割り当てたパレット

このパレットは、円周方向ではなく半径方向に色相環を割り当てている（図 5.18, 図 5.19）。先に紹介した中央が見えるパレットと縁が見えるパレットでは、カテゴリ内の値の偏りはただ透明度としてしか表されない。値の偏りをより詳細に見たい場合は、透明度の代わりに色相を使ったこのパレットを利用出来る。

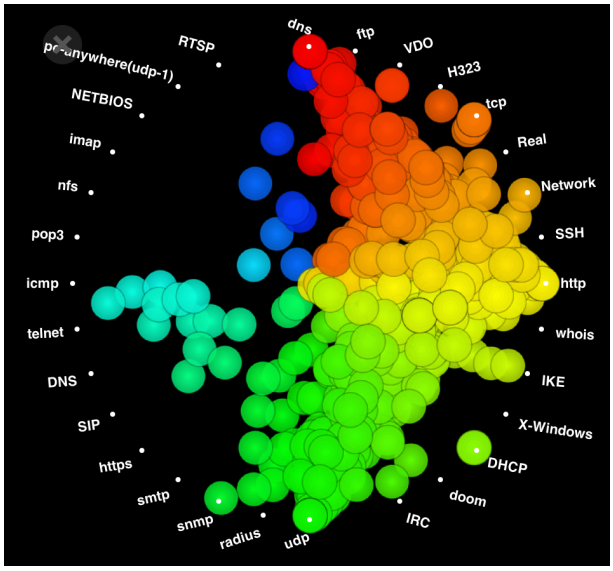

図 5.12: 通常のパレットによる着色

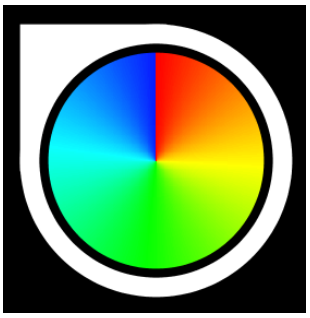

図 5.13: 通常のパレット

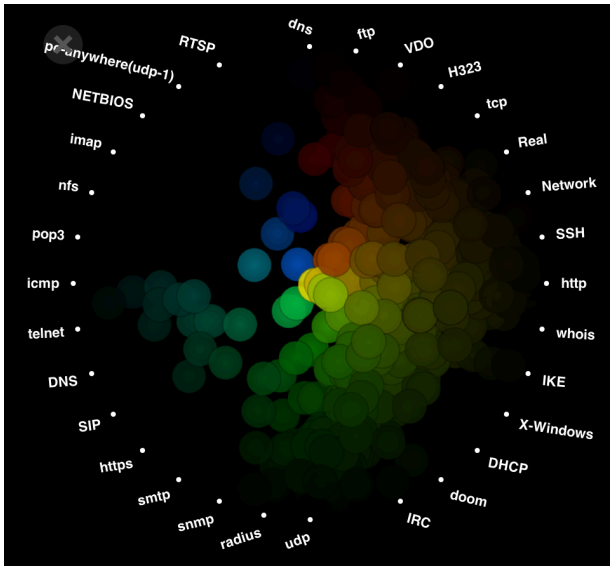

図 5.14: 中央が見えるパレットによる着色

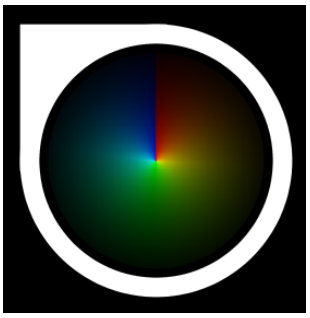

図 5.15: 中央が見えるパレット

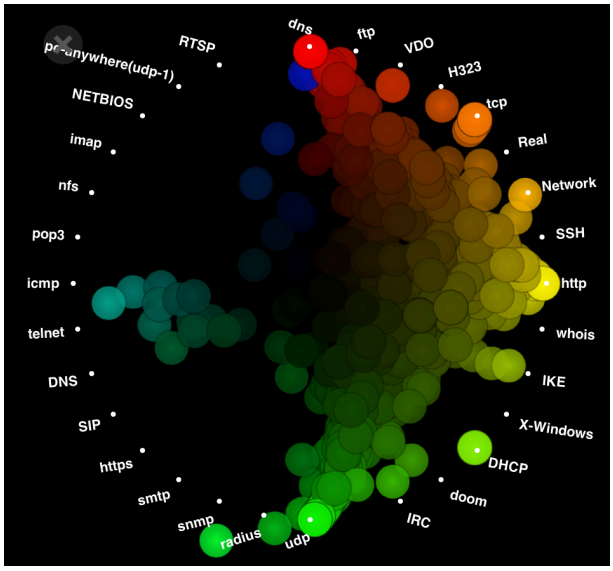

図 5.16: 縁が見えるパレットによる着色

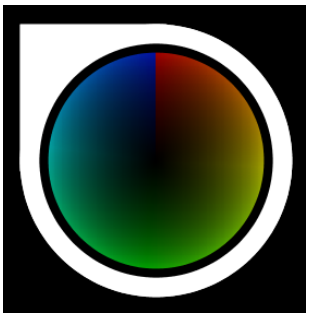

図 5.17: 縁が見えるパレット

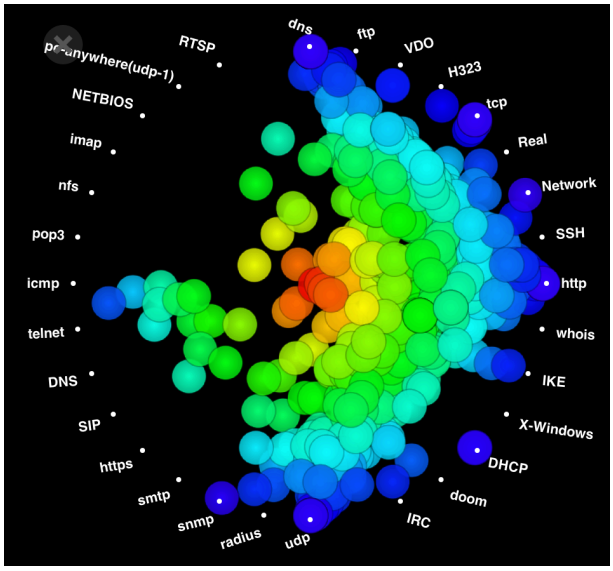

図 5.18: 偏りを色に割り当てたパレットによる着色

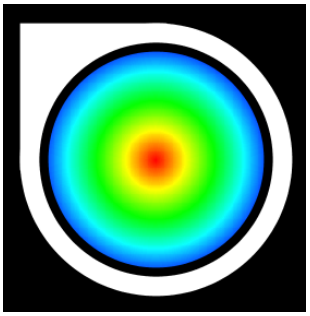

図 5.19: 偏りを色に割り当てたパレット

5.5　プロット

本手法では、1 つのカテゴリを 1 つの小さな円として表現する。これをプロットと呼ぶ。プロットの描画は、高速化のため OpenGL を利用している。

5.5.1　ブレンドモード

OpenGL は、すでに描画されているオブジェクトの上に新たにオブジェクトを描画する際、重なり合う部分の着色をどのようにするかを決めることが出来る。本手法では、プロット同士が重なりあった部分着色のブレンドモードとして、上書き（図 5.20）と加算（図 5.21）の 2 種類を用意した。

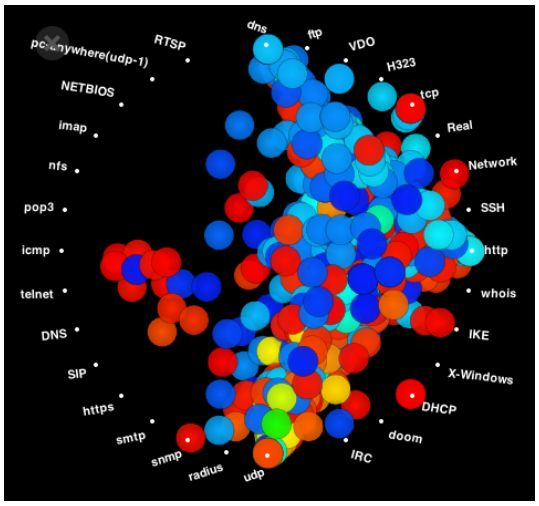

図 5.20: ブレンドモード：上書き

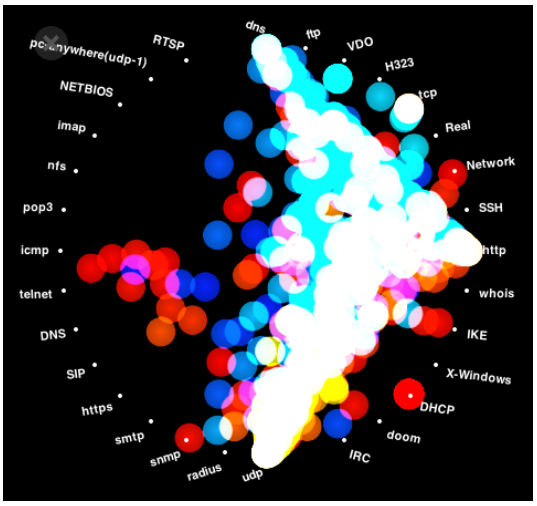

図 5.21: ブレンドモード：加算

上書き

上書きは、すでに描画されているオブジェクトの上に重ねて描画する。このモードは古い描画が埋もれていくため、表面に見えているプロットは新しいものとなる。

加算

加算は、すでに描画されているオブジェクトの色と、新しく描画する色を加法混色によってブレンドする。上書きでは大きなプロットの下に小さなプロットが隠れてしまうことがあるが、加算では隠れたプロットも見通すことが出来る。RGB チャンネル同士を加算した色で描画していくため、大量にプロットした場合は色の値が最大値に達し、図 5.21 のように白くなってしまうことがある。

ブレンドモードを加算にしている場合、着色の明度を下げると、1 つのプロットの RGB の値が小さくなり、描画結果が白くなってしまうのを抑えることが出来る（図 5.22）。また、明度をさらに下げると、プロットの色の傾向が見えるようになる（図 5.23）。

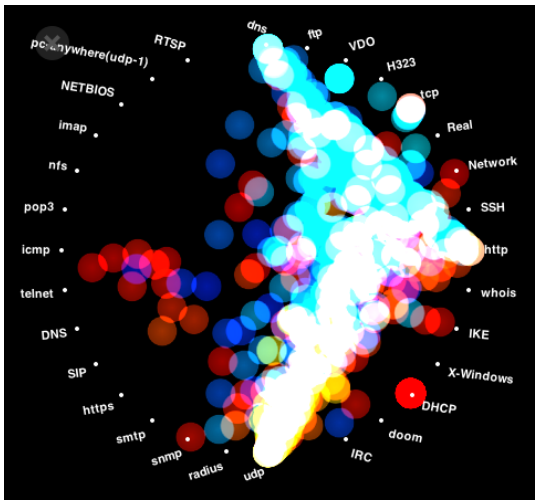

図 5.22: 加算で明度を 60%で描画

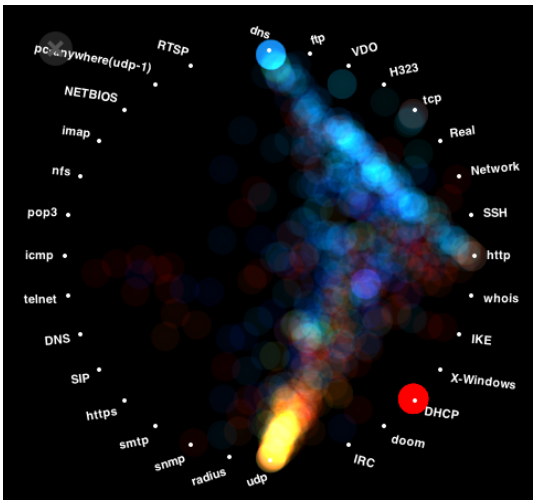

図 5.23: 加算で明度を 10%で描画

5.5.2 テクスチャ

OpenGL を使ったプロットの描画には、円形のテクスチャを用いている。このテクスチャは、ユーザー操作によって自由に変更できる。テクスチャやプロットサイズ、5.5.1 で述べたブレンドモードの設定によって、プロットの分布が見やすくなる、色の量が見やすくなるなどの変化が生まれる（図 5.24）。

5.5.3 魚眼レンズ

本提案手法では、プロットを大量に描画する。この時、プロット同士の位置関係を詳しく見たい場合や、大量のプロットを掻き分けて見たい場合にこの魚眼レンズ機能を用いる。次元ビューの円内にマウスカーソルが侵入すると、マウスカーソルの近くにあるプロットは周囲に押しのけられるように移動する（図 5.25）。このツールはリアルタイムにプロットを描画するため、マウスカーソルを動かすと逐次押しのけられた結果が表示される。マウスカーソルを円の外に出すと、プロットは元の位置に戻る。また、魚眼レンズの半径はユーザーが任意に変えることが出来る。

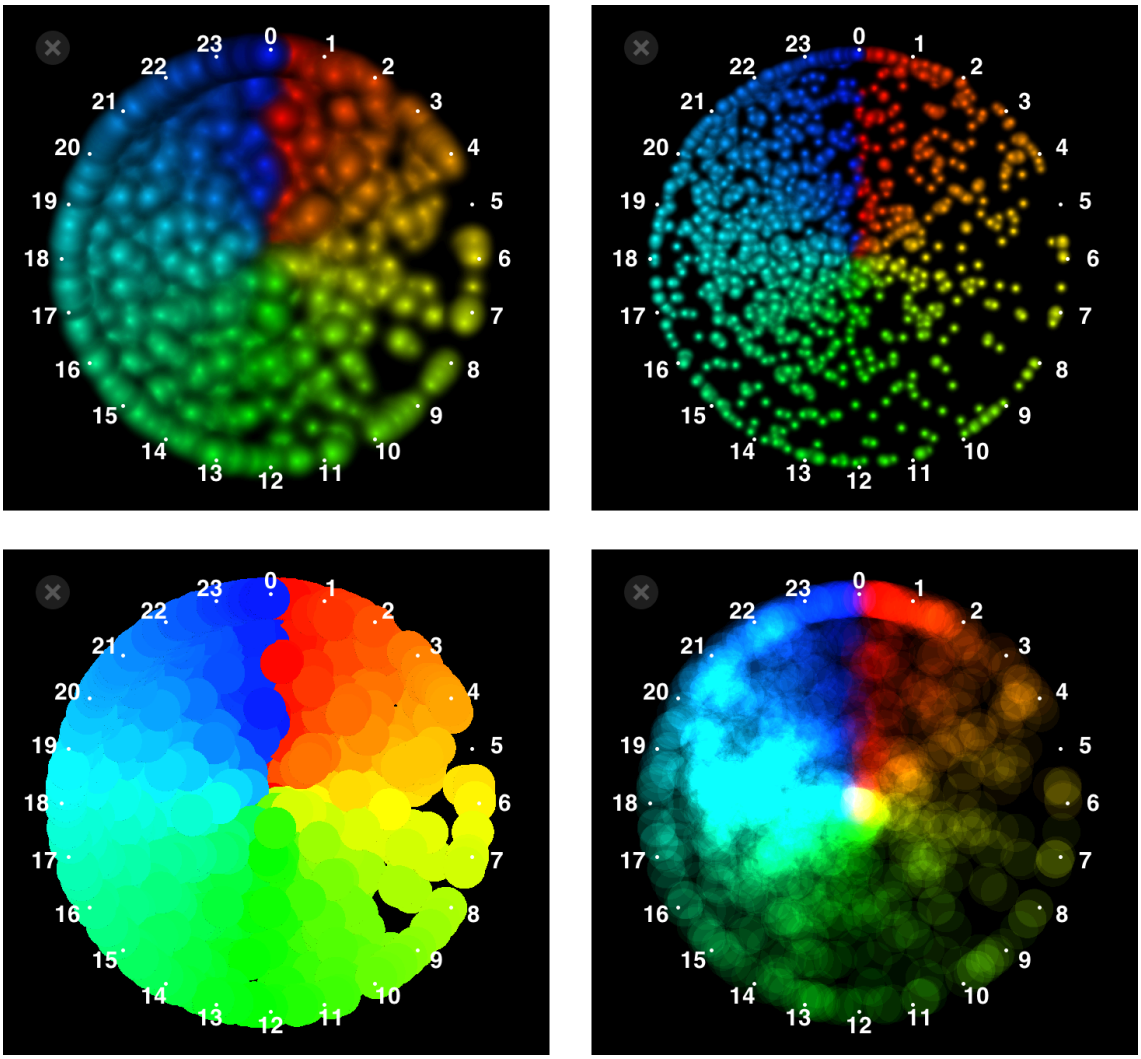

図 5.24: テクスチャとブレンドモードによるバリエーション

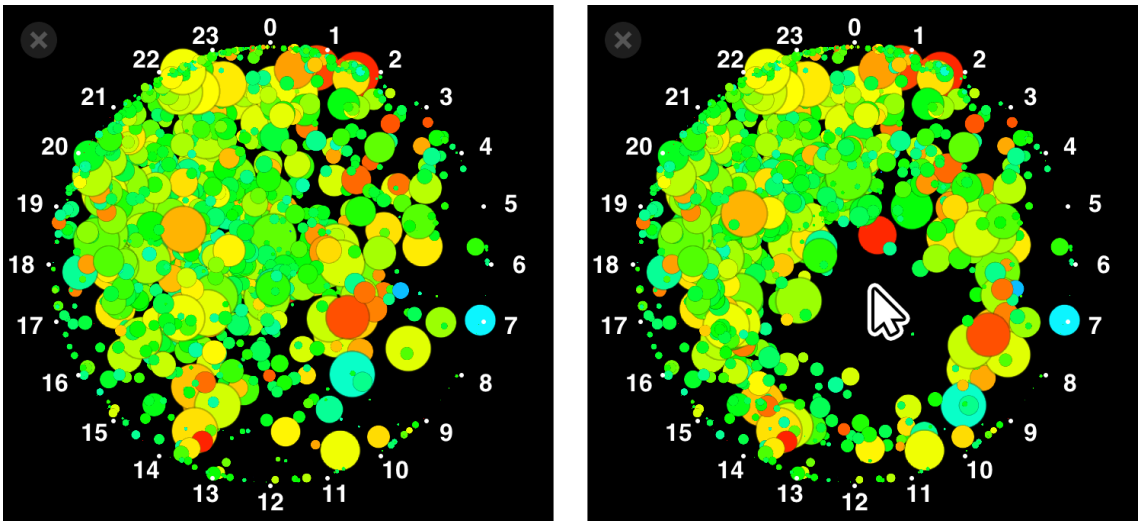

図 5.25: 魚眼レンズを使ったインタラクション

5.5.4　データの詳細表示

本手法ではレコードをカテゴリデータとしてまとめている。カテゴリは重心の他にカテゴリの名前、カテゴリに属するレコード数の合計、重みの次元が設定されている場合は重みの合計を詳細データとして表示することが出来る。重みの合計とは、次元ビューに対して重みの次元が設定されていた時、カテゴリに属するすべてのレコードの、重みとして指定された次元の値の合計である。

魚眼レンズを使ったインタラクション中、マウスのすぐ近くにあるプロットのカテゴリ名、レコード総数、重みの合計、の順番でカテゴリの詳細が表示される（図 5.26）。マウスが侵入している次元ビューでは、マウスを始点としたプロット位置の延長線上に詳細データを表示してプロットと線で結ぶ。それ以外の次元ビューでは、マウスが侵入しているビューにおける位置と同じ位置に詳細データを表示し、プロットと線で結ぶ。

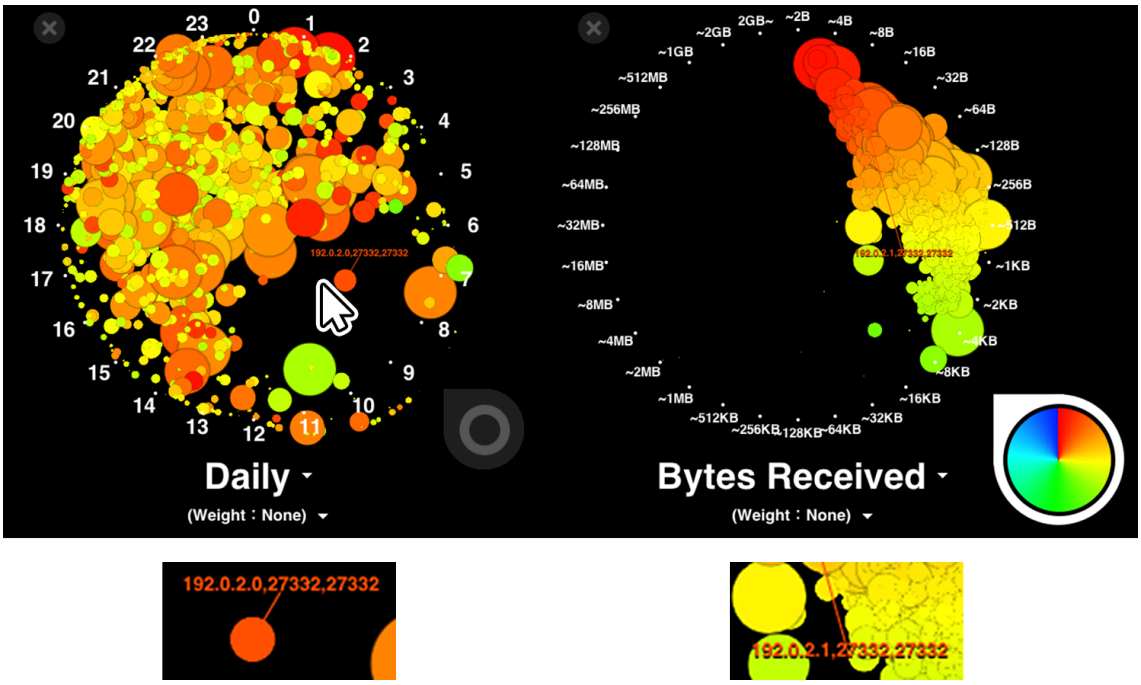

図 5.26: データの詳細表示

5.6　フィルタリング

アルファ値の調整

5.4 の中央が見えるパレットや縁が見えるパレットは、徐々にアルファ値を変えて行く具合を調節することが出来る。パレットのアルファ値は、パレットの半径を 1 とし、中心からの半径を $r(0 \leq r \leq 1)$、調整の係数を $p(0 \leq p)$、計算結果のアルファ値を $\alpha(0 \leq \alpha \leq 1)$ とすると、式 5.1 によって導かれる。

$$\alpha = \begin{cases} 1 - \dfrac{log_e(r \cdot p + 1)}{log_e(p + 1)} & (\text{中央が見えるパレット}) \\ 1 - \dfrac{log_e((1 - r) \cdot p + 1)}{log_e(p + 1)} & (\text{縁が見えるパレット}) \end{cases} \tag{5.1}$$

パレットの透明度を変えることで、データの重心位置によるフィルタリングを調節することが出来る。図 5.27 では、透明度の変化を通常の中央が見えるパレットよりも強くした。また、図 5.28 では変化をさらに強くし、重心が円のほぼ中心にあるものしか表示されなくしている。

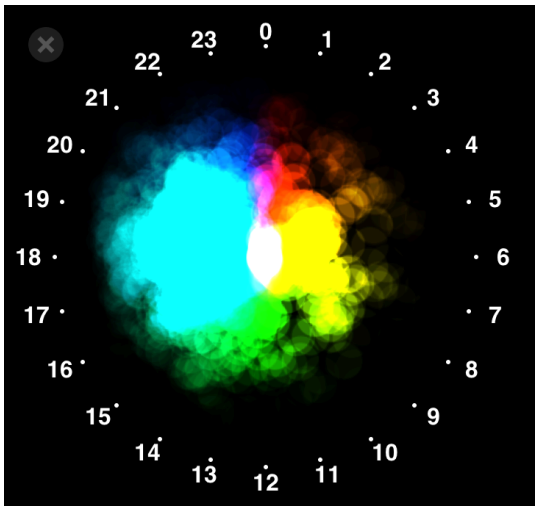

図 5.27: $p = 3$ のパレットによる着色

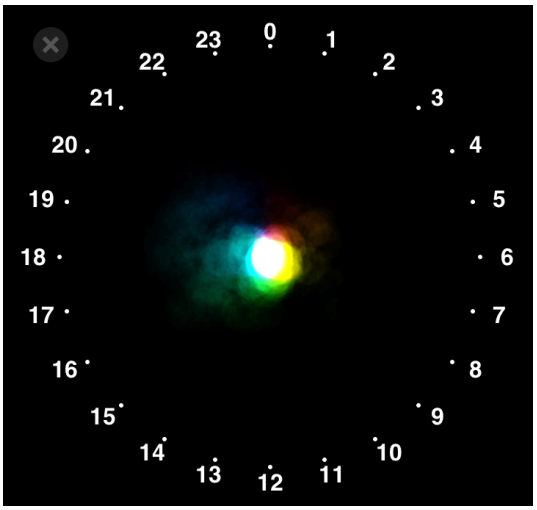

図 5.28: $p = 6000$ のパレットによる着色

角度フィルタ

本ツールはパレットの透明度の調整によるフィルタリングとは別に、角度方向のフィルタリングが可能である。このフィルタリングは 4 種類すべてのパレットで利用できる。角度フィルタを使うと、角度方向の範囲を絞ることが出来、範囲外のカテゴリがすべての次元ビューにおいて表示されなくなる。例えば図 5.30 のパレットでは、時計の 3 時付近に重心があるデータしか表示されなくなる。

図 5.29: 角度フィルタなし

図 5.30: 角度フィルタあり

この機能は、単に値の範囲を絞ることで、視覚的な混雑度を減らす目的があるが、データが一箇所に集中した際に色相環を割り当てし直すという目的もある。

図 5.31 は、左の次元を色付けの次元に設定しており、データは時計の 3 時付近に集中して

いる。このようにデータが集中した次元で着色をすると、すべての次元において、プロットが近い色で塗られてしまい、データの分散が見えづらい。

そこで、パレットの角度による絞り込みを行う。図 5.32 は、図 5.30 のパレットを適用したものである。パレットの角度範囲を狭めると、それに伴って角度に対する色相環の割り当てが大きくなる。図 5.31 でほとんど黄色く塗られてしまっていた右の次元が、色相環の再割当てによって図 5.32 のように分散が見えるようになる。

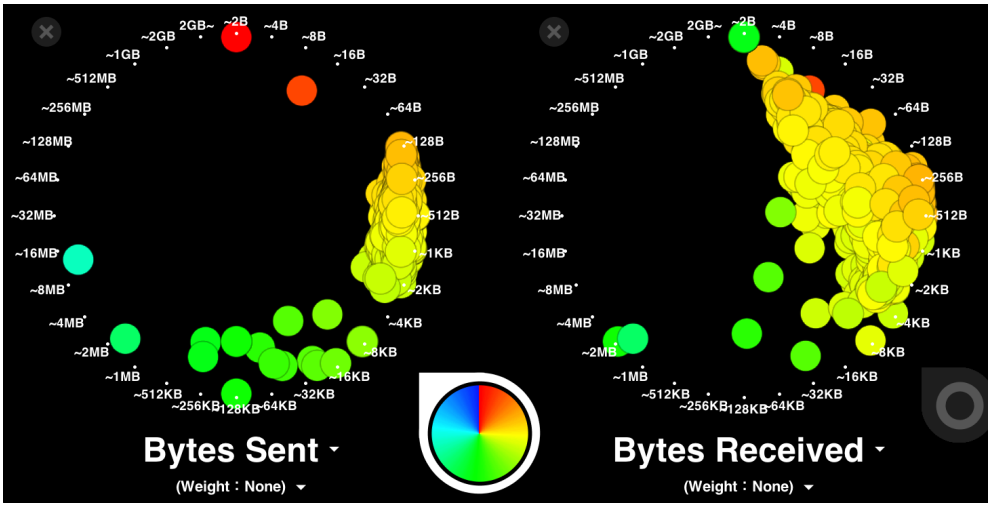

図 5.31: データが一箇所に固まった例

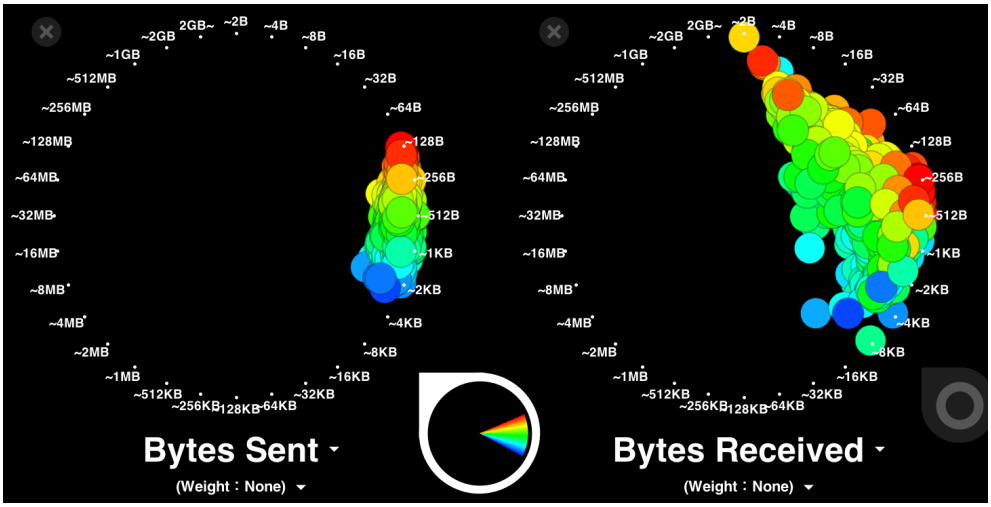

図 5.32: 色相環の割り当てを限定した場合

第6章　ユースケース

本章では、開発した分析ツールを用いてネットワークの通信ログを分析することで、ネットワークの利用傾向や不正利用を調べる。

6.1　分析に用いるデータ

分析する対象のデータとして、大学の宿舎に引かれているインターネットのアクセスログを用いた。アクセスログは、NetScreen[1]のファイアウォールが記録するログである。1 件の通信ログは、通信に関する様々な情報を持つ多次元データの構造になっている。データの例を表 6.1 に示す。

表 6.1: アクセスログの例

No.	時刻 通信時間	サービス名 プロトコル	送信元 IP 送信先 IP	受信バイト数 送信バイト数	送信元ポート番号 送信先ポート番号
1	1293807603 61	udp 17	192.0.2.5 198.51.100.21	0 152	9826 7140
2	1293807603 62	udp 17	198.51.100.50 203.0.113.125	148 152	1581 11595
3	1293807603 61	udp 17	192.0.2.68 203.0.113.1	556 472	1791 9493
:	:	:	:	:	:

解析用のデータには NetScreen のファイアウォールログを tsv 形式に変換したものを用いる。ファイアウォールログの期間は 2011 年 1 月 1 日から 2011 年 12 月 31 日までで、1 年間で 93 億余りのレコードがある。本ユースケースでは、2011 年 1 月 1 日の 24 時間と、2011 年 1 月 1 日〜2011 年 12 月 31 日の 1 年間を読み込んだ 2 通りの結果から考察を行う。

ネットワークの利用者毎の利用傾向を分析するため、カテゴリデータとして集計する次元には送信元 IP を選んだ。宿舎のネットワークは日が変わるタイミングで全住民の IP が再割当てされる。従って、1 年間のデータを分析する際は、一つのプロットが必ずしも同じ利用者を表している訳ではないことを留意しなければならない。

なお、画像中の IP アドレスは実際のものではなく、ダミーに置き換えて表示している。

[1] Juniper Networks, http://www.juniper.net/

6.2　結果と考察

6.2.1　1 月 1 日の分析

2011 年 1 月 1 日のデータ総数は 2431 万 2976 件で、送信元 IP は 1268 種類であった。

一般的な傾向

図 6.1 は、プロトコル（左）と通信時間（右）を比較したものである。プロトコルの次元では、よく使われるプロトコルであるピンク色で示した tcp と udp の間と、青色で示した dns と http の間に、それぞれ直線のシルエットが見える。この 2 組のプロトコル間の通信に関係があることを示している。また、通信時間からは、4 秒から 2 分までの通信が一般的であることがわかる。ここで、通信時間によって色付けを行う。この場合、通信時間が少ないほど赤く、長いほど黄、緑、青と変わっていく。この着色でプロトコルを見ると、先に挙げたメジャーなプロトコルの中で最も赤いもの、すなわち通信時間の最も短いものが http で、逆に最も長いものが udp であるという傾向がわかる。

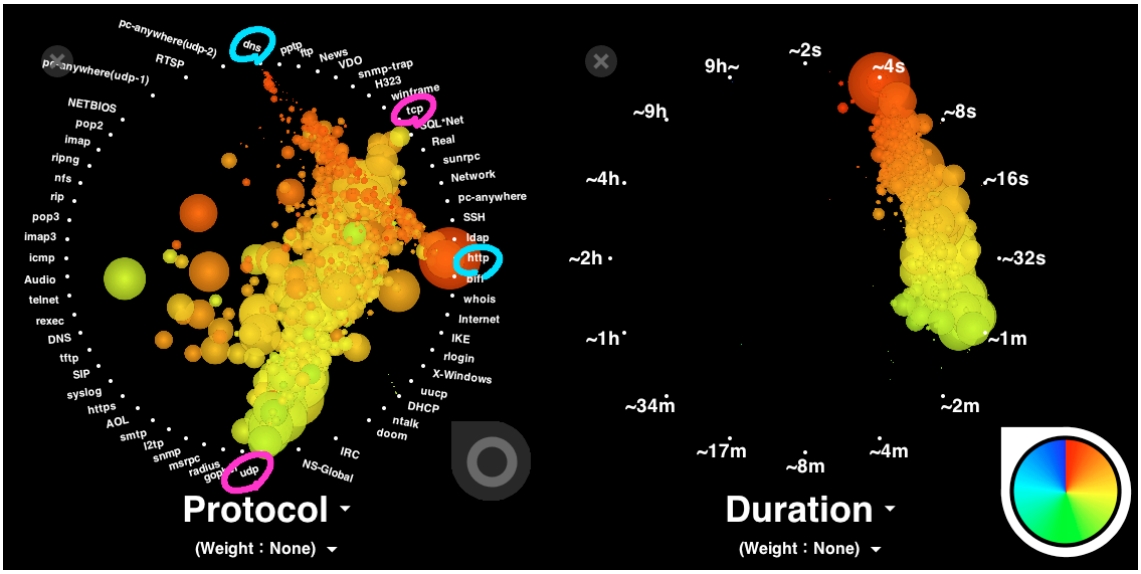

図 6.1: プロトコルと通信時間の関係

不正利用の発見

図 6.2 の赤い丸で示したプロットは、元旦の通信回数が最も多いユーザーを表している。この利用者の通信回数は 170 万回で、その通信は午前 10 時前後に行われたことがわかる。また、毎回 2KB〜4KB を送信し、4KB〜8KB を受信しており、その通信のプロトコルは毎回 http であったことが分かる。この挙動は明らかに不自然であり、この利用者が DoS 攻撃[2]のようなことを行なっていたと考えられる。

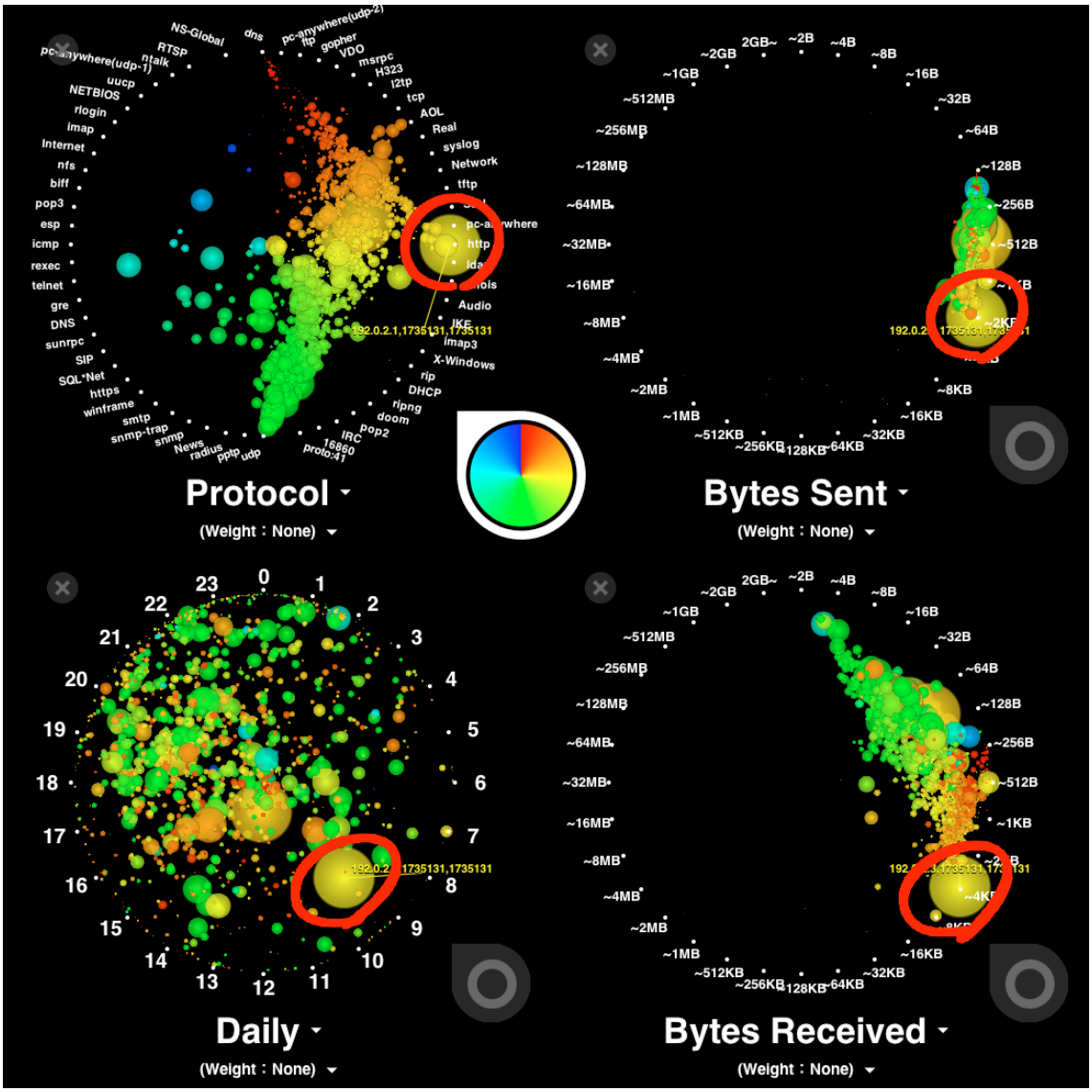

図 6.2: DoS 攻撃の発見

[2]ネットワーク越しのマシンに大量のパケットを送る攻撃方法

6.2.2　2011 年の 1 年分の分析

2011 年の 1 年分のデータ総数は 93 億 6953 万 2312 件で、送信元 IP は 4447 種類であった。本ユースケースで扱うデータは、学生への送信元 IP の割り当てが毎日変わるネットワーク環境における通信ログである。そのため、特定の利用者の不正利用を見つけることは出来なかった。

図 6.3 は、一周を 24 時間として 365 日分のすべてのレコードの重心位置の密度を明度によって表している。全体的な傾向として、最も大きな塊のある位置から、1 日で最もネットワークが利用されるのは 17〜23 時の時間帯であることが分かる。また、円周上にあるプロットは、その時間しかネットワークを利用しなかったユーザーを表している。これを見ると、1 日のうちで 4〜8 時の時間帯のみでネットワークを利用したユーザーはほとんどいないことが分かる。図 6.4 は、時計回りに曜日を並べたものである。最も大きな塊から、曜日によるネットワークの利用頻度の違いがほとんどない事がわかる。しかし、この塊の周囲の少し薄くプロットされた部分を見ると、休日よりも平日の方がインターネットの利用頻度が高い傾向が見える。図 6.4 の黄色い円で囲った位置は、日曜日の深夜から月曜日の早朝の時間帯である。平日の同じ時間帯では、黄色い円の時間帯が最も利用頻度が低い。週明けの

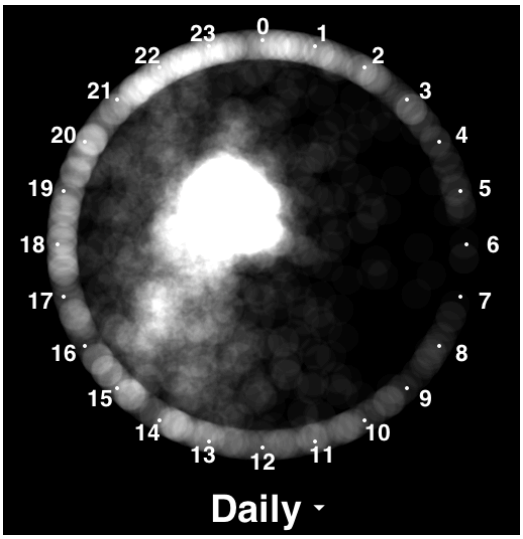

図 6.3: 時間毎の頻度

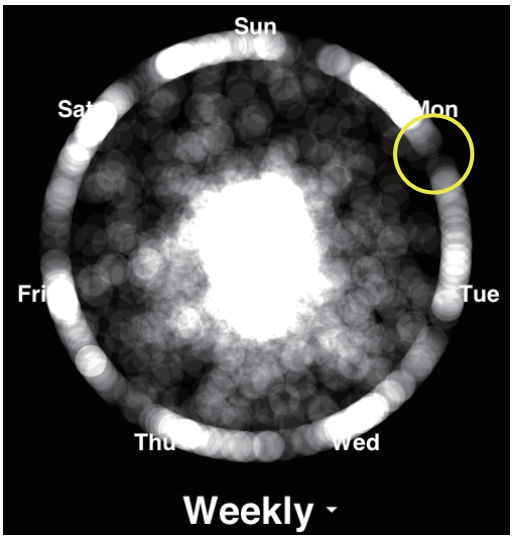

図 6.4: 曜日毎の頻度

第7章　結論

本研究では、レコードが数十億件に上る大規模な多次元データの可視化手法を開発した。この手法を用いると、大規模なデータの特徴を残しつつレコード数を削減し、大規模データを読み込んでいる間にも分析をリアルタイムに行うことが可能となる。

多くの可視化手法では、ファイルの内容をすべてメモリに読み込んでからデータの加工を行い、描画する。しかし、大規模データでは加工のためにすべてをメモリに読み込むことが難しい上、大量のデータをそのまま描画してもデータの傾向を見ることは困難である。本研究では、ある次元の値をカテゴリとしてとらえることで、カテゴリ毎に次元の傾向を算出しながらデータの数を減らし、メモリに乗る容量に抑えつつデータの傾向を表現する手法を提案した。提案した手法を基に、一般的なパーソナルコンピュータで大規模な多次元データを分析できるツールを開発した。

開発したツールを用い、実際に学生宿舎ネットワークの 1 年間のファイアウォールログを分析した。その結果、ネットワークの利用傾向の把握や不正利用と思われる挙動を発見することが出来た。また、本研究で開発したツールで数十億規模のデータを可視化することが出来た。

謝辞

本研究を進めるにあたり、指導教員である三末和男先生をはじめ、志築文太郎先生、高橋伸先生、田中二郎先生には、ゼミなどを通して丁寧なご指導や貴重なご意見をいただきました。深くお礼申し上げます。

また、インタラクティブプログラミング研究室の皆様には、ゼミでの発表や研究室生活の中で、様々な貴重なご意見を頂きました。特に、NAIS チームの皆様には、ゼミ以外でも研究の相談をして頂いたり、研究生活においてもたくさんのご意見やご指摘を頂きました。心よりお礼申し上げます。

参考文献

[1] A. Inselberg, B. Dimsdale. The plane with parallel coordinates. *The Visual Computer*, Vol. 1, No. 4, pp. 69–91, 1985.

[2] A. Inselberg, B. Dimsdale. Parallel Coordinates: A Tool for Visualizing Multi-dimensional Geometry. *Proceedings of the 1st conference on Visualization (VIS '90)*, pp. 361–378, 1990.

[3] Z. Geng, Z. Peng, R. S. Laramee, R. Walker, J. C. Roberts. Angular Histograms: Frequency-Based Visualizations for Large, High Dimensional Data. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 17, pp. 2572–2580, 2011.

[4] J. Heinrich, J. Stasko, D. Weiskopf. The Parallel Coordinates Matrix. *Eurographics Conference on Visualization (EuroVis2012)*, pp. 37–41 2012.

[5] D. B. Carr, R. J. Littlefield, W. L. Nicholson, J. S. Littlefield. Scatterplot Matrix Techniques for Large N. *Journal of the American Statistical Association*, Vol. 82, No. 398, pp. 424–436, 1987.

[6] F. Bendix, R. Kosara, H. Hauser. Parallel sets: visual analysis of categorical data. *Information Visualization (InfoVis2005)*, pp. 133–140, 2005.

[7] J. Hartigan, B. Kleiner. Mosaics for contingency tables. *Proceedings of the 13th Symposium on the Interface*, pp. 268–273, 1981.

[8] F. Fischer, J. Fuchs, F. Mansmann. ClockMap: Enhancing Circular Treemaps with Temporal Glyphs for Time-Series Data. *Eurographics Conference on Visualization (EuroVis2012 Short Papers)*, pp. 97–101 2012.

[9] M. Krstajic, E. Bertini, D. A. Keim. CloudLines: Compact Display of Event Episodes in Multiple Time-Series. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 17, pp. 2432–2439, 2011.

[10] Satoko Shiroi, Kazuo Misue, Jiro Tanaka. ChronoView: Visualization Technique for Many Temporal Data. *16th International Conference Information Visualization*, 2012.

[11] L. Nováková, O. Štěpánková. Multidimensional clusters in RadViz. *SMO'06 Proceedings of the 6th WSEAS International Conference on Simulation, Modelling and Optimization*, pp. 470–475, 2006.

[12] J. Sharko, G. Grinstein, K. A. Marx. Vectorized Radviz and Its Application to Multiple Cluster Datasets. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 14, No. 6, pp. 1444–1451, 2008.