数理解析研究所講究録
第1644巻 2009年 12-19

VAULT: Visualization and Analysis Utility for Large Tree structure

宮村（中村） 浩子
Hiroko Nakamura Miyamura

日本原子力研究開発機構 システム計算科学センター
Center for Science and e-Systems, Japan Atomic Energy Agency

1 背景と目的

分枝限定法を用いて整数計画問題を解く際に，どのような分枝戦略を選択するかは重要な問題である．この分枝戦略が結果に及ぼす影響を把握するには，分枝限定木の生成過程で出力されるログデータを調べることが有効である．そこで著者は分枝戦略が結果に及ぼす影響を直感的に把握するため，分枝限定木の生長過程をツリーグラフで可視化するシステムを提案した [1]．しかし，可視化対象となるログデータが大規模である場合，ツリーグラフはその構成要素であるノードやリンクの数が膨大になる．膨大なノード，リンクから構成されたツリーグラフは，ノードやリンクを表現するプリミティブ同士が重なり合うオクルージョンやプリミティブの近接によるちらつきに代表されるクラッタリングが生じ，データの特徴解析や認識が困難になる（図 1）．さらに描画に膨大な時間を要することもある．

近年，計算機の性能の向上に伴い，出力されるログデータは大規模化しており，前述のような問題は深刻になりつつある．そこで本研究では，ツリーグラフを少ない要素で精度よく可視化することを試みる．その際，ツリーグラフは領域によって構成要素の密集度が異なることに着目し，密集度が高いところではオクルージョンやクラッタリングを考慮した表現形式を，密集度が低いところでは直感的に階層データの構造を認識できる表現形式を採用する．さらに，描画スペースの問題から，重ね描きしなければならないほどグラフの構成要素が密集している領域では，ツリーグラフの要素数を削減する簡略化を施す．

なお，従来のツリーグラフではどの領域でも同一の形式で表現していたのに対し，本手法はツリーグラフの表現形式そのものを密集度に応じて領域ごとに変えて表現する．これは，各領域でユーザが認識できるレベルに合わせて情報を提示するため，ユーザに対して最大限に情報を提示する試みといえる．

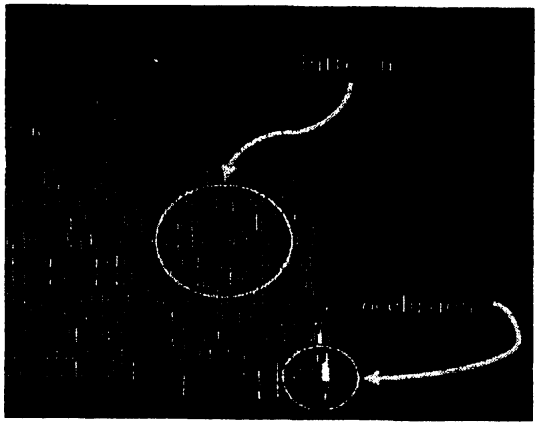


図 1: オクルージョン，クラッタリング

2 ツリーグラフ可視化の先行研究

並列分枝限定法の計算過程を視覚的に捉えるために，Sapal-BB [2] が開発された．このシステムは，生成される分枝限定木を動的に可視化するため，逐一変化する分枝木の生成の様子を観察できる．また，大規模な巡回セールスマン問題を分枝カット法により解いた際の分枝限定木を可視化する Explanation of Branch-and-Cut Tree Pictures [3] も開発された．このシステムでは，各ノードの評価関数値を木構造の高さ方向の座標と対応付けてノードを配置することで，値が大きく変化する枝を発見できる．これらの研究例からもわかるように，分枝限定木の可視化は計算過程を把握する上では有効である．しかし，大規模な分枝限定木を可視化した場合，ノードが混雑してしまうため，可視化結果から計算過程を解析することは困難になる．本研究では，この問題に対処するため，大規模分枝限定木を効果的に可視化するシステムの開発を目指す．

大規模分枝限定木の可視化を実現するためには，大規模ツリーグラフの可視化技術が不可欠である．大規

模ツリーグラフの可視化技術は，バイオインフォマティクスの分野を中心に注目されており，いくつかの取り組みがなされている [4, 5, 6]．例えば，同じ画素に重ね描きする際，描画順を操作し重要な情報が隠れないようにする TreeJuxtaposer [4] が提案された．これはハイライト表示とよばれ，ユーザが指定した領域は他の領域によって背後に隠れないようにする．しかし，すべてのノードを表示できないほど大規模なツリーグラフに対してはハイライト領域を指定すること自体が困難である．さらに，指定したハイライト領域もまた大規模である場合にはオクルージョンやクラッタリングが発生してしまう．

大規模ツリーグラフのすべての要素を表示するのではなく，簡略化によって選択的に表現する方式では，階層データのフラクタル性を利用した Fractal Views [7] が挙げられる．この手法は，大局的特徴と局所的特徴の繰り返し構造を利用して表示するノード数を調整している．これはノード数を一定範囲内に抑えられることから，大規模データを効率的に表示できる．しかし，ツリーグラフのフラクタル性が条件となる．著者らはツリーグラフを特徴に応じて簡略化する手法 [1] を提案し，オクルージョン，クラッタリングの発生を回避した．しかし，簡略化による表示では，表示されていない情報の取得が難しい，簡略化の処理自体に時間を要するなどの問題が残る．

木構造の表現方式自体を工夫する H-tree, radial layout, balloon layout [8]，また，radial layout で作成したツリーグラフを非ユークリッド空間に投影する手法 [9] が提案された．これらは描画空間を効率的に利用しているが，やはり大規模な階層データに対してノードが密集してしまう問題は残る．階層データのリンクの表現形式を工夫する研究も行なわれている．TreeMaps [10] は，ノード間の階層関係を入れ子状で表現し，直接的なリンクの表示を略している．限られた空間に多数のノードを配置できる利点はあるが，ノード間の関係やノードの深さを直感的に捉えにくいだけでなく，上位階層のノードに与えられた情報は示せないため，分枝限定木には不向きである．同様の入れ子による階層データの可視化手法として Data Jewelry Box [11] が提案されたが，TreeMaps と同様にノード間の関係や上位階層の情報を把握しにくい．

ユーザの操作によって選択的に表示する手法も提案されている．例えば，3 次元空間にツリーグラフを配置する ConeTree [12] では，ユーザは必要な階層以下のノードを表示しないで観察することができる．また，radial layout を採用し，リンク情報を領域の隣接状態で示すことで空間効率を高めた Information Slices [13] では，選択したノードの下位階層を新たに新しいグラフとして作成する．これらの手法では，ユーザの選択操作を必要とするが，ユーザに興味ある領域を選択させるためには全体を表示する必要があり，その際にはオクルージョンやクラッタリングが発生する．

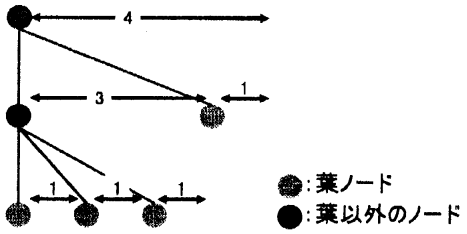


図 2: ノード配置

本研究ではこれら先行研究とは異なり，ひとつのツリーグラフに対して複数のグラフ表現形式を用いて表現する適応的ツリーグラフを採用する．その際のグラフ表現形式は，先行研究の特長を利用する．

3 適応的ツリーグラフ

大規模ツリーグラフを表示するための先行研究では，ノードやリンクを表現するプリミティブの形状やレイアウトを工夫したり，ユーザの対話的操作によって興味ある領域を選択的に表示する方式がとられてきた．これらに対して，本研究では領域ごとに適した表現形式を選択する適応的ツリーグラフを提案する．

まず，オクルージョン，クラッタリングを抑えたツリーグラフの表現を目指していくつかの表現方式を実装する．本システムでは，top-down layout と radial layout を採用し，それぞれに対する効果的なノードの配置とリンクの提示方法を検討する．また，それぞれのツリーグラフの表現形式で，空間効率，情報の提示量について考察する．考察にしたがってノードの密集度に応じて適したツリーグラフの表現形式を選択する適応的表現を提案する．

3.1 ツリーグラフの表現形式バリエーション

ツリーグラフのレイアウトは，top-down layout と radial layout [8] それぞれについて検討する．Top-down layout では，自身より下の階層に存在する葉ノード数分だけ x 軸方向に領域を確保しながら配置する (図 2)．この配置を「始点揃え」とよぶ．始点揃えでは，兄弟ノード間で親子の距離が異なる．そこで親子関係をより明確に示すために，親ノードを子ノードの中心に再配置する．この配置を「中央揃え」とよぶ．

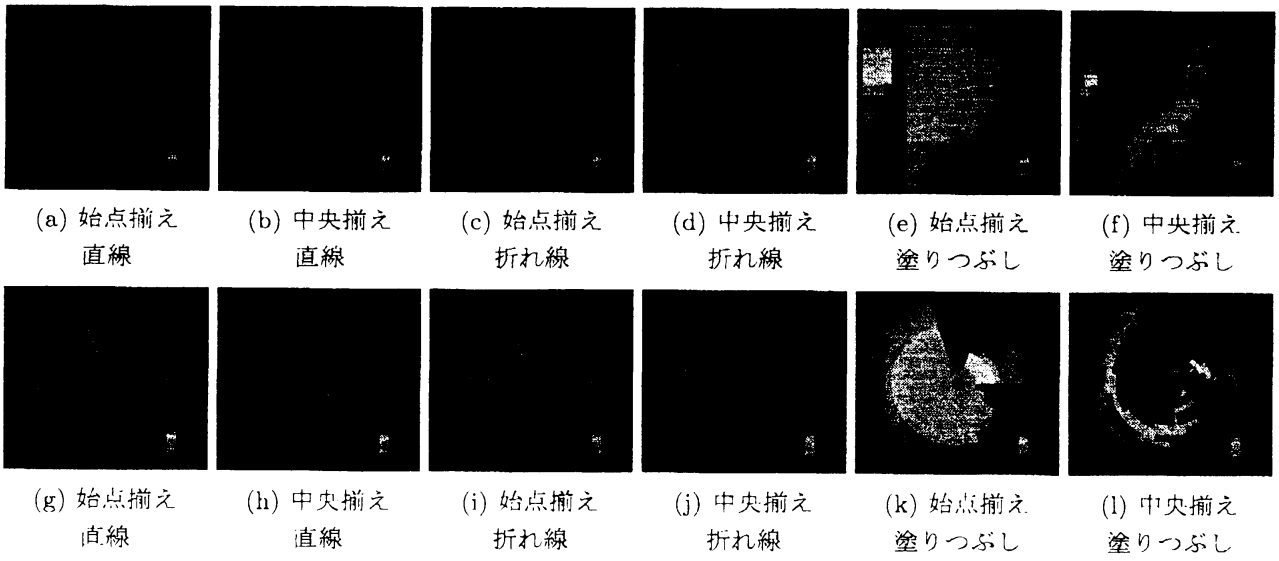
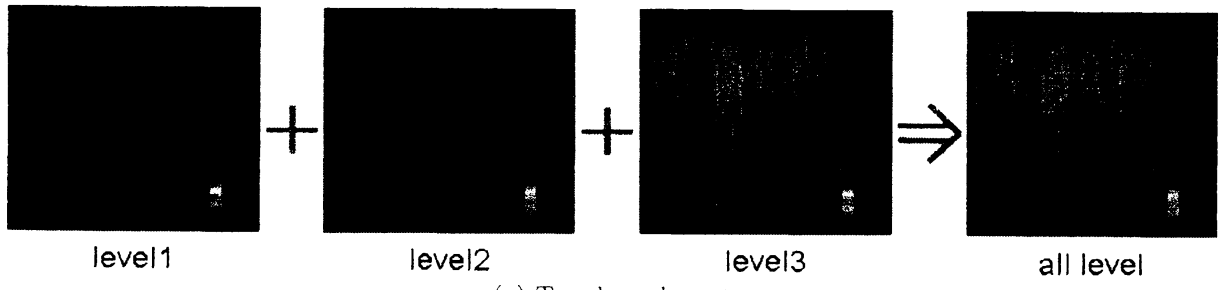
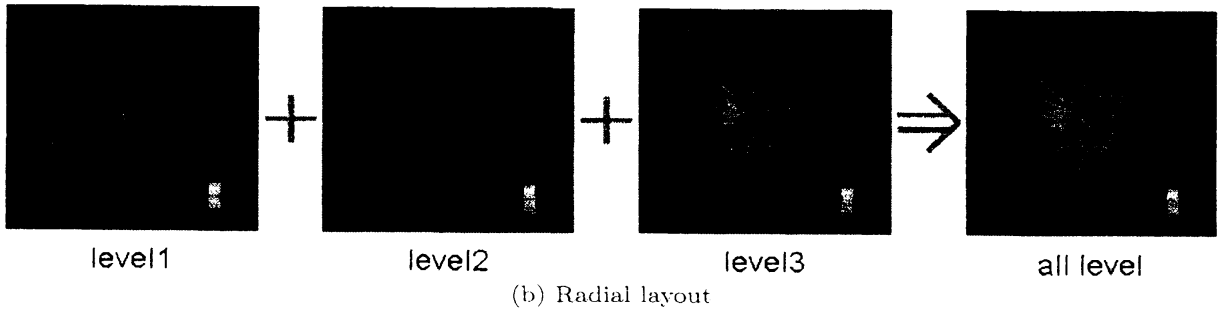


図 3: ノード配置，リンク提示によるツリーグラフの表現形式；上段：top-down layout，下段：radial layout



(a) Top-down layout



(b) Radial layout

図 4: レベル別適応的ツリーグラフ

なお，階層の深さは y 座標で表す．Radial layout では，自身より下の階層に存在する葉ノード数分だけ角度を確保する．中央揃えでは，角度が子ノードの中心になるように親ノードを移動する．階層の深さはルートノードからの距離で表す．

リンクの提示では，一般的にはノード間を直線によって提示する方法が用いられる．この提示方法を「直線」とよぶ．この提示方法では，直感的に親子関係を認識できるが，radial layout に適用するとルートノードを発見しにくいという問題がある．そこで，top-down layout でしばしば用いられる，座標軸に沿った折れ線でリンクを提示する方法を radial layout に適用することを提案する．これによって，空間効率を高めた radial layout でもルートの発見が容易になるだけでなく，同階層のノードの特定も容易になる．この提示方法は「折れ線」とよぶ．また，線を利用することで生

じるクラッタリングを回避するため，線のようなオブジェクトを用いず，領域の塗りつぶしからリンクを提示する「塗りつぶし」も用いる．

3.2 ツリーグラフの表現形式による効果

Top-down layout，radial layout それぞれに対して，ノードは始点揃え，中央揃えで配置する．リンクは，直線，折れ線，塗りつぶしで提示する(図 3)．ノード配置に関しては，中央揃えでは親子間の距離が短くなるため，親子関係の把握を助ける．また，リンクが短くなるため表示空間内の混雑が緩和できる．リンクの提示に関しては，直線による提示では一般的に大規模ツリーグラフを表現する際にクラッタリングが発生する．データが大規模である場合には折れ線を採用することで，クラッタリングの発生が抑制できる．radial layout ではルートノードを把握しやすい特長も併せもつ．塗りつぶしによる親子関係の提示では，クラッタリングを抑えられるだけでなく，色の変化が大きいノードの発見が容易である．これは，スカラデータの変動に着目しながら階層データを観察する際に有効といえる．

3.3 密集度に応じたグラフ表現形式の選択

前項で議論したツリーグラフの表現形式による特長をふまえ，ノードの密集度に応じてグラフ形式を選択する(図 4)．レベル 1 では，隣接するノード間を識別できるだけのスペースが確保できることを条件とし，隣接するノードを識別するだけのスペースを確保できない領域はレベル 2 とする．さらに，ノードを重ねずに描画できない領域をレベル 3 とする．これらのレベルに対して，適切なツリーグラフの表現形式を選択する．

レベル 1 では，階層方向の密集度によって 2 種類のグラフを使い分ける．親子ノード間にスペースを確保できる場合は，ノードを点，リンクを線で表現する形式(図 3(a–d, g–j))を利用し，スペースが確保できない場合は，塗りつぶしの中央揃え(図 3(f, l))を適用する．次に，レベル 2 では，クラッタリングやオクルージョンを抑えるため，塗りつぶし始点揃え(図 3(e, k))を用いる．最後に，レベル 3 は各ノードに領域を割り当てられないので統合して表現する．ここでは，親ノードが確保する領域を調べ，描画領域が確保できるノードまで階層を順々上げながら調べる．領域が確保できるノードに到達したら，そのノード以下の部分木は上位階層との値の差が大きいノードに同階層のノードを統合する(図 5)．ただし，最適解が求まったノードが存在する場合には，そのノードにマークをつけた上で同階層のノードを統合する．これによって，最適解が求まったノードは常に表示されるようにする．

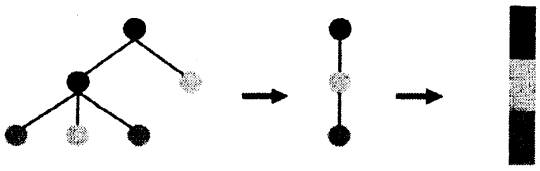
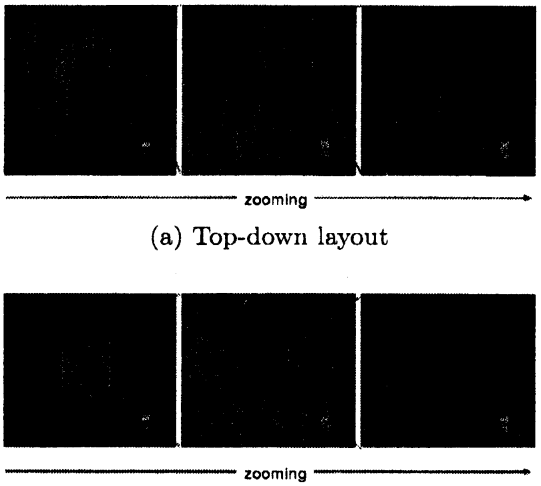


図 5: 部分木の簡略化



(a) Top-down layout

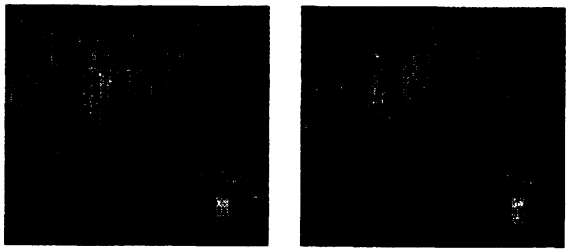
(b) Radial layout

図 6: ズームによる表現形式の変更

このようにして，ノードの密集度に応じて表現形式を 3 レベルに分け，それぞれの領域でできるだけ有効にスペースを利用したツリーグラフの表現形式を利用する．また，最適解のような重要な情報の提示は保証する．なお，ズーミング機能などによって対話的に観察している間，各領域でツリーグラフの表現形式は変えて表現する(図 6)．

3.4 詳細情報の提示

描画要素が密集している領域では，複数のノードが統合されて描画される．このような領域に焦点を当てて観察したい場合は，ズーミング機能によって拡大することで，充分な描画領域を確保することができる．しかし，ズーミングすることによって，木構造の一部しか描画画面内に表示されなくなり，全体像と共に注目領域を観察することが難しくなる．そこで，ノード



(a) 全体表示　　(b) 選択領域の拡大

図 7: 注目ノードの選択と拡大

の選択機能，選択ノードの復元機能，選択ノード周辺の拡大機能を追加する．

図 7 に選択の様子を示す．全体像（図 7(a)）から例えば最適解が求まっているノードを選択する．選択されたノードの詳細な情報はテキストとして取得でき，また選択ノードからルートノードまでの経路はハイライト表示される．このとき，選択ノードから親ノードまでの経路上のノード，および選択ノードの子孫ノードが統合されていた場合（レベル 3 領域であった場合），木構造表現（レベル 1 の表現形式）で表示しなおす．これによって，注目ノード，およびその先祖，子孫ノードの情報損失を抑える．さらに，これらの領域の情報を把握しやすくするために，選択領域周辺を拡大する（図 7(b)）．

3.5 並列環境での実行結果

本システムは，ユーザがクライアント PC でログデータを解析することを前提としている．しかし，整数計画問題の計算に関しては並列環境でも行なわれることを考慮しており，サブツリーに分割された分枝限定木にも対応する．まず，サブツリーをノードとして木構造を生成する．その後，各サブツリー内で木構造を生成する．このとき，各サブツリーでは，サブツリー内の葉ノード数を保持する．この葉ノード数を，サブツリーの親子関係から上位階層に送ることで各ノードの葉ノード数を更新する．そして，これによって得られた葉ノード数を用いてツリーグラフを描画する．

今後，一度に読み込めないような大規模ログデータが生成された際には，まずサブツリーをノードにするようなツリーグラフを表示し，次にユーザが指定したサブツリーの情報を読み込んでツリーグラフを表示するなど工夫する必要がある．

4 評価実験

実験環境は，CPU: Pentium M 1.6GHz, RAM: 1GB，表示サイズは縦 400，横 300pixel である．またユーザインタフェースの構築には AutoGL [14] を使用している．ここでは，ノード間隔 3 画素以上をレベル 1 とし，1 画素以上をレベル 2，それ未満をレベル 3 として適応的表示を用いた．

4.1 完全二分木構造データでの評価

完全二分木構造のツリーグラフを作成し，グラフのサイズと描画時間の関係を評価する．ノードを四角形，リンクを線で描画する場合（図 3(a) の方式）と適応的に表現する提案法で描画した場合の描画時間を表 1 に示す．従来の表示法の場合，描画時間はノード数に比例するのに対し，提案法では描画プリミティブがある一定量からほとんど増加しなくなるため，描画時間を抑えられる．あらかじめ簡略化した木構造データを用意しておく場合には，再描画の度に簡略化操作が必要なくなるため，リアルタイム描画が可能となり，対話的操作によるデータ探索もできる．

4.2 分枝限定木への適用

図 8 に ILOG-CPLEX ソルバ [15] により，整数計画問題を解いた際に生成される分枝限定木（ノード数 : 842,827，階層数 : 176）を可視化した結果を示す．ノードの色は評価値を表している．また，赤線による矩形は解が求まったノードを表している．図 8(a, c) はすべての領域を点と線で表現し（図 3(b) の形式），図 8(b, d) は提案する適応的な表現を用いた結果である．図 8(a, c) では中位階層でリンクを示す線による模様が生じ，値の変動を捉えにくい．また，描画順に依存して表示されないノードが存在する．提案手法では，ノードが密集する領域ではリンクを表示しないため，リンクがノード値の変動の把握を妨げることはない．また，値の変動が大きいノードの表示は保証される．図 8(a) では矢印位置に存在する値の高いノードが表示されていないが図 8(b) では表示されている．このように相殺される情報を把握しながら観察できる．さらに，上位階層のように表示スペースに余裕がある場合はリンクが表示されるので，親子関係を的確に捉えられる．

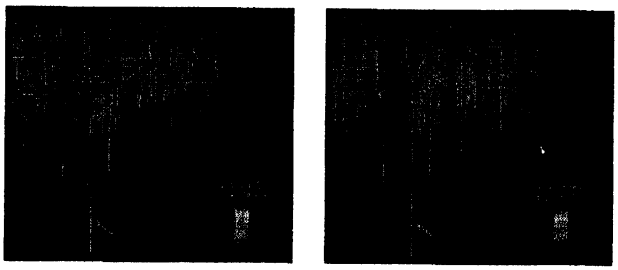
描画プリミティブ数は，図 8(a) は長方形 842,827 個，直線 842,826 本であるのに対し，図 8(b) は，長方形 21,955 個，直線 487 本である．描画時間も 7.670 秒から 0.447 秒（簡略化に必要な時間を含む）に短縮で

表 1: ノード数と描画時間，簡略化処理時間（CPU 秒）

階層数	ノード数	処理時間 描画	処理時間 簡略化	描画プリミティブ数 四角形	描画プリミティブ数 直線
従来法によるツリーグラフ表示 (図 3(a) の形式)					
5	31	0.00	-	31	30
10	1,023	0.03	-	1,023	1,022
15	32,767	0.27	-	32,767	32,766
16	65,535	0.53	-	65,535	65,534
17	131,071	1.06	-	131,071	131,070
18	262,143	2.14	-	262,143	262,142
19	524,287	4.16	-	524,287	524,286
20	1,048,575	8.83	-	1,048,575	1,048,574
21	2,097,151	16.47	-	2,097,151	2,097,150
22	4,194,303	35.25	-	4,194,303	4,194,302
階層数	グラフノード数	処理時間 描画	処理時間 簡略化	描画プリミティブ数 四角形	描画プリミティブ数 直線
適応的ツリーグラフ表示					
5	31	0.00	0.00	31	30
10	1,023	0.02	0.00	767	254
15	32,767	0.03	0.00	2047	254
16	65,535	0.03	0.00	2303	254
17	131,071	0.04	0.01	2559	254
18	262,143	0.04	0.05	2815	254
19	524,287	0.04	0.11	3071	254
20	1,048,575	0.05	0.27	3327	254
21	2,097,151	0.05	0.56	3583	254
22	4,194,303	0.05	1.33	3839	254

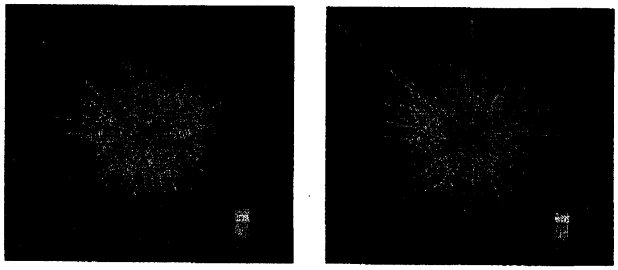
きた．

また探索戦略による違いを ILOG-CPLEX ソルバ [15]mas76 のログデータを使用して比較する（図 9）．整数計画問題は，実行可能解（整数条件を含めて条件を全て満足する解）を得ることですら NP-困難な問題である．一般に，分枝限定法において実行可能解の発見を優先して探索すると，探索される領域は広くなり最適性保証のためには計算時間がかかる．逆に，最適性の保証を優先するように探索すると，一般に各ノードでの処理時間が長くなり，さらに実行可能解が見つかりにくい．どのような探索戦略が適しているかは，解く問題に依存する．そこで，CPLEX では，探索戦略を実行時に指定できる．図 9 は，同一の問題を異なる探索戦略を用いて解いた結果である．図 9(a) は，実行可能解の発見に重点を置いた探索戦略を取ったとき，図 9(c) は，最適性の保証に重点を置いた探索戦略を取ったときの結果である．図 9(b) はそれらのバランスを取った探索戦略の結果である．解いている問題は最小化問題で，赤色側が目的関数値が大きく，悪い評価値を与えたノードである．図 9(a) では，実行可能解の発見を優先しているため探索ノード数は多く，深さの浅い部分で赤や黄色の部分が目立つ．分枝限定木の浅い部分でも実行可能解を発見できているが，その評価値は悪い．一方，図 9(c) では，分枝限定木の浅い部分での赤や黄色の部分はあまり見当たらず，確実に評価値の良いノードを優先して探索していることがわかる．最後に，バランスを取った探索戦略による結果（図 9(b)）は，前者 2 つを融合した探索戦略であることが読み取れる．なお，図中の赤枠で囲まれたノードは発見した実行可能解を示している．



(a) 従来法　　(b) 提案法

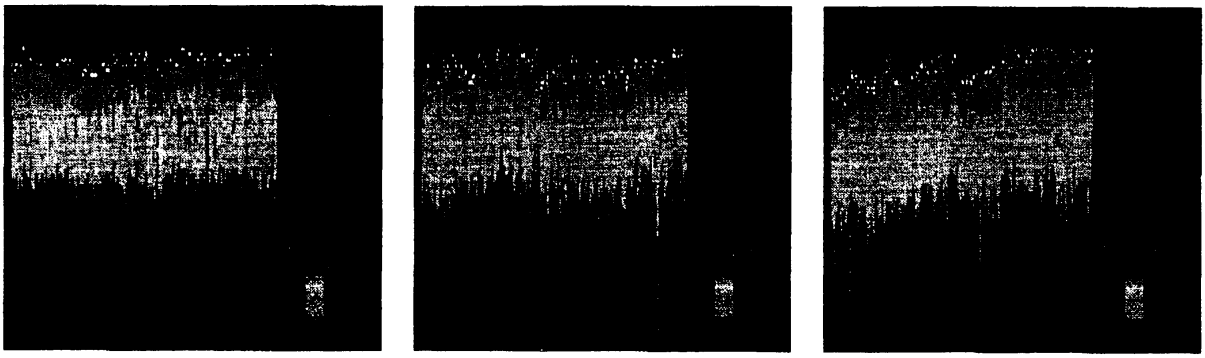
top-down layout によるツリーグラフ表示



(c) 従来法　　(d) 提案法

Radial layout によるツリーグラフ表示

図 8: 従来法と適応的ツリーグラフとの比較



(a) 許容解優先型
emp1
973,237 ノード

(b) バランス型
emp0
375,729 ノード

(c) 最適性保証優先型
emp3
747,953 ノード

図 9: 分枝戦略の違いによる違いの比較

5 まとめ

大規模な分枝限定法の計算過程を可視化するシステムを提案した．大規模ツリーグラフを効果的に表示するために，ツリーグラフの表現形式をいくつか実装し，ノードの密集度に応じて適応的に使い分ける表現方式を提案した．この適応的ツリーグラフは，ユーザがズーミングなどの操作で表示環境を変更したときにも逐一更新するため，注目箇所は木の形状で観察できる．さらに，注目する領域に対して詳細な情報を提示するため，ユーザが選択したノードとその先祖，子孫ノードに対して，ノード間の親子関係も把握できるように表現しなおす機能を追加した．これによって，従来のズーミング機能では大局的かつ局所的情報を同時に把握できなかった問題を解決した．

本提案手法は，大規模分枝限定木に適用し，情報の提示精度，描画時間の側面から評価し，検証した．その結果，表示精度，描画時間双方で，すべての領域を単一の表現形式で表示する方式よりも優れていることが確認できた．

今後の課題としては，一度に読み込むことができないほど大規模なデータの可視化が挙げられる．並列環境で計算されたログデータに関しては，サブツリーを先にツリーグラフで表示し，必要に応じて個々のサブツリー情報を読み込むことで対処する予定である．しかし，さらに大規模化が進捗とが予測される．その場合には，本手法を応用的に利用して前処理として簡略化ツリーグラフを作成し，これを入力データとする方法を取ることで解決できると考えられる．ただし，ズーミングによって詳細情報を提示するときに，ズーミングの度に元データに戻って情報を取得する必要があり，処理時間に問題が生じる．そこで，適切な詳細度変更の方式の確立を検討する必要がある．

謝辞

本研究を進めるにあたり，貴重なご意見をいただいた京都大学 永持 仁 教授に深く感謝いたします．本システムを開発するにあたり，ログデータを提供してくださり，有意義な議論をしてくださった東京農工大学 品野 勇治 准教授，宮代 隆平 助教に深く感謝いたします．可視化手法の開発にあたり，貴重なご意見をいただいた東京農工大学 斎藤 隆文 教授，満倉 靖恵 准教授に深く感謝いたします．

参考文献

[1] 宮村（中村） 浩子，品野 勇治，宮代 隆平，斎藤 隆文：分枝限定法における分枝戦略選択のための計算過程の可視化，情報処理学会論文誌 数理モデル化と応用（TOM23）（掲載予定）2009.

[2] 大西 克実，榎原 博之，中野 秀男：並列分枝限定法に対するビジュアリゼーションシステム，電子情報通信学会論文誌，vol. J79-D-1，no. 7，pp. 400–408，1996.

[3] W. Cook: Traveling Salesman Problem, Sweden Home,
http://www.tsp.gatech.edu/sweden/

[4] T. Munzner, F. Guimbertiere, S. Zhang, and Y. Zhou: "TreeJuxtaposer: Scalable Tree Com-

parison Using Focus+Context with Guaranteed Visibility," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2003)*, vol. 22, no. 3, pp. 453–462, 2003.

[5] C. M. Zmasek and S. R. Eddy: "ATV: Display and Manipulation of Annotated Phylogenetic Trees," *Bioinformatics*, vol. 17, no. 4, pp. 383–384, 2001.

[6] U. Rost and E. Bornberg-Bauer: "TreeWiz: Interactive Exploration of Huge Trees," *Bioinformatics*, vol. 18, no. 1, pp. 109–114, 2002.

[7] H. Koike: "Fractal Views: A Fractal-Based Method for Controlling Information Display," *ACM Transactions on Information System*, vol. 13, no. 3, pp. 305–323, 1995.

[8] E. M. Reingold and J. S. Tilford: "Tidier Drawings of Trees," *IEEE Transactions on Software Engineering*, vol. 7, no. 2, pp. 222–228, 1981.

[9] J. Lamping, R. Rao, and P. Pirolli: "A Focus+context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies," In *Proceedings of SIGCHI '95: the Conference on Human Factors in Computing Systems*, pp. 401–408, 1995.

[10] B. Johnson and B. Shneiderman: "Treemaps: A Space-filling Approach to the Visualization of Hierarchical Information," In *Proceedings of IEEE Visualization '91*, pp. 284–291, 1991.

[11] T. Itoh, Y. Yamaguchi, Y. Ikehata, and Y. Kajinaga: "Hierarchical Data Visualization Using a Fast Rectangle-Packing Algorithm," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 3, pp. 302–312, 2004.

[12] G. G. Robertson, J. D. Mackinlay, and S. K. Card: "ConeTree: Animated 3D Visualizations of Hierarchical Information," In *Proceedings of SIGCHI '91: the Conference on Human Factors in Computing Systems*, pp. 189–194, 1991.

[13] K. Andrews and H. Heidegger: "Information Slices: Visualizing and Exploring Large Hierarchies Using Cascading, Semicircular Discs," In *Proceedings of IEEE Information Visualization '98*, pp. 9–12, 1998.

[14] H. Kawai: "ADVENTURE AutoGL: A Handy Graphics and GUI Library for Researchers and Developers of Numerical Simulations," In *Computer Modeling in Engineering and Sciences*, vol. 11, no. 3, pp. 111–120, 2006.

[15] ILOG: CPLEX: `htpp://www.ilog.co.jp/`