階層型時系列タスクデータの可視化の一手法

内田悠美子　(指導教官：伊藤貴之)

1. 概要

情報可視化とは実体を持たない抽象的なデータを，コンピュータグラフィックスで表現することにより，ユーザの情報認識を手助けする技術である．文字や数値のみで表された情報より，直感的にデータを認識することが出来るのが利点である．

情報可視化では，階層型データや時系列データが対象となることが多い．ここで階層型データとは，各要素がグループ化され，木構造を形成しているデータを指す．また時系列データとは，ある現象の時間的変化を記録したデータを指す．一般的に，文字情報として記述された時系列データから時間経過に伴う数値変化を読み取るのは難しい．よって時系列データの可視化は，意義が大きいと言える．本報告では階層構造と時系列情報を共に持つデータを階層型時系列データと呼び，その可視化を試みる．また階層型時系列データの一例として，タスクデータの可視化を試みる．

本報告ではタスクデータの各タスクを木構造の葉，タスクをグループ化したものを中間ノードとすることで，階層型データとして表現する．本報告では，タスクデータを格納する表をタスク表と呼ぶ．多くの企業では，タスクの分担やそれぞれのタスクの開始日時，終了日時，進捗度などをタスク表に記録することで，業務を円滑に管理する．また並列計算の実行状況なども，タスクデータとして取り扱うことが可能である．本報告では，上記のタスクデータの階層構造と時系列情報を同時に表現するための手法を提案する．本システムの利用目的としては，タスク表において進捗に遅れの出ている部門の発見や，並列計算のスケジューリングが適切かどうかの判断，などが挙げられる．さらに目標としては，さまざまな状況を想定したタスクスケジュールのシミュレーションなども可能にしたい．

2. 関連研究

以前よりプロジェクト管理や生産管理などには，タスクを帯グラフで表現する Gantt Chart[1]を利用した工程管理図が用いられており，Gantt Chart を発展させた研究も数多く行われている[2]．また，帯グラフと折れ線グラフ，ネットワークを示す図を同時にディスプレイに表示することにより，時系列情報と階層構造を表現する手法などがある[3].

平安京ビュー[4]は階層型データの葉ノードをアイコンで，階層構造を長方形の枠で表示する可視化手法である．階層構造を表しながら，一画面内に多くのノードを表示出来るので，大規模な階層型データを可視化するのに適している．

そこで本報告の提案手法では，既存の研究と同様に帯グラフを用いてタスクの時系列情報を表し，平安京ビューと同様に長方形の入れ子を用いることで階層構造を表すものとする．

3. 提案内容

3.1 データ構造

提案手法で扱うデータの一例を，表 1 に示す．この例では，ユーザが指定した階層構造として，

Project ⊃ Function ⊃ Task ⊃ STask

という包含関係がある．提案手法では，各タスクの最小単位（表 1 の場合 STask）を，階層構造の葉と称し，各タスクの上位単位(表 1 の場合 Project, Function, Task）を階層構造の中間ノードと称する．

表 1：タスク表の一例

タスク	消化 (%)	予定 開始日	予定 終了日
Project		2006/08/01	2006/08/31
Function1		2006/09/01	2006/11/30
Task1.1		2006/09/01	2006/11/30
STask1.1.1	40%	2006/09/01	2006/11/30
STask1.1.2	60%	2006/10/01	2006/11/30
Task1.2		2006/10/01	2006/10/31
STask1.2.1	80%	2006/10/01	2006/10/31
Function2		2006/08/01	2006/08/31
Task2.1		2006/08/01	2006/08/31
STask2.1.1	100%	2006/08/01	2006/08/31
：	：	：	：

3.2 データの表現形式

提案手法では，所要期間(表 1 の場合，開始日～終了日までの日数)に比例した長さを持つ帯グラフとして葉を表現し，帯グラフを包括する長方形の枠で中間ノードを表現するものとする．

提案手法の処理手順は以下の通りである．まず，画面の水平方向を時間軸とし，時間軸に平行に基準軸を設ける．そして基準軸の上に順に，重ならないように帯グラフを配置していく．次にそれらを囲うように長方形を形成する．この作業を再帰的に繰り返すことでデータ全体の配置を決定する．

3.3 長方形の配置方法

以下，すでに配置されている長方形 R_j の頂点の座標を図 1 のように表すとする．また，これから配置する長方形 R_i の頂点の x 座標値はタスクの始点と終点（表 1 の場合，開始日と終了日）から求めることが出来る．

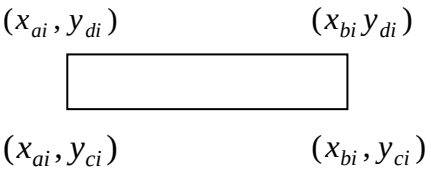

図１：　長方形の頂点の座標

ここで R_i と干渉する可能性のある R_j は，頂点の座標値について $x_{aj} < x_{bi} \cap x_{ai} < x_{bj}$ …(1)を満たすもののみである(図 2).

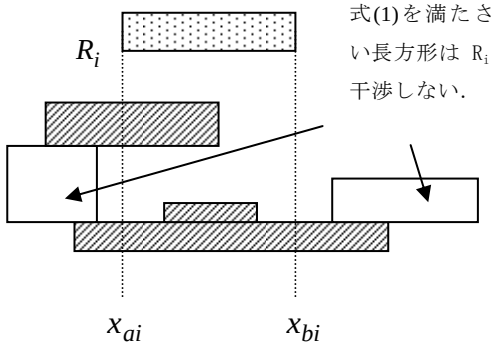

図２：　R_i と干渉する可能性の有無の判定

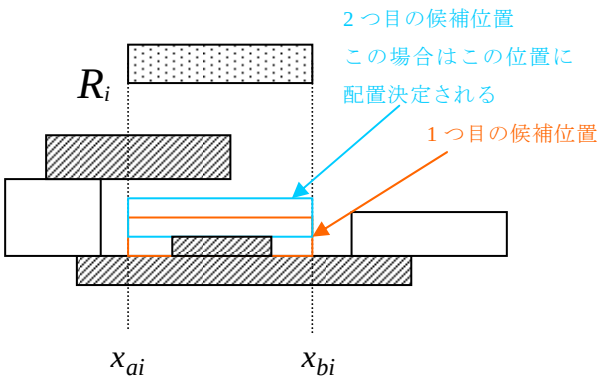

図３：　他の長方形と干渉しない位置の探索

このような R_j が k 個あったとき，y_{cj} を小さい順にソートしたものを $y'_{c1}, y'_{c2}, \cdots y'_{ck}$ とする．同様に y_{dj} を小さい順にソートしたものを $y'_{d1}, y'_{d2}, \cdots y'_{dk}$ とする．

$s = 1$ から順に

ⅰ)　R_i の下辺の y 座標を $y_{di} = y'_{ds}$ として仮配置する．

ⅱ)　$y'_{cj} < y_{di} \cap y_{ci} < y'_{dj}$ を満たす R_j があるかどうか調べる．

・1 個でもある場合には R_i は既に配置された長方形と干渉していると判定し，s を 1 増やしてⅰ)へ戻る．

・1 つもない場合には $y_{di} = y'_{ds}$ として長方形の配置を決定する(図 3).

4. 実行結果

提案手法によってタスク表を可視化した実行結果を，図 4 に示す．ここではタスク表に記述された消化率に比例した長さで，各々の帯グラフを 2 分割している．そして消化率に比例する部分に明度の高い色，他方の部分に明度の低い色を割り当てることで，消化率をパロメータ表示している．また，進捗度という概念を導入し，進捗度から帯グラフの色相を算出している．ここで進捗度とは，タスクの所要期間とタスクの開始日から現在までの経過日数の比として求めた理論上の消化率と，タスク表に記述された実際の消化率との比である．進捗度が低ければスケジュールに遅れが生じていることを示している．この例では進捗度の高いタスクほど青みの強い色相で，進捗度の低いタスクほど赤みの強い色相で表示されている．

提案手法を用いることにより，どの部門に遅れが生じているかを一目で判断することが出来る．

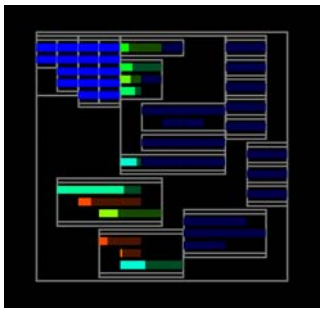

図４：　タスク表の可視化結果

5. まとめと今後の課題

本報告では，階層型時系列タスクデータを可視化する一手法を提案した．今後はタスクの期間を変更した場合に，長方形群を出来るだけシームレスに配置移動するための手法を提案したいと考えている．さらに前後関係を持ったタスクを可視化し，タスクの期間変更があった場合に，前後関係に従った長方形群の配置移動を可能にしていきたい．

参考文献

[1] Clark, Wallace,　The Gantt Chart, a Working Tool of Management, second edition,　Sir Isaac Pitman & Sons, Ltd., London, 1942.

[2] Jeffrey W. Herrmann, A History of Decision-Making Tools for Production Scheduling,　Multidisciplinary Conference on Scheduling: Theory and Applications, July 18-21, 2005.

[3] Russell, A.D., Udaipurwala, A., Construction Schedule Visualization, Proceedings of the International Workshop on Information Technology in Civil Engineering, 2-3 November 2002, pp.167-178, 2002.

[4] 伊藤，山口，小山田，長方形の入れ子構造による階層型データ視覚化手法の計算時間および画面占有面積の改善，可視化情報学会論文集, 26, 6, 51-61, 2006.