数理解析研究所講究録
第 1799 巻 2012 年 118-122

8 次格子モデルによる表の行/列操作

日本大学　髙加 晋司 (Shinji Koka)
Nihon University

電気通信大学　後藤 隆彰 (Takaaki Goto)
The University of Electro-Communications

東洋大学　土田 賢省 (Kensei Tsuchida)
Toyo University

電気通信大学　西野 哲朗 (Tetsuro Nishino)
The University of Electro-Communications

日本大学　夜久 竹夫 (Takeo Yaku)
Nihon University

概要

不均一矩形分割のための 8 次格子グラフモデルに基づく 1 行削除，複数行削除，1 列削除，複数列削除のアルゴリズムについて述べ，計算時間の評価を行う．

1 はじめに

表編集ソフトウェアは数多く存在する．これらのソフトウェアの編集操作ではユーザが予期しない動作を引き起こすことがある．また多くの計算時間を必要としていると思われる．このため，信頼性の高い効率的な表（矩形分割）編集処理の実現が望まれている．そこで本論文では，表編集に適したグラフモデルの導入と，そのグラフモデルに基づく効果的な表編集アルゴリズムを提案する．

R. A. Finkel と J. L. Bentley[1]は，1974 年に 4 分木を導入した．さらに，K. Kozminsky と E. Kinnen[2]は，1985 年に矩形分割のデータ構造である双対グラフの性質を導入している．Yaku 等(例えば，[3], [4], [5])は，8 次格子グラフの導入と，それを用いた壁の移動，セルの合併と 1 つの列の挿入などのシンプルなアルゴリズムを提案している．しかし，必要な操作の中で，まだ研究されていないものがある．

そこで，本論文では，必要な表編集アルゴリズムを提案し，計算時間の評価を行う (*cf.* [6], [7]).

本稿は，2 節で準備として 8 次格子グラフモデルについて解説し，3 節で，表編集アルゴリズムを提案し，計算時間の比較を行う．

2 準備

2.1 矩形分割 (例えば，[2])

共通部分のない幾つかの矩形による平面上の矩形領域の分割のことを，**矩形分割**という．矩形は 2 種類に分類され，分割されていない矩形を**セル**と呼び，行もしくは列を表すための周辺の矩形を**周辺セル**と呼ぶ．また，セルの境界を形成している線を**罫線 (壁)** という．1 つのセルに対して，上下左右に位置する壁をそれぞれ**北壁** (*nw*), **南壁** (*sw*), **西壁** (*ww*), **東壁** (*ew*) という．

図 1 は，矩形分割の例である．太線の矩形 1 つが**内部セル**であり，太線で形成している矩形の周りに存在する矩形が行や列を表す周辺セルである．図 1 の数値は“座標値”である．例えば，セル*c*の北壁，南壁，西壁，東壁は，それぞれ 0，2，0，2 である．

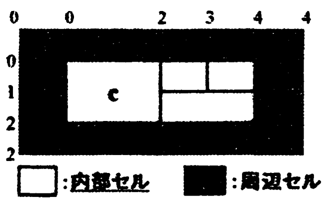

図 1：矩形分割の図.

2.2 8次格子グラフ[5]

矩形分割から図 2 のように構成されるグラフを，矩形分割に対する 8 次格子グラフと言い，以下で定義する．

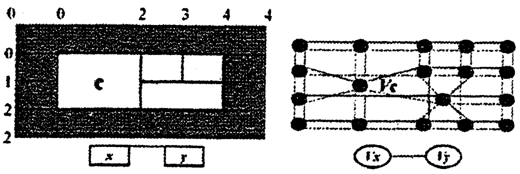

図 2:矩形分割（左）に対する 8 次格子グラフ（右）．

定義 2.2.1

Dを矩形分割とする．**Dに対する 8 次格子グラフ**は，多重無向グラフ$G_D=(V_D,L,E_D,A_D,\alpha_D)$である．ただし，$V_D$は内部セルもしくは周辺セルを表し，セル$c$はノード$v_c$に対応している．$L$は，辺のラベル集合として，$L=\{enw,esw,eew,eww\}$とする．$E_D$:$E_D\subseteq V_D\times L\times V_D$は，ラベル付きの無向辺の集合である 6. ただし頂点v_cとv_d，ラベルlに対して，$[v_c,l,v_d]$と表し，E_Dは次のルール 1~4 によって定義する．

ルール 1

$nw(c)=nw(d)$（セルc，dの北壁が共通）で，セルc，dが最も近い位置にあるとき，辺$[v_c,enw,v_d]$はE_Dに属し，**北壁辺**と呼ぶ．

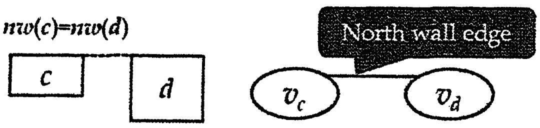

図 3：北壁辺の接続．

ルール 2

$sw(c)=sw(d)$（セルc，dの南壁が共通）で，セルc，dが最も近い位置にあるとき，辺$[v_c,esw,v_d]$はE_Dに属し，**南壁辺**と呼ぶ．

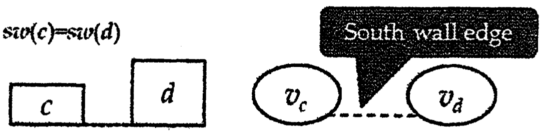

図 4：南壁辺の接続．

ルール 3

$ew(c)=ew(d)$（セルc，dの東壁が共通）で，セルc，dが最も近い位置にあるとき，辺$[v_c,eew,v_d]$はE_Dに属し，**東壁辺**と呼ぶ．

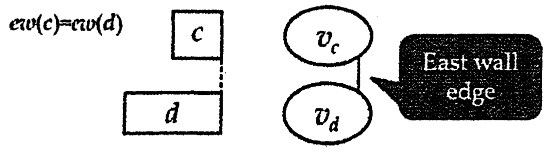

図 5：東壁辺の接続．

ルール 4

$ww(c)=ww(d)$（セルc，dの西壁が共通）で，セルc，dが最も近い位置にあるとき，辺$[v_c,eww,v_d]$はE_Dに属し，**西壁辺**と呼ぶ．

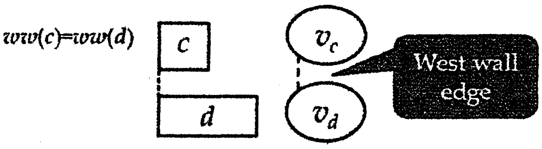

図 6：西壁辺の接続．

そして，A_Dは属性集合$\subseteq R^4$（セルの左上隅のxy座標と，幅と高さの集まり）であり，α_Dを頂点に属性を持たせる写像」とし，$\alpha_D=(nw(c),sw(c),ew(c),ww(c))$とする．

8 次格子グラフの頂点の次数は最大で 8 であることに注意する．

グラフGが **8 次格子グラフ**であるとは，Gに対応する矩形分割が存在すると定義する．

2.3 8次格子グラフの実装[4]

8 次格子グラフの実装の場合，H3CODE と呼ばれるファイル形式が導入され，図 7 に示すのが H3CODE のリスト構造である．

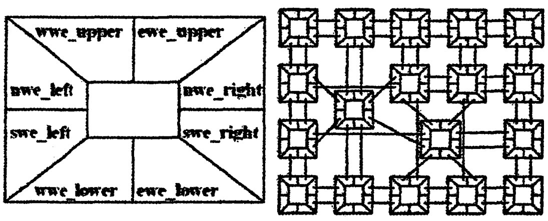

図 7：H3CODE のレコード（左）とリスト（右）．

3 準備

3.1 1行削除アルゴリズム

ここでは，焦点のセルと北壁を共有する行

を削除するアルゴリズムを示す．図 8 がその例である．

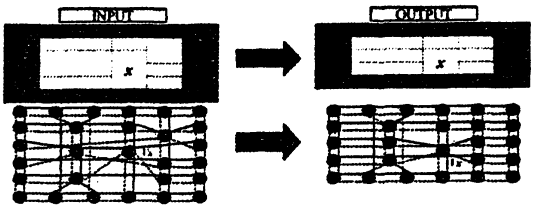

図 8：入力例と出力例．

アルゴリズム

$DeleteSingleRow(G_D, v_x, G_E)$

[入力]

G_D：$n \times m$矩形分割Dに対する 8 次格子グラフ($n \geq 4, m \geq 3$)

v_x：焦点のセルx（G_Dの内部セル）に対応する頂点

[出力]

G_E：G_Dから，v_xと交差する行の中の一番上の 1 行を削除した 8 次格子グラフ

[方法]

1. v_xから北壁辺をたどって西側の周辺セルに対応する頂点にv_0と置く．
2. v_0から北壁辺をたどって全ての頂点に"N"とマークする．

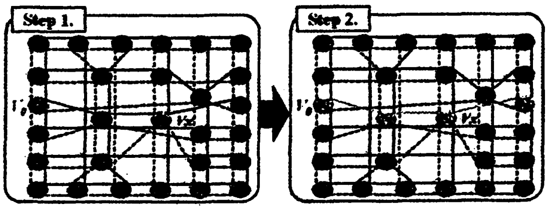

図 9：*DeleteSingleRow*の Step1~2.

3. v_0から南壁辺をたどって全ての頂点に"S"とマークする．
4. v_0から北壁辺にリンクした頂点の内，"S"とマークされた頂点を"D"とマークする．

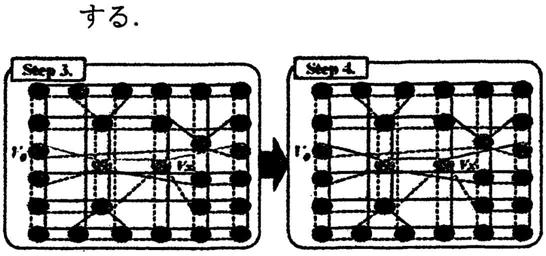

図 10：*DeleteSingleRow*の Step3~4.

5. v_0からリンクした内部セルに対応する頂点の北壁辺を変える．
6. v_0からリンクした内部セルに対応する頂点の南壁辺を変える．

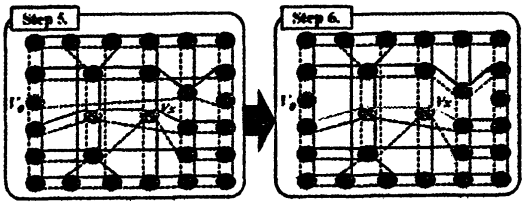

図 11：*DeleteSingleRow*の Step5~6.

7. "D"とマークされた頂点の東西のリンクを変え，"D"とマークされた頂点を削除する．
8. 削除した行の高さを合わせて，高さを変える．

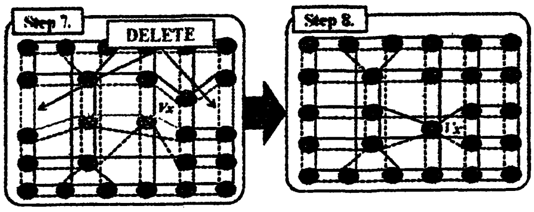

図 12：*DeleteSingleRow*の Step7~8.

Step1, 2, 3, 4 のそれぞれのリンクをたどって，高々m個の頂点を訪問する．Step2~3, Step3~4 の間では，右端の頂点から高々m個のリンクをたどって目的とする左端の頂点に戻る．各頂点の次数は高々8 なので，各頂点における時間計算量は定数である．

したがって，全体としての時間計算量は，$O(m)$となる．

3.2 複数行削除アルゴリズム

ここでは，焦点のセル（高さk）に交差する全ての行を削除するアルゴリズムを示す．図 13 がその例である．

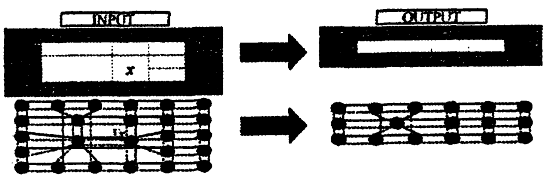

図 13：入力例と出力例．

アルゴリズム

$DeleteMultipleRows(G_D, v_x, G_E)$

[入力]

G_D : $n \times m$矩形分割Dに対する 8 次格子グラフ($n \geq 5, m \geq 3$)

v_x : 焦点のセルx（高さk）に対応する頂点（北・南壁に隣接するセルが周辺セルでないG_Dの内部セル）

[出力]

G_E : G_Dから，v_xと交差する全ての行を削除した 8 次格子グラフ

[方法]

1. v_xから南壁辺をたどって西側の周辺セルに対応する頂点にv_0と置く.
2. v_0から南壁に隣接した頂点にv_hと置く.

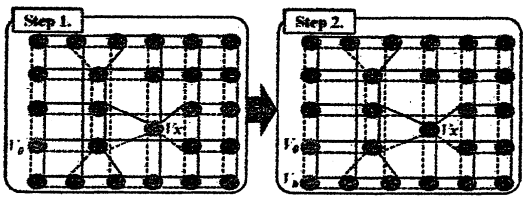

図 14 : *DeleteMultipleRows*の Step1~2.

3. v_xから北壁辺をたどって西側の周辺セルに対応する頂点にv_iと置く.
4. v_iから隣接した下の頂点にv_{i+1}と置く.

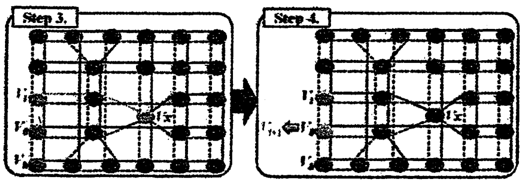

図 15 : *DeleteMultipleRows*の Step3~4.

5. *DeleteSingleRow*アルゴリズムを用いる.

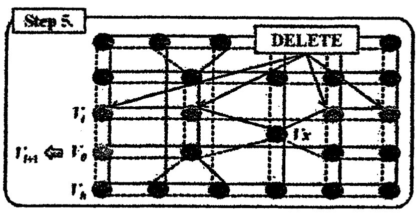

図 16 : *DeleteMultipleRows*の Step5.

6. $i++$.
7. もし$sw(v_i) < sw(v_h)$のとき, Step4 から繰り返す.

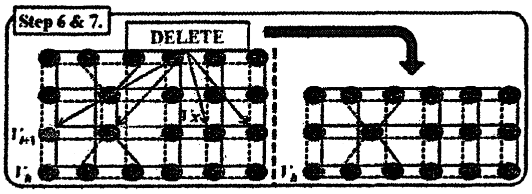

図 17 : *DeleteMultipleRows*の Step6~7.

Step1, 3 のそれぞれのリンクをたどって，高々m個の頂点を訪問する．Step2, 4 では，2 個の頂点を訪問する．Step2~3 の間では，高々m個のリンクをたどって目的とする左端の頂点に戻る．各頂点の次数は高々8 なので，各頂点における時間計算量は定数である.

以上をxの高さ$k \leq n$だけ繰り返す.

したがって，全体としての時間計算量は，$O(m \times k)$となる.

3.3 1 列削除アルゴリズム

8 次格子グラフは，水平方向と垂直方向の構造を同等に扱うことが出来る為，1 行削除と同様のアルゴリズムで削除することができる.

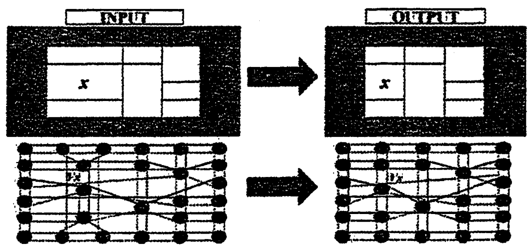

図 18 : 入力例と出力例.

1 行削除アルゴリズムと同様に，全体としての時間計算量は$O(m)$である.

3.4 複数列削除アルゴリズム

8 次格子グラフは，水平方向と垂直方向の構造を同等に扱うことが出来る為，1 行削除と同様のアルゴリズムで削除することができる.

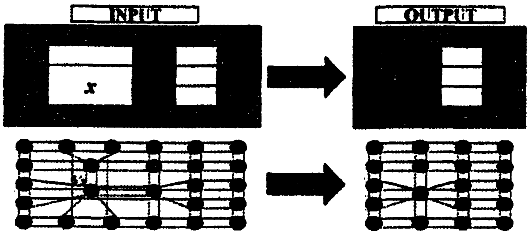

図 19：入力例と出力例.

1 行削除アルゴリズムと同様に，全体としての時間計算量は$O(m \times k)$である.

3.5 計算時間の比較

8 次格子グラフと，既存のデータ構造である 4 分木と双対グラフについての時間計算量の比較を行う.

n行m列の表に対して，1 行削除を行う為の時間計算量は，4 分木の場合，グラフの性質上，想定外なので考えていない．頂点数Nの双対グラフの場合，列数mに対して，最大で全ての頂点を見て回る必要があるので，$O(m \times (N))$である．1 列削除を行う為の時間計算量も同様なので，以下のようになる.

表 1：1 行削除，1 列削除の計算時間.

	1 行削除，1 列削除
4 分木	—
双対グラフ	$O(m \times (N))$
8 次格子グラフ	$O(m)$

n行m列の表に対して，複数行削除を行う為の時間計算量は，4 分木の場合，グラフの性質上，想定外なので考えていない．頂点数Nの双対グラフの場合，列数mに対して，最大で全ての頂点を見て回る必要がある．さらに，高さ$k \leq n$だけ繰り返す．したがって，$O(m \times (N) \times k)$である．複数列削除を行う為の時間計算量も同様なので，以下のようになる.

表 2：複数行削除，複数列削除の計算時間.

	複数行削除，複数列削除
4 分木	—
双対グラフ	$O(m \times (N) \times k)$
8 次格子グラフ	$O(m \times k)$

4 おわりに

本論文では，1 行削除，複数行削除，1 列削除，複数列削除のアルゴリズムを提案した．これらのアルゴリズムの計算時間は，既存のデータ構造より早いことがわかった．今後は，8 次格子グラフを特徴付けるグラフ文法を構成する.

謝辞

貴重なコメントを頂いた早稲田大学高等学院の穴田浩一先生，日本大学の神藤悠希氏に深く感謝いたします.

参考文献

[1] R. A. Finnkel and J. L. Bentley, Quad Trees: A Data Structure for Retrieval on Composite Keys, *Acta Informatica*, Vol. 4, No. 1, 1974, pp.1-9.

[2] K. Kozminsky and E. Kinnen, Rectangular Duals of Planar Graphs, *Networks* 16, 1985, pp.145-157.

[3] T. Yaku, "Representation of Heterogenenous Tessellation Structures by Graphs", WAAP 108, research report, 1–6, Dec. 2001. URL:http://www.waap.gr.jp/waap-rr/waap-rr-01-013.pdf

[4] T. Kirishima, T. Motohashi, K. Tsuchida, and T. Yaku, Table Processing based on Attribute Graphs, *Proc. 6th IASTED International Conference on Software Engineering and Applications*, 2002, pp.317-322.

[5] 本橋友江，谷聖一，土田賢省，夜久竹夫，表編集のアルゴリズム，**数理解析研究所講究録**, Vol.1325, 2003, pp.152-157.

[6] 土田賢省，本橋友江，夜久竹夫，山澤聡，吉住寿洋，表の格子グラフモデルと編集アルゴリズム，**情報処理学会研究報告**, 2009-MPS-073, 2009, pp.225-228.

[7] T. Yaku, K. Anada, S. Koka, Y. Shindo, and K. Tsuchida, Row Manipulation in the Heterogeneous Tabular Forms with an Octal Grid Model, *Proc.* 2011 *IEEE Symposium on Viasual Languages and Human-Centric Computing*, 2011, pp.269-270.