表データのカット&ペーストの一方式

6R－3

山下晶夫、平山唯樹
日本アイ・ビー・エム株式会社 東京基礎研究所

1 はじめに

紙でしか存在しない文書からテキストDBを作成する場合には、図表の情報を保存するためにページ単位の文書画像を関連付けて同時に保管する場合が多い。このような時には、特に図表を含むページではテキストよりも画像を表示する方が直感的でわかりやすい。一方テキストを再利用する時には、画像ではなくページに該当するテキストを別のビューワに呼び出し、ユーザー操作により他のアプリケーションに引き渡すといった方法が一般的である。

本稿では、表示してある文書画像、特に表画像をマウスでドラグしてデータのカット&ペーストを行う時、従来のユーザーインターフェースのようにビットマップデータをコピーするのではなく、マウス座標から該当する表のカラムデータを引き渡す方式について述べる。このようなユーザーインターフェースによれば、データの登録時に表の構造情報を保存するとともに、表の項目データをCSV形式等で蓄えることによって、画像ビューワ上の表の必要なデータ部分をドラグして、スプレッドシートのアプリケーション等に行、列を揃えて貼りこむことが可能となる。

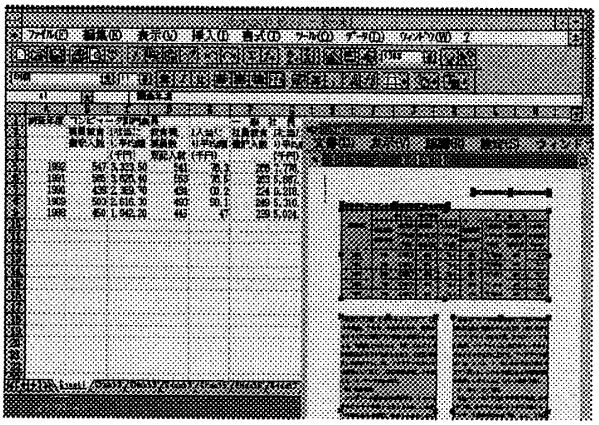

図1: Cut & Paste ユーザーインターフェースの例

A Method of Cut & Paste from Table Data
A.Yamashita, Y.Hirayama
IBM Research, Tokyo Research Laboratory, IBM Japan, Ltd.

2 ユーザーインターフェースの概要

図1に本稿で実現されたユーザーインターフェースを示す。図中示したように、表示されている文書画像の一部(例えば表の一部)をマウスでドラグすると、マウス座標から画像中のどの部分のテキストデータに対応するか計算され、該当するデータが例えばクリップボードにコピーされる。このデータは通常のテキスト或はテキストとしても貼りつけられるカラム情報を含んだデータ形式になっているので、クリップボードをサポートしている他のアプリケーションにペーストできる。このように画像から対応するテキストの該当部分を切り取るには、文書画像中のテキストの位置座標をテキストに関連付けして保存しなければならない。OCRを使って画像データからテキストを認識する場合には、レイアウト解析や文字切り出しの情報を保存しておくことで、これが可能になる。

3 表画像の解析方法

文書画像を認識してテキストデータに変換するOCRでは、認識に先立ってテキスト部分の位置を検出し、文字列や個々の文字イメージを切り出すので、各文字の画像中での位置座標を求めることができる。さらに、表の解析機能を備えているOCRの場合には、縦横罫線、カラムの位置や関係を求めることができる。表の形式は必ずしも格子状とは限らず、カラムが統合されていたり、細分されている場合もある。複雑な形状の表は通常単純な格子状モデルにはあてはまらないが、近似的にこのモデルをあてはめた表画像の解析を行いデータを抽出すれば、他のアプリケーションにとって再利用しやすい形式でデータを保存できる(1)。表画像の解析は以下のようなプロセスで行なわれる。

1. 表画像を解析し、線分(罫線)、文字列を抽出する
2. 実罫線を表の外周にぶつかるまで延長した仮想罫線を引き、表全体を仮の格子状に分割する
3. 表のカラムのデータを格子のセルにマッピングする。1個の文字列が複数のセルに分割される場合

には、最も上かつ左のセルにのみマップし、残りは空セルとする

このように表の縦横の関連を求め、カラム別に文字列を切り出すことができるため、縦横整列し、罫線部分に区切り記号 (ここではカンマ又はタブを用いている) を入れた認識結果を出力できる。

4 保存データの構造

一般にOCRからの出力 (認識結果) には、文字列や表の位置情報は含まれないが、本稿で提案するユーザーインターフェースを実現するためにはテキストと同時に位置情報も必要である。そこで、図2に示すように、表を処理した結果について、OCRが内部的に持っている位置情報をタグ付きテキストで出力するものとする。図中の表に対する保存データの構造のサンプルを示す。表の部分は<table>, < \ table>で囲まれている。<table>タグに続く4個の数値は表の位置と大きさ (XY, 幅と高さ)、続く数値はそれぞれ列の区切りとなる罫線の位置のX座標である。<column>は表の行を区切る罫線の始点のY座標を示す (長さは格子状モデルを仮定しているので、表の幅に等しいものとしている)。カラムのデータはCSV形式 (カンマでカラム毎のデータが区切られた形式) のようにカンマまたはタブ等で区切って記述しておく。カンマの位置は、仮想セルの境界である。図に示した表は、格子状ではないが、格子モデルに基づいた表解析により各セルに近似的にマップされている。

item1		item2	
data11	value11	data12	value12
data21	value21	sum of items	
data31	value32		

```
<table
size='x_pos, y_pos, x_size, y_size'
raw_pos='raw1_x_pos, raw2_pos, raw3_pos'>
  item1,          ,   item2,        ,
<column column_pos='column1_y_pos'>
  data11, value11, data21, value21,
<column column_pos='column2_y_pos'>
  data21, value21, sum of items,   ,
<column column_pos='column3_y_pos'>
  data31, value31,        ,        ,
<\table>
```

図 2: 表をあらわすタグ付きテキストの例

5 保存データを利用したユーザーインターフェース

マウスで表画像をカット&ペーストした時の処理の流れは例えば次の通りである。

1. 画像上で切り出したい表画像の範囲をドラグする
2. ウインドウのマウスの座標を変換し、画像中の絶対座標を計算する
3. 関連付けられたタグ付きテキストを呼び出して、マウスで囲った矩形内に包含されるデータをタグで示した表のカラムの位置座標から求める。
4. 該当するデータをタグを外してクリップボードにコピーする。

表画像を対象とした場合には、必ず表解析処理が可能なOCRを使って情報を抽出する必要があるが、例えばワープロに罫線や特殊記号を使って記述された表についても、同じような仕組みでCSVデータとしてカット&ペーストすることができる。この場合には、表画像の解析方法で説明した仮想罫線による格子分割をワープロデータ上で行えばよい。表画像の解析と異なり解析誤りが無いので、わざわざデータを保存しておかないでも、即時に処理することで上記ユーザーインターフェースを可能にすることができる。

6 おわりに

テキストベースのビューワよりは、画像ベースの方が特に表画像については全体像をつかみやすい。本稿で提案したユーザーインターフェースによれば、画面では表画像を表示しながら、表示データの再利用の時には必要なテキストや数値データを画像上で指定し、取り出すことができる。しかもあらかじめ表画像を格子状モデルで解析し、CSV形式等で保存していれば、直接スプレッドシートやデータベースのアプリケーションにカラムを合わせて貼りこめる形でデータを引き渡すことができる。

参考文献

[1] Y.Hirayama, "A Method for Table Structure Analysis Using DP Matching", Proc. of ICDAR'95, Canada (1995, to appear).