Эвристический метод обнаружения таблиц в разноформатных документах*

И. В. Бычков, Г. М. Ружников, А. Е. Хмельнов, А. О. Шигаров
Институт динамики систем и теории управления СО РАН, Иркутск, Россия
e-mail: shigarov@icc.ru

Предлагается эвристический метод обнаружения таблиц, ориентированный на особенности публикуемых статистических таблиц. В качестве входных данных в предлагаемом методе используются метафайлы, что позволяет применять его к разноформатным документам. В предлагаемом методе процесс обнаружение таблиц строится как сегментация страницы документа снизу вверх: от более простых элементов страницы к более сложным. Экспериментальная оценка этого метода показывает эффективность его использования для широкого круга статистических таблиц.

Ключевые слова: анализ и распознавание и документов, извлечение информации, извлечение и обработка таблиц.

Введение

Для решения многих научных и практических задач требуется извлекать данные из таблиц, содержащихся в различных документах. Методы и системы извлечения таблиц из документов позволяют автоматизировать этот процесс. Обзоры работ по извлечению и обработке таблиц [1 – 4], появившиеся за последние годы, показывают растущий интерес к данной проблематике. В обзоре [1] выделяются следующие этапы процесса извлечения таблиц:

обнаружение таблиц в документах — поиск на страницах документов областей, являющихся таблицами;

сегментация таблиц — выделение столбцов, строк и ячеек таблицы;

функциональный анализ — определение роли ячеек таблицы (являются ли они заголовками, подзаголовками, или значениями);

структурный анализ — определение зависимостей между заголовками, подзаголовками и значениями;

интерпретация — преобразование полученного на предыдущих этапах описания табличных данных к целевому представлению (например, преобразование такого описания таблицы к отношению в терминах реляционной модели).

*Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 08-07-00163-а) и Президентской программы “Ведущие научные школы РФ” (грант № НШ-1676.2008.1).

Таким образом, чтобы извлечь таблицы из документов, прежде всего необходимо их обнаружить в этих документах. Существующие методы обнаружения таблиц ориентируются на определенные форматы документов (как правило, это растровые изображения, ASCII-текст или HTML). В то же время при извлечении таблиц из документов в качестве представления входных данных можно использовать один из обменных форматов, например, PostScript [5], PDF [6] или EMF [7]. Эти обменные форматы более информативны по сравнению с растровыми изображениями и ASCII-текстом. Кроме текста и графики, они содержат шрифтовые метрики выводимого текста, а также информацию о порядке вывода этого текста на печать. Эта полезная информация может применяться для более эффективного и аккуратного обнаружения таблиц. Извлекать таблицы из PDF предлагается в работе [8], причем ее авторы отмечают, что им неизвестны другие методы извлечения таблиц из PDF. В работе [9] указывается возможность использования формата PostScript в качестве представления входных данных, но при этом не рассматривается сам процесс извлечения информации из PostScript.

В отличие от других обменных форматов EMF может интерпретироваться с помощью стандартных функций GDI API [7] (часть Windows API). Это делает обработку EMF достаточно простой и доступной. При этом документы различных форматов (например, DOC, XLS, PDF, ASCII-текст, HTML) могут быть напечатаны в метафайлы EMF. В настоящей работе предлагается эвристический метод обнаружения таблиц, ориентированный на метафайлы EMF. При этом предполагается, что таблицы в документах, напечатанных в метафайлы, не являются растровыми включениями. Стоит отметить, что в указанных обзорах нет ссылок на работы, в которых бы рассматривалась такая возможность. Авторам также неизвестно о существовании систем и методов извлечения таблиц, ориентированных на метафайлы.

1. Особенности рассматриваемых таблиц

Разнообразие форм таблиц очень велико. Многие из существующих методов обнаружения таблиц ориентируются на их различные особенности, которые обычно определяются стандартами и соглашениями, принятыми в некоторой предметной области. Предлагаемый в данной работе метод ориентируется на особенности структуры статистических таблиц. Таблицы с такой структурой используются в государственных статистических

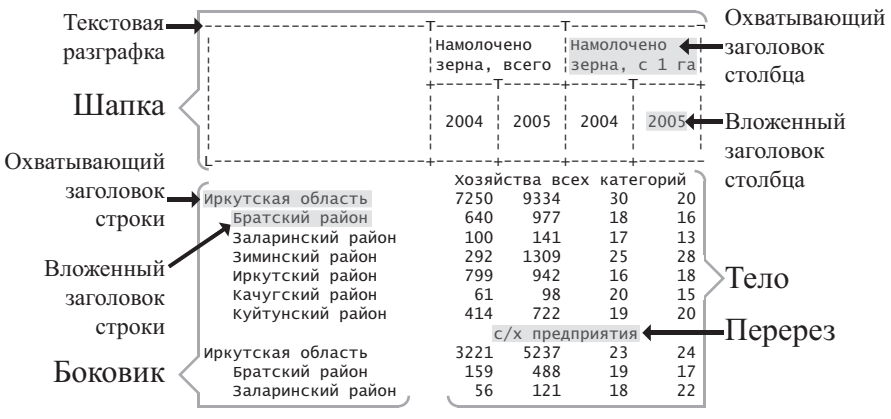


Рис. 1. Пример статистической таблицы

отчетах России, США, Евросоюза, Китая, Японии, а также в финансовых отчетах различных компаний. Проведенный авторами анализ публикуемых статистических таблиц показал, что они обладают достаточно схожей структурой, вне зависимости от национальных или корпоративных особенностей.

На рис. 1 показан пример статистической таблицы, где обозначены основные элементы ее структуры. Описываемый метод предполагает, что таблица может иметь шапку, боковик, тело, а также перерезы внутри тела. Заголовки столбцов таблицы могут образовывать иерархию, причем охватывающие заголовки всегда расположены над соответствующими им вложенными заголовками. Текст таблицы должен иметь горизонтальное расположение. Кроме того, предлагаемый метод учитывает, что таблица может иметь полную или частичную разграфку или не иметь ее вовсе. При этом разграфка таблицы может быть образована как графическими примитивами (линиями, прямоугольниками), так и символами псевдографики и некоторыми другими символами, в последнем случае будем говорить, что таблица имеет *текстовую разграфку*.

2. Получение данных из метафайлов

Прежде всего электронные документы необходимо преобразовать в метафайлы. Для этого можно использовать виртуальный EMF-принтер (например, свободно распространяемый проект EMFPrinter[1]). Также для некоторых форматов (например, PDF) можно использовать специализированные конвертеры. Каждый полученный таким образом метафайл соответствует одной странице исходного документа. Полученные метафайлы обрабатываются с помощью функций GDI. В метафайлах инструкциям вывода текста соответствуют записи типов EMR_EXTTEXTOUTW и EMR_SMALLTEXTOUT (существуют и другие типы записей EMF для вывода текста, но на практике они не применяются). Инструкциям вывода линеек часто соответствуют записи типа EMR_BITBLT. Указанные типы записей метафайлов описаны в MSDN [7], кроме недокументированного типа записей — EMR_SMALLTEXTOUT. С помощью контекста метафайла выполняется интерпретация данных из этих записей (определяются позиции вывода текста, межсимвольные расстояния, шрифтовые метрики, цвета фона и текста) в соответствии с системами координат и режимами отображения, используемыми в метафайле. При этом игнорируются записи, соответствующие инструкциям, которые выводят текст вне области страницы либо выводят текст с тем же цветом, что и цвет фона области, ограничивающей этот текст.

Стоит отметить, что в предлагаемом методе применяется прямоугольная система целочисленных координат, в которой ось X направлена слева направо, а ось Y — сверху вниз. Кроме того, все рассматриваемые в данной работе ограничивающие прямоугольники задаются координатами своих сторон, при этом если некоторый рассматриваемый объект o имеет ограничивающий прямоугольник, то функции $x_l = x_l(o)$, $y_t = y_t(o)$, $x_r = x_r(o)$ и $y_b = y_b(o)$ определяют координаты сторон этого прямоугольника: левой, верхней, правой и нижней соответственно.

В результате описанной обработки метафайла для каждой записи, соответствующей инструкции вывода текста, можно получить одну или несколько структур, определяющих отдельные последовательности непробельных символов этого текста. Будем называть эти структуры *текстовыми элементами*. Определим текстовой элемент как чет-

[1] http://emfprinter.sourceforge.net

Рис. 2. Пример текстового элемента.

верку $e = (C, S, M, R)$, где $C = \{c_1 \dots c_n\}$, $n \in \mathbb{N}$, — упорядоченный набор непробельных символов; $S = \{s_1 \dots s_n\}$ — упорядоченный набор межсимвольных интервалов; $M = \{m_{el}, m_{il}, m_a, m_d, m_{fp}, m_{si}\}$ — набор текстовых метрик: $m_{el} = m_{el}(e)$ — внешний зазор, $m_{il} = m_{il}(e)$ — внутренний зазор, $m_a = m_a(e)$ — надстрочный интервал, $m_d = m_d(e)$ — подстрочный интервал, $m_{fp} = m_{fp}(e)$ — шаг шрифта (фиксированный или переменный), $m_{si} = m_{si}(e)$ — межсимвольный интервал пробела; R — ограничивающий прямоугольник.

Дополнительно определим, что $w(e) = x_r(e) - x_l(e)$ — ширина ограничивающего прямоугольника текстового элемента e, а $h(e) = y_b(e) - y_t(e)$ — высота этого прямоугольника. Причем $w(e) = \sum_{i=1}^{n} s_i$, где s_i, $i = \overline{1,n}$, — полностью составляют набор межсимвольных интервалов текстового элемента e, а $h(e) = m_a(e) - m_{il}(e)$. На рис. 2 показан текстовой элемент и некоторые его метрики.

Кроме того, в результате обработки метафайла из структур, соответствующих инструкциям вывода графики, извлекаются *линейки* (линии разграфки). При этом каждая линейка r задается своим ограничивающим прямоугольником.

3. Предобработка полученных данных

На рис. 1 шапка таблицы ограничена текстовой разграфкой. Такая разграфка часто используется в таблицах, являющихся ASCII-текстом. Поскольку такая разграфка является текстом, ее наличие может затруднить обнаружение и извлечение таблиц. В предлагаемом методе в процессе предобработки выполняется разделение текстовой разграфки и остального текста. Из текстовых элементов исключаются символы, составляющие текстовую разграфку. При этом текстовые элементы, содержащие такие символы, могут быть изменены, разделены на несколько или полностью удалены, а исключенная текстовая разграфка преобразуется в линейки, которые объединяются с остальной, не текстовой, разграфкой страницы. На рис. 3 показан пример разделения текстовой разграфки и текста на фрагменте страницы.

Кроме того, если понимать *слово* как последовательность подряд идущих непробельных символов в тексте, то можно заметить, что при печати документа в метафайлы разным частям одного слова могут соответствовать разные инструкции вывода текста на графический контекст. В этом случае после выполнения описанного процесса получения данных из метафайлов разным частям этого слова будут соответствовать разные текстовые элементы. Поэтому предобработка включает в себя также процедуру объединения нескольких текстовых элементов в единый текстовой элемент в тех случаях,

а *б* *в*

Рис. 3. Пример разделения текстовой разграфки и текста: *а* — фрагмент таблицы с текстовой разграфкой; *б* — на фрагменте выделены текстовые элементы; *в* — на фрагменте выделены текстовые элементы, оставшиеся после разделения текста и текстовой разграфки

когда боковые стороны соответствующих им ограничивающих прямоугольников вплотную прилегают друг к другу, а проекции этих прямоугольников на ось Y полностью совпадают. В результате выполнения данной процедуры большинство слов будет восстановлено, т. е. большинству текстовых элементов будут соответствовать целые слова, а не их отдельные части.

Помимо списанной задачи восстановления слов из их частей, существует и обратная задача — разделения текстового элемента на несколько частей. Поскольку в некоторых случаях для разделения между собой отдельных слов вместо пробелов используются удлиненные межсимвольные интервалы последних символов слов, то несколько слов может оказаться в одном текстовом элементе. Поэтому предобработка включает в себя также процедуру обнаружения таких удлиненных межсимвольных интервалов и разделения по ним текстовых элементов на несколько частей, соответствующих отдельным словам.

4. Существующие методы обнаружения таблиц

В литературе упоминается несколько методов обнаружения таблиц, которые используют структуры, схожие с рассматриваемыми в данной работе текстовыми элементами. Такие структуры соответствуют отдельным словам и имеют ограничивающие прямоугольники (wbb). К таким методам авторы обзора [1] относят рассматриваемые в работах [10–12] (метод из [11] описывается также в работе [13]). Кроме того, к таким методам можно отнести рассмотренные в работах [8, 14]. Методы [10–12, 14] ориентируются на растровые изображения, методы [10, 11, 13] годятся и для ASCII-текста.

Авторы работы [12] прибегают к поиску в тексте документа ключевых слов (например, “Table”), чтобы обнаружить строки текста, в которых расположены таблицы. Стоит отметить, что отсутствие таких ключевых слов в начале таблиц может отрицательно повлиять на результаты обнаружения таблиц методом [12]. В работе [10] для обнаружения таблиц решается оптимизационная задача поиска наилучшего разбиения строк текста на последовательности, которые являются либо таблицами, либо просто текстом. Из того, как в данном методе строятся оценки корреляции строк, можно сделать вывод, что многие таблицы, имеющие в шапках многоуровневую иерархию заголовков столбцов, будут обнаруживаться неточно. Более того, данный подход, как и метод из [12], не учитывает, что внутри таблицы может применяться вертикальное выравнивание текста по середине высоты отдельных ячеек, — в таких случаях обнаружение таблиц этими методами может выполняться неточно. В работах [11, 13] предлагается

подход к кластеризации слов снизу вверх и выбору кластеров, составляющих таблицы. Недостатки этого подхода обсуждаются в работе [14].

В работе [14] отдельные слова рассматриваются как компоненты, для которых определены ограничивающие прямоугольники. Эти компоненты предлагается объединять в структуры, называемые в работе [14] *word blobs*, в том случае, если они расположены на одной строке текста и горизонтальное расстояние промежутка между ними не превышает определенного порога. Если на странице использовался только один шрифт, то значение этого порога скорее всего будет приближено к интервалу одного пробела. В результате, строка текста наверняка будет содержать единственную структуру word blobs, а строка таблицы — несколько таких структур. Это позволяет в дальнейшем достаточно просто идентифицировать строки таблицы и отличать их от строк текста. Изложенная в работе [14] идея — об объединении отдельных близко расположенных в одной строке слов в более крупные структуры — заимствуется нашим методом. Стоит отметить, что в дальнейшем процессе обнаружения метод [14] использует слишком упрощенные предположения о нескольких таблицах на странице: предполагается, что между таблицами должны располагаться пустые строки или строки текста. В противном случае обнаружение будет выполняться неточно. На точность обнаружения в этом методе также могут повлиять строки с выравниванием текста по ширине.

5. Сегментация страницы

5.1. Формирование текстовых блоков

На этой стадии текстовые элементы группируются в более крупные структуры, которые являются некоторым аналогом структур word blobs из метода [14]. Будем называть эти структуры *текстовыми блоками*. Определим текстовой блок как пару $b = (E, R)$, где $E = \{e_1, \ldots, e_n\}$, $n \in \mathbb{N}$, — набор текстовых элементов; R — ограничивающий прямоугольник. Причем, ограничивающий прямоугольник такого текстового блока вычисляется следующим образом:

$$R = \Big(\min_{1 \leqslant i \leqslant n}\{x_l(e_i)\}, \min_{1 \leqslant i \leqslant n}\{y_t(e_i)\}, \max_{1 \leqslant i \leqslant n}\{x_r(e_i)\}, \max_{1 \leqslant i \leqslant n}\{y_b(e_i)\}\Big).$$

Пусть e — текстовой элемент. Определим для него две прямоугольные области: $A_{\text{top}}(e)$ — вокруг вершины в правом верхнем углу его ограничивающего прямоугольника текстового элемента e, $A_{\text{bottom}}(e)$ — вокруг вершины в правом нижнем углу этого прямоугольника, — следующим образом:

$$A_{\text{top}}(e) = \big(x_r(e) - v_1,\ \ y_t(e) - h\,v_2,\ \ x_r(e) + m_{si}(e)c,\ \ y_b(e) - h\,v_3\big),$$
$$A_{\text{bottom}}(e) = \big(x_r(e) - v_1,\ \ y_t(e) + h\,v_3,\ \ x_r(e) + m_{si}(e)c,\ \ y_b(e) + h\,v_2\big),$$

где v_1: $v_1 \in \mathbb{R}$ и $0 \leqslant v_1 < w(e)$ (по умолчанию $v_1 = 0$); v_2: $v_2 \in \mathbb{R}$ и $0 \leqslant v_2 \leqslant 1$ (по умолчанию $v_2 = 0,7$); v_3: $v_3 \in \mathbb{R}$ и $0 \leqslant v_3 \leqslant 1$ (по умолчанию $v_3 = 0,1$) — задаваемые пользователем параметры, которые настраивают размеры и расположение соответствующих областей $A_{\text{top}}(e)$ и $A_{\text{bottom}}(e)$; величина c выбирается в зависимости от шага шрифта, зафиксированного в текстовом элементе: $c = 1$, если $m_{fp}(e) = 0$, и $c = 2$, если $m_{fp}(e) = 1$ (выбор значения c может быть настроен иначе). На рис. 4 эти прямоугольные области выделены штриховкой.

Кроме того, определим прямоугольную область промежутка между двумя текстовыми элементами — e_i, e_j: $x_l(e_i) < x_l(e_j)$, следующим образом:

$$W(e_i, e_j) = \big(x_r(e_i),\ \min\{y_t(e_i), y_t(e_j)\},\ x_l(e_j),\ \max\{y_b(e_i), y_b(e_j)\}\big).$$

Обозначим множество линеек, расположенных на странице, как R_{pg}. Будем считать, что два текстовых элемента e_i, e_j: $x_l(e_i) < x_l(e_j)$, принадлежат одному текстовому блоку, если выполняется следующее условие:

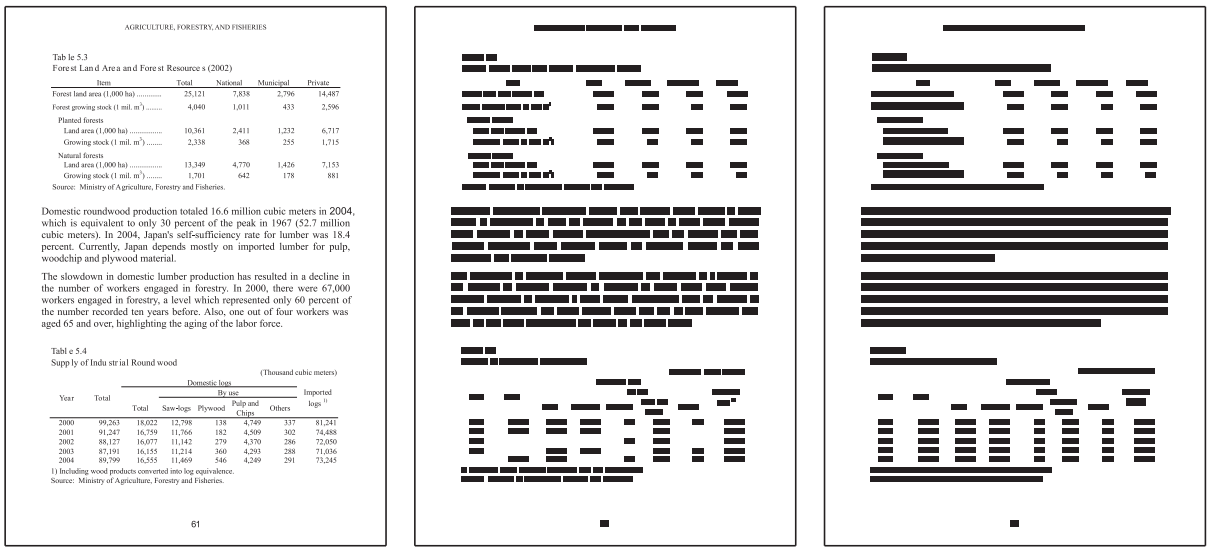
$$\begin{cases} \big(x_l(e_j), y_t(e_j)\big) \in A_{\text{top}}(e_i), \\ \big(x_l(e_j), y_b(e_j)\big) \in A_{\text{bottom}}(e_i), \\ r \cap W(e_i, e_j) = 0\ \forall\ r : r \in R_{pg}. \end{cases} \tag{1}$$

На рис. 4 показаны два случая расположения ограничивающих прямоугольников двух текстовых элементов e_i и e_j на странице.

Используя условие (1), формируем текстовые блоки. Текстовые элементы, для которых не существует ни одной пары, удовлетворяющей условию (1), образуют отдельные текстовые блоки. Множество текстовых элементов страницы обрабатывается до тех пор,

а *б*

Рис. 4. Примеры расположения двух текстовых элементов на странице: *а* — текстовые элементы принадлежат одному текстовому блоку; *б* — текстовые элементы принадлежат двум разным текстовым блокам



а *б* *в*

Рис. 5. Пример формирования текстовых блоков: *а* — исходная страница; *б* — ограничивающие прямоугольники текстовых элементов; *в* — ограничивающие прямоугольники текстовых блоков

пока для каждого текстового элемента не будет определен текстовой блок. На рис. 5 показан пример формирования текстовых блоков из текстовых элементов на одной странице.

В отличие от метода, описанного в работе [14], предлагаемый метод использует шрифтовые метрики для группировки слов, а также линии разграфки, если они есть. Это позволяет более аккуратно формировать текстовые блоки.

5.2. Формирование строк

С помощью текстовых блоков выполняется сегментация страницы на строки, которые могут оказаться как строками текста, так и строками таблиц. При этом предполагается, что строки таблиц набраны на странице в одну колонку. Каждая непустая строка может охватывать один или несколько текстовых блоков. Внутри строки текстовые блоки отделены друг от друга вертикальными промежутками *белого пространства* (т. е. пространства страницы, не занятого ограничивающими прямоугольниками текстовых блоков). К примеру, на рис. 6 показан фрагмент страницы, содержащий пять строк.

Определим *строку* как тройку $l = (B,\ G,\ R)$, где $B = \{b_1, \ldots, b_n\}$, $n \in \mathbb{N}$, — набор текстовых блоков; $G = \{g_1, \ldots, g_m\}$, $m \in \mathbb{N}$, — набор вертикальных промежутков; R — ограничивающий прямоугольник.

Пусть B_{pg} — множество всех текстовых блоков некоторой страницы. Пусть отрезок $P_y(b) = \big[y_t(b), y_b(b)\big]$ — проекция на ось Y текстового блока b. Чтобы выделить на этой странице строки, прежде всего определяется разбиение множества B_{pg} на подмножества $B_1, \ldots, B_n$, $n \in \mathbb{N}$: $B_1 \cup \cdots \cup B_n = B_{pg}$, так, что в каждом подмножестве все принадлежащие ему текстовые блоки находятся в транзитивном замыкании отношения: $P_y(b) \cap P_y(\tilde{b}) \neq 0$, $b, \tilde{b} \in B_{pg}$. Будем считать, что каждое полученное подмножество B_i, $i = \overline{1,n}$, определяет набор текстовых блоков в отдельной строке. В результате на странице формируются строки, содержащие хотя бы по одному текстовому блоку. Далее на странице для каждой ее строки с соответствующим набором текстовых блоков $B = \{b_1, \ldots, b_n\}$, $n \in \mathbb{N}$, определяется ограничивающий прямоугольник следующим образом:

$$R = \Big(x_l(p), \min_{1 \leqslant i \leqslant n}\{y_t(b_i)\}, x_r(p), \max_{1 \leqslant i \leqslant n}\{y_b(b_i)\}\Big),$$

где p — ограничивающий прямоугольник этой страницы, координаты сторон которого заданы ее соответствующими краями. При этом, в силу алгоритма формирования строк, их ограничивающие прямоугольники не пересекаются. После чего внутри белого пространства каждого такого прямоугольника выделяются вертикальные промежутки

(Thousand cubic meters)

Year	Total	Domestic logs					Imported logs [1)]
		Total	By use				
			Saw-logs	Plywood	Pulp and Chips	Others	
2000	99,263	18,022	12,798	138	4,749	337	81,241
2001	91,247	16,759	11,766	182	4,509	302	74,488

Рис. 6. Пример расположения строк на странице: строки выделены своими ограничивающими прямоугольниками; вертикальные промежутки белого пространства выделены штриховкой

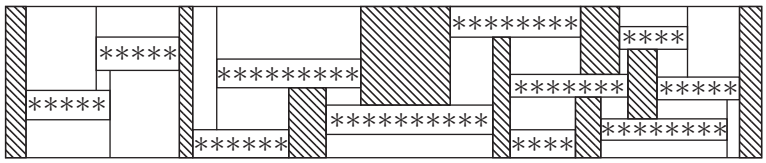


Рис. 7. Пример сегментации белого пространства строки: штриховкой выделены области, определенные как вертикальные промежутки; остальные прямоугольные области белого пространства не закрашены

соответствующей строки. Каждый *вертикальный промежуток* g задается своим ограничивающим прямоугольником.

Для определения вертикальных промежутков в строках используется алгоритм сегментации белого пространства строки. Алгоритм проходит по всем текстовым блокам строки, из вершин их ограничивающих прямоугольников протягиваются вертикальные линии вверх и вниз до тех пор, пока они не упрутся либо в ограничивающий прямоугольник другого текстового блока, либо в верхнюю или нижнюю границу этой строки. Далее все белое пространство этой строки делится на прямоугольные области, боковые стороны которых совпадают с проведенными вертикальными линиями, а верхняя и нижняя стороны проходят либо по верхним или нижним сторонам ограничивающих прямоугольников текстовых блоков, либо по верхней или нижней границе этой строки. При этом полученные прямоугольные области белого пространства строки не пересекаются между собой. Далее среди них выбираются вертикальные промежутки так, что проекция на ось Y любого вертикального промежутка не перекрывается полностью проекцией на ось Y любой из соседних областей белого пространства, т. е. областей, между которыми нет текстовых блоков. На рис. 7 показан результат сегментации белого пространства строки.

Все полученные строки составляют множество строк страницы, причем внутри этого множества строки упорядочены в соответствии со своим расположением на странице (сверху вниз). Следующая задача сегментации страницы на строки состоит в определении пустых строк (т. е. строк, не содержащих ни одного текстового блока), которые могут располагаться между непустыми строками. Количество строк, расположенных между двумя соседними непустыми строками (т. е. строками, между которыми нет других непустых строк), — l_k и l_{k+1}: $y_b(l_k) < y_t(l_{k+1})$, вычисляется следующим образом:

$$\big(y_t(l_{k+1}) - y_b(l_k)\big) / \big(m_{el}(l_k) + m_a(l_k) + m_d(l_k)\big),$$

где $m_{el}(l_k)$, $m_a(l_k)$ и $m_d(l_k)$ — средние значения, соответственно, внешнего зазора, надстрочного и подстрочного интервалов текстовых элементов, содержащихся в текстовых блоках строки l_k. Полученные таким образом пустые строки добавляются в множество строк страницы, не нарушая при этом его упорядоченности.

5.3. Формирование табличных регионов

Рассматривая документы, включающие таблицы, можно заметить, что почти каждая строка любой таблицы охватывает по нескольку текстовых блоков (даже самая простая таблица содержит хотя бы одну такую строку). Кроме того, подряд идущие строки, составляющие одну таблицу, коррелируют друг с другом по расположению своих вертикальных промежутков. В данном разделе описывается анализ связности строк

и формирования структур, охватывающих последовательности идущих подряд связных строк. Будем называть эти структуры *табличными регионами*. На рис. 8 показан фрагмент страницы, содержащий три табличных региона.

Определим табличный регион как тройку $t = (L, G, R)$, где $L = \{l_1, \ldots, l_n\}$, $n \in \mathbb{N}$, — упорядоченный (сверху вниз по расположению на странице) набор строк, которые охватывает табличный регион; $G = \{g_1, \ldots, g_m\}$, $m \in \mathbb{N}$, — набор вертикальных промежутков белого пространства; R — ограничивающий прямоугольник.

Причем, ограничивающий прямоугольник этого табличного региона вычисляется следующим образом: $R = \big(x_l(p), y_t(l_1), x_r(p), y_b(l_n)\big)$, где p — ограничивающий прямоугольник страницы, определенный в предыдущем разделе.

Для обнаружения последовательностей строк, определяющих текстовые регионы, используется несколько эвристик. Прежде всего для того, чтобы строка l с набором вертикальных промежутков $G(l) = \{g_1, \ldots, g_n\}$, $n \in \mathbb{N}$, являлась строкой табличного региона, она должна удовлетворять трем следующим условиям.

Во-первых, строка табличного региона должна содержать хотя бы два текстовых блока и соответственно иметь более двух вертикальных промежутков, т. е.

$$\big|G(l)\big| > 2. \tag{2}$$

Во-вторых, строка табличного региона должна иметь такую компоновку вертикальных промежутков строки, при которой нижняя граница любого вертикального промежутка этой строки проходит по нижней границе этой строки, т. е.

$$y_b(g) = y_b(l) \;\forall\; g : g \in G(l). \tag{3}$$

В-третьих, ширина белого пространства строки табличного региона относительно всей ширины данной строки не должна превышать заранее определенной пороговой величины, т. е.

$$\frac{1}{x_r(l) - x_l(l)} \sum_{i=1}^{n} \big(x_r(g_i) - x_l(g_i)\big) \geqslant v_4, \tag{4}$$

где величина v_4: $v_4 \in \mathbb{R}$ и $0 < v_4 < 1$ (по умолчанию $v_4 = 0,1$) определяет минимально допустимую ширину белого пространства строки табличного региона.

(METRIC TONS AND $1000)

COUNTRY	January - July				July			
	Quantity		Value		Quantity		Value	
	2004	2005	2004	2005	2004	2005	2004	2005
European Union								
Germany	11,662	10,684	86,690	81,784	1,118	415	6,525	1,685
Belgium-Luxembourg	9,505	5,284	67,820	37,930	27	123	146	648
Netherlands	2,775	4,875	21,429	39,694				
France	5,612	3,030	15,889	12,923				
Other Markets								
Japan	13,352	9,117	90,901	52,604	107	76	550	313
Russian Federation	6,406	8,801	29,026	47,781	1,173		5,123	
Switzerland	1,902	2,899	13,713	21,090	144	58	878	447

Рис. 8. Пример расположения табличных регионов на странице: табличные регионы выделены прямоугольными рамками; вертикальные промежутки табличных регионов выделены штриховкой

Сформулируем общее условие, необходимое для того, чтобы последовательность подряд идущих строк являлась набором строк табличного региона. Пусть L_{pg} — множество всех строк некоторой страницы. В соответствии с описанными эвристиками определяется подмножество строк, из которых формируются табличные регионы на этой странице:

$$L_{tab} = \big\{l : l \in L_{pg} \text{ и } l \text{ удовлетворяет условиям (2)–(4)}\big\}.$$

Определим подмножество вертикальных промежутков строки l, у которых верхняя граница совпадает с верхней границей ограничивающего прямоугольника данной строки следующим образом:

$$G_{\text{top}}(l) = \big\{g : g \in G(l) \text{ и } y_t(g) = y_t(l)\big\}.$$

Кроме того, определим функцию, которая принимает положительные значения, если проекции на ось X ограничивающих прямоугольников двух вертикальных промежутков g и $\tilde{g}$ пересекаются следующим образом:

$$wp(g, \tilde{g}) = \min\{x_r(g), x_r(\tilde{g})\} - \max\{x_l(g), x_l(\tilde{g})\}.$$

Тогда последовательность строк табличного региона $L(t) = \{l_1, \ldots, l_n\}$ должна удовлетворять следующему условию:

$$\begin{cases} l_i \in L_{tab}\ \forall\ i = \overline{1,n}, \\ \forall\ i : i = \overline{1,n-1},\ \forall\ j : j = \overline{i+1,n},\ \forall\ g \in G(l_i)\ \exists\ \tilde{g} \in G_{\text{top}}(l_j) : wp(g,\tilde{g}) \geqslant v_5, \end{cases} \tag{5}$$

где v_5: $v_5 \in \mathbb{R}$ и $v_5 \geqslant 0$ — пороговая величина, которая задает минимально допустимое пересечение проекций на ось X двух заданных вертикальных промежутков (по умолчанию определяется как среднее значение среди интервалов пробелов текстовых элементов, принадлежащих текстовым блокам данных строк l_i и l_j).

Пусть $\{l_p, \ldots, l_{p+q}\}$ — последовательность подряд идущих строк страницы, удовлетворяющая условию (5), причем:

если $p > 1$, то $\{l_{p-1}, l_p, \ldots, l_{p+q}\}$ не удовлетворяет условию (5);

если $p + q < n$, то $\{l_p, \ldots, l_{p+q}, l_{p+q+1}\}$ не удовлетворяет условию (5).

Тогда будем считать, что последовательность $\{l_p, \ldots, l_{p+q}\}$ определяет набор строк табличного региона.

Строки страницы проходятся сверху вниз в поиске последовательностей строк, определяющих табличные регионы. Как только обнаружена такая последовательность, ее строки исключаются из дальнейшего поиска. Таким образом, ограничивающие прямоугольники полученных табличных регионов не будут пересекаться. Далее для каждого табличного региона формируется множество его вертикальных промежутков по следующей схеме. Будем обозначать последовательность строк $\{l_p, \ldots, l_{p+q}\}$ как $l[p, p+q]$. Если при этом последовательность $l[p, p+q]$ удовлетворяет условию (5), то множество вертикальных промежутков, проходящих через последовательность, будем обозначать как $G\big(l[p, p+q]\big)$. Определим, что

$$G\big(l[p,i]\big) = \begin{cases} G(l_p), & i = p\,; \\ app\big(G\big(l[p, i-1]\big), G(l_i)\big), & i = \overline{p+1, p+q}. \end{cases}$$

Функция $app\big(G\big(l[p, i-1]\big), G(l_i)\big)$ формирует множество $G\big(l[p,i]\big)$ по следующей схеме. Пусть $G\big(l[p, i-1]\big) = \{g_1, \ldots, g_n\}$, $n \in \mathbb{N}$, тогда для каждого g_j, $j = \overline{1,n}$, из этого

множества определяется подмножество вертикальных промежутков из $G(l_i)$, у которых верхняя граница совпадает с верхней границей ограничивающего прямоугольника строки l_i и пересечения проекций на ось X каждого из них с проекцией вертикального промежутка g_j превышают порог, определенный в условии (5):

$$\tilde{G}(g_j, l_i) = \{\tilde{g} : \tilde{g} \in G_{\text{top}}(l_i) \text{ и } wp(g, \tilde{g}) \geqslant v_5\}$$

(так как при этом требуется выполнение условия (5), то $\tilde{G}(g_j, l_i) \neq 0$). Пусть для g_j получено следующее подмножество $\tilde{G}(g_j, l_i) = \{\tilde{g}_1, \ldots, \tilde{g}_m\}$, $m \in \mathbb{N}$, тогда далее строится множество вертикальных промежутков, которые должны заменить g_j в $G\big(l[p,i]\big)$ следующим образом:

$$G'(g_j, l_i) = \{g'_k\}, \quad k = \overline{1,m},$$

где $g'_k = \big(\max\{l(g_j), l(\tilde{g}_k)\}, t(g_j), \min\{r(g_j), r(\tilde{g}_k)\}, b(\tilde{g}_k)\big)$.

В результате выполнения данной процедуры для всех g_j получим:

$$G\big(l[p,i]\big) = \Big(\bigcup_{j=1}^{n} G'(g_j, l_i)\Big) \cup \Big(G(l_i) / \bigcup_{j=1}^{n} \tilde{G}(g_j, l_i)\Big).$$

Если $l[p, p+q]$ является последовательностью строк табличного региона t, то будем считать, что $G\big(l[p, p+q]\big)$ составляет его набор вертикальных промежутков.

Полученные табличные регионы могут быть таблицами или частями таблиц. Но также они могут являться текстом (например, текстом с выравниванием по ширине и достаточно разряженным пространством между словами), или колонтитулами страниц с табличной компоновкой, или текстовыми подписями графиков.

5.4. Выделение границ таблиц

Рассматривая полученные табличные регионы, можно заметить, что те из них, которые составляют одну таблицу, коррелируют друг с другом по расположению своих вертикальных промежутков. Например, это выполняется для табличных блоков, показанных на рис. 8. Эта особенность и ряд эвристик о строках, расположенных между текстовыми регионами, составляющими одну таблицу, используются в предлагаемом методе для анализа связности табличных регионов и определения табличных границ. Кроме того, между табличными регионами, составляющими одну таблицу, могут располагаться пустые строки и строки с единственным текстовым блоком. В этом случае непустые строки соответствуют перерезам и/или частям заголовков строк таблицы, а пустые строки изначально могли содержать текстовую разграфку.

Формирование последовательностей связных табличных регионов, являющихся частями одной таблицы, выполняется по аналогии с описанным в предыдущем разделе формированием последовательностей строк, составляющих табличные регионы, но при этом используются иные условия связности.

Пусть t и $\tilde{t}$: $y_t(t) < y_t(\tilde{t})$ — два табличных региона с соответствующими наборами вертикальных промежутков $G(t)$ и $G(\tilde{t})$. Для вертикальных промежутков g: $g \in G(t)$ определим функцию

$$gcorr(g, \tilde{t}) = \begin{cases} 1, & \text{если } \exists\, \tilde{g}\text{: } \tilde{g} \in G(\tilde{t}\,) \text{ и } wp(g, \tilde{g}) \geqslant v_6; \\ 0, & \text{в противном случае,} \end{cases}$$

где v_6 : $v_6 \in \mathbb{Z}$ и $v_6 \geqslant 0$ выполняет функцию, аналогичную пороговой величине v_5, но с другим определением по умолчанию (по умолчанию определяется как среднее значение среди интервалов пробелов текстовых элементов, принадлежащих текстовым блокам строк табличного региона $\tilde{t}$).

Кроме того, пусть $G(t) = \{g_1, \ldots, g_n\}$, $n \in \mathbb{N}$, определим функцию

$$tcorr(t, \tilde{t}) = \sum_{i=1}^{n} gcorr(g_i, \tilde{t}).$$

Будем считать, что два табличных региона t и $\tilde{t}$ являются *связными*, если каждая строка, расположенная между ними, удовлетворяет условию (4), а количество подряд идущих пустых строк между ними не превышает пороговой величины v_7: $v_7 \in \mathbb{Z}$ и $v_7 \geqslant 0$ (по умолчанию $v_7 = 2$) и выполняется следующее условие:

$$tcorr(t, \tilde{t})/|G(t)| \geqslant v_8, \tag{6}$$

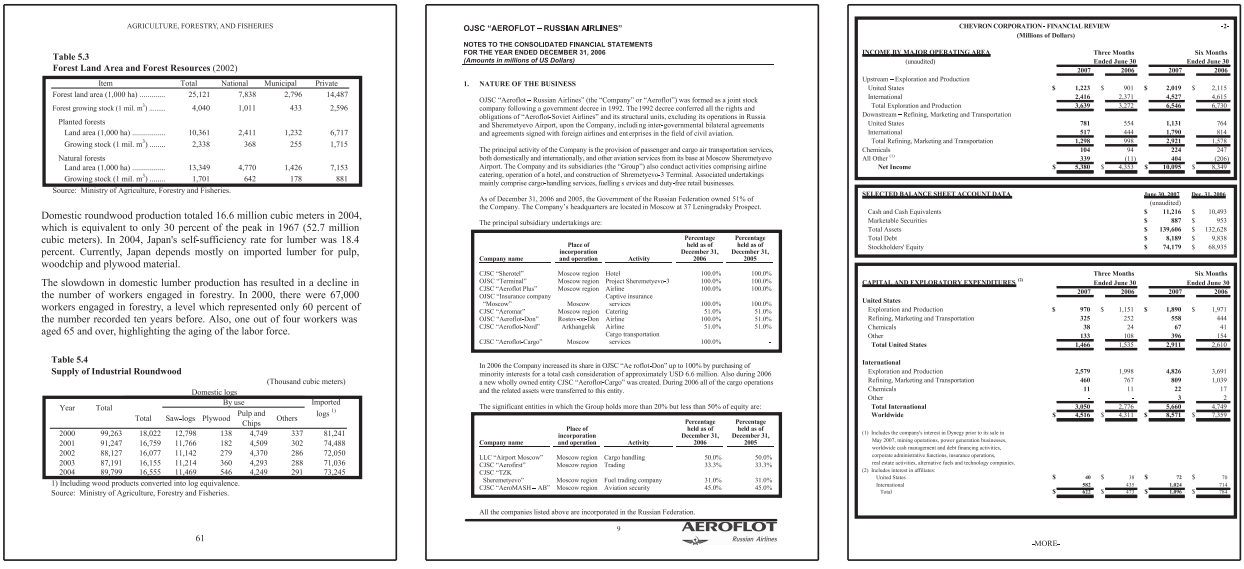
где величина v_8: $v_8 \in \mathbb{R}$ и $0 < v_8 \leqslant 1$ (по умолчанию $v_8 = 0,8$).

Будем считать, что последовательность табличных регионов

$$t_p, \ldots, t_{p+q} : y_t(t_i) < y_t(t_{i+1}), \quad i = \overline{p, p+q-1}, \quad p, q \in \mathbb{N},$$

в совокупности со строками, расположенными между ними, составляет таблицу, если в каждой паре (t_i, t_{i+j}), $i = \overline{p, p+q-1}$, $j = \overline{p+1, p+q}$ табличные регионы являются связными и существуют две таких пары (t_{i-1}, t_k), (t_k, t_{p+q+1}), $k \in [p, p+q]$, в которых табличные регионы не являются связными.

С помощью данного условия на странице выполняется определение прямоугольников, ограничивающих таблицы. При этом если последовательность состоит из единственного табличного региона, содержащего единственную строку, то такая последовательность интерпретируется как текст и исключается из дальнейшего рассмотрения.



а *б* *в*

Рис. 9. Результаты применения предлагаемого метода обнаружения таблиц: прямоугольными рамками выделены области страниц, определенные как таблицы

На рис. 9 показаны примеры результатов применения предлагаемого метода обнаружения таблиц.

Стоит отметить, что в результате применения описанного алгоритма формирования таблиц из последовательностей связных табличных регионов иногда возможны случаи, когда границы сформированных таблиц будут пересекаться (т. е. таблицы будут иметь общие строки). В таких случаях реализация достаточно универсального и точного автоматического разделения общих для пересекающихся таблиц строк требует привлечения анализа и понимания содержания этих таблиц. В описываемом методе в таких случаях предполагается, что пользователь уточняет границы пересекающихся таблиц или выбирает из них те, которые в действительности являются таблицами (поскольку лучше найти лишнюю таблицу, чем не найти имеющуюся).

6. Экспериментальные результаты

Экспериментальная оценка результатов использования предлагаемого метода проводилась в соответствии с подходом, предложенным в работе [10]. Таблица считается корректно обнаруженной, если по крайней мере корректно обнаружено ее тело, т. е. каждая строка в теле таблицы идентифицирована как часть данной таблицы. При этом не допускается, чтобы в качестве строк тела данной таблицы были идентифицированы какие-либо строки, не принадлежащие данной таблице. В этом подходе используются две оценки эффективности метода обнаружения: *точность* (precision) обнаружения — процент количества корректно обнаруженных таблиц к общему количеству обнаруженных таблиц; и *полнота* (recall) обнаружения — процент количества корректно обнаруженных таблиц к общему числу существующих таблиц.

Экспериментальные данные были составлены из статистических отчетов, публикуемых Росстатом[2] и Территориальным управлением Росстата по Иркутской области[3]; из государственных статистических отчетов США[4], Евросоюза[5], Японии[6], а также из финансовых отчетов различных компаний[7]. Указанные документы были представленны в форматах: PDF, DOC, XLS, HTML. Всего было обработано 345 страниц из указанных документов. Каждая страница содержала от одной до четырех таблиц. Всего документы содержали 440 таблиц, из них 134 имели текстовую разграфку. Кроме того, эти страницы содержали текст (включая текст с выравниванием по ширине) верхние и нижние колонтитулы, имеющие табличную форму, графики и диаграммы с текстовыми подписями. В таблице приведены измерения точности и полноты обнаружения для каждого формата.

Экспериментальные результаты показывают применимость предлагаемого метода для обнаружения статистических таблиц в разноформатных документах. Стоит отметить, что точность обнаружения предлагаемого метода можно улучшить после выполнения функционального анализа таблиц. Так, текст, имеющий табличную форму (на-

[2]"Регионы России, социально-экономические показатели 2002" и др.

[3]"Сельское хозяйство Иркутской области 1993–1998" и др.

[4]"Tobacco: World Markets and Trade 2005" и другие отчеты, доступные по адресу www.fedstats.gov.

[5]"Eurostat yearbook 2006-07".

[6]"Statistical Handbook of Japan 2007".

[7]"Boeing Co., Annual Report 2006", "OJSC AEROFLOT—RUSSIAN AIRLINES, Consolidated Financial Statements For the Year Ended December 31, 2006", "ОАО АК ТРАНСНЕФТЬ. Консолидированная финансовая отчетность за год, закончившийся 31 декабря 2006 года" и др.

Точность и полнота обнаружения

Формат	Количество таблиц	Точность, %	Полнота, %
PDF	132	84.1	96.2
DOC	248	80.9	91.9
XLS	45	93.0	88.8
HTML	15	87.5	93.3

пример, подписи к графикам, колонтитулы, списки, оглавления), можно игнорировать, если заметить, что его содержание не соответствует содержанию статистических таблиц, в теле которых преобладают числовые данные.

Заключение

Статистические таблицы имеют существенное сходство в структуре расположения своих компонентов. Это сходство позволило сделать некоторые предположения о таких таблицах и сформулировать эвристики, используемые предлагаемым методом обнаружения таблиц. Использование метафайлов в качестве источника данных в предлагаемом методе позволяет применить этот метод к документам, представленным в разных форматах (например, PDF, DOC, XLS, HTML и др.).

На основе предлагаемого метода разработано приложение для извлечения таблиц из разноформатных документов, выполняющее обнаружение и сегментацию таблиц в документах, представленных в виде метафайлов. В дальнейшем на основе предлагаемого метода может быть построена система извлечения таблиц, конечная цель которой — автоматическое преобразование таблиц из документов в реляционное представление.

Список литературы

[1] Silva A.C., Jorge A.M., Torgo L. Design of an end-to-end method to extract information from tables // Intern. J. on Document Analysis and Recognition. 2006. Vol. 8, N 2. P. 144–171.

[2] Embley D.W., Hurst M., Lopresti D., Nagy G. Table-processing paradigms: a research survey // Intern. J. on Document Analysis and Recognition. 2006. Vol. 8, N 2. P. 66–86.

[3] Lopresti D., Nagy G. A tabular survey of automated table processing // Lecture Notes in Computer Science. Springer. 2000. Vol. 1941. P. 93–120.

[4] Zanibbi R., Blostein D., Cordy J.R. A survey of table recognition: Models, observations, transformations, and inferences // Intern. J. on Document Analysis and Recognition. 2004. Vol. 7, N 1. P. 1–16.

[5] PostScript Language Reference, Third Edition // Addison-Wesley, 1999.

[6] PDF Reference. Fifth edition. Adobe.

[7] Microsoft Developer Network. http://msdn.microsoft.com

[8] Hassan T., Baumgartner R. Table recognition and understanding from pdf files // Proc. 9th Intern. Conf. on Document Analysis and Recognition (ICDAR 2007), IEEE Computer Society. 2007. P. 1143–1147.

[9] Ramel J.-Y., Crucianu M., Vincent N., Faure C. Detection, extraction and representation of tables // Proc. 7th Intern. Conf. on Document Analysis and Recognition (ICDAR 2003), IEEE Computer Society. 2003. Vol. 2. P. 374–379.

[10] Hu J., Kashi R., Lopresti D., Wilfong G. Medium-independent table detection // Document Recognition and Retrieval VII. IS&T/SPIE Electronic Imaging. San Jose, 2000. P. 291–302.

[11] Kieninger T. Table structure recognition based on robust block segmentation // Proc. Document Recognition V, IS&T/SPIE Electronic Imaging. 1998.

[12] Tupaj S., Shi Z., Chang C. H., Alam H. Extracting tabular information from text files // Tufts University, USA, Medford, 1996. http://citeseer.nj.nec.com

[13] Kieninger T., Dengel A. The T-Recs table recognition and analysis system // Lecture Notes in Computer Science. Springer. 1999. Vol. 1655. P. 255–270.

[14] Mandal S., Chowdhury S.P., Das A.K., Chanda B. A simple and effective table detection system from document images // Intern. J. on Document Analysis and Recognition. 2006. Vol. 8, N 2. P. 172–182.

Поступила в редакцию 1 июля 2008 г.