基于标记树的 Web 页面区域划分和搜索方法

胡 飞

(重庆教育学院 重庆 400067)[1] (南京大学计算机科学与技术系 南京 210093)[2]

摘 要 Web 页面的布局可以分为:主要内容、单位标识、导航信息、交互信息和版权申明。我们在处理这些页面时往往只关心主要内容,而且可以从语义上快速定位到主要内容,但是软件系统要做到这一点就非常困难。本文提出一种基于标记树的 Web 页面区域划分和搜索方法,让软件系统可以忽略别的区域,快速定位到主要内容。对于大量 Web 页面处理而言,这种方法可以起到减少时间,缩小空间的作用,Web 页面越多,效果就越显著。

关键词 Web 页面布局,页面结构,页面区域,标记树,标记树模式

How to Get the Main Part of Web Pages

HU Fei

(Chongqing Educational College, Chongqing 400067)[1]

(Department of Computer Science and Technology, Nanjing University, Nanjing 210093)[2]

Abstract A Web page can be divided into several parts, they are "the main part, the department logo, the navigation bar, the hyperlinks and the copyright". How to get the main part of Web pages. It's easy for humankind, but hard for computer pocessing. In this paper we tackle the problem by exploring a tag tree, which can suitably express the structure and the layout of Web pages. Here we propose a method to build the tag tree, in addition to develop a single path tag tree named tag tree model, which only describe the main part of Web pages.

Keywords Web page layout, Web page structure, Web page area, Tag tree, Tag tree model

1 引言

互联网络已成为巨大的、分布广泛的信息源,其主要构成形式是大量的 Web 页面。页面从结构和内容上来看,可以划分为:主要内容、单位标识、导航信息、交互信息和版权申明。用户在访问页面,或者说软件系统在处理页面的时候,往往关心的只是主要内容。图 1 是我们截取的网易新闻(news. 163. com)上的一个页面,其中区域 1 为主要内容,区域 2、3、4 分别为单位标识、导航信息、版权申明,剩下的区域是交互信息。我们在处理这个页面的时候,关心的只是区域 1 中的信息,而忽略别的区域,这样可以减少页面处理在时间和空间上的不必要花费。余勤等[1]提出了一种基于"图像定位"的页面区域划分方法,通过对页面中各部分内容在屏幕中位置、长和宽的不同,来划分出多个区域,该方法能很好地运用到 PDA 等小屏幕设备中,但是对于 Web 挖掘、信息检索(IR)等领域而言,要从"图像定位"的角度对页面结构进行分析处理还比较困难,鉴于此,本文提出了基于标记树的 Web 页面区域划分方法,通过标记树来对页面结构进行分析,然后再划分为多个区域。

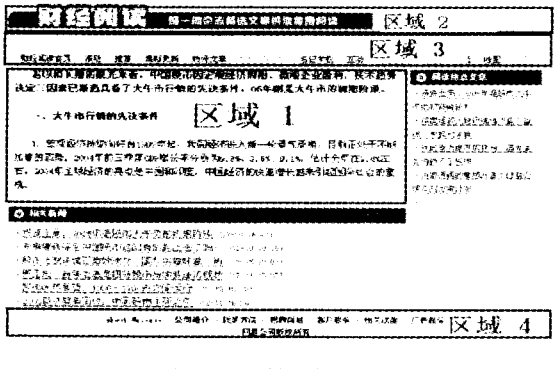

图 1 网易新闻页面

这种方法可以运用于 Web 挖掘、信息检索、PDA 页面访问等多个领域,减少 Web 挖掘中生成的不必要的模式,提高信息检索的查准率和查全率,实现 PDA 等小屏幕设备中页面的智能化区域选择与显示。

2 基于标记树的 Web 页面结构与区域划分方法

2.1 基本定义

为了对一个页面进行结构和布局分析,并最终找到该页面的主要内容区域,应弄清楚什么是主要内容区域? 对于人而言这个问题非常容易理解,我们在浏览一个页面的时候,能立即定位到主要内容区域并获取信息,如图 1 所示,是一个网易上的新闻页面,我们能立即知道区域 1 中的信息是我们所需要的。但是这个过程让软件系统来如何实现呢? 在对这个问题进行探讨之前,我们必须弄清楚如下几个定义。

定义 1 在一个页面中,每个区域可以用六个变量来描述:width、height、left_start、right_end、top_start、bottom_end,分别表示区域的左边界、右边界、上边界、下边界等在页面中的位置,单位为像素。

定义 2 对于某个区域,当 Left_start $<\chi<$ right_end、top_start $<\delta<$ bottom_end 的时候,定义该区域为主要内容区域,或者为主要内容区域里面的小区域,其中 χ 和 δ 是常量,单位为像素。

因为主要内容区域中可能还会由许多更小的区域构成,仅仅(χ,δ)不能定位到整个主要内容区域上,而可能定位到里面更小的区域上,所以这里要加入一个阈值 θ,如下面定义 3。

定义 3 函数 $f=width\times\alpha+height\times\beta$,其中($width$, $height$)定义为某一区域的宽度和高度,$0\leqslant\alpha\leqslant1$,$0\leqslant\beta\leqslant1$,对于主要内容区域里面嵌套的区域,其 $f<\theta$,而对于主要内容区域,其 $f\geqslant\theta$,其中 θ 为阈值。

在定义 1 中,变量 left_start 和 right_end 还可以用百分

比(%)作为其单位,在一些页面中也是这样做的,这时候,我们只要把定义2中常量 χ 的单位也改成百分比,并且重新修正这个值即可。下面为了讨论方便,都以像素作为默认单位。另外变量 *width* 和 *height* 只是在计算其余四个变量的时候起辅助作用,描述一个区域的位置只要其余四个变量即可,关于这个辅助作用将在2.3节中进行叙述。

2.2 页面结构分析

Web页面语言,如HTML,是标记语言,即通过标记来标注文本文档。这些标记反映了页面文档的数据结构,同时能通过这些标记来分析页面文档的布局,把页面文档划分为多个区域,正如图1所示,该页面就被划分成5个区域:主要内容、单位标识、导航信息、版权申明、交互信息。如何得到一棵标记树,使之既能表达页面的结构,也能表达页面的布局。一旦这棵能表达页面结构和布局的标记树生成之后,剩下的工作就仅仅是沿着这棵标记树自上而下寻找一个主要内容的标记树模式了,关于这部分工作拟在第3节中讨论。

在文[2,3]中提到通过页面链接(links)信息来区分具有不同页面结构的页面,这种方法可以很好地运用于页面结构的分析上。另外在文[4, 5]中都提到过标记树的构建方法,这些标记树只能表达页面结构,标记语言中有许多标记符号,如〈table〉、〈ul〉、〈iframe〉、〈image〉等,其中以〈table〉标记符,包括其内部嵌套的〈tr〉和〈td〉标记符,最能体现页面的结构布局,而且互联网上绝大部分网站页面都是用〈table〉标记符来对页面布局的,例如下面的标记符例子,表示一个表格,该表格宽300像素,高400像素。

〈table width="300" height="400"〉〈tr〉〈td〉〈/td〉〈/tr〉〈/table〉

本文中仅仅用〈table〉标记符来构建标记树,而该标记树足以表达页面结构和布局。

图2是图1中页面的简化标记树,去掉了许多细节,只保留了6个区域的数据结构(其中交互信息区域被划分成了两个区域:交互信息1、交互信息2。),节点root是假想的根节点,代表0级,也就是最上面一级,你可以把它看作标记〈body〉。

该标记树还不能反映页面布局,关于这一点我们将在后面继续改造这棵标记树。

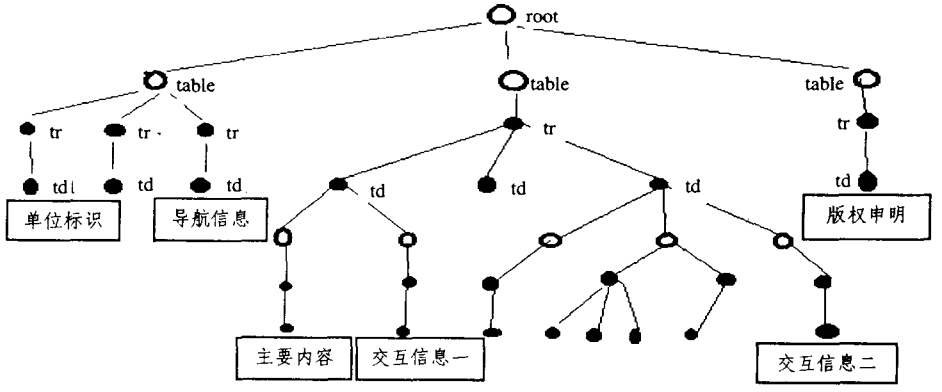

图2 简化标记树

2.3 页面区域划分方法

为了让图2中的标记树能表达图1中页面的布局,引入定义1中的六个变量:width、height、left_start、right_end、top_start、bottom_end,这六个变量赋予标记树中的〈td〉和〈table〉节点,因为所有区域都可以认为是放在某一个单元格里面的,而一个〈td〉就表示了一个单元格;对于〈table〉对应的变量值而言,可以由其内嵌的所有〈td〉标记推断出来。所以下面要做的就是怎样为〈td〉节点计算出其相应的六个变量值,这是本节的重点所在。下面就以图2中的标记树为例来说明如何计算各〈td〉相关的六个变量值。

首先,采用自下而上的方法计算所有〈td〉的width和height两个变量值。以图2中主要内容区域所对应的〈td〉节点为例,该区域的HTML代码如下(为了版面整洁和便于理解,对其中内容做了一些修改,删除了许多文本内容):

```
〈TD width="538"〉
〈P〉若以稍长期的眼光来看,〈/P〉
〈P〉一. 大牛市行情的先决条件〈/P〉
〈img width="125" height="46" src="image/icon. gif"border="0"〉
〈P〉〈/P〉
〈P〉1、宏观经济持续向好自1999年起。〈/P〉〈/TD〉
```

从〈TD width="538"〉可以初步判断该区域width="538"、height="0"。再看里面的内容,一个图片占据的宽度为125,得到 width′=125,则我们得到本区域的真实宽度 width=Max(width, width′)。对于高度,首先计算字体的高度,由前面的样式表可知,字体高度为12(即12像素),而一段连续的字"若以稍长期的眼光来看"共有10个,则该段字所占的高度为⌈10×12÷538⌉×12=12,一个〈p〉〈/p〉的高度为25像素,一个图片的高度为46,则本内容区域的高度 height′=12×3+25×3+46=157,则得到本区域的真实高度 height=Max(height, height′)。所以本区域所对应〈td〉的width和height值为(538, 157)。

通过同样的方法也可以得到本区域的兄弟区域"交互信息一"的width和height值,在此为了叙述简便,不再做计算,假设其值为(438, 350)。现在再来看看上一级〈td〉所对应的HTML代码(已简化,其中左边为计算前的HTML代码,右边为计算后带有width和height修正值的HTML代码),如表1所示。

嵌套在最里面的两个〈td〉〈/td〉标记的width和height属性的字体为斜体,表示其标注的值不是HTML代码中的初始值,而是通过前次计算后得到的修正值,真实反映了这两个〈td〉〈/td〉所对应区域在页面中的宽度和高度。首先计算里面两个〈table〉的宽度和高度,第一个〈table〉的宽度 width_table1=Max(556, 556),高度 height_table1=(0, 157),得到第一个〈table〉修正后的 width_table1 和 height_table1 为(556,157),同理,第二个〈table〉的 width_table2 和 height_table2 修正值为(556, 350),则本级〈td〉里面的内容的宽度

width′ = Max (width_table1， width_table2)，height′ = height_table1 + height_table2，为(556，507)，而本级〈td〉的标注宽度和高度为(556，304)，则最终的到本级〈td〉的修正宽度 width 和高度 height 为(556，507)。如此递归，从下往上计算出所有〈td〉的修正(width，height)值。最后我们得到一棵标记树，该树的所有〈td〉节点都包含修正后的(width，height)属性值，如图 3 所示，为了便于理解，对一些标记做了标签，例如 table(1)、table(2)它们都表示〈table〉标记符，而 td(1)、td(2)都表示〈td〉标记符。

表 1 〈td〉所对应的 HTML 代码

计算前	计算后，width 和 height 已修正
〈TD width="556" height="304"〉 〈TABLE width="556"〉 〈TR〉 〈TD width="556" height="157"〉 主要内容区域〈/TD〉〈/TR〉〈/TABLE〉 〈TABLE height="25"〉 〈TR〉 〈TD width="556" height="350"〉 信息交互一〈/TD〉〈/TR〉〈/TABLE〉〈/TD〉	〈TD width="556" height="507"〉 〈TABLE width="556" height="157"〉 〈TR〉 〈TD width="556" height="157"〉 主要内容区域〈/TD〉〈/TR〉〈/TABLE〉 〈TABLE width="556" height="350"〉 〈TR〉 〈TD width="556"height="350"〉 信息交互一〈/TD〉〈/TR〉〈/TABLE〉〈/TD〉

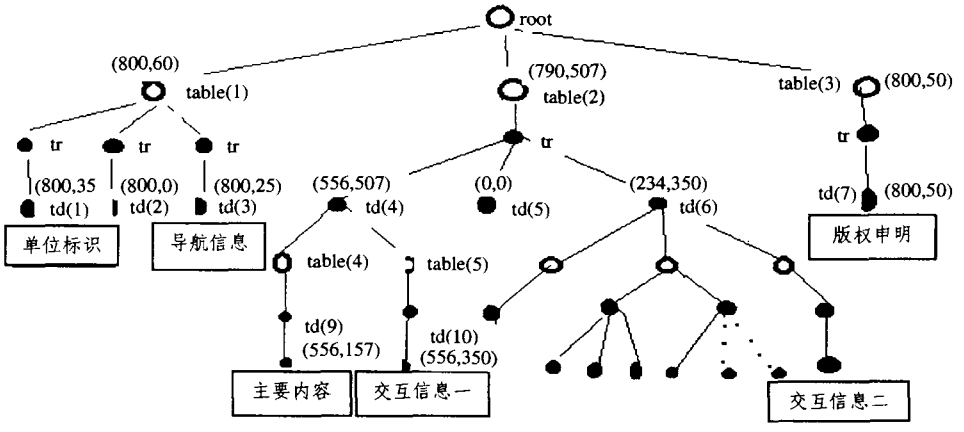

图 3 标记树

在图 3 中还有两个节点是通过虚线和其上一级节点相连的，即两个〈td〉节点，那么它们表示什么意思呢？这两个节点实际上是不存在的，我们称其为虚节点，它们是为了后面方便对其他实节点计算服务的。下面通过表 2 中的 HTML 代码(本代码不来自于任何页面，是为了对虚节点概念进行说明而随机编写的代码)，来详细说明一下什么是虚节点。

从表 2 中我们可以看出，虚节点分为三种情况。第一种情况是当〈td〉节点中既有文本内容和图片，也嵌套有〈table〉节点的时候，我们要把该文本内容和图片转换为一个〈table〉虚节点，作用是文本内容和图片所在区域的宽度和高度能在标记树中体现出来；第二和第三种情况是〈td〉节点中的 colspan 和 rowspan 两个属性，该属性是单元格合并属性，破坏了标记树中结构的完整性，让人无法理解其真实的结构，在这里我们引入〈td〉虚节点的目的就是恢复标记树的完整性，该虚节点不占空间位置，其六个变量值都为 NULL。

表 2 虚节点

	真实 HTML 代码	加入虚节点之后修正的 HTML 代码
HTML 代码	〈table〉〈tr〉〈td〉 这是一段文本内容和图片：〈br〉〈/br〉 〈img width="120" height="120" src=test. jpg〉 〈table width="400"〉 〈tr〉 〈td colspan="2" width="390"〉 这是合并的单元格 colspan=2〈/td〉〈/tr〉 〈tr〉 〈td rowspan="2" width="273"〉 这是合并的单元格 rowspan=2〈/td〉 〈td width="111"〉〈/td〉〈/tr〉 〈tr〉 〈td width="111"〉〈/td〉〈/tr〉〈/table〉 〈/td〉〈/tr〉〈/table〉	〈table〉〈tr〉〈td〉 〈table width="120" height="140"〉 〈tr〉 〈td〉〈/td〉〈/tr〉〈/table〉 〈table width="400"〉 〈tr〉 〈td width="390"〉 这是合并的单元格 colspan=2〈/td〉 〈td〉〈/td〉〈/tr〉 〈tr〉 〈td width="273"〉 这是合并的单元格 rowspan=2〈/td〉 〈td width="111"〉〈/td〉〈/tr〉 〈tr〉 〈td〉〈/td〉 〈td width="111"〉〈/td〉〈/tr〉〈/table〉 〈/td〉〈/tr〉〈/table〉

下一步要做的就是依照图 3，从上往下计算所有〈td〉节点和〈table〉节点的(left_start，right_end，top_start，bottom_end)四个变量值。在计算之前，我们发现一些有趣的规律，由此给出下面的定义。

定义 4 〈td〉节点(包括〈root〉节点)内嵌套的所有〈table〉节点，将继承〈td〉节点中的 left_start 变量值；〈table〉节点中的所有〈tr〉节点，将继承〈table〉节点的 left_start；〈tr〉节点中的所有〈td〉节点，将继承〈tr〉节点中的 top_start。

定义5 同一个〈tr〉节点下的所有〈td〉节点，其变量height的值都统一为最大值。

由定义5可知，〈td(4)〉、〈td(5)〉、〈td(6)〉的(width, height)在修正后分别为(556,507)、(0,507)、(234,507)。

在此我们采用广度优先搜索的算法。首先〈root〉节点的四个变量值为(0, 0, 0, 0)，依照广度优先搜索算法，我们先遍历与〈root〉节点相邻的三个节点，按照从左到右的顺序〈table(1)〉、〈table(2)〉、〈table(3)〉。〈table(1)〉的left_start和top_start继承其上一级节点，为0，right_end=left_start+width，bottom_end=top_start+height，则其四个变量值为(0, 800, 0, 60)；依照定义4，〈table(2)〉和〈table(3)〉都集成了上一级的left_start，〈table(2)〉的top_start=〈table(1)〉的bottom_end，〈table(3)〉的top_start=〈table(2)〉的bottom_end，这样我们就得到了〈table(2)〉的四个变量值为(0, 800, 60, 567)，〈table(3)〉的四个变量值为(0, 800, 567, 617)。

然后再依次得到〈table(1)〉下面的〈td(1)〉、〈td(2)〉、〈td(3)〉和〈table(2)〉下面的〈td(4)〉、〈td(5)〉、〈td(6)〉。表3是图3中通过计算得到的部分〈table〉和〈td〉节点的(left_start, right_end, top_start, bottom_end)。

表3 节点(left_start, right_end, top_start, bottom_end)

标记节点		标记节点		标记节点	
〈table(1)〉	(0,800,0,60)	〈table(2)〉	(0,800,60,567)	〈table(3)〉	(0,800,567,617)
〈table(4)〉	(0,556,60,217)	〈table(5)〉	(0,556,217,567)		
〈td(1)〉	(0,800,0,35)	〈td(2)〉	(0,800,35,35)	〈td(3)〉	(0,800,35,60)
〈td(4)〉	(0,556,60,567)	〈td(5)〉	(556,556,60,567)	〈td(6)〉	(556,800,60,567)
〈td(7)〉	(0,800,567,617)	〈td(9)〉	(0,556,60,217)	〈td(10)〉	(0,556,217,567)

至此，标记树所有〈table〉和〈td〉节点的(left_start, right_end, top_start, bottom_end)四个变量值都已得到，这时候的标记树除了能反映页面的数据结构以外，还能清晰地反映出页面的各区域布局。在下面的工作中我们将从上到下，沿着该标记树找到页面的主要内容区域。

3 搜索主要内容区域

依照定义2，为了找到主要区域，我们得人为设定两个参数：χ和δ。(χ,δ)为页面中的一个点，χ表示该点距页面左边距的宽度，δ表示页面距页面上边距的高度。在这里我们假设点(χ,δ)必然落在，也仅仅落在页面主要区域中。以图3和表3所对应的页面为例，假设$(\chi,\delta)=(400, 200)$。

仍依照广度优先搜索算法，首先比较与顶级节点〈root〉相邻的三个节点，发现(χ,δ)落在〈table(2)〉所对应的区域中；然后再比较与〈table(2)〉节点相邻的三个〈td〉节点，发现(χ,δ)落在〈td(4)〉所对应的区域中；然后是落在〈table(4)〉所对应区域中；〈td(9)〉所对应区域中。

〈td(9)〉节点下面可能还会有更多，更深层次的内嵌节点，我们如何判断该到什么时候结束向下搜索呢？这时候我们引入定义3。当计算得知(χ,δ)落在某个节点所对应区域中时，我们还要计算该节点所对应f值，当$f>\theta$时，还要继续往下寻找更小区域，当$f<\theta$时，回溯到上一级节点，并认为该上一级节点所对应的区域就是主要内容区域。

这里假设$\theta=300$，$\alpha=0.5$，$\beta=0.5$，由〈td(9)〉节点计算得知其$f=356$，比θ大，则继续往下搜索，当发现下一级没有同时符合定义二和定义三条件的节点时，就回溯到〈td(9)〉，并认为〈td(9)〉所对应区域就是主要内容区域。而沿着〈root〉、〈table(2)〉、〈td(4)〉、〈table(4)〉、〈td(9)〉的一棵单路径树(single path tree)就是标记树模式。

对于同一网站同一主题的所有页面而言，例如网易新闻的所有新闻页面，标记树都是唯一的。我们可以从中选取一部分页面(如50个)作为训练集，人为设定χ、δ、α、β、θ五个参数，通过这个训练集得到的各个标记树模式和训练集里面的页面比较对比，不断调节五个参数，以期得到较好的参数值，并最终得到一个比较统一的标记树模式。然后再把这个标记树模式运用于所有页面。

结束语 本方法在Web挖掘、信息检索、PDA小屏幕设备显示方面都有积极的意义，尽管事先要人为确定χ、δ、α、β、θ五个参数，这对于新手而言可能有一定难度，但是我们还会从待处理页面中选取一小部分作为训练集，通过训练集来不断调整更新这五个参数。所以即使这五个参数在初始设置的时候不太精确，也能在后面的调整中得到比较满意的值。

对于一些HTML页面的标记结构不严谨可能对页面结构分析带来的困难，可以事先把这些页面转换为DHTML或者XML格式，在文[6,7]中都有提到这种转换的过程和方法。

鸣谢： 拙作是在南京大学计算机科学与技术系博士生导师陈世福悉心指导下完成的，在此表示由衷感谢。

参考文献

1 Chen Yu, et al. Detecting web page structure for adaptive viewing on small form factor devices. In: Proc. of the 11th World Wide Web Conf. (WWW 12), 2003

2 Crescenzi V, Merialdo P, Missier P. Fine-grain web site structure discovery. In: Proc. of the fifth ACM intl. workshop on Web information and data management, New Orleans, Louisiana, USA, 2003

3 Gedov V, et al. Matching web site structure and content. WWW (Alternate Track Papers & Posters)2004. 286～287

4 张文斌，等. 一种基于多叉树的HTML到XML的转换方法. 小型微型计算机系统，2004，24(4)

5 常育红，等. 基于标记树表示方法的页面结构分析. 计算机工程与应用，2004，16

6 唐翔弘，等. 基于Web的数据采集. 计算机科学，2004，31(8)：24～26

7 吴扬扬，等. 识别和抽取Web列表中的关系信息. 计算机科学，2004，31(6)：86～88