基于规则的网页分割预处理算法研究

彭红超[1] 童名文[1] 邹军华[2] 郝秋红[1]

(华中师范大学信息与新闻传播学院 武汉 430079)[1] (湖北大学教育学院 武汉 430070)[2]

摘 要 针对国家精品课程网站中网页内容和样式独立设计,网页分割算法难以运行的问题,基于规则提出了一种网页分割预处理算法,建立了网页标签和样式信息的关联。算法包括3个步骤:第一,获取样式信息;第二,关联样式信息和标签;第三,输出HTML和PerfectNode关联类列表。随机选取了100个国家精品课程网站的网页运行预处理算法,实验结果表明该算法可以有效地融合网页标签和样式信息,解决了网页分割算法无法运行的问题。

关键词 网页分割,预处理算法,级联样式表,样式信息

中图法分类号 TP311.1 **文献标识码** A

Rule-based Preprocessing Algorithm for Web Page Segmentation

PENG Hong-chao[1] TONG Ming-wen[1] ZOU Jun-hua[2] HAO Qiu-hong[1]

(College of Information Technology, Journalism and Communications, Central China Normal University, Wuhan 430079, China)[1]

(Faculty of Education, Hubei University, Wuhan 430070, China)[2]

Abstract Since the independent design between web contents and styles of National Level Excellent Courses, web page segmentation algorithm can hardly run. We present a rule-based preprocessing algorithm of web page segmentation to create correlation between tags and style information. The algorithm consists of three steps: first, get the style information; second, associate styles with tags; third, output HTML and PerfectNode which is associated class list. We selected 100 pages from the National Level Excellent Courses randomly to run the preprocessing algorithm. Experimental results show that the algorithm can associate tags with styles efficiently, which can solve the problems that web page segmentation algorithm cannot run.

Keywords Web page segmentation, Preprocessing algorithm, Cascading style sheets, Style information

1 引言

自2003年以来我国已经建设了3835门国家级精品课程(NLECs)。随着微电子技术、移动通信技术的迅速发展和非正式学习理论在我国的迅速流行,学习者开始通过移动终端访问NLECs网页。然而,NLECs网页绝大多数针对个人电脑而设计,不适合小尺寸移动终端呈现和处理,学习者几乎不能随时、随地地正常访问这些网页。针对网页在移动终端无法正常呈现和处理的问题,国内外学者提出将网页分割成较小子块的解决方法,并提出了一些网页分割算法。其中多数算法都基于样式信息或视觉信息实现网页分割[1-8]。这些算法要求样式信息包含在网页标签中,以标签属性的形式与标签紧密关联。然而,随机选取50个不同的NLECs网站,对每个网站随机获取一个网页,对得到的50个网页进行统计分析,有92%的网页采用级联样式表实现网页样式设计,这些网页的标签和样式信息失去了直接的关联,因此网页分割算法无法直接对它们进行分割处理。另外,由于CSS处理样式信息的灵活性和多样性,致使样式信息存在冲突,通过对上述50个网页中的样式信息进行统计,发现存在1863个冲突,这些冲突将影响网页分割算法的效果,如一个CSS文件中有样式信息"p{font:12px/1.5;}",另一CSS文件中有样式信息"p{font-size:16px;}",这两条样式信息符合CSS标准,却存在冲突,目前的网页分割算法无法正确地判断标签〈p〉的font-size属性应该取何值。所以,网页在分割前需要做预处理,以解决样式信息冲突和标签与样式信息分离问题,从而为网页分割算法正常运行提供条件。目前为止,网页分割预处理多为人工处理,文献研究中尚未发现实现自动预处理的算法。但是研究网页分割预处理算法对于网页自动分割尤其是大批量分割处理非常重要。

基于CSS级联样式规则提出网页分割预处理算法,算法包括3个步骤:第一,获取样式信息;第二,关联样式信息和标签;第三,输出HTML和PerfectNode关联类列表。随机选取100个不同的NLECs网站,对每个网站随机获取一个网页,对得到的100个NLECs网页采用预处理算法进行处理。

本文受教育部人文社科基金项目:移动学习服务适配决策技术及优化策略研究(10YJC880113),国家科技支撑计划课题:全媒体在线编辑与适配推送数字出版技术研究及应用示范(2013BAH30F01),中央高校基本科研业务费项目:泛在多媒体服务中内容适配决策模型及优化研究,中央高校基本科研业务费项目:数字化学习环境及工具的典型应用(CCNU10C01003)资助。

彭红超(1987—),男,硕士生,主要研究方向为教育信息资源设计与开发,E-mail:hongchao5d@qq.com;童名文(1975—),男,博士,副教授,硕士生导师,主要研究方向为多媒体内容适配、教育资源管理与服务;邹军华(1972—),男,博士,副教授,硕士生导师,主要研究方向为算法设计与分析;郝秋红(1991—),女,硕士生,主要研究方向为教育信息资源设计与开发。

实验结果及分析表明，该处理算法可以有效地对网页进行预处理，准确地将 NLECs 网页和样式信息进行融合，这有利于后续对 NLECs 进行准确的分割。

2 预处理算法设计

2.1 样式信息获取

预处理算法第一步获取 HTML 网页中的样式信息。HTML 使用样式信息的方法和位置非常灵活，主要有以下 3 种情况：(1)使用〈link〉标签引用 CSS 文件，〈link〉标签只能位于〈head〉标签中，数量可以是多个；(2)使用〈style〉标签定义样式信息，〈style〉标签的位置不确定，数量可以是多个；(3)通过 HTML 标签的 style 属性指定样式信息，但并不是每个标签都有 style 属性。

通过分析上述 3 种情况，提出样式信息获取算法以便获取 HTML 中的样式信息，其伪代码如图 1 所示。该算法依次从〈link〉标签、〈style〉标签、style 属性值中获取样式信息。这是因为，〈link〉标签引用的样式信息优先级最低，style 属性值定义的样式信息优先级最高，只有按照优先级从低到高依次获取，才能避免由优先级引起的冲突。此外，采用该顺序获取样式信息，为后面样式冲突处理做准备。

```
Input:domTree ▷DOM tree of web page
Output: styleInfos = {(selector1, styleInfoList1), …, (selectorn, styleInfoListn)} ▷StyleInfomation

Procedure getStyleInfos(domTree)
begin
  styleInfos ← {};
  rootNode ← getRootNode(domTree);
  headNode ← getHeadNode(rootNode);
  linkNodes ← getLinkNodes(headNode);
  for each linkNode ∈ linkNodes do
    styleInfos ← styleInfos ∪ getStyleInfoFrom(linkNode);
  enddo
  styleNodes ← getLinkNodes(rootNode);
  for each styleNode ∈ styleNodes do
    styleInfos ← styleInfos ∪ getStyleInfoFrom(styleNode);
  enddo
  tagAndStyleAttrValues ← getTagAndStyleAttrValues(rootNode);
  for each tagAndStyleAttrValue ∈ tagAndStyleAttrValues do
    styleInfos ← styleInfos ∪ getStyleInfoFrom(tagAndStyleAttrV-
    alues);
enddo
return styleInfos
end
```

图 1 样式信息获取算法伪代码

〈link〉标签与〈style〉标签中的样式信息，每条样式规则都含有选择器，以便对应标签，如“body{color:red;}”，该条样式规则的选择器是“body”。然而，style 属性值中所定义的样式规则却没有选择器。所以，在获取 style 属性值中的样式信息时(图 1 中的 getStyleInfoFrom(tagAndStyleAttrValues)方法)，需要构造选择器。因为 style 属性值中的样式信息的优先级最高，构造选择器算法依据标签的所有祖先标签及其自身(按辈分从大到小排列)来构造选择器。对每个祖先标签，优先考虑标签的 id 属性值，再考虑 class 属性值，如果这两个属性值都为空，才考虑标签的名称；而对自身标签，为了保证标签具有唯一性，只考虑它的 id 属性值，如果没有 id 属性，须给该标签设置一个。只有这样，构造的选择器才会准确无误地对应到原标签，且不会因优先级的问题而被其他样式信息覆盖。

2.2 关联样式信息和标签

由于网页内容和样式独立设计，网页内容和样式在结构上分离，网页中几乎不含样式信息；另外，分割后，形成块的 DOM 树结构发生了改变，样式信息与标签无法正确对应。因此网页分割要求样式信息、标签在结构上具有高度相关性。针对该问题，提出样式信息-标签关联算法，将得到的样式信息与网页标签关联，实现网页和独立样式信息融合，以便后续 NLECs 准确地分割。该算法分 3 步：①查找对应标签；②定义“样式信息-标签”关联类；③样式冲突检测与处理。

2.2.1 查找对应标签

关联样式信息和标签，首先需要将结构上分离的样式信息与标签进行对应，即根据样式信息查找对应标签。根据 CSS 级联样式规则，每条样式规则根据其选择器与标签对应，这些选择器可归为 2 类：①基本选择器：即不含“:”或“::”的选择器，如属性选择器(E[attr])等；②特殊选择器：即含“:”或“::”的选择器，如伪类选择器、伪元素选择器等。每种选择器根据不同的规则对应标签，为此，提出查找对应标签算法以准确地查找标签。该算法有 5 个步骤：①判断是否是特殊选择器；②分割选择器；③设置优先级；④查找对应标签；⑤合并选择器。

特殊选择器较普通选择器“特殊”的地方在于：普通选择器仅用于对应标签，而特殊选择器除对应标签的部分(普通部分)外，有非对应标签的部分，即“特殊部分”。所以，查找对应标签算法第①步判断当前选择器是否为特殊选择器，如果是，则把特殊选择器分为两部分：“普通部分”和“特殊部分”，分割符为“:”或“::”。“普通部分”用来执行第②步；如果不是特殊选择器，直接进行第②步。

在第②步分割选择器阶段，首先判断选择器是否为群组选择器，如果是，则将该条样式规则拆分为多条等价非群组选择器的样式规则。然后对非群组选择器进行分割。分割算法步骤如下：①将选择器去掉头尾无效空格后转换成字符数组；②如果某字符在“[”和“]”中，则不分割；③如果某字符在“(”和“)”中，则不分割；④其余字符执行 doSplit 方法，该方法定义的规则如表 1 所列，selectorInfoList 用于存储分割后的选择器信息，sb 用于临时存储字符，preChar 与 postChar 分别表示前一个字符与后一个字符。经过上述分割选择器算法之后，便以运算符为分割符将选择器分为多个片段，例如选择器“body a”经过分割形成“body”、“ ”、“a” 3 个部分，其中“ ”是运算符。

表 1 doSplit 方法规则

规则 1	如果 preChar ∈{字符，‘#’，‘.’，‘:’，‘-’，‘]’，‘*’}并且 postChar ∈{字符，‘#’，‘.’，‘:’，‘-’，‘]’}，那么 postChar 加入 sb 中；
规则 2	如果 preChar ∈{字符，‘#’，‘.’，‘:’，‘-’，‘*’}并且 postChar ∉{字符，‘#’，‘.’，‘:’，‘-’}，那么将 sb 中的字符加入到 selectorInfoList 中，清空 sb 后，把 postChar 加入到 sb 中；
规则 3	如果 preChar 是空格而 postChar 是‘>’或‘+’，则把 sb 最后一个字符用 postChar 替换。

在第③步设置优先级阶段，根据分割形成的片段设置优先级，该操作主要为后面样式冲突处理做准备。CSS 级联样

式规则对优先级的规定为：〈link〉中的样式信息＜〈style〉中的样式信息＜style 属性值。而对于同一位置的样式规则，其优先级 p 如下：

• 每个 id 选择器(#idName)，加 100；

• 每个 class 选择器(.className)、每个属性选择器(形如[attr=value]等)、每个伪类(形如:hover 等)，加 10；

• 每个元素或伪元素(:firstchild)，加 1；

• 其它选择器包括通配符选择器 *，加 0。

根据上述规定，设计优先级设置算法，该算法的规则定义如表 2 所列。这样，样式规则的优先级最终值 priorityLevel=基础优先值+p。

表 2　优先级设置算法规则

规则 1	如果该选择器来自〈link〉标签，则加 0 作为基础优先级；
规则 2	如果该选择器来自〈style〉标签，则加 1000 作为基础优先级；
规则 3	如果该选择器来自 style 属性值，则加 2000 作为基础优先级；
规则 4	对于具体的选择器则用基础优先级加上 p。

第④步查找对应标签，根据分割形成的片段在 DOM 树中查找对应标签。在 HTML 转换成的 DOM 树中，某一节点最多只有一个父节点，其子节点则不确定。所以，当选择器为后代选择器时，采用先子后父的查找方法，这样可以提高查找效率。最后，执行第⑤步，将分割后的选择器进行合并，以便后续使用。

2.2.2　定义“样式信息-标签”关联类

〈link〉、〈style〉标签中，样式信息的每条样式规则含有选择器，而 style 属性值中的样式信息只有样式属性，由于特殊选择器(如，伪类选择器、伪元素选择器等)的存在，不可能将样式信息去掉选择器后加入到 style 属性中。例如，样式规则“a:link {color:red;}”，表示点击链接之后，〈a〉标签的字体变为红色。如果把该样式规则直接去掉“a:link”选择器后加入到 style 属性值中，它的作用变为设置〈a〉标签字体为红色。这会造成不可预知的后果。因此，需要一种特殊的方法，将样式信息与标签关联。

针对该问题，定义新类型 PerfectNode 来存储之前所有操作的结果，并建立“样式信息-标签”关联。PerfectNode 的类图如图 2 左图所示。PerfectNode 类中的 tagNode 属性存放查找到的标签，styleInfos 属性存放 tagNode 的样式规则。这样，在 PerfectNode 类中，就建立了“样式信息-标签”的一一对应关系。

由于每个标签的样式规则一般只有几个，所以，在 styleInfos 属性中使用数组列表类型。另外，每条样式规则用新定义的 StyleRule 类型存储，如图 2 右图所示。在 StyleRule 类中，selector 属性存放合并后的选择器，priorityLevel 属性存放设置好的优先级，styleAttrs 存放该选择器所有的样式属性。需要注意，由于每个样式规则会有较多的样式属性，加之，网页分割时会频繁地搜索样式规则，所以，采用 Hash 链表存储样式属性，这样可以提高预处理算法的性能。

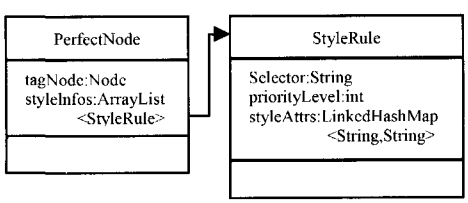

图 2　PerfectNode 类图与 StyleInfo 类图

2.2.3　样式冲突检测与处理

经过上述步骤之后，会得到一个 PerfectNode 关联类列表。该列表包含某一 HTML 网页所有样式规则及其对应的标签，且实现了“样式信息-标签”的关联。不幸的是，由于样式信息具有灵活性和多样性，其本身存在显性冲突和隐性冲突。在解决样式冲突之前，这些信息不能为网页分割服务，否则会得到无法预计的结果。针对这个问题，设计样式冲突检测与处理算法来解决样式冲突，伪代码如图 3 所示。该算法将显性冲突分为以下 3 类：

• PerfectNode 列表中 tagNode 属性相同，而 StyleInfos 属性相同或者不同；

• StyleInfos 列表中的 selector 属性相同，而 styleAttrs 属性相同或者不同；

• 样式属性名相同，而样式属性值相同或者不同。

隐性冲突指属性名不相同，但依然会引起的冲突，该冲突与因属性名相同引起的冲突相同。该类冲突由简单属性和复合属性的混合使用造成。在实际开发中，Web 编程人员经常混合使用这两类样式属性，当某一复合属性包含另一简单属性时，就会产生隐性冲突。

```
Input:perfectNodes
Output:perfectNodes

Procedure styleConflictProcessing(perfectNodes)
begin
  perfectNodes ← splitCompositeAttr(perfectNodes);
  for perfectNode1,perfectNode2 ∈ perfectNodes do
    if perfectNode1. tagNode==perfectNode2. tagNode then
      if not eachSelector in perfectNode1. styleInfos and in perfect-
      Node2. styleInfos is different then
        styleRule1 ∈ perfectNode1. styleInfos;
        styleRule2 ∈ perfectNode2. styleInfos;
        if selector1 in styleRule1 is same as selector2 in styleRule2
        then
          if not eachAttr in styleRule1 and in styleRule2 is different
          then
            if attr1 in styleRule1 is same as attr2 in styleRule2 then
              merge(attr1,attr2);
            endif
          endif
        merge(styleRule1,styleRule2);
        endif
        merge(perfectNode1. styleInfos,perfectNode2. styleInfos);
      endifa
      merge(perfectNode1,perfectNode2);
    endif
  enddo
  return perfectNodes
end
```

图 3　样式冲突检测与处理算法伪代码

由于这些样式冲突的存在，如果直接获取某标签样式信息时，就会得到无法预计的结果。例如样式规则“body{font-size:16px; font: 12px/1. 5;}”，font 指定的字体大小为“12px”，font-size 指定的字体大小为“16px”。在网页分割阶段，如果通过 font-size 获取字体大小，其值为 16px，而通过 font 获取字体大小，其值却为“12px”；如果通过“line-height”

属性获取行高，其值却为空，但事实上，行高属性是存在的，其值为“1.5”。

样式冲突检测与处理算法首先将隐性冲突显性化。采用的方法是将复合属性分为多个等价的简单属性，该方法也简化了样式属性的获取操作，如上例样式规则“body{font-size:16px;font:12px/1.5;}”可以线性化为“body{font-size:16px;font-size:12px;line-height:1.5;}”。该算法将隐性冲突显性化后，接下来对样式冲突进行检测与处理。样式冲突检测与处理如图3所示，主要包括以下3步骤：①如果PerfectNode列表中两PerfectNode的tagNode属性相同，则进行合并；②如果styleInfos中两选择器相同则进行合并；③如果同一样式规则中两属性名相同，则进行合并，合并原则是后面样式属性覆盖前面样式属性。经过上述步骤后，显性化后的样式规则最终结果为“body{font-size:12px;line-height:1.5;}”，该样式规则呈现的效果与原效果一致，且不含样式冲突。

2.3 输出HTML和PerfectNode关联类列表

样式冲突处理完成之后，得到的PerfectNode关联类列表中不含任何样式冲突，且所有的样式属性都是简单样式属性。PerfectNode列表中的样式属性和标签在结构上是一一对应的，在后续的NLECs网页分割中，当需要获取样式信息时，只要对PerfectNode列表检索即可。所以，该预处理算法最后的操作为“输出HTML和PerfectNode关联类列表”。需要注意，在输出HTML网页前，还需要将HTML进一步简洁化(即，去掉HTML原有的样式信息和〈link〉、〈style〉标签)。

3 实验设计与结果分析

为了测量该预处理算法的准确性，随机选取100个不同的NLECs网站，对每个网站，随机获取1个网页(可能是主页面，也可能是子页面)，对得到的100个NLECs网页进行预处理，并进行定性和定量实验。在定性实验中，从这100个NLECs网页中随机选取20个网页，从视觉效果角度将处理后呈现出的视觉效果与未进行预处理的视觉效果进行比较，从而定性地测量该预处理的准确性；在定量实验中，从“NLECs网页预处理后样式信息准确度”、“样式冲突的解决程度”、“分割形成块样式信息的准确度”3个角度对该预处理算法进行评价。

另外，有些样式属性具有继承性，而进行继承性处理需要一定的计算量。考虑到网页分割时不需要对所有标签进行继承性处理，本预处理算法并没有将继承的样式属性加入到StyleInfos中，但为了避免干扰因素的影响，在进行试验时，假设继承性问题已经得到解决。

3.1 定性实验与分析

图4所示的是3组具有代表性的预处理前后的视觉效果图对比，前两组是整个网页的部分视觉效果图(其中a(1)、b(1)是原图，a(2)、b(2)是处理后的效果图)，后一组是假设分割后，其中两个块的视觉效果图(c(1)、c(2))。从a(1)与a(2)的比较中发现，经过预处理算法之后，网页可以准确无误地呈现，并和预处理前的视觉效果一样；从b(1)与b(2)的比较中发现，有些样式信息丢失了，这可能是由于该NLECs网页的样式信息和HTML代码不规范造成的；观察c(1)、c(2)，可以发现c(1)块的视觉效果准确无误，而c(2)块的尺寸变大了(虚线是该块的大小)。之所以块会变大，是因为该处的块该标签为〈table〉标签(〈div〉标签也有此特殊特性)，当〈table〉未指定宽度，且作为块的根节点时，它的宽度是不确定的(样式属性width不具有继承特性)。经过上述分析，可以认为该预处理算法有较高的准确度，可以有效地对网页进行预处理。

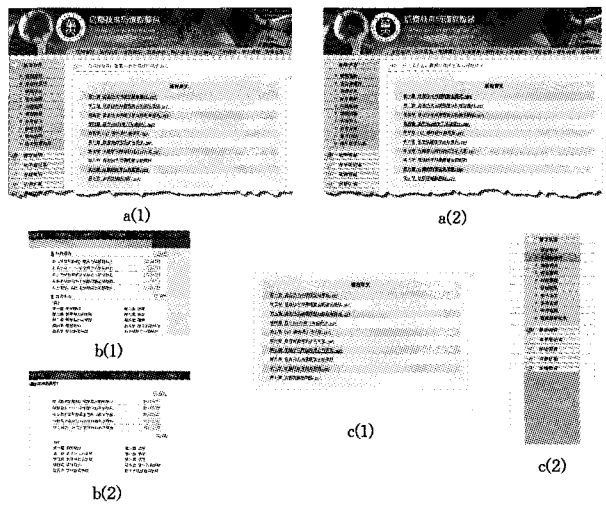

图4 预处理前后的视觉效果图

3.2 定量实验与分析

定性实验从视觉角度定性地对预处理算法进行了评价，然而，从图4b(2)中可以发现，该预处理算法并没有完美地预处理这100个NLECS。为此，需要一些准确的数据来衡量它的准确度，所以，还需进行定量实验来更客观地评价。

3.2.1 定量实验评价标准制定

NLECs网页预处理后样式信息准确度、样式冲突的解决程度、分割形成块样式信息的准确度直接影响后续NLECs网页分割的准确度，所以结合检索领域的评价标准体系，该实验从这3个角度来定量地测量预处理算法的准确性。

对于第1个角度，由于经过预处理算法后，样式信息较预处理前的样式信息发生了很大变化，因此需要特殊的方法(式(1))来进行衡量。其中，分子“预处理后的样式规则数－出错的样式规则数”是“(预处理后的样式规则数＋丢失的样式规则数)－(出错的样式规则数＋丢失的样式规则数)”的简写。

$$r1=\frac{\text{预处理后的样式规则数}-\text{出错的样失式规则数}}{\text{预处理后的样式规则数}+\text{丢失的样式规则数}} \quad (1)$$

对于第2个角度，采用式(2)来衡量。

$$r2=\frac{\text{原有冲突数}-\text{剩余冲突数}}{\text{原有冲突数}} \quad (2)$$

对于第3个角度，考虑到每个NLECs网页分割后会形成多个块，从每个NLECs网页中随机取5个块(假设是网页分割形成的块)作为样本。对这些样本，采用式(3)来衡量。

$$r3=\frac{5-\text{不准确的块数}}{5} \quad (3)$$

3.2.2 定量实验结果分析

通过对这100个NLECs进行预处理，得到了一系列的实验数据，进行统计之后，得到的结果如表3所列，该表从“最小值”、“最大值”、“平均值”3个方向记录了该实验的结果。

从表3的“平均值”一列可知$r1=95.11\%$，$r2=100\%$，$r3=90.82\%$。由此，可以初步判定：该处理算法可以有效地对网

(下转第388页)

结束语 基于 ARM 微处理器 S3C2410 的嵌入式实时控制系统相对于单片机具有运算速度快、易扩展等优势，而且通过数据库对太阳运动轨迹的记录及双轴跟踪系统可以提高太阳跟踪范围，实现对太阳的高精度跟踪。硬件方面通过利用步进电机与嵌入式核心板构成闭环控制，直到成对的硅光电池输出信号差值在允许的范围内，电机才停止运动，是一种完全可行的方案。软件方面通过在视日运动轨迹和追光跟踪两种方式下的高度角、方位角计算，利用 QT 软件实现系统跟踪控制。该系统的实现具有结构简单、安装方便、跟踪精度高、跟踪范围广、运行可靠、在任何天气状况下工作等特点。

参 考 文 献

[1] 徐成，谭曼琼，徐署华，等. 嵌入式 Linux 系统实训教程[M]. 北京：人民邮电出版社，2010：1-78

[2] 周立功. ARM 微控制器基础与实战[M]. 北京：北京航空航天出版社，2003：1-27

[3] 嵌入式太阳能光伏板自动跟踪器关键技术研究与产品开发_可行性报告[R]

[4] 周琪，万青. 基于 S3C2440 的智能型太阳跟踪系统[J]. 计算机系统应用，2011(10)：33-35

[5] 杨培环. 高精度太阳跟踪传感器与控制器的研究[D]. 武汉：武汉理工大学，2010

[6] 周培涛，李成贵，刘绍宗，等. 高精度双轴太阳跟踪控制器设计[J]. 电光与控制，2011(4)：81-84

[7] 刘学富. 基础天文学[M]. 北京：高等教育出版社，2004：1-64

[8] 吴士力，刘奇，朱兰. 嵌入式 Linux 应用开发全程解析与实战[M]. 北京：机械工业出版社，2009：1-124

(上接第 382 页)

页进行预处理，为后续网页分割服务。

表 3 实验数据统计表

数据%/统计项 / 评价角度	最小值	最大值	平均值
r1	87.96%	100%	95.11%
r2	100%	100%	100%
r3	20.00%	100%	90.82%

观察表中“最小值”一列发现 $r1=87.96\%$，$r3=20.00\%$，这两个值并没有达到期望的值。这是由于本预处理算法是基于 CSS 级联样式规则和 DOM 设计的，因此，该算法对样式信息和 HTML 代码的规范性要求比较高。然而，由于 Web 开发者的开发水平和开发风格不同，加上为了减少开发成本，很多 NLECs 是在现有网页基础上进行多次修改的产物，造成有些 NLECs 网页的样式信息和 HTML 代码存在很多不规范。此外，目前的浏览器具有强大的容错功能，即使是不规范的 NLECs 网页，浏览器也能正常显示。这可能就是 $r1$ 和 $r3$ 出现上述情况的原因。通过观察发现，$r3$ 最大值和最小值的跨度比较大，主要是因为每个网页只随机取了 5 个块造成的，这是根据网页分割的实际情况而定的。另外，由于〈div〉、〈table〉等标签未指定宽度而又作为块的根节点时，它的宽度不确定，这可能就是 $r3$ 略小于 $r1$ 的原因。

对于 $r2$ 的准确率，从式(3)中可知它只保证了未丢失的样式信息不包含样式冲突，这就是即使有些 NLECs 不规范，也能达到 100%的原因。这恰好充分证明，该算法可以彻底解决掉网页分割时因样式冲突而引起的不便。综上分析，可以得出结论：该处理算法可以有效地对网页进行预处理，准确地将 NLECs 网页和样式信息进行融合，这有利于后续对 NLECs 进行准确的分割。

结束语 针对国家精品课程网站中网页内容和样式独立设计，网页分割算法难以运行的问题，基于 CSS 级联样式规则提出了一个网页预处理算法。实验结果及分析表明，该处理算法可以有效地对网页进行预处理，准确地将 NLECs 网页和 CSS 样式信息进行融合。通过实验可以发现，该预处理算法对样式信息和 HTML 代码的规范性比较敏感。这将是后续研究工作需要改进的不足，之后将对 NLECs 网页进行分割，以便学习者可以通过移动设备正常地访问国家级精品课程。

参 考 文 献

[1] Sano H, Shiramatsu S, Ozono T, et al. A Web Page Segmentation Method based on Page Layouts and Title Blocks[J]. International Journal of Computer Science and Network Security, 2011, 11(10): 84-90

[2] Chibane I, Doan B L. A Web page topic segmentation algorithm based on visual criteria and content layout[C]// Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007: 817-818

[3] Simon K, Lausen G. ViPER: augmenting automatic information extraction with visual perceptions[C]// Proceedings of the 14th ACM international conference on Information and knowledge management. ACM, 2005: 381-388

[4] Cai D, Yu S, Wen J R, et al. VIPS: a visionbased page segmentation algorithm[R]. Microsoft technical report, MSR-TR-2003-79. 2003

[5] Gupta A, Kumar A, Tripathi V N, et al. Mobile web: web manipulation for small displays using multi-level hierarchy page segmentation[C]// Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology. ACM, 2007: 599-606

[6] Yang S J H, Zhang J, Chen R C S, et al. A unit of information-based content adaptation method for improving web content accessibility in the mobile Internet[J]. ETRI journal, 2007, 29(6): 794-807

[7] Chen Y, Xie X, Ma W Y, et al. Adapting web pages for small-screen devices[J]. Internet Computing, IEEE, 2005, 9(1): 50-56

[8] Artail A, Raydan M. Device-aware desktop web page transformation for rendering on handhelds[J]. Personal and Ubiquitous Computing, 2005, 9(6): 368-380