

# Segmenting Webpage with Gomory-Hu Tree Based Clustering

Xinyue Liu<sup>1,2</sup>

1. School of Computer Science and Technology, Dalian University of Technology, Dalian, China  
Email: xylu@dlut.edu.cn

Hongfei Lin<sup>1</sup> and Ye Tian<sup>2</sup>

2. School of Software, Dalian University of Technology, Dalian, China  
Email: hflin@dlut.edu.cn

**Abstract**—We propose a novel web page segmentation algorithm based on finding the Gomory-Hu tree in a planar graph. The algorithm firstly distills vision and structure information from a web page to construct a weighted undirected graph, whose vertices are the leaf nodes of the DOM tree and the edges represent the visible position relationship between vertices. Then it partitions the graph with the Gomory-Hu tree based clustering algorithm. Experimental results show that, compared with VIPS and Chakrabarti et al.'s graph theoretic algorithm, our algorithm improves upon the other two with much higher precision and recall, and its running time is far lower than that of Chakrabarti et al.'s graph theoretic algorithm.

**Index Terms**—Webpage segmentation, DOM tree, Gomory-Hu tree, Planar graph

## I. INTRODUCTION

Nowadays, people rely on search engine to find useful information from the web and web pages are treated as the basic information unit by major commercial search engines. However, they are not atomic units for information search on the Web, since web pages are usually embedded with various kinds of valuable semantic information. Understanding of the structure and the semantics of web pages could significantly improve people's searching experience [1-6]. Web page segmentation, whose aim is to partition a web page into continuous and cohesive regions with unified theme in content and purpose, is an essential task of web page understanding.

There are several applications of web page segmentation. (1) Ranking: segments discriminate between different types of information and can produce more accurate ranking results for different queries with block lever link analysis [1, 2]. (2) Duplicate detection: web pages with identical content information may be presented using different web page layouts, web page segmentation is important to deal with this kind of content duplication. (3) Content Extraction: segments demarcate informative and non-informative content on a

web page; removing template noise might also increase classifier performance. Besides web searching and mining tasks, web page segmentation also plays an important role in displaying web pages on small screen devices.

Traditional approaches of web page segmentation usually use heuristic rules [7] or machine learning algorithms either by visual analysis or interpreting the meaning of tag structures. While these approaches might work well on some kind of pages, they still have problems that limit them to be universal applicable. Very recently, Chakrabarti et al.[8] dealt with web page segmentation from a graph-theoretic perspective. They cast the problem as a minimization problem on a weighted directed graph, whose nodes are the DOM tree nodes and the arc-weights express the cost of placing the end points in same/different segments. Chakrabarti et al.'s [8] approach is shown to be superior to previous approaches, however, the underlying graph of this approach is too complex, making the task of solving the minimization problem very difficult. Thus this approach is not efficient and does not suit for real applications.

Our contributions:

(1) We formulate the problem of web page segmentation to the clustering problem on an undirected graph, in which the vertices are leaf nodes of the DOM tree, and the edges represent the neighborhood relationship between vertices when they are rendered on the screen. The graph constructed by both vision and structure information is a planar graph, on which most optimization problems are much easier than those on general graphs.

(2) We deploy the Gomory-Hu tree based algorithm to solve the underlying clustering problem in our formulation. The use of Gomory-Hu tree based algorithm not only because it has theoretical clustering quality guarantees, but also because it can benefit from the graph's planarity to gain very efficient algorithm. However, Gomory-Hu tree based algorithm tends to generate outliers. We use a simple pre-processing technique to remedy this problem.

(3) We evaluate our approach through experiments. The experimental results show that our algorithm

Manuscript received December 28, 2010; revised March 1, 2011; accepted March 28, 2011.

Corresponding author: Xinyue Liu

improves upon the other two typical algorithms with much higher precision and recall, and its running time is far lower than that of Chakrabarti et al.'s graph theoretic algorithm.

## II. RELATED WORK

There is plenty of past work on web page segmentation. Prior work falls into four categories. (1) Heuristics approaches: Cai et al. [7] Proposed the VIPS algorithm that uses rule based heuristics to segment the visual layout of a webpage. Bar-Yossef and Rajagopalan's [9] algorithm identifies template blocks by finding common shingles. Kao et al. [10] presented a webpage segmentation algorithm that relies on content based features. (2) Machine learning approaches: Baluja [5] recast web page segmentation into a machine learning framework using decision tree. Chakrabarti et al.[13] proposed a regularized isotonic regression based algorithm to determine the templates of DOM nodes. (3) Pattern recognition approaches: Xiang et al. [14] define some common patterns beforehand, and then search them in the layout tree.(4) Graph-theoretic approach: Chakrabarti et al. [8] dealt with web page segmentation from a graph-theoretic perspective.

Our approach is also a graph-theoretic approach, thus Chakrabarti et al.'s [8] algorithm is the most closely related one to us. Ours differs from theirs in two ways. Firstly, their algorithm work on a directed graph in which nodes are DOM tree nodes, our algorithm's underlying graph is undirected graph whose vertices are leaf nodes of the DOM tree. What's more, the graph used in our algorithm is planar graph, this property helps to find much more efficient algorithm. Secondly, although the essential tasks of both approaches are graph partition or graph clustering, clustering on directed graph seems much more difficult than clustering undirected graph, and different algorithms are deployed. The Gomory-Hu tree algorithm used in our approach, which is shown to have theoretical quality guarantee, is not applicable to directed graphs.

## III. UNDERLYING GRAPH CONSTRUCTION

Different from Chakrabarti et al.'s approach [8] which uses DOM tree structure information to construct the underlying graph, we use both the DOM tree structure information and the vision information to construct the graph. Initial steps of graph construction are shown as Fig. 1.

### A. Vertex Selection

A web page can contain many types of information: content information, vision information, construct information and other inner information, and we segment the web page is to segment the content of the web page, so when choosing vertices we only choose the nodes with real message, such as text, picture and video. In a HTML DOM tree, only leaf nodes contain semantic information, the upper level nodes contain only structural information. Since web page segmentation is to segment semantic information, we use only leaf nodes of the DOM tree as

graph vertices. Thus the deduced graph has much smaller number of vertices than Chakrabarti et al.'s graph, and is more informative.

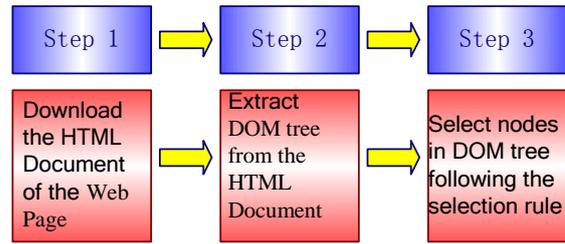


Figure 1. Initial steps of graph construction

We first download the HTML document from the web, second extract the HTML DOM tree, and then choose vertices from the HTML DOM tree depending on the rules. The main principle of vertex selection is choosing the content information, and the rules for vertex selection are showed in Table 1.

TABLE I.  
RULES FOR VERTEX SELECTION

Rules	Content
Rule 1	DOM tree nodes containing text content, link and picture information which appear in the rendered page are vertices of the graph
Rule 2	DOM tree nodes containing web page layout, text font format information are not vertices of the graph
Rule 3	DOM tree nodes containing text content, link and picture information which do not appear in the rendered page are not vertices of the graph
Rule 4	Visual nodes whose width and height on the browser screen is smaller than threshold $a$ are not graph vertices
Rule 5	Visual nodes that are non-informative blank regions on the browser screen are not graph vertices

### B. Edge Adding

The visual layout information of the web page is very important. We use this information to construct the undirected graph: if two vertices are neighbors on the browser screen, we add an edge between them. Since each such node is a rectangle on the screen, and only the four borders of the rectangle can produce edge, thus the obtained graph is a planar graph.

The edge adding steps are as follows.

(1) Obtain the locate information of DOM tree nodes which are graph vertices: offsetTop, offsetBottom, offsetLeft and offsetRight.

(2) Find neighbors for each vertex. This is done by finding all nodes that are on the top, bottom, left and right of the current, and then select the nearest among them as the neighbor.

(3) Add edges to vertices with the relationship of neighbor, the flowing is an example of adding edges.

Figure 2 shows an example web page: www.qq.com, in which A, B, C, D, E, F, G, H, I and J are vertices of the corresponding graph. Firstly we obtain the reality location of the vertices and then find the neighbors for all vertices. After adding edges for vertices the graph is as Figure 3.

C. Edge Weighting

We use structural information of the DOM tree to add weights to edges of the graphs. Since each vertex in the graph is a leaf node of the DOM tree, which corresponds to a path from the root to the node. We use path similarity to add the edge weight between two vertices.

Given two vertices in the graph  $V_i$  and  $V_j$ , which corresponds to two nodes of the DOM tree,  $L_i=D_{i1}, \dots, D_{im}E_i$  and  $L_j=D_{j1}, \dots, D_{jn}E_j$ , where  $D_{ix}$  and  $E_i$  are used to denote the  $x$  tag and entities corresponding to the nodes  $i$ . The weight, which is the similarity of the two paths, is defined as:

$$Sim(L_i, L_j) = 1 - EditDist((D_{i1}, \dots, D_{im}), (D_{j1}, \dots, D_{jn})) / Max(D_i, D_j) \quad (1)$$

Where EditDist is the edit distance[17],  $Max(D_i, D_j)$  is the max length of the path of  $D_i$  and  $D_j$ .  $L_i$  and  $L_j$  represent the length of the path.

The edit distance EditDist used to calculate the min time of insert, delete and replace from the source string (s) to the target string (t). It can weigh the similarity of two strings. For example, string  $s_1="12433", s_2="1233"$ ,  $EditDist(s_1, s_2)=1$ , because we can insert 4 into 1233 to get 12433. The integrity algorithm can be seen in [17].



Figure 2. An example web page

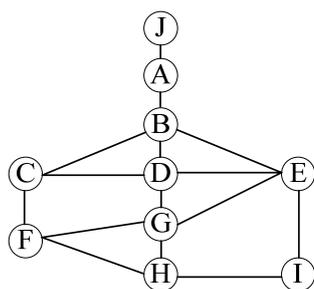


Figure 3. The graph for the web page in Fig. 2

IV. WEB PAGE SEGMENTATION

The task of web page segmentation is essentially to partition or clustering the graph constructed in the above section into groups such that the inner group similarity is maximized while inter group similarity is minimized. Each cluster of the clustering result is a segment of the web page. There are many established clustering algorithms for undirected graphs, we deploy Gomory-Hu tree partition algorithm for both quality and efficiency consideration.

A. Gomory-Hu tree graph partition

Gomory-Hu's[18] partition is a classic graph theoretic algorithm, which partitions the graph by computing minimum cuts and has theoretical quality guarantees[19].

Give an undirected weighted graph  $G = (V, E, W)$ , where  $V = (V_1, V_2, \dots, V_n)$  is the set of vertices,  $E = \{e_{ij} = (v_i, v_j)\}$  is the set of edges, and  $W = (w_{ij})$  is the set of weights.

**Definition 1** (Minimum cut tree or Gomory-Hu tree) [18]: For an undirected weighted graph G, a tree is a minimum cut tree if it follows,

- (1) Nodes of the tree are vertices of the graph;
- (2) Each edge in the tree has a non-negative weight  $w_{ij}$ ;
- (3) For each pair of nodes  $s$  and  $t$ , let  $e_{ij}$  be the edge on the path from  $s$  to  $t$  with the minimum weight, then  $e_{ij}$ 's is equal to the capacity of the minimum cut between in  $s$  and  $t$  in the graph.

The Gomory-Hu's algorithm constructs the minimum cut tree based on the maximum flow minimum cut theorem. It computes the maximum flow between each pair of vertices to find the minimum cut between them, and constructs the minimum cut tree using these minimum cuts. Gomory-Hu's algorithm requires  $O(n)$  maximum flow computations. In a general graph, using the fastest maximum flow algorithm, the computational complexity of Gomory-Hu's algorithm is  $O(n^2 m \log(n^2 / m))$  [20] where  $n$  is the number of vertices and  $m$  is the number of edges. However, this time bound can be significantly reduced to  $O(n^2 \log n)$  [21] when the graph is planar. Thus, the use of planar graph in our algorithm can obtain very high efficiency.

For the graph deduced from a web page, in which the vertices represent the visual leaf nodes of the DOM tree, the edges represent neighborhood relationship between them, and edge weight represent similarity between neighbors. Then during the construction of the minimum cut tree, each SupperNode represents continuous region of the web page. The procedure of minimum cut tree construction can be stopped following a best cluttering criteria, thus we can get  $k$  SupperNodes, i.e.,  $k$  sub-graphs that maps to  $k$  blocks in the web page, such that the intra block semantic similarity is maximized, and the inter block semantic similarity is minimized. Thus a good segmentation of the web page is got.

B. Dealing with outliers

Theoretically Gomory-Hu algorithm is an optimal algorithm [18]. However, since it partitions the graph following the minimum cut criteria, it tends to generates outliers, i.e., very small sub-graph or singleton vertices. This is not desirable since in practical applications we usually need the cut to be balanced with some extent. To deal with outliers, we here propose a preprocessing method.

**Definition 2** (Outlier): If the sum of edge weights associated to a vertex  $i$  is smaller than the edge weight sum of its neighbor's by some ratio, i.e,

$$\sum_{j \in V} W_{ij} \leq \beta \sum_{j,k \in V} W_{jk} \quad (2)$$

Where  $i$  is an outlier, and  $\beta$  is a predefined threshold.

To avoid outliers in the final clustering result, potential outliers are merged with their neighbors. The principle for merging is:

$$\max_{i \in S, j \in V} W_{ij} \geq \alpha \sum_i W_{ij} \quad (3)$$

Where  $i$  is a generated vertex merged by a sub set of the vertices;  $\alpha$  is a weighted ratio and is set to be 0.1 by experiences. If a vertex  $j$  follows the formula, then  $j$  is merged into the class  $i$ ; otherwise,  $j$  belongs to another class. Figure 4 is an example of the merging process.

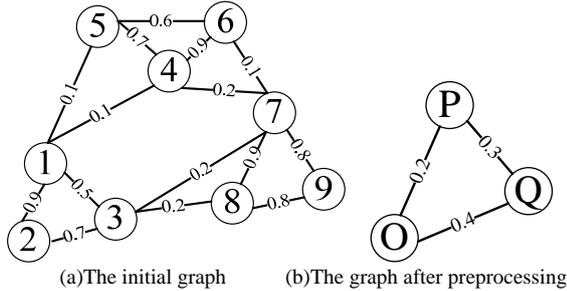


Figure 4. Graph preprocessing

### C. Gomory-HuPS Algorithm

The whole Gomory-HuPS algorithm is described as follows:

#### Gomory-HuPS Algorithm

Input:  $G$  is an undirected graph with  $n$  vertices;

Output:  $k$  parts of  $G$ ;

- 1 vertex  $i$  is one class of  $G$ , the number of vertices reduce 1;
- 2 while(vertex  $j$  in all vertices of  $G$ ){
- 3 if( $i, j$  satisfy formulae (3))  $j$  is in the class of  $i$ , the number of vertices reduce 1;
- 4 else {
- 5 if( $j$  satisfy formulae (2)){
- 6  $j$  is in the class of  $i$ , the number of vertices reduce 1;
- 7 if(the number of vertices is 0) using Gomory-Hu algorithm to get the Gomory-Hu tree of the new graph, taking out the edges ascending, then get  $k$  parts of  $G$ ;
- 8 else  $j$  is a new class, the number of vertices reduce 1;
- 9 }
- 10 else  $j$  is a new class, the number of vertices reduce 1;
- 11 }
- 12 }

## V. EXPERIMENTS

To evaluate the proposed algorithm, which is denoted as Gomory-HuPS, we did experiments on 100 pages with 2726 blocks. The data set is crawled automatically and the blocks are manually segmented. We compared our algorithm with VIPS and Chakrabarti et al.'s graph theoretic approach.

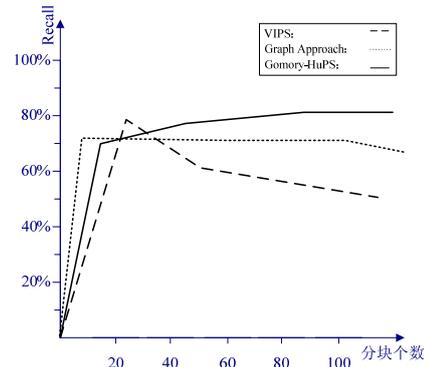


Figure 5. Recall comparison

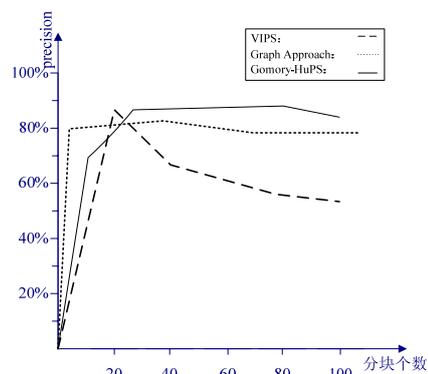


Figure 6. Precision comparison

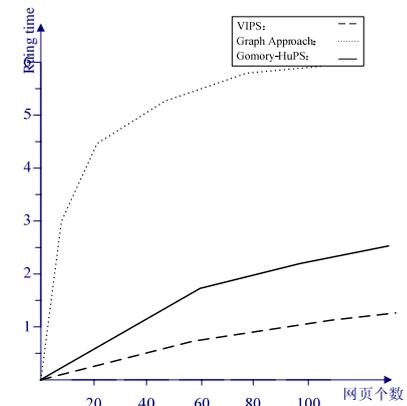


Figure 7. Running time comparison

The experimental results show that our algorithm improves upon the other two with much higher precision and recall, and its running time is far lower than that of Chakrabarti et al.'s graph theoretic algorithm. One reason is VIPS is base on heuristic rules, it might work well on some kind of pages, but it cannot apply to all types of web pages. Gomory-HuPS and Chakrabarti et al.'s graph approach are both base on the graph theory, can apply to any types of web pages. Our approach Gomory-HuPS modifies the graph construct method, uses the Gomory-Hu algorithm to segment the web page, and makes the running time much lower than Chakrabarti et al.'s graph approach.

## VI. CONCLUSION

In this paper we have addressed the web page segmentation problem by proposing an algorithm based on Gomory-Hu tree. The algorithm uses vision and structure information from a web page to construct a graph with the Gomory-Hu tree based clustering algorithm. Since the graph is a planar graph, the algorithm is very efficient. Experimental results show that, our algorithm outperforms both VIPS and Chakrabarti et al.'s algorithm in terms of precision and recall, and is faster than Chakrabarti et al.'s graph theoretic algorithm.

## ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under Grant No. 60873180 and the Fundamental Research Funds for the Central Universities.

## REFERENCES

- [1] D. Cai, S.P. Yu, J.R. Wen, and W.Y. Ma, "Block Level Web Search", In *Proc. The 27th Annual International ACM SIGIR Conference (SIGIR'2004)*, 2004.
- [2] D. Cai, X.F. He, J.R. Wen, and W.Y. Ma, "Block Level Link Analysis", In *Proc. The 27th Annual International ACM SIGIR Conference (SIGIR 2004)*, 2004.
- [3] Z.Q. Nie, J.R. Wen, and W.Y. Ma, "Webpage understanding: beyond page-level search", *ACM SIGMOD Record*. 2009, 37(4):48-54.
- [4] J. Zhu, Z.Q. Nie, J.R. Wen, B. Zhang, and H.W. Hon, "Webpage Understanding: an Integrated Approach", In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD07)*, 2007, 903-912.
- [5] S. Baluja, "Browsing on Small Screens: Recasting Web-Page Segmentation into an Efficient Machine Learning Framework", In *Proc. The 15th International Conference on World Wide Web (WWW2006)*, 2006.
- [6] X.Qi and B.D.Davison, "Web Page Classification: Features and Algorithms", *ACM Computing Survey*, 2009, 41(2): Article 12.
- [7] D. Cai, S.P. Yu, J.R. Wen, and W.Y. Ma, "VIPS: a Vision-based Page Segmentation Algorithm", *Technical Report MSR-TR-2003-79*, 2003.
- [8] D. Chakrabarti, R. Kumar and K. Punera, "A Graph-Theoretic Approach to Webpage Segmentation", *The 17th International Conference on World Wide Web (WWW 2008)*, 2008.
- [9] Z. Bar-Yossef, and S. Rajagopalan, "Template detection via data mining and its applications", In *Proc. 11th International Conference on World Wide Web (WWW2002)*, 2002, 580-591.
- [10] H.Y.Kao, J.M. Hoand and M.S. Chen, "WISDOM: Web intra-page informative structure mining based on document object model", *IEEE Transactions on Knowledge and Data Engineering*, 2005, 17(5):614-627.
- [11] Y. Liu, Q. Wang and Q.X. Wang, "A Heuristic Approach for Topical Information Extraction from News Pages", *The 7th International Conference on Web Information Systems Engineering (WISE2006)*, 2006.
- [12] D. Debnath, S. Mitra, N. Pal and C.L. Giles, "Automatic Identification of Informative Sections of Web Pages", *IEEE Transactions on Knowledge and Data Engineering*, 2005, 17(9): 1233-1246.
- [13] D. Chakrabarti, R. Kumar and K. Punera, "Page-level template detection via isotonic smoothing", In *Proc. The 16th International Conference on World Wide Web (WWW2007)*, 2007, 61-70.
- [14] P.F.Xiang, X. Yang and Y.S. Shi, "Web Page Segmentation based on Gestalt Theory", In *Proc. IEEE International Conference on Multimedia and Expo*, 2007, 2253-2256.
- [15] H. Srinivasan , R. Rupesh and M. Amit, "Web Page Layout Optimization Using Section Importance", *The 17th International Conference on World Wide Web (WWW 2008)*, 2008.
- [16] Sandip Debnath, Prasenjit Mitra, Nirmal Pal, and C. Lee Giles. Automatic Identification of Informative Sections of Web Pages. IEEE Computer Society.2005.
- [17] D. Gusfield, *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, 1997.
- [18] R.E. Gomory and T.C. Hu, "Muti-Terminal Network Flows", *Journal of the SIAM*, 1961, 9: 551-570.
- [19] A.V. Goldberg and R.E. Tarjan, "A new approach to the maximum-flow problem", *Journal of the ACM*, 1988, 35(4): 921-940.
- [20] G. Borradaile, and P. Klein, "An O(nlogn) Algorithm for Maximum st-Flow in a Directed Planar Graph", *Journal of the ACM*, 2009, 56(2): Article 9.
- [21] G.W. Flake, R.E. Tarjan and K. Tsioutsouliklis, "Graph Clustering and Minimum Cut Trees", *Internet Mathematics*, 2002, 1(4): 385-408.

**Xinyue Liu** received the M.S degree in Computer Science and technology from Northeast Normal University, China, in 2006. She is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Dalian University of Technology, Dalian, China. Her research interests include multimedia information retrieval, web mining and machine learning.

**Hongfei Lin** received the Ph.D degree from Northeastern University, China. He is a professor in the School of Computer Science and Technology, Dalian University of Technology, Dalian, China. His professional interests lie in the broad area of information retrieval, web mining and machine learning, affective computing.

**Ye Tian** received her M.S. degree in Software Engineering from Dalian University of Technology in 2008. Her major interests lie in web page segmentation.