Harnessing the Wisdom of the Crowds for Accurate Web Page Clipping*

Lei Zhang[†‡], Linpeng Tang[♭‡], Ping Luo[‡], Enhong Chen[†],
Limei Jiao[‡], Min Wang[‡] and Guiquan Liu[†]
[†]University of Science and Technology of China, [♭]Shanghai Jiao Tong University, [‡]HP Labs China
[†]{stone, cheneh, gqliu }@ustc.edu.cn,
[‡]{linpeng.tang, ping.luo, li-mei.jiao, min.wang6}@hp.com

ABSTRACT

Clipping Web pages, namely extracting the informative clips (areas) from Web pages, has many applications, such as Web printing and e-reading on small handheld devices. Although many existing methods attempt to address this task, most of them can either work only on certain types of Web pages (e.g., news- and blog-like web pages), or perform semi-automatically where extra user efforts are required in adjusting the outputs. The problem of clipping any types of Web pages accurately in a totally automatic way remains pretty much open. To this end in this study we harness the wisdom of the crowds to provide accurate recommendation of informative clips on any given Web pages. Specifically, we leverage the knowledge on how previous users clip similar Web pages, and this knowledge repository can be represented as a *transaction database* where each transaction contains the clips selected by a user on a certain Web page. Then, we formulate a new pattern mining problem, *mining top-1 qualified pattern*, on transaction database for this recommendation. Here, the recommendation considers not only the pattern *support* but also the pattern *occupancy* (proposed in this work). High support requires that patterns appear frequently in the database, while high occupancy requires that patterns occupy a large portion of the transactions they appear in. Thus, it leads to both precise and complete recommendation. Additionally, we explore the properties on occupancy to further prune the search space for high-efficient pattern mining. Finally, we show the effectiveness of the proposed algorithm on a human-labeled ground truth dataset consisting of 2000 web pages from 100 major Web sites, and demonstrate its efficiency on large synthetic datasets.

Categories and Subject Descriptors

H.2.8 [**Database Application**]: Data Mining

General Terms

Algorithms, Design, Experimentation

*The work was done when Lei Zhang and Linpeng Tang were visiting HP Labs China.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
KDD'12, August 12–16, 2012, Beijing, China.
Copyright 2012 ACM 978-1-4503-1462-6 /12/08 ...$15.00.

Keywords

Occupancy, Frequent and Dominant Pattern, Web Page Clipping, Wisdom of the Crowds

1. INTRODUCTION

Many people use the World Wide Web as the primary information source in their daily lives. However, Web pages encoded by the HTML language are designed for viewing on PC screen and may not be suitable for other purposes, such as Web printing and e-reading on small handheld devices. Thus, extracting informative *clips* (referring to content areas in Web pages) attracts many research works [12, 14, 3, 10, 9]. However, most of these previous works focus on extracting *article* Web pages, such as news stories, encyclopedia entries, and blog posts. In these article pages, the informative contents include the title, text-body, content-related images and image captions and so on. These content items usually appear in the similar layouts and formats in the article Web pages. For example, the article body usually consists of contiguous paragraph blocks, which occupy the main area of the Web page. The article title is usually placed above the text-body in a more standout font. Thus, these commonly-shared visual features can be used to extract these informative items with high accuracy [3].

Besides article pages, there are a large number of *non-article* Web pages (e.g., online shopping pages, job recruiting pages and recipe pages). For these *non-article* pages the informative clips are scattered irregularly, thus it is hard to summarize the visual features, which makes the automatic extraction of these informative clips impossible. For example, in a product Web page users are usually interested in the product name, brief description, product image, and price. However, these informative items are formatted in diverse manners in different Web sites. Thus, an expedient solution to clipping non-article Web pages is a semi-automatic process, where users can manually select the informative areas in an interactive interface. This is exactly what the tool of *Smart Print*[1] provides for Web page printing [9].

Let us see how *Smart Print* works using the example in Figure 1. When a user opens the recipe Web page[2] and clicks *Smart Print* button, it recommends a valuable area, which is highlighted in a white rectangle (Figure 1(a)). The user can also preview the recommended print content (Figure 1(b)) at this time. Then, the user can interactively add or remove any content blocks on this page. For example, as shown in Figure 1(c) the recipe title "sausage-strata" is added and some uninformative clips, i.e. the six black rectangles

[1]An extension of Web browser, download from http:www.hp.com/go/smartprint
[2]http://www.diabeticlifestyle.com/recipes/breakfast/sausage-strata

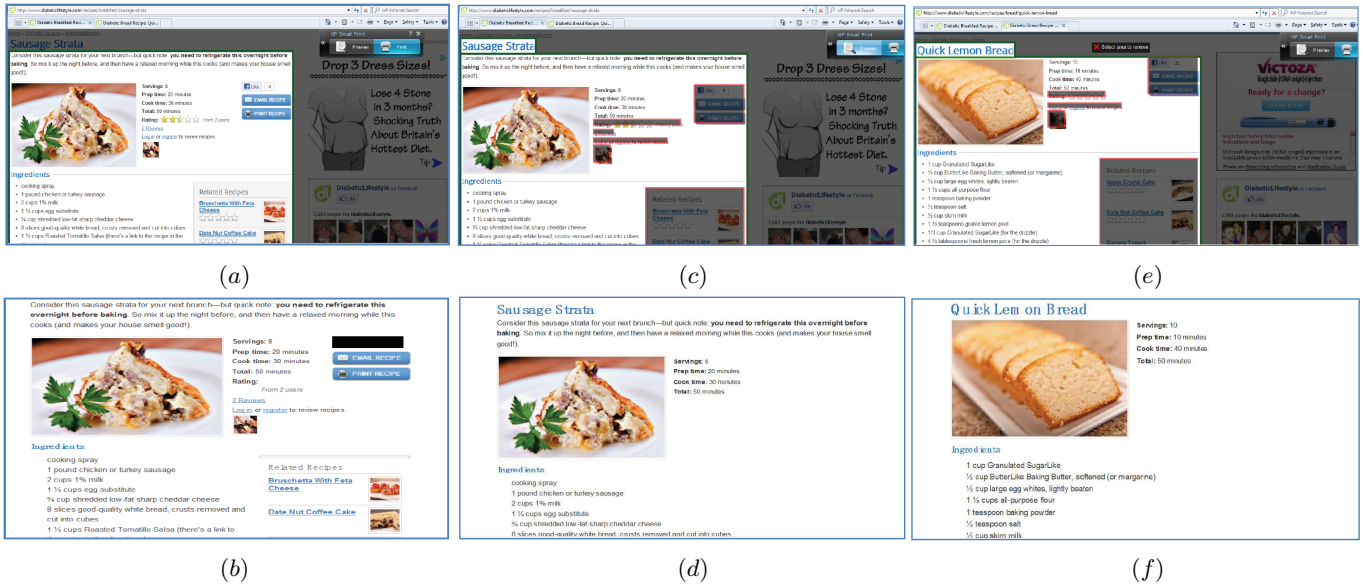


Figure 1: The Screenshoots of SmartPrint.

inside the big white rectangle, are removed. Finally, you can see that we get exactly what we want by the current print preview as shown in Figure 1(d)). Through this whole process, we see clearly that, many tedious user efforts are still required in *Smart Print*. And what is more, when we want to print similar Web pages, e.g. print another recipe from the same Web site, we have to do the tedious selection again. It means that in current *Smart Print* users' efforts are utilized only once and then discarded. But in fact these users' selections are very valuable to help us to understand users' interests and printing intentions.

Therefore, a more intelligent solution to clipping Web pages is in urgent need, where the user efforts in selecting the content areas are eliminated or greatly alleviated. Ideally, only one click is required in Clipping any Web page. To this end we leverage the wisdom of the crowds for more accurate recommendation. Here, the wisdom of the crowds comes from the log data on how previous users clip the Web pages. The print logs can be collected by *Smart Print* with users' permission. At a high level, we want to learn from the print log what other users were printing for a particular Web page and then recommend the learned selection to the current user. For example, assume that the selection on the "sausage-strata" Web page (Figure 1(c)) is shared to our knowledge repository. Then, if another user wants to print another recipe page about "quick-lemon-bread"[3], by leveraging the print logs our method can directly recommend the perfect informative clips (as shown in Figure 1(e)(f)) without manual selection. Interested readers can also watch a demo video[4] of this work to taste its effectiveness.

Table 1: The transaction database for Figure 1(c) and 1(e)

Transaction	Items
Figure 1(c)	{A B C D E F G H}
Figure 1(e)	{A B C D E F G}

In fact, these log data can be represented as a *transaction database* where each transaction contains the content clips selected by a user on a Web page. For example, Table 1 shows a transaction database for the content clips selected in Figure 1(c) and 1(e), where items A and B represent the two white rectangles (clips) and items C,D,E,F,G,H represent the six black rectangles. Then, with this transaction database we formulate a novel pattern mining problem, *mining top-1 qualified pattern*, for accurate informative clips recommendation on any given Web pages. It considers both the pattern *support* and the pattern *occupancy*. Previous studies widely consider support as the primary measure of pattern interestingness. In this study we introduce an another measure, *occupancy*, which requires that patterns should occupy a large portion of the transactions they appear in. Thus, intuitively the patterns with both high support and high occupancy may lead to both precise and complete recommendation. We summarize our contributions in this study as follows:

- We harness the wisdom of the crowds for more accurate Web page clipping. We formulate this task as a pattern mining problem on transaction databases. Besides the widely-studied pattern support, we propose a new concept, pattern *occupancy*, which measures the occupancy degree of a pattern inside the transactions it appears in. Thus, pattern support together with occupancy may lead to both precise and complete recommendation.
- We propose an efficient algorithm for this pattern mining problem with the constraints of both support and occupancy. Specifically, we explore the properties on occupancy, which can further prune the search space compared with the search process of frequent pattern mining.
- We demonstrate the effectiveness of the proposed algorithm on a human-labeled ground truth dataset consisting of 2000 Web pages from 100 major print-worthy Web sites. The average precision and recall of the proposed method reach more than 90%. Compared with the baseline method without the support of the wisdom of the crowds the precision increases 23.8% and the recall increases 5.1%. In addition, we also show the efficiency of the proposed algorithm on large synthetic datasets. Our algorithm runs faster than the baseline method by several orders of magnitude.

The rest of the paper is organized as follows. We describe the related work in Section 2. We formulate the task of clipping Web pages as a pattern mining problem in Section 3 and propose the efficient algorithm to solve it in Section 4. We address the issues on how to identify and represent informative clips in Web pages in Section 5 and summarize the whole solution in Section 6. We report an empirical study in Section 7 and finally conclude the paper in Section 8.

[3]http://www.diabeticlifestyle.com/recipes/bread/quick-lemon-bread

[4]http://goo.gl/VixPl

2. RELATED WORKS

Web page clipping has received substantial interests in literature [12, 14, 3, 10, 8, 16, 9]. Most existing methods only attempt to focus on extracting *article* Web pages. For example, Pasternack and Roth [12] proposed a method based on the maximum subsequence segmentation to generate scores for words using features of tri-grams and tags. Wang et al.[14] proposed a SVM-based template-independent wrapper with the content and spatial features to identify two minimum sub-trees containing title and article body. Luo et al. [10] utilized the visual and DOM tree based features to identify paragraphs as basic elements and then find the region for the main content by the maximum subsequence algorithm. Furthermore, some other heuristics are used to filter out the non-informative clips within the main content region. Fan et al. proposed a full-fledged system to extract title, text-body, content-related images and image captions from any article Web page [3] and achieved a high extraction accuracy.

However, such methods make use of the some prior assumptions on the layouts and formats of article Web pages and only work well for text-dominant Web pages such as news and blogs. They may not work well on non-article Web pages such as shopping, recipes or other types of Web pages where the informative clips are scattered irregularly in different formats. For non-article pages Lim et al. [9] proposed a semi-automatic tool *Smart Print* which can automatically select the main content of the web page in the first round and then allows the users to make adjustments. All of these previous works leverage the features of DOM tree, text or visual information on the given Web page. To our best knowledge, this work is the first study to show the log data on how previous users clip the Web pages can greatly improve the accuracy in clipping any Web page.

Pattern mining algorithms are adopted for the recommendation of informative clips. Frequent pattern mining [1] has been well recognized to be fundamental to many important data mining tasks. There is a great amount of work that studies efficient mining of frequent patterns [5, 6, 7, 2, 11]. These algorithms can be classified into mining frequent patterns [5, 6, 7], frequent maximal patterns [2], and frequent closed patterns [11]. To reduce the number of frequent patterns some interestingness measures and constraints are proposed [13, 4]. The concept of occupancy proposed in this paper can be viewed as a new interestingness measure. However, it is not *anti-monotonic*, *monotonic*, *convertible*, and *succinct* [4], thus, no previous methods can be leveraged. In this study we explored the properties on occupancy and show how it can be used to greatly prune the search space. The experimental result shows its superiority in efficiency over the other methods.

3. WEB PAGE CLIPPING BASED ON PATTERN MINING

In this section we will detail how to transform our task of Web page clipping based on the print logs into a pattern mining problem. First, we will briefly describe this task as follows.

3.1 Brief Description

The task of Web page clipping is actually to recommend a set of informative clips for the given Web page. In this application we assume that a log database is given for the Web pages from a Web site, where each piece of log records all informative clips a previous user selected on one of these Web pages. If we consider each clip as an *item*, then any piece of log becomes a *transaction* (set) of items and the log database becomes a *transaction database* [1]. The details on how to identify a clip in a Web page will be addressed later in Section 5.

Although the selections from different users may be different due to their own use habits and interests, generally their selections on similar Web pages may reach certain degree of consensus since the user intensions in clipping certain Web pages may be similar. Then, our task is to mine users' interests and intensions in clipping these Web pages and then provide more accurate recommendation.

The key question to this task is how to measure the *quality* of a clipping pattern (a set of items). On one hand, the more frequently a pattern appears in the database (it means a large number of users select this set of informative clips), the better it is. Thus, the *support* of a pattern is a key factor to its quality. On the other hand, we prefer the pattern which occupies a large portion of the transactions it appears in. Note that besides the items in a given pattern a transaction may include some other ones. If we apply the pattern onto the Web page corresponding to this transaction, all these other items will be missed. Intuitively, the patterns with high occupancy may lead to more complete recommendations. Thus, it is another factor of its quality. Altogether, patterns with bigger values of support and occupancy are preferred. With this motivation we give some definitions and notations as follows, and then formulate the problem.

3.2 Definitions and Notations

A *transaction database*[5] is a set of transactions, where each *transaction* is a set of items. Let $\mathcal{I}$ be the complete set of distinct items and $\mathcal{T}$ be the complete set of transactions. Any non-empty set of items is called an *itemset* and any set of transactions is called a *transaction set*. The transactions that contain all the items in an itemset X are the *supporting transactions* of X, denoted as $\mathcal{T}_X$. The *frequency* of an itemset X (denoted as $freq(X)$) is the number of transactions in $\mathcal{T}_X$.

The following two definitions are adopted from [1].

Definition 1 (Support): The *support* of X is defined as

$$\sigma(X) = freq(X)/|\mathcal{T}| \qquad \blacksquare$$

Definition 2 (Frequent Itemset): For a given minimum support threshold α ($0 < \alpha \leq 1$), X is said to be *frequent* if $\sigma(X) \geq \alpha$. ■

In this application, the itemset we intend to find should also occupy a large portion of the transactions in which it appears. We can calculate the occupancy degree as follows. For an itemset X we identify all its supporting transactions $\mathcal{T}_X$. For each transaction $t \in \mathcal{T}_X$ we calculate the ratio of $\frac{|X|}{|t|}$, where $|X|$ and $|t|$ represents the number of items in X and t, respectively. We then aggregate these ratios to compute a single value of occupancy for X. In this paper we focus on the *harmonic* average of these ratios while other aggregate functions such as *quantile* or *min* may also be considered. The definition of occupancy is given as follows.

Definition 3 (Occupancy): Formally, the *occupancy* of an itemset X is defined as

$$\phi(X) = \frac{freq(X)|X|}{\sum_{t \in \mathcal{T}_X} |t|}$$

where t is any support transaction of X. ■

In other words, the occupancy of an itemset X is the ratio of the occurrences of the items in X to the total number of the items in $\mathcal{T}_X$ (the supporting transactions of X). The high value of the

[5]This term is commonly used by the frequent pattern mining community, such as [1].

occupancy indicates that besides the items in X there are only a small number of items left inside the supporting transactions of X. Thus, the itemset with high occupancy is preferred.

Take the transaction database in Figure 2 as an example. Consider the itemset $\{BC\}$. Its supporting transactions are $\{1, 2, 4, 6\}$, the support $\sigma(\{BC\}) = \frac{4}{6}$, and the occupancy $\phi(\{BC\}) = \frac{4\times 2}{3+2+2+3} = \frac{4}{5}$.

In some sense, occupancy describes the *relative size* of an itemset to its supporting transactions. One may compare it to the *absolute size* of an itemset, i.e., the number of items it contains. For example, in [15], Wang et al. formulated the problem of finding the top-k frequent itemsets that have absolute size larger than min_l, so the absolute size of a itemset was used as measure on its quality. However, in many cases where transactions in the database varies in sizes, occupancy provides a more accurate and normalized score on the quality of the itemset. Later we will also empirically prove the superiority of occupancy over absolute size of an itemset on the application of web-print recommendation.

One may think that the itemset with a larger absolute size leads to a bigger value of occupancy. However, it is not always true. Here, we generate another transaction database based on the one in Figure 2. There are also 6 transactions in it, where the first 5 ones are the same with the ones in Figure 2 and the last one changes to $\{BCDEFGHI\}$. In this new transaction database $\phi(\{BC\})$ changes to $\frac{4\times 2}{3+2+2+8} = \frac{8}{15}$ while $\phi(\{BCD\}) = \frac{3}{8}$. Clearly, $\phi(\{BCD\}) < \phi(\{BC\})$. The reason is that BCD only appears in large transactions where it only occupies a small fraction, while BC appears in many smaller transactions where it occupies a large fraction. Thus, occupancy does not always increase monotonically when we add more items to an itemset. Similarly, we can show that occupancy does not always decrease monotonically when we add more items to an itemset either. This non-monotonic property of occupancy is in contrast to that of support in frequent pattern mining.

Definition 4 (Dominant Itemset): For a given minimum occupancy threshold β ($0 < \beta \leq 1$), X is said to be *dominant* if $\phi(X) \geq \beta$. ∎

With the definition of support and occupancy we can measure the *quality* of an itemset by combining these two factors.

Definition 5 (Quality): The *quality* of an itemset X is defined as $q(X) = \sigma(X) + \lambda\phi(X)$, where occupancy weight $0 \leq \lambda < +\infty$ is a user defined parameter to capture the relative importance of support and occupancy. ∎

Definition 6 (Qualified Itemset): For a given minimum support threshold α and a minimum occupancy threshold β ($0 < \alpha, \beta \leq 1$), X is said to be *qualified* if $\sigma(X) \geq \alpha$ and $\phi(X) \geq \beta$. ∎

Assume that we have the log database which records how previous users clipped the Web pages from a Web site. Given a Web page from the same Web site, we aim to recommend the informative clips for this Web page. More specifically, let $\mathcal{I}$ be the complete set of distinct clips in the database. For a given Web page we can get a set $Q \subseteq \mathcal{I}$ of clips which are included in this Web page. It is meaningless to recommend the informative clips which are not in the given Web page (how to determine Q will be addressed in Section 5). Thus, our task is to select a subset of Q for the clip recommendation.

In this application it is required that interesting patterns should be both frequent and dominant. On one hand, if a pattern F is frequent it means that there are enough cases such a pattern appears in the transaction database. Thus, it may improve the recommendation precision. On the other hand, if X is dominant it indicates that the recommendation of X is complete enough. Therefore, with the definition of support and occupancy, we formulate the problem of mining top-1 qualified patterns as follows.

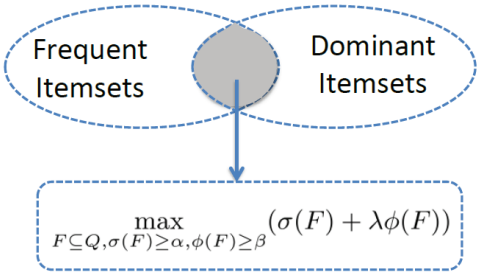


Figure 3: The problem of mining top-1 qualified pattern

3.3 Problem Formulation

Mining Top-1 Qualified Pattern. With the definition of support, occupancy and quality, we describe our recommendation method as follows. The recommended itemset is $F \subseteq Q$ which has the maximal quality value among the qualified itemsets (for a given support threshold α and a occupancy threshold β). Formally, it is

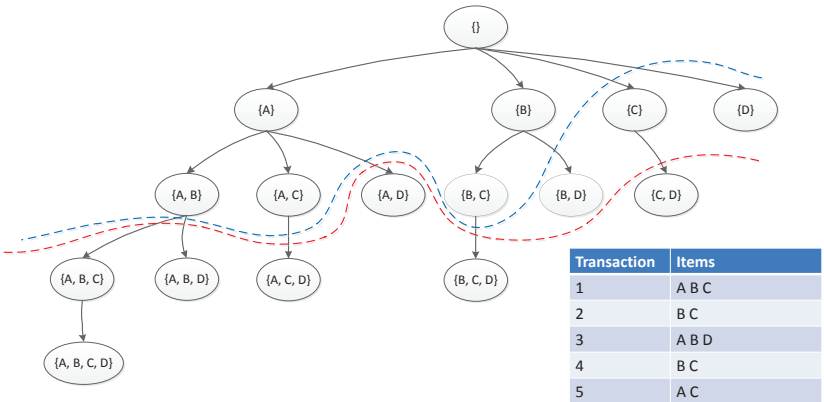
$$\arg\max_{F:F\subseteq Q,\sigma(F)\geq\alpha,\phi(F)\geq\beta}(\sigma(F)+\lambda\phi(F)) \quad (1)$$

This problem formulation is illustrated in Figure 3. Actually, we aim to find the top-1 qualified itemset among the intersection area of frequent and dominant itemsets.

There are three parameters in the definition of the top-1 qualified pattern, namely α, β, λ. If there is no itemset that is both frequent and dominant with respect to α, β, the top-1 qualified pattern does not exist and it will not output any result since no non-empty pattern exists that meets the quality requirements. Parameter λ, the occupancy weight, is a user defined parameter to capture the relative importance of support and occupancy.

One may think that dominant patterns with high occupancy usually contain a large number of items and thus methods based on *Maximal Frequent Itemset* (an itemset X is a maximal frequent itemset if X is frequent and no superset of X is frequent [2]) mining may also work in this application. Specifically, given a support threshold we can get multiple maximal frequent itemsets. Among them we can select the one with the largest number of items for the recommendation. Is this a valid method for our task? The answer is "no" for the following reasons. First, in methods based on mining maximal frequent itemsets, the number of items in a pattern is used as a measure in pattern selection. Compared with the concept of occupancy, this is actually the *absolute* size of an itemset while occupancy is the *relative* size of an itemset to the number of items in its supporting transactions. Among all the frequent itemsets, the maximal frequent itemset selected usually has very low support, thus leading to a low precision in recommendation. Secondly, in mining top qualified pattern a weighted sum of both support and occupancy is used as the interestingness measure, which may lead to better recommendation performance compared to the patterns selected by methods based on maximal frequent itemsets. The experimental results in Section 7 further validate our analysis here.

As mentioned before, Wang et al. [15] formulated a similar pattern mining task, i.e. the top-k frequent itemset with sizes larger than a minimum threshold. Here we can immediately see the differences of our problem to that problem: first, we use occupancy, the relative size, rather the absolute size of an itemset as an interesting-



Transaction	Items
1	A B C
2	B C
3	A B D
4	B C
5	A C
6	B C D

Figure 2: The example of the transaction database and lexicographic subset tree ($\alpha = 0.2, \beta = 0, \lambda = 2$)

ness measure; second, we use the the weighted sum of support and occupancy, rather than just the support as the quality of an itemset.

Next we will propose an efficient algorithm to this new pattern mining problem.

4. EFFICIENT PATTERN MINING ALGORITHM

The straightforward solution to the pattern mining problem is to first generate all the frequent itemsets, calculate the occupancy value for each frequent itemset, and then select the qualified itemset with maximum quality value. In this section we will show how the properties on occupancy and quality measures can be injected deeply into the search process and greatly prune the search space.

Pattern mining algorithms usually adopt the *lexicographic subset tree* [2, 11] to guide the search process. See Figure 2 as an example for four items A, B, C, D. The top element in the lattice is the empty set and each lower level l contains all the l-itemsets (itemsets with exactly l items). The l-itemsets are ordered lexicographically on each level. Generating children in this manner enumerates all the distinct itemsets to be considered without redundancy. As there are 4 items, there are in total $2^4 = 16$ itemsets for consideration. Thus, in the lexicographic subset tree there are 16 nodes, each of which corresponds to an itemset. Frequent pattern mining usually leverages the *monotonic decreasing* property of support values when adding more items to a given itemset. That is, if an itemset X is not frequent then all the supersets of X are not frequent either. Thus, the traversal in the tree is to find a cut (the red line in Figure 2) such that all the nodes (itemsets) above the cut are frequent, and all the nodes below the line are infrequent.

Note in our task of clip recommendation we only consider the items in Q, namely the clips identified in the given Web page (here, $\mathcal{I}$ may contain the distinct clips in the logs from the same Web site). Thus, the lexicographic subset tree only includes the items in Q. The search space becomes much smaller compared with the one on the complete set of distinct items.

4.1 Algorithm Overview

With the proposed occupancy and quality measure we can further prune the search space. Specifically, we can give the upper bounds of the occupancy and quality values for all the nodes in a subtree. In other words, the occupancy or quality of any node in a given subtree will be no bigger than its upper bound. If the upper bound on the occupancy is smaller than the occupancy threshold β, the corresponding subtree will be pruned. Also, we can maintain the current biggest quality value in the search process, denoted by $q*$. Then, the subtrees with the upper bounds less than $q*$ will be pruned.

Take the subtree with the root $\{C\}$ in Figure 2 as an example. We can give an upper bound of the quality for all the nodes in the subtree, including $\{C\}, \{CD\}$. Currently, this upper bound is less than the quality of node $\{BC\}$. Thus, this subtree is trimmed. The blue line (which is above the red one) in Figure 2 is the cut line for our problem. Clearly, it further reduces the search space.

Next, we will propose the properties to show how the upper bounds on occupancy and quality are computed.

4.2 The Upper Bounds on Occupancy and Quality

First, we give the notations which will be widely used in the following properties. For any subtree, let X be the itemset for the subtree root, Y be the itemset including all the new items which will be extended in all the descendants of this subtree. For example, for the subtree root $X = \{A\}$ there are three new items B, C, D which will appears in the descendants. Thus, $Y = \{BCD\}$. Then, we have the following properties.

Lemma 1: For any itemset W in the subtree of X let u be the frequency of W. Then, the occupancy of W satisfies that

$$\phi(W) \leq \frac{u|X| + \max_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap Y|}{\min_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap (\mathcal{I} - Y)| + \max_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap Y|}, \tag{2}$$

where t_{l_i} is any supporting transaction for X and $\mathcal{I}$ is the complete set of distinct items. ■

Proof: Remember that Y is the itemset including all the new items which will be extended in the descendants, X is the root of the subtree. Now we consider the occupancy of W.

$$\phi(W) = \frac{u|W|}{\sum_{t \in \mathcal{T}_W} |t|} \tag{3}$$

$$= \frac{u|X| + \sum_{t \in \mathcal{T}_W} |t \cap (W - X)|}{\sum_{t \in \mathcal{T}_W} |t|} \tag{4}$$

$$\leq \frac{u|X| + \sum_{t \in \mathcal{T}_W} |t \cap Y|}{\sum_{t \in \mathcal{T}_W} |t \cap (\mathcal{I} - Y)| + \sum_{t \in \mathcal{T}_W} |t \cap Y|} \tag{5}$$

$$\leq \frac{u|X| + \max_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap Y|}{\sum_{t \in \mathcal{T}_W} |t_{l_i} \cap (\mathcal{I} - Y)| + \max_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap Y|} \tag{6}$$

$$\leq \frac{u|X| + \max_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap Y|}{\min_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap (\mathcal{I} - Y)| + \max_{l_1,\cdots,l_u} \sum_{i=1}^{u} |t_{l_i} \cap Y|} \tag{7}$$

where $t_{l_1}, \cdots, t_{l_u}$ be any supporting transactions of X.

Inequality (5) comes from $Z - X \subseteq Y$. Inequality (6) comes from the inequality that if $b > a > 0, \gamma \geq 0$, then $\frac{a}{b} \leq \frac{a+\gamma}{b+\gamma}$. ■

Note that the right hand side of Equation (7) is only dependant on u, X, Y. Thus, it can be denoted as $F(u, X, Y)$. Then, we also have

Lemma 2: $F(u+1, X, Y) \leq F(u, X, Y)$ ∎

Proof: Let α_k be the k-th largest value in the vector $\langle |t \cap Y| : t \in \mathcal{T}_W \rangle$, and β_k be the k-th smallest value in the vector $\langle |t \cap (\mathcal{I} - Y)| : t \in \mathcal{T}_W \rangle$. Then, by this definition we have $\alpha_{k+1} \leq \alpha_k$, $\beta_{k+1} \geq \beta_k$. Clearly,

$$F(u+1, X, Y) = \frac{\sum_{i=1}^{u+1}(|X| + \alpha_i)}{\sum_{i=1}^{u+1}(\beta_i + \alpha_i)} \tag{8}$$

For any k ($1 \leq k \leq u$) we have

$$\frac{|X| + \alpha_k}{\beta_k + \alpha_k} \geq \frac{|X| + \alpha_k}{\beta_{u+1} + \alpha_k} \geq \frac{|X| + \alpha_{u+1}}{\beta_{u+1} + \alpha_{u+1}} \tag{9}$$

Then,

$$F(u, X, Y) = \frac{\sum_{i=1}^{u}(|X| + \alpha_i)}{\sum_{i=1}^{u}(\beta_i + \alpha_i)} \geq \frac{|X| + \alpha_{u+1}}{\beta_{u+1} + \alpha_{u+1}} \tag{10}$$

So

$$F(u, X, Y) \geq \frac{\sum_{i=1}^{u}(|X| + \alpha_i) + |X| + \alpha_{u+1}}{\sum_{i=1}^{u}(\beta_i + \alpha_i) + \beta_{u+1} + \alpha_{u+1}} = F(u+1, X, Y) \tag{11}$$

Inequality (10) comes from the property that if $1 \geq \frac{a_i}{b_i} \geq \frac{c}{d}$ for any i, then $\frac{\sum a_i}{\sum b_i} \geq \frac{c}{d}$.
Inequality (11) comes from the property that if $1 \geq \frac{a}{b} \geq \frac{c}{d}$, then $\frac{a}{b} \geq \frac{a+c}{b+d}$. ∎

It is required that itemset W be frequent. Thus, the frequency of W is not smaller than $\alpha \cdot |\mathcal{T}|$. As shown in Lemma 2, $F(u, X, Y)$ will not decrease along the decrease of u. Then, we have

Theorem 1:

$$\phi(W) \leq F(u, X, Y) \leq F(u', X, Y)$$

where u' is the minimal integer which is not smaller than $\alpha \cdot |\mathcal{T}|$. ∎

In other words, u' is the minimal frequency value which satisfies the support threshold α. Theorem 1 actually gives an upper bound on the occupancy values of the itemsets in the subtree.

Theorem 2:

$$q(W) \leq \max_{\alpha \cdot |\mathcal{T}| \leq k \leq |\mathcal{T}_X|} \left(\frac{k}{|\mathcal{T}|} + \lambda F(k, X, Y)\right),$$

where k is an integer between $\alpha \cdot |\mathcal{T}|$ and $|\mathcal{T}_X|$. ∎

Proof: Let u be the frequency of W. Then,

$$q(W) = \frac{u}{|\mathcal{T}|} + \lambda\phi(W) \leq \frac{u}{|\mathcal{T}|} + \lambda F(u, X, Y) \tag{12}$$

It is required that itemset W be frequent. Thus, $\alpha \cdot |\mathcal{T}| \leq u \leq |\mathcal{T}_X|$. Then,

$$\frac{u}{|\mathcal{T}|} + \lambda F(u, X, Y) \leq \max_{\alpha \cdot |\mathcal{T}| \leq k \leq |\mathcal{T}_X|} \left(\frac{k}{|\mathcal{T}|} + \lambda F(k, X, Y)\right) \tag{13}$$

It follows the conclusion. ∎

Theorem 2 actually gives an upper bound on the quality values of the itemsets in the subtree.

5. IDENTIFICATION OF WEB PAGE CLIPS

So far we assume that the transaction database of print logs is provided in advance. In this section we will detail how to identify the clips in a Web page. We first give some preliminaries on Web page analysis before we address these details.

5.1 Preliminaries on Web Page Analysis

Here, we provide some preliminaries for DOM (Document Object Model) tree, DOM tree path and Web page rendering, which are the key concepts in Web page analysis.

DOM Tree. A DOM tree for any HTML or XML document is defined as a tuple $T = \{V, E, L_V, F_V\}$, where V is the set of vertices, $E \subseteq V \times V$ is the set of edges, L_V is the set of vertex labels, and $F_V : V \rightarrow L_V$ maps the vertices to their labels. Figure 4(a) shows an example of DOM tree.

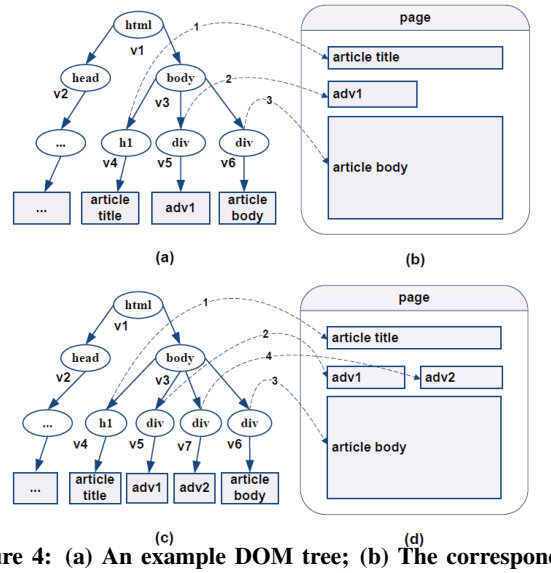


Figure 4: (a) An example DOM tree; (b) The corresponding Web page for Figure 1(a); (c) Another example DOM tree with one more adv; (d) The corresponding Web page for Figure 1(c).

DOM Tree Path. A DOM Tree Path (DTPath for short) for a node v in a DOM tree T is a sequence of label-position pairs, i.e. $P_v = ((l_1, k_1), \cdots, (l_n, k_n))$, where l_i is the combination of HTML tag and attributes of $node_i$ and k_i is the relative position among its sibling nodes. It records the path from the root of T to node v. For example, the DTPath of the node v_5 and v_6 in Figure 4(a) is *((html,1),(body,1),(div,1))* and *((html,1),(body,1),(div,2))* respectively. Note that in this example the attributes on each node are omitted from the DTPath just for description convenience. Since the node attributes contain much format and style information especially when CSS (Cascading Style Sheets) is more and more popular in Web page design, practically they help to identify the informative clips more precisely.

Matching. The DTPath P is *matched* in a DOM tree T, denoted by $P \sqsubset T$, if and only if there exists a path from the root node to an internal node in T such that the sequence along this path is exactly P. Consider Figure 4(a) again. We can say the DTPath $((html, 1), (body, 1), (h1, 1))$ can be matched in this tree since along this path we reach node v_4. However, the DTPath $((html, 1), (body, 1), (div, 3))$ can not be matched in this tree.

Web Page Rendering. The whole HTML document appears in a Web browser after *rendering*. The rendering process actually assigns each DOM tree node a *bounding rectangle*, identifying where

the content inside the node is displayed in the browser. Thus, a node v also corresponds to a unique bounding rectangle. Figure 4(b) shows the Web page corresponding to the DOM tree in Figure 4(a). As you can see, v_4, v_5, v_6 corresponds to the three rectangles of *article title*, *adv1*, and *article body* respectively.

The tool of Smart Print provides the interactive interface in a Web browser, where users can select the informative clips in a Web page. Actually, each clip is identified by a bounding rectangle, which then corresponds to a node in the DOM tree of the Web page. Thus, any clip can be identified by the DTPath of the selected node in the DOM tree. Then, a piece of clipping log actually contains a set of DTPaths.

5.2 Match a Set of DOM Tree Paths

To recommend informative clips we need to get all the clips which are included in a given Web page. Recommending the informative clips which are not in the given Web page is meaningless. As each clip is represented by its DTPath, this task is to match a set of DTPaths to a given DOM tree. More specifically, let $\mathcal{I}$ be the complete set of distinct DTPaths in the database. We aim to find all the DTPaths in $\mathcal{I}$, which can be matched in the given DOM tree T. Formally, this set Q is $\{P|P \in \mathcal{I} \cap P \sqsubset T\}$. These clips are the recommendation candidates in the given Web pages.

A straightforward method to this task is to consider each DTPath separately. Then, its complexity to n DTPaths is $o(\sum_{i=1}^{n} N_i)$ where N_i is the number of nodes in the i-th DTPath.

5.3 Generalized DOM Tree Paths

So far the DTPath considers the HTML tag and attributes of a node, and also its relative position among its siblings. However, the match of such DTPaths is not robust to the change of DOM tree structures. See the examples in Figure 4. The Web page in Figure 4(b) contains one advertisement, and the DTPath for *article body* is

$$P : ((html, 1), (body, 1), (div\{id = \text{“}article\_body\text{”}\}, \mathbf{2}))$$

The Web page in Figure 4(d) has the same Web page template with the one in Figure 4(b). However, one more advertisement is inserted into the page. Thus, the DTPath for *article body* in Figure 4(d) changes to

$$P' : ((html, 1), (body, 1), (div\{id = \text{“}article\_body\text{”}\}, \mathbf{3}))$$

P and P' are two distinct items in the transaction database even though they refer to the same type of content clips. It indicates that the transaction database generated by such DTPathes will be very sparse. Additionally, the clip for *article body* in Figure 4(d) will be missed if we apply P to match it. Thus, such DTPathes are vulnerable to the change of DOM tree structures.

After some empirical studies we find that this situation happens quite often, and is worth careful consideration. We address this issue by proposing *generalized* DTPaths by which we do not consider the relative positions for some nodes during the matching process. Specifically, we require that two distinct DTPathes should be combined as a generalized one when they are different only at the position indexes for some nodes in the pathes. For example, except the position indexes of the third nodes the other parts of the two paths P and P' are the same. Thus, they should be combined as a generalized DTPath

$$P^* : ((html, 1), (body, 1), (div\{id = \text{“}article\_body\text{”}\}, *))$$

where $*$ can be any integer. It indicates that we will not consider the position index of the third node in this path when matching. Using the generalized DTPath P^* the clips of *article body* in Figures 4(b) and (d) can be identified successfully. Readers may also worry that some more clips will be matched by the generalized DTPaths since a generalized DTPath can match multiple clips. We argue that this situation is acceptable since the identified clips by a generalized DTPath appear in the same format and style (remember that the attributes of the HTML nodes are used here).

After this combination the number of distinct generalized DTPaths is much smaller than that of the original DTPaths. Thus, the item size of the corresponding transaction database becomes much smaller, leading to more efficient pattern mining. Additionally, the database becomes more dense, which definitely helps to find the patterns in clipping the Web pages by users.

6. SOLUTION SUMMARY

We summarize the solution framework in Figure 5.

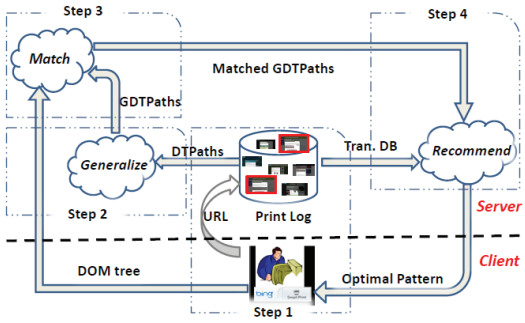


Figure 5: The Solution Framework.

Step 1. After a clipping request is fired, the client first sends the url and DOM tree of the current Web page to the server. Then, the server looks up in the log database to collect the print logs from the same Web site of the give URL. Based on these logs, it construct the corresponding transaction database $\mathcal{T}$ and all the distinct DTPaths $\mathcal{P}$ in these logs are also identified.

Step 2. Given the complete set $\mathcal{P}$ of distinct DTPaths we combine them to a set $\mathcal{P}^*$ of generalized DTPaths. Accordingly, we generate the transaction database based on the generalized DTPaths. See the details in Section 5.3.

Step 3. Identify the generalized DTPaths which is matched in the current DOM tree T. The set of the matched generalized DTPaths is denoted by Q. See the details in Section 5.2.

Step 4. Generate the pattern F in Equation 1 using the efficient algorithm in Section 4. Finally, the clips identified by the generalized DTPaths in F are recommended to the given Web page.

7. EXPERIMENTAL STUDY

In this section we present an empirical evaluation of our proposed method. Specifically, we show the effectiveness of the proposed pattern mining algorithm on a human-labeled ground-truth dataset including 2000 Web pages, discuss the impact from the method parameters and show the efficiency of our method on the synthetic datasets.

7.1 Data and Baseline Methods

A human-labeled ground truth dataset [6] consisting of 2000 Web pages was collected from 100 major print-worthy Web sites. These Web sites were selected from the 10 Web site types, namely *DIY, Recipe, Shopping, News, Jobs, Science, Article, Leaning, Travel and Sports*. They are top Web site types, which attract large numbers of Web page printing. For each Web site type we selected 10 Web sites, and for each Web site we selected 20 Web pages. The

[6]http://home.ustc.edu.cn/ stone/files/Publications/Data.zip

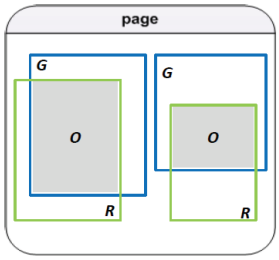


Figure 6: The region overlap

Web pages from the same Web site may have the different Web templates. On each Web page we manually select the informative clips as the ground truth.

We compare the proposed solution Clipping via Crowd Intelligence (**CIC** for short) with two baseline methods. The first one is the method used in the tool of Smart Print [9] (**SP** for short). This tool analyzes the visual features of a given Web page and aims to select one clip for the *main content* as recommendation. This recommendation usually covers a big area, inside which it may contain un-informative junks. Note that this is a method without the support of the wisdom of the crowds. The other baseline method is based on Maximal Frequent Itemset mining (**MFI** for short) over the transaction database for the clipping log data. Specifically, we can first generate all the maximal frequent itemsets and among them select the one with the largest number of items for recommendation.

7.2 Evaluation Method and Measure

We use leave-one-out cross validation to evaluate the accuracy of the methods on the data-sets. For the 20 Web pages from a Web site, we iteratively select one page as query and the log data on the remaining 19 Web pages are used to generate the transaction database for recommendation. The result is then compared against the ground truth of the query page.

A recommendation result actually refers to a set of content clips on the given Web page. Thus, we can evaluate its effectiveness by calculating the overlap area between the recommended clips and the ground truth. Note that each clip corresponds a bounding rectangular (area) with four coordinates (i.e. x, y, height and width) after rendering in the browser and we can easily get those coordinates. See the evaluation example in Figure 6. The two blue rectangles refers to the two clips selected by a user, and the two green ones refers to the two clips recommended by our methods. Thus, the Area O is their overlap area. Then, with the overlap area we can calculate the precision P, recall R and $F1$ score of the recommendation in terms of area size. Specifically,

$$P = \frac{|A_G \cap A_R|}{|A_R|}, R = \frac{|A_G \cap A_R|}{|A_G|}, F1 = 2 \times \frac{P \times R}{P + R}$$

Where A_G is the clipping region of ground truth, A_R is the clipping region of the recommendation result and $|\dots|$ denotes the region size. If the precision is less than 1, it means that we need to remove some areas from the recommendation. If the recall is less than 1, it indicates that we need to add some contents to get the exact clipping areas.

7.3 Experimental Results

Effectiveness. Table 2 records the average precision, recall and F1 score of the three methods on each Web site type. Specifically, we average the performance values over the 200 Web pages from each Web site type. In the recommendation we set $\alpha = 0.01$, $\beta = 0.1$ and $\lambda = 2$. The average precision and recall over all the 2000 Web pages of CIC reach 92.46% and 94.93% respectively. Compared with the SP method, CIC increases 23.81% in precision and 5.14% in recall. This clearly shows that leveraging the wisdom of the crowds greatly improves the recommendation performance. Compared with the best MFI result ($\alpha = 0.1$) in Table 3, CIC increases 1.34% in recall and 0.42% in precision. It indicates that CIC helps to find better patterns for recommendation. The reason why CIC can find better pattern has been detailed discussed in Section 3.3.

Table 2: The average recommendation performance (%) of CIC and SP on each Web site type ($\alpha = 0.01$, $\beta = 0.1$ and $\lambda = 2$)

WebSite Type	CIC			SP		
	P	*R*	*F1*	*P*	*R*	*F1*
DIY	89.34	93.7	89.85	70.14	93.31	76.43
Recipe	92.82	96.7	94.23	70.94	96.91	78.56
Shopping	80.96	89.69	83.11	34.81	81.61	41.47
News	93.3	94.82	93.36	68.99	78.14	66.23
Jobs	95.51	94.48	94.16	85.17	97.21	89.11
Science	91.56	92.95	90.96	71.02	89.4	72.01
Article	93.51	95.9	94.3	70.06	89.37	72.13
Learning	95.82	95.28	94.16	69.88	96.77	74.62
Travel	97.44	98.79	97.82	69.25	83.08	64.73
Sports	94.38	97.0	94.76	76.19	92.09	79.59
Average	**92.46**	**94.93**	**92.67**	**68.65**	**89.79**	**71.49**

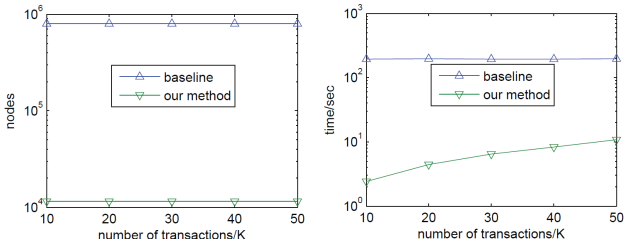
Table 3: The average recommendation performance (%) of MFI and CIC with different α on the 2000 Web pages

α	MFI			CIC ($\beta = 0.1$,$\lambda = 2$)		
	P	*R*	*F1*	*P*	*R*	*F1*
0.01	91.51	93.16	90.73	**92.46**	**94.93**	**92.67**
0.1	**92.04**	**93.69**	**91.85**	91.64	93.58	91.66
0.2	90.77	90.78	89.52	91.36	93.09	90.97
0.3	90.25	90.63	88.89	89.78	91.27	89.15
0.4	87.8	87.56	86.06	87.79	88.61	86.65
0.5	83.26	81.07	79.36	82.84	83.45	81.32

Table 4: Results of CIC (%) with different λ ($\alpha = 0.01$ and $\beta = 0.1$)

λ	*P*	*R*	*F1*
0.1	91.23	78.66	77.08
0.2	91.54	83.01	80.89
0.4	92.66	85.29	83.54
0.8	92.96	89.9	87.86
1.0	92.49	92.86	90.59
2.0	**92.46**	**94.93**	**92.67**
4.0	92.37	94.19	92.25
8.0	92.15	94.12	92.08
$+\infty$	90.77	91.67	89.53

Impact from λ. The parameter λ decides the trade-off between support and occupancy in the quality measure. Here, we also evaluate its impact on the recommendation performance. Table 4 shows the average performance over 2000 Web pages when λ changes from 0.1 to $+\infty$. $\lambda = +\infty$ means that we only consider occupancy in the quality measure. Clearly, we can see when λ increases from 0.1 to 2 both the precision and recall increase. When λ continues to increase from 2, both the precision and the recall drop. Usually, too much emphasis on frequency will typically recommend informative but uncomplete itemset, in other word, the recommend area R in Figure 6 is small and almost overlap with the overlap area O (Note that the ground truth area G is fixed). Thus, the precision is high and the recall is low when λ is small. With the increase



(a) Number of Nodes (b) Running Time

Figure 7: The effects of the number of transactions on MAFIA and our method

of λ from 0.1 to 2, both recommend area R and the overlap area O grow, leading to the large improvement on recall and the slight increase on precision. In contrast, too much emphasis on occupancy will recommend an over-complete itemset with too many items. However, in our application of *Smart Print* there are actually two kinds of items in the database—the positive items (the clips that users select) and the negative items (the clips that users de-select). When λ is increasing from 2 to $+\infty$, the recommendation will include many such negative items, leading to the decrease of overlap area O. Thus, both the precision and the recall drop. For our 2000 Web page data set, $\lambda = 2$ achieves the best result. In practice, λ can be chosen by cross validation to achieve the best performance in similar manners.

Efficiency. To show the efficiency of the proposed pruning methods in Section 4 we compare its running time with the straightforward method to our problem on the synthetic transaction databases with different numbers of transactions. The data sets used in the experiment are generated by the IBM synthetic data generator[7] for itemset patterns. Here, the straightforward method is simply frequent pattern mining with the check on occupancy. The implementation of MAFIA [2], a highly efficient method for finding frequent patterns, is adopted in this comparison. Our method leverages the properties in Section 4 to further prune the search space, thus may achieve better efficiency.

Here, we generate a database of $N = 10000$ transactions and scale it up by vertical concatenation of the database. The results are shown in Figure 7, including the running time and the number of nodes in the subset tree searched by the baseline and our method. Note that in this experiments we only duplicate the database to increase its size. Thus, the number of nodes visited in the subset tree is expected to stay unchanged. As can be seen from Figure 7(a), our method only searches about 1.4% nodes of those searched by the baseline. With respect to running time, our algorithm's running time grows linearly, from 2.45 seconds for 10000 transactions to 10.34 seconds for 50000 transactions. What is interesting is that the baseline's running time remains stable when the number of transactions grows. After careful investigation, we think it is due to the extreme efficiency of bit operators used extensively by MAFIA. However, since it takes the baseline about 195 seconds to run the experiment for each data set, our algorithm is still much faster.

8. CONCLUSIONS

In this work we harness the wisdom of the crowds for accurate Web page clipping. The crowd intelligence comes from the log data on how previous users clip Web pages. We formulate this problem as a new pattern mining task, *mining top-1 qualified pattern*, which considers both the support and occupancy in pattern recommendation. Along this line, we propose the properties on occupancy which helps to greatly prune the search space for high efficiency. In addition, we propose the generalized DOM tree paths to identify content clips in Web pages, and show its robustness to the change of DOM tree structures. Compared with previous works our method significantly improves the accuracy in Web page clipping, which is shown by the experimental study on the 2000 Web pages.

It is worth mentioning that the proposed problem of mining top-K qualified pattern can be useful to other pattern mining applications where each transaction corresponds to a set of actions for a task. Also, we can extend this concept of occupancy in sequential mining.

[7]http://www.cs.rpi.edu/ zaki/www-new/pmwiki.php/Software

9. ACKNOWLEDGEMENTS

The authors Lei Zhang, Enhong Chen and Guiquan Liu are supported by grants from Natural Science Foundation of China (Grant No. 60775037), The National Major Special Science & Technology Projects (Grant No. 2011ZX04016-071), The HeGaoJi National Major Special Science & Technology Projects (Grant No. 2012ZX01029001-002) and Research Fund for the Doctoral Program of Higher Education of China (20093402110017).

10. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD*, pages 207–216, 1993.

[2] D. Burdick, M. Calimlim, and J. Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Proceedings of the IEEE ICDE*, pages 443–452, 2001.

[3] J. Fan, P. Luo, S. H. Lim, S. Liu, P. Joshi, and J. Liu. Article clipper-a system for web article extraction. In *Proceedings of the ACM SIGKDD*, pages 743–746, 2011.

[4] K. Gade, J. Wang, and G. Karypis. Efficient closed pattern mining in the presence of tough block constraints. In *Proceedings of the ACM SIGKDD*, pages 138–147, 2004.

[5] B. Goethals and M. J.Zaki. Frequent itemset mining implementations. In *Proceedings of the ICDM workshop on Frequent Itemset Mining Implementations*, 2003.

[6] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of the ACM SIGMOD*, pages 1–12, 2000.

[7] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, pages 53–87, 2005.

[8] Y. Hu, G. Xin, R. Song, G. Hu, S. Shi, Y. Cao, and H. Li. Title extraction from bodies of html documents and its application to web page retrieval. In *Proceedings of the ACM SIGIR*, pages 250–257, 2005.

[9] S. H. Lim, L. Zheng, J. Jin, H. Hou, J. Fan, and J. Liu. Automatic selection of print-worthy content for enhanced web page printing experience. In *Proceedings of the ACM DocEng*, pages 165–168, 2010.

[10] P. Luo, J. Fan, S. Liu, F. Lin, Y. Xiong, and J. Liu. Web article extraction for web printing: a dom+visual based approach. In *Proceedings of the ACM DocEng*, pages 66–69, 2009.

[11] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proceedings of the ACM ICDT*, pages 398–416, 1999.

[12] J. Pasternack and D. Roth. Extracting article text from the web with maximum subsequence segmentation. In *Proceedings of the ACM WWW*, pages 971–980, 2009.

[13] J. Pei, J. Han, and L. Lakshmanan. Mining frequent itemsets with convertible constraints. In *Proceedings of the IEEE ICDE*, pages 433–442, 2001.

[14] J. Wang, C. Chen, C. Wang, J. Pei, J. Bu, Z. Guan, and W. V. Zhang. Can we learn a template-independent wrapper for news article extraction from a single training site? In *Proceedings of the ACM SIGKDD*, pages 1345–1354, 2009.

[15] J. Wang, J. Han, S. Member, Y. Lu, and P. Tzvetkov. Tfp: An efficient algorithm for mining top-k frequent closed itemsets. *IEEE Trans. on Knowledge and Data Engineering*, 17:2005, 2005.

[16] J. Xiao and J. Fan. Printmarmoset: Redesigning the print button for sustainability. In *Proceedings of the ACM CHI*, pages 109–112, 2009.