An Experimental Approach to Detect Similar Web Pages Based on 3-Levels of Similarity Clues*

WOOSUNG JUNG¹, EUNJOO LEE+ AND CHISU WU²

¹Software Capability Development Center
LG Electronics
Seoul, 137-130 Korea
²School of Computer Science and Engineering
Seoul National University
Seoul, 151-742 Korea
³School of Computer Science and Engineering
Kyungpook National University
Daegu, 702-701 Korea

It is hard to maintain web applications due to rapid changes and the proliferation of various techniques applied to web applications. Several approaches, such as clustering or refactoring web applications, have been suggested to improve their maintainability. The similarity measure is one of the principal criteria in these approaches. Existing studies on web similarity focused on semantic or context similarity. Most of the existing clone detection techniques concentrated on general applications, not web applications. In this paper, WSIM has been suggested to measure similarity in web applications, based on the usage degree of clues and two linking directions. The similarity clues include page relations, source and target entities, and parameters. WSIM can be classified in three levels and two directions. Six kinds of WSIMs are defined, and each WSIM has its own purpose. Finally, several experiments were conducted on simulated data and real open sources to validate the proposed WSIM.

Keywords: web application, similarity, page clone, clues, maintainability

1. INTRODUCTION

Maintaining web domains has become more cumbersome, as web environments evolve rapidly and requirements change frequently. It is hard to establish systematic development methodologies due to the varying techniques and languages used in Web Applications (WAs). Several approaches, such as clustering similar pages or refactoring web pages using duplicated codes and link information, have been proposed to support the maintenance of WAs and improve understandability. Similarity is a main criterion of these approaches.

Several studies have been made on similarity between web pages [1-6]. In [1-3], page similarity was defined based on the links between pages; however, the target is not the web application but web pages, and the defined similarity focus on semantic likeness. Lee et al. proposed a similarity measure in WAs based on cosine similarity, where the terms in the vector have been computed using the relations’ weight values between pages.
In that approach, pages that are only directly related are considered. That is, pages that are not directly related cannot be detected, such as the parallel structures in Fig. 2. In [4-6], similarity measures were defined to cluster similar pages in WAs, but the data used were mainly HTML tags and the defined similarity measures were mostly used to obtain the UI (user interface) similarity.

Existing pages are often reused as basic structure, when new web pages are created [4]. That is, a new page is created by copying and modifying the base page. The two example pages in Fig. 1 which share link structure and have different content, correspond to this case. However, it is difficult to detect this situation, because existing studies focus on finding similar code blocks or similar UIs. Developers often search code via search engines, such as koders [8], and google search [9], for code to reuse. There is little support for web code search, because web pages are composed of various entities, such as script code, HTML, and CSS. Therefore, several meaningful elements should be selected and abstracted to be utilized as search inputs. The elements may include the relation types or parameters between pages. Existing studies on detecting code clones or finding similar pages cannot be used in that way. Additionally, most of the existing studies on finding similar pages are based on the edit distance approach, where the target pages are changed into strings and the differences between two strings are utilized as a similarity measure. However, as web pages consist of many heterogeneous entities, it is not easy to select appropriate elements to make the string in the edit distance approach. In sum, existing similarity measures have been defined based on different criteria, and they incorporate a limited view on similarity. Consequently, a similarity measure that covers the various viewpoints of similarity in web applications is required.

In this paper, WSIM (Web SIMilarity) is proposed to detect similar pages in Web Applications. WSIM is based on [10]. In WSIM, several clues are identified. These clues include the source and target in the relations, relation types, and parameters. WSIM is classified in three levels, according to the usage of the clues. The relation types include hyperlink, submit, include, redirect, and load [11]. In this paper, ‘link’ refers to the general term that includes those relation types. The parameters are usually copied to reduce mistakes in handling parameter passing that causes several faulty cases [12]. Therefore, the number of parameters, and the parameter itself can be used as a meaningful clue to estimate similarity. Most existing studies do not fully utilize the parameters. For instance, [7, 11] consider only the number of parameters. In brief, clues include to-entity and from-entity, relationships such as submit, hyperlink, include, redirect, and so on, and set of parameters used in some relations. When two pages are identical, the clues between two pages are the same. When the clues between two pages are identical, it is highly possible that the two pages are similar, which is shown in the experimental results in section 5. The similarity between the two sets of clues does not always tell the similarity of the two pages. However, the usage of clues enables to reduce the cost of comparing pages. In addition, it is possible to compare the structural similarity between pages by using the clues, where the contents are ignored and only the structures between two pages are compared. In level 1, the relations in the selected entity are considered as to find similar pages which have similar structure to the entity. Parameters or target entities are not concerned in level 1. The clues in level 2 incorporate entity, relations, and parameters. As level 2 includes more concrete clues than level 1, the precision becomes higher and the number of target pages to be compared becomes lower than those of level 1. Level 3
utilizes relations, parameters, and source or target entity, which is the strongest condition among the three levels. The precision and recall are considered to select the appropriate level. That is, level 3 is suitable to find very similar pages though the number of detected pages is small. When every similar page is expected to be retrieved, level 1 is used. The directions of relations are also applied to WSIM in this paper. OUT direction is used to determine structural similarity between pages and IN direction is used to find usage similarity between pages. For example, assume that two similar library functions exist and they do not depend on other entities, that is, they do not use other entities. The OUT-similarity cannot detect this case, but IN-similarity may detect it when they are used in similar ways. In this case, refactoring may be applied further, for combining them to reduce redundancy. Four ways, connected to the three levels of WSIM, are introduced to use WSIM. The tool, WANA [11, 13], has been modified and extended to apply WSIM to simulated WAs and real world WAs. Several experiments have been conducted that present a variety of views. Experimental results show the applicability of WSIM.

The contributions of this paper are as follows: (1) WSIM comprehends various aspects of similarity in WAs, because it has been defined using clue levels and link directions. Thus, it can complement the missing points of existing similarity; (2) WSIM measures the similarity by link structure, not by UI level, so it will be helpful to developers who are interested in reusing the link structure in pages and handling parameter passing between pages; (3) WSIM is also applied to refactoring or clustering to improve the maintainability of WAs.

The remaining sections of this paper are organized as follows. Section 2 describes the motivation and background of WSIM. Section 3 presents the concept of clues, levels, and WSIM definitions. Section 4 explains four usages of WSIM. Section 5 presents the results of several different experimental methodologies. Section 6 describes related work to page similarity and code clone detection. Section 7 outlines the summary, contributions, and future work of this study.

2. MOTIVATION AND BACKGROUND

Figs. 1 (a) and (b) have the same link structure, such as five hyperlinks and one submit, but their contents are different. When the existing similarity measures are applied, the similarity between the two pages becomes low, although two pages are ‘similar’ from a link structure view. That situation was the starting point of this work. Some reference pages are analyzed and new pages are created based on the reference pages to achieve a rapid development of WA [4]. An extreme case is multilingual pages, such as Korean and English pages, in the web sites of an organization. Another example is add/delete/modify product or customer information in an e-commerce system. The ‘add’, ‘delete’, and ‘modify’ pages of customers and the pages of products have a good chance to be structurally similar, although the inner data differ. If the customer pages are first created, then the product pages are very likely to be developed based on customer pages, and vice versa.

Parameters are frequently used in FORMs, hyperlinks, etc. To our knowledge, there is a lack of studies on web similarity that actively use parameters, although parameters are semantically meaningful identifiers, such as ‘name’ and ‘address’ in the registration
Fig. 1. Motivation example.

Fig. 2. Example of parallel structure in B2B.

page. Assume that ‘add.jsp/delete.jsp/edit.jsp’ exist in the product and in the customer folder of the example in Fig. 2. It is quite likely that ‘product/add.jsp’ is similar to ‘customer/add.jsp’. In this case, two ‘add.jsp’s are referred to as sharing a ‘parallel structure’, in this paper. Two pages are also referred to as ‘parallel pages’. Fig. 2 shows the open source B2B [14]’s folder structure. Ten folders in ‘admin’ folders have similar files, as shown in Fig. 2. Each ‘add.jsp’ in ten folders is a parallel page, and the code in each page is very similar. Thus, it seems probable they have been created based on the same reference page.

However, existing clone detection techniques and web similarity measures have limitations in finding those cases. In most approaches where the edit distance is applied [4, 6], it is a key point to transform a page into an abstract representation, such as a tree or string. If the web pages are not well structured, then it is difficult to extract data to be abstracted. For example, if HTML tags are used, as in [4], then the similarity is limited in UI level. Existing clone detection techniques, such as CCFinder [15] and CP-Miner [16, 17] do not focus on the web domain, although they are regarded as ‘state of the art’. Most web page similarity metrics are also close to semantic similarity based on hyperlinks. They can find only directly related similar pages and the parallel structure cannot be detected. Most web page similarity metrics do not utilize parameters, so the precision becomes low. When page \( a \) is copied to page \( b \) and the content in \( b \) is modified, it is difficult to determine whether or not the page is similar using existing clone detection tech-
niques, although the linking structure in the two pages is similar.

The target entity, as well as the relationship types and parameters, are essential clues used to determine the similarity. When there are links from \( a.jsp \) to \( b.jsp \), \( b.jsp \) is the target entity, in terms of the ‘OUT’ direction. In terms of the ‘IN’ direction, \( a.jsp \) is the target entity. In the case of parallel pages, the targets of out links are different. That is, ‘product/add.jsp’ passes FORM parameters to ‘product/list.jsp’ and ‘customer/add.jsp’ passes parameters to ‘customer/list.jsp’. In two relations, it will be a powerful clue that two target entities are strongly similar. Assume that \{category1.jsp, category2.jsp\} are related to \{product1.html, product2.html, product3.html\} in the OUT direction and the parameters on each relation are similar. It is possible that ‘category1.jsp’ and ‘category2.jsp’ are very similar. That is, it can be said that links and parameters are copied and pasted from ‘category1.jsp’ to ‘category2.jsp’, or vice versa. This approach to use target entities is slightly related to bibliographic similarity [18]; however, the latter focuses on the content similarity of web pages, not web applications. It is helpful to detect near-miss clones [19] that are mostly similar, but not exactly the same, using the out links.

Most existing studies on similarity or clone detection techniques have focused on ‘OUT’ direction. However, the ‘IN’ direction can also be used to detect the similarity of usage. That is, similar pages on ‘IN direction’ tend to be used in similar ways, such as header or footer pages that are commonly included in a group of pages. A few studies on web similarity have considered ‘IN direction’, such as similarity of co-citation analysis; however, the purpose of those studies differs from that of this work in that they focused on only content similarity between static pages and they did not consider usage similarity in web applications where static pages and dynamic pages are mixed. To take the case of header and footer pages, it is highly possible that the content in the two pages differ. The existing clone detection techniques cannot incorporate this case, and the existing similarity studies based on IN direction are unsuitable for web applications.

Fig. 3 presents the concrete example, where each node is a page and the relations are all ‘include’. That is, pages \( a1, b1, c1 \) include headerKR and footerKR, while pages \( a2, b2, c2 \) include headerEN and footerEN. It seems quite probable that four pages, headerKR, footerKR, headerEN, and footerEN have similar in-links structures. Furthermore, assume that each header page has parameters, \( p1 \) and \( p2 \). Each footer page also has parameters, \( q1, q2 \) and \( q3 \). Consequently, headerKR and headerEN may have high similarity and footerKR and footerEN may have high similarity in this proposed approach. In brief, parallel files that use similarly can be detected, though an existing link-based similarity, such as SIMRANK [1], hardly finds the similarity.

The source entity can be included to the similarity clues, like the target entities in
the similarity based on the OUT direction. When the source pages are considered, similar
groups \(\{\text{headerKR}, \text{footerKR}, \text{headerEN}, \text{footerEN}\}\) are divided into \(\{\text{headerKR}, \text{footerKR}\}\) and \(\{\text{headerEN}, \text{footerEN}\}\).

In summary, not only OUT direction, but also IN direction is utilized in this paper.
Besides, the similarity clues including relations, parameters, and source/target entities are
variously concerned with showing the diverse aspects in the similarity measure, referred
to as ‘level’ in this paper. A more detailed explanation follows,

**Directions:** Existing studies on link-based similarity mostly considered one direction, IN
or OUT [1, 2]. Extended PageSim [20] considers bi-direction, but it is applicable to web
pages, not web applications. IN and OUT directions have a distinct meaning in web
applications. For IN direction, each page is regarded as a black-box, and similar pages in-
dicate that they are used in similar ways. Both \(a\) and \(b\) in Fig. 4 (a) correspond to that
case. In addition to pages, the target of similarities includes the dynamic links derived
from static analysis. In the case of OUT direction, two pages are considered as similar
when the two pages have similar link structures, like \(c\) and \(d\) in Fig. 4 (b). That is, the
OUT direction shows the similarity degree of the link structures between the web pages.

![Fig. 4. In-links vs. out-links.](image)

**Levels:** Most existing studies on similarity tune thresholds to obtain results. Practically,
empirical skill is sometimes required to control the thresholds; therefore, a more system-
atic approach may be needed to improve the quality of results. In this paper, the proposed
similarity, WSIM, has three levels according to the usage degree of the clues. Fig. 5 il-
lustrates the concept of levels. The three levels of similarity provide multi-views. Higher
level uses a stronger condition for similarity than the lower ones, which introduces the
trade-off between recall and precision. That is, a high similarity value in the case of high
level implies the high possibility of real clones, which results in high accuracy. However,
the high level misses some types of clones, because of its stringent conditions, resulting
in low recall. Additionally, each level also provides specific views. Fig. 6 represents
the clues used to calculate the similarity for each level. For a simple explanation, only in-
links are assumed for three levels. For level 1, the similarity is calculated based on the
types and relation count. It shows the highest coverage of similarity types, because it has
the weakest condition among three levels. For level 2, the parameter types and numbers
are considered for similarity, in addition to level 1’s condition. Finally, the condition of
level 3 includes comparing the source and target pages and previous conditions, as well.
Leveling is a kind of filtering process for each view. For example, the parallel structure of clone clusters, which have relatively similar relations and nodes but different physical pages, can be found with level 2’s condition. A base page is often copied and modified in web development [4], and the parallel structure is one such case. In the latter case, the modified parts in the clone page are the target entities of the relations. Level 2 enables the detection of the latter case. Conversely, level 3’s conditions require the same source or target pages, which is the case of using the same file or being used by a same file. Level 3 is more effective for this form of cloning. In short, each level has its distinct purpose, and it is hard to obtain the effect of leveling by only controlling the threshold of existing similarity metrics.

3. WSIM: THREE LEVELS OF PAGE SIMILARITY

Fig. 7 overviews obtaining the WSIM value between foo1.jsp and foo2.jsp using several clues the pages have. A clue is a set of 4-dimensional vectors that includes abstract information, such as entities, relations, and parameters. It is used to indirectly calculate the similarity between two concrete web pages. The more positive clues between two pages exist, the more similar the two pages are. For further details on clues, see Definition 1 in section 3.1.
3.1 Terms and Definitions

This subsection formally defines the basic concepts and terms in this work.

**Definition 1** Clues

A clue is a set of 4-dimensional vectors that includes abstract information such as two entities, relations, and parameters. It is used to indirectly calculate the similarity between two concrete web pages. Thus, it is independent of the contents and languages of target web applications.

The clue is represented as “e→r→es→et→p”, as is shown in Fig. 7. “r” is the type of relation, and “es”, “et” are the source and target page of the relation, respectively. Finally, “p” is a parameter that is submitted from the relation. In the case of missing parameters, we assume that a clue has null as its special parameter. In the case of multiple parameters, we separate them into multiple distinctive clues. One relationship between two pages can have many clues depending on the number of parameters. When pages a and b are related to ‘submit’ and three parameters are passed for that submit relation, a minimum of three clues is created. Fig. 8 presents this situation. Assume that the relation between a and b is ‘submit’ and p, p1, and p2 are parameters. Consequently, the clues of (a), (b), and (c) in Fig. 8 are a→submit→null→b, a→submit→p→b, and {a→submit→p1→b, a→submit→p2→b}, respectively.

**Definition 2** Clue-Patterns (CP) and Meta-Clue-Patterns (MCP)

A clue, which includes “?” as its special element, is termed a pattern. “?” is a form of wildcard and can be replaced with any other possible values. However, if the pattern is a form of a lambda function, then it is termed a meta-pattern, because its elements are not yet fully determined. For example, let meta-pattern $\varphi = \lambda e.(e→p→?)$, then $\varphi$ is a meta-pattern that has wildcards as its relation, and a parameter and target entity after the source entity are determined. A meta-pattern can be shifted to a lower level by $\Delta$ level like $\varphi_{\Delta$level. The concept of level is explained in section 2. For example, $\varphi_{\Delta = 1} = (?→p→?)$. Thus, if the level is shifted down, then the wildcards replace the previous values in a clue by the count of $\Delta$ level. In this case, the number of variables of the pattern decreases as the level becomes lower, because “?” replaces the variable “e”. In contrast, the number of variables increases, as the level becomes higher. The order of replacement is source entity, relation, parameter, and target entity. The case that all elements of a clue are replaced by “?” is considered as a special one that satisfies a condition for both clue-pattern and meta-pattern, even though the pattern is not a lambda function.

If a meta-pattern is given full values, then it finally becomes a clue-pattern, and it is equivalent to a bag of possible combinations of clues that satisfy the given pattern. Therefore, $\psi(e_1) = (e_1→p→?) = [e_1→p→e_1]_{\Delta=0}$, where r, p, and e are all relations in $e_1$, all parameters in $e_1$, and the target entity of $e_1$. $|\psi(e_1)|$ is the size of the bag. Finally, $\psi_{\Delta}(e)$ is a set of possible combinations of meta-patterns at the level of $\Delta$, which has only one input parameter as a source page. It is termed a meta-pattern (MCP) generator. For example, if
an entity “e” has two relations “r₁” and “r₂”, then \( \psi_l(e) = \{ \lambda x.(x \rightarrow r₁? \rightarrow ?), \lambda x.(x \rightarrow r₂? \rightarrow ?) \} \). Each CP for three levels is defined in section 3.2.

**Definition 3  WSIM-Comparability**

For given \( e_1 \) and \( e_2 \), which are different entities of a target web application, if they satisfy “\( \exists \varphi \in \psi_e \cap \psi_{e'} \cdot \varphi(e) \cup \varphi(e') \neq \varnothing \)”, then \( e_1 \) and \( e_2 \) are termed “WSIM-comparable at the level of \( L \)”. \( \psi(k) \) is the MP set of entity \( k \) and \( \varphi(k) \) is \( k \)'s CP. In brief, it can be said that two entities are comparable when common MPs exist between two entities and one or more CPs satisfy the corresponding MP for any of the two entities. WSIM can be calculated in this case. However, in the other case, it is incalculable, and the entities are not considered similar because no clues exist to determine it. Thus, only WSIM-comparable pairs are considered in detecting clone pages.

\( K_L(e) \) is the set of WSIM\( _L \)-comparable pages with “\( e \)”.

\[
K_L(e) = \{ e' \mid \exists \varphi \in \psi_e \cap \psi_{e'}, \varphi(e) \cup \varphi(e') \neq \varnothing \}. \]

**Definition 4  vSet\( _L \)**

\( vSet_L(e) \) is a function that returns the set of 3-dimensional vectors in the form of \(<e, e', WSIM_L(e, e')>\), where \( e \) is an entity, \( e' \) is an element of \( K_L(e) \), and WSIM\( _L(e, e') \) is a similarity measure between \( e \) and \( e' \). \( vSet_L(e) \) is sorted in descending order based on the measure. In short, \( vSet_L(e) \) is the sorted list by WSIM value of \( e \) and similar entities of \( e \) at level \( L \). For a detailed example, see Fig. 11.

\[
vSet_L(e) = \text{Sort} \nabla wSIM_L(\cup_{e \in K_L(e)} <e, e', wSIM_L(e, e')>)
\]

\( vSet_L(e)[i] \) (\( 1 \leq i \leq |vSet_L(e)| \)) denotes \( i \)th vector of \( vSet_L(e) \). The range, e.g., \( 1 \ldots 3 \), can replace the number, resulting in a partial ordered set. The subset of columns, \( vSet_L(e).c(r) \), represents the \( vSet_L(e) \) that is only composed of its \( r \)th elements. Thus, \( vSet_L(e).c(r) \) is a partial ordered set of WSIM\( _L \) values.

**Definition 4.1  vdSet\( _L \)**

\( vdSet_{e_1, e_2} \) is a function that returns the set of 4-dimensional vectors in the form of \(<e_1, e_2, WSIM, \Delta WSIM>\), where \( e_1 \) and \( e_2 \) are entities. WSIM, \( \Delta WSIM \) are used for visualization later.

**Definition 5  sSet\( _L \)**

\( sSet_L \) is similar to \( vSet_L \), except that the third element in the vector is the standardized value, \( std_L \), not WSIM. That is, \(<e, e', std_L(e, e')>\). Therefore, \( std_L \), not WSIM, is the criterion value to sort the standardized value. Step 2 (b) in Fig. 13 illustrates a detailed example. The WSIM values in each cluster tend to follow some distribution. The standardized value, \( std_L \), is calculated by Eq. (1). To get \( std_L(a, b) \), the difference between WSIM\( (a, b) \) and average WSIM of the cluster is divided by the standard deviation of the cluster. In short, \( std_L \) is used to find the outlier value in a cluster.

\[
std_L(e, e') = \frac{WSIM_L(e, e') - \nu AVG_L(e')}{\sigma_L(e')}
\]

where
\(e_i, e_j\): entities. Specially, \(e_i\) is the common target of the cluster, such as \(a\) or \(b\) in Fig. 12. 
\(e_j\): The comparable entities to \(e_i\) that belong to the same cluster, such as, \(\{x, y, z, w\}\) or \(\{p, q, r, s\}\) in Fig. 12.

\(\text{WSIM}_{L}(e_i, e_j)\): the WSIM value between \(e_i\) and \(e_j\). For a detailed definition, see Fig. 9.

\[vAVG_{L}(e_i) = \frac{\sum \text{WSIM}_{L}(e_i, e_j)}{|vSet_{L}(e_i)|},\] the average WSIM in a cluster.

\[\sigma_{L}(e_i) = \sqrt{\frac{\sum (\text{WSIM}_{L}(e_i, e_j) - vAVG_{L}(e_i))^2}{|vSet_{L}(e_i)|}},\] the standard deviation in a cluster.

\(sSet_{L}(e_i)\) is defined similarly to \(vSet_{L}(e_i)\), as follows,

\[sSet_{L}(e_i) = \text{Sort} \backslash \text{std}_{L}(\bigcup e' \in K_{L}(e_i) < e, e', \text{std}_{L}(e, e')>).\]

### 3.2 WSIM Formula

It is assumed that page similarity is based on two conditions, how specific the clues are and how many common clues they have; they represent the level of clues and the result of WSIM, respectively. Thus, different types of similarity can be measured based on the levels of clues.

**Level 1 (L1):** \(e \xrightarrow{\text{re}} \) is used as a clue-pattern. Thus, WSIM is calculated based on the types, structure, and number of relations. Therefore, it is independent of the parameters or target pages. Thus, level 1 has the highest coverage of similarity types and can be applied between heterogeneous web applications. However, it has low accuracy because it is very sensitive to noise. Level 1 is also helpful in finding similar pages between heterogeneous web applications that cannot be detected with other levels because it calculates the similarity only from the structural information based on the relation and source/target entities without considering the parameters.

**Level 2 (L2):** \(e \xrightarrow{\text{pre}} \) is used as a clue-pattern. Thus, the parameter types and numbers are considered to calculate WSIM, in addition to level 1. Thus, it can be applied to parallel pages because it still does not consider the target page conditions. Level 2 has the best practical qualities among the three levels because it is more robust to noise than level 1, and it covers the case of level 3. However, it is unprofitable for heterogeneous web applications. Level 2 is applicable to reveal parallel structure.

**Level 3 (L3):** \(e \xrightarrow{\text{rpee}} e'\) is used as a clue-pattern. It is the strongest condition among the three levels. Thus, the highest similarity values are given to calculate WSIM when pages have common clues in this case. However, level 3 lacks flexibility compared to the other levels, so the coverage of clone types is very small. Thus, level 3 is the best choice for “copy&paste” clones, but it will fail to detect the clones, if there have been modifications. The cost of calculating WSIM at level 3 is lower than level 2 because the set of satisfying level 3 conditions is rare.

\(\text{WSIM}_{L}(e_1, e_2)\) is defined in Fig. 9. The WSIM value can be calculated from the sum
DETECTING SIMILAR WEB PAGES USING THREE LEVELS OF CLUES

of the products of point($P_L$), weight($W_L$), and scarcity($S_L$), based on the level conditions. When the direction is IN(OUT), the in-links(out-links) to(from) $e_1$ or $e_2$ are counted.

$$WSIM_L(e_1, e_2) = \sum_{\phi \in PL(e_1) \cap PL(e_2)} P_L(\phi, e_1, e_2) \cdot W_L(\phi, e_1, e_2) \cdot S_L(\phi)$$

where

$$P_L(\phi, e_1, e_2) = C_L \cdot \left[ \frac{|\phi_L(e_1)|}{|\phi_L(e_1) \cup \phi_L(e_2)|} \right] \cdot \left[ \frac{|\phi_L(e_2)|}{|\phi_L(e_1) \cup \phi_L(e_2)|} \right]$$

$$C_L = 0.1 + \frac{L-1}{2}$$

$$W_L(\phi, e_1, e_2) = \frac{|\phi_L(e_1) \cup \phi_L(e_2)|}{|\phi_L(e_1) \cup \phi_L(e_2)|}$$

$$S_L(\phi) = (1 + \ln \left| \frac{q(\phi)}{\varphi(?) | \varphi(?) \|} \right|) \cdot \left[ \frac{q_{L,2}(?) \|}{|\varphi(?) |} \right]$$

Fig. 9. Definitions for WSIM formula.

A point, $P_L$, can have a plus or minus value based on $C_L$, a constant value determined by the level. When $e_1$ and $e_2$ have a similar number of clues under the same clue condition, the $P_L$ value approximates $C_L$ as a positive value. Conversely, when the clue numbers of $e_1$ and $e_2$ are very different, $P_L$ becomes negative. As the level becomes higher, $C_L$ becomes higher, because a high level puts high weighting on similar clues.

A weight, $W_L$, is the rate of participating clues on the total pages. $W_L$’s denominator is the total value of clues that $e_1$ and $e_2$ have, under the primitive level, and $W_L$’s numerator is the number of clues under the corresponding level. For example, in the case of level 2, $W_L$ is calculated by dividing the number of cases, where the relation type and parameter name are identical, by the number of cases where only the relation type is equal.

Scarcity, $S_L$, is related to each clue-patterns probability. $W_L$ focuses on the target entities $e_1$ and $e_2$, but $S_L$ focuses on all entities. Assume a case in which the level is 2, relation is submit, and parameter is $p1$. $S_L$ is derived by dividing the number of whole CP that has <submit, $p1$> by the number of all clue patterns that have submit relations. In brief, $S_L$ indicates CP’s scarcity on all relations.

The first process for WSIM is to filter out candidates for clone pages among all comparable pairs, because the cost is too high if all pairs are considered. If Page is assumed as the set of total target pages, then Fig. 10 represents the overall approach. In Fig. 10, $xSet_L(e)$ denotes $vSet_L(e)$ or $sSet_L(e)$.

Thus far, we focused on the out-WSIM. In case of in-WSIM, it is sufficient to re
serve the direction to out-WSIM, so the repeated explanation is omitted. In sum, out-WSIM utilizes relations, parameters, and target entity based on the source entity. However, in-WSIM uses those clues based on the target entity to find similar pages in view of “referenced”, not “referring”.

4. FOUR WAYS TO USE WSIM

This section suggests four practical approaches to calculate WSIM. Some factors differentiate each method. “global” implies that the method uses all pairs when selecting clone candidates, while “clustered” focuses on the set of groups based on each entity. “static” means that the size to be selected for the candidates is fixed, while “dynamic” is not. Additionally, WSIM, delta-WSIM, and std are used to decide candidates. Each case is explained in detail.

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>10</td>
</tr>
<tr>
<td>y</td>
<td>x</td>
<td>9.5</td>
</tr>
<tr>
<td>a</td>
<td>x</td>
<td>9.3</td>
</tr>
<tr>
<td>p</td>
<td>q</td>
<td>8.6</td>
</tr>
</tbody>
</table>

Fig. 11. Example of global dynamic selection.

4.1 Global Dynamic Selection

This is the simplest method among four approaches; it selects the top rank pairs after being sorted by WSIM in descending order for all comparable pairs in vSet. This approach is very easy, intuitive, and especially effective in the case of level 3, due to its clear distribution of clone pairs. However, the high rate of false positives for other levels is one of the disadvantages of this method; this results in lower rate of recall and precision than those of other clustered methods. Thus, this method is effective only for level 3, because it requires the lowest cost to calculate its WSIM. The number of used clues in level 3 is larger than level 1 or level 2 and other techniques in sections 4.2, 4.3, and 4.4 requires more effort than this global approach. Therefore, only WSIM suffices to get similar pages in level 3.

4.2 Clustered Static Selection

This is a method to obtain the upper K pairs from vSet(e), which is a cluster where entity e is the criterion entity. A suggested list size, K, is a pre-set constant number. Therefore, when the number of clusters is N and the suggested list size is K, the number of candidate pairs is N • K. In most cases, actual similar pairs tend to be located at the upper ranks in a cluster, and this method is useful to improve the recall rate. However, it is highly possible that several false positive or false negative cases can be generated, because each cluster has various distributions of similar pairs. Some clusters may have few similar pairs, but others may have more similar pairs than those of K. Consequently, this
method is applicable for simulation to check the performance and usefulness of each level with simulated data. This method can be practically applied to level 1 because it is hard to make a distinction between clues in level 1. That is, although the dynamic method is generally more effective than the static one, it is hard to select an outlier WSIM that can be used as a cutline in the case of level 1.

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>x</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>9.5</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>9.1</td>
</tr>
<tr>
<td></td>
<td>w</td>
<td>8.5</td>
</tr>
<tr>
<td>b</td>
<td>p</td>
<td>9.3</td>
</tr>
<tr>
<td></td>
<td>q</td>
<td>7.8</td>
</tr>
<tr>
<td></td>
<td>r</td>
<td>8.1</td>
</tr>
<tr>
<td></td>
<td>s</td>
<td>7.0</td>
</tr>
</tbody>
</table>

Fig. 12. An example of clustered static selection.

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>x</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>9.5</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>9.1</td>
</tr>
<tr>
<td></td>
<td>v</td>
<td>8.8</td>
</tr>
<tr>
<td></td>
<td>w</td>
<td>4.5</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>p</td>
<td>9.3</td>
</tr>
<tr>
<td></td>
<td>q</td>
<td>7.8</td>
</tr>
<tr>
<td></td>
<td>r'</td>
<td>3.1</td>
</tr>
</tbody>
</table>

Fig. 13. Example of delta-WSIM based clustered dynamic selection.

4.3 Delta-WSIM based Clustered Dynamic Selection

The static selection method has two main problems. First, many false negative pairs can be generated because the number of similar pairs exceeds the suggested list size $K$. That is, actual similar pairs cannot be retrieved. Second, false positive pairs can be generated because the number of similar pairs is below the suggested list size. This method uses WSIM and delta-WSIM to find the outlier pair to handle those problems. After finding the outlier, each cluster sets the cutline based on the outlier. Consequently, the dynamic selection technique is applied, because each cluster has a different cutline, according to its distribution. That is, the pairs whose WSIM values are relatively large are first targeted, and the delta-WSIM values are then calculated, which means the difference between two adjacent pairs in total ranks, such as between $<a, x>$ and $<a, y>$ in Fig. 13. The point where the delta-WSIM is extraordinarily large can be the cutline to retrieve similar pairs. In Fig. 13, $<a, v>$ and $<b, q>$ can be the cutline pairs in each cluster. This method is a two-dimensional approach compared to the methods described in sections
4.1 and 4.2, because it utilizes both WSIM and delta-WSIM. This improves the precision of the results.

There can be several ways to use WSIM and delta-WSIM to find cutting points, such as the tables-based method in Fig. 13. However, it is inefficient; a more appropriate visualization technique is needed. This paper proposes a graphical approach, where WSIM and delta-WSIM individually correspond to the x-axis and y-axis and the distribution scales are indicated by circled areas. Figs. 19 and 20 show the cases. In the graph, the points distant from the origin, that is, the points whose WSIM and delta-WSIM value are high can be the candidate cutlines. Section 5.1.3 provides a more detailed explanation of this graph. However, it is not easy to determine the appropriate cutline in the selection in the graph. Therefore, it is desirable to substitute this method with the std-based method of section 4.4.

4.4 Std-based Global Dynamic Selection

The clustered static selection method in section 4.2 improves the recall rate, but it is highly likely to result in many false positives or false negatives. It is hard to practically apply, a better method, the delta-WSIM based method in section 4.3 [10], because the graph is not intuitive and the secondary effort is required to find the cutline. First, the candidate cutline points are determined as the red oval area in Fig. 19. Second, the upper pairs on the cutline are retrieved.

In this work, the distribution in each cluster is considered to incorporate the merit of the clustered dynamic selection and global dynamic selection. That is, this method uses not the original WSIM but standardized WSIM (std) per cluster. As mentioned above, a cluster indicates a set of pairs composed of one fixed entity and other entities related to the fixed entity. A std value of a pair shows the distance between the average WSIM of the cluster and the WSIM of the pair, using the standard deviation of the pair by times. For example, the std of \(<a, b>\) in Fig. 14 is calculated using the following formula:

\[
\text{std} = \frac{4.000 - 2.460}{1.226} = 1.256.
\]

Conclusively, std of a pair shows a degree of outlier. This approach is helpful to get the outlier in each cluster, because the aspect of distribution in each cluster varies. Low WSIM can be an outlier; however, previous approaches tend to ignore this case. For example, the WSIM of \(<a, b>\) in Group #1 in Fig. 14 is 4.0, which is twice that of \(<p, q>\) in Group #2 in Fig. 14. However, the std of \(<a, b>\) is lower than that of \(<p, q>\), which implies that the WSIM of \(<p, q>\) is a probable outlier. Therefore, it is highly possible that \(<p, q>\) are similar pairs.

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
<th>std</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>4.0</td>
<td>1.256</td>
</tr>
<tr>
<td>c</td>
<td>3.3</td>
<td>0.685</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>2.5</td>
<td>0.033</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>1.3</td>
<td>−0.946</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>1.2</td>
<td>−1.028</td>
<td></td>
</tr>
<tr>
<td>average (WSIM)</td>
<td></td>
<td>2.460</td>
<td></td>
</tr>
<tr>
<td>stdev (WSIM)</td>
<td></td>
<td>1.226</td>
<td></td>
</tr>
</tbody>
</table>

(a) Group #1.

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
<th>std</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>q</td>
<td>2.0</td>
<td>1.780</td>
</tr>
<tr>
<td>r</td>
<td>0.3</td>
<td>−0.292</td>
<td></td>
</tr>
<tr>
<td>s</td>
<td>0.2</td>
<td>−0.414</td>
<td></td>
</tr>
<tr>
<td>v</td>
<td>0.1</td>
<td>−0.536</td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>0.1</td>
<td>−0.536</td>
<td></td>
</tr>
<tr>
<td>average (WSIM)</td>
<td></td>
<td>0.540</td>
<td></td>
</tr>
<tr>
<td>stdev (WSIM)</td>
<td></td>
<td>0.820</td>
<td></td>
</tr>
</tbody>
</table>

(b) Group #2.

Fig. 14. Example of std-based global dynamic selection (I).
DETECTING SIMILAR WEB PAGES USING THREE LEVELS OF CLUES

This method is based on both WSIM and std, as is shown in Fig. 15, which is the result of Fig. 14. This approach considers both WSIM and std values. Each entity pair has its WSIM and std value pair, for example, entity pair <a, x> has value pair <4.0, 1.780>. The pairs whose WSIM and std are high, become similar pairs with a high possibility. Then, the pairs with high WSIM and low std, or the pairs with high std and low WSIM, are secondly considered. Section 5.1.4 details the selection process.

When delta-WSIM is used, as described in section 4.3, a complex graph like Fig. 19 is necessary to find the appropriate threshold. After finding the threshold, the list of clusters, where pairs are sorted by WSIM, is required to find similar pairs using the thresholds. This is a cumbersome task. Compared to the delta-WSIM based method, this std-based approach finds not threshold but similar pairs directly using WSIM and std, incorporating the advantage of the global approach and static approach. This method is useful to find the outlier per cluster, which may be missed in the global approach in section 4.1. Although the cluster-based approach is used in the methods in sections 4.2 and 4.3, it is hard to find the case <p, q> in Fig. 14 in both approaches. This shows the case that lower WSIM does not always indicate lower similarity between two entities. The method in section 4.2 also has a weakness, because it is a static approach. The global approach generally has several advantages, such as high practicality due to its simplicity. It is also relatively simple to find the thresholds. However, the recall rate and precision are high in the std-based approach. The std-based method displays good results, especially in level 2 and level 3, which are shown in the next section; however, additional computational cost is incurred by calculating std. Due to the additional std computation, it can be concluded that this method is appropriate for level 2, although the quality of results is better than those of other methods. When the number of pairs in a cluster is insufficient to determine the tendency of the distribution, std may not show an exact outlier. After all, the experimental results show that the std-based method makes up for the weakness in the global approach, where only WSIM is used. The experimental results also show that the std-based method is superior to the delta-WSIM based method. The global approach shown in section 4.1 is sufficient to level 3, because most clues are used in level 3, which is an offset to std usage. Practically, the static approach in section 4.2 is applicable to level 1, although the static approach has some weaknesses.

In this paper, similarity is classified by levels and directions. To understand the meaning of levels and directions, refer section 2. The directions are used to detail each level, such as IN/OUT of level 1, IN/OUT of level 2, and IN/OUT of level 3. Thus, it is important to match each level to one of four ways. As said above, clue conditions become weaker as level is lower. The global dynamic method (in section 4.1), referred as

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>x</td>
<td>4.0</td>
</tr>
<tr>
<td>a</td>
<td>c</td>
<td>3.3</td>
</tr>
<tr>
<td>a</td>
<td>d</td>
<td>2.5</td>
</tr>
<tr>
<td>p</td>
<td>q</td>
<td>2.0</td>
</tr>
</tbody>
</table>

- (a) by WSIM.

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>std</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>q</td>
<td>1.780</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td>1.256</td>
</tr>
<tr>
<td>a</td>
<td>c</td>
<td>0.685</td>
</tr>
<tr>
<td>a</td>
<td>d</td>
<td>0.033</td>
</tr>
</tbody>
</table>

- (b) by std.

Fig. 15. Example of std-based global dynamic selection (II).
naïve approach, requires the least cost and is the simplest way to detect clone pages among the four approaches. However, the precision and the recall of the approach are too low to be used practically except for level 3 type clone pages. The delta-WSIM based method (in section 4.3) and std-based approach (in section 4.4) require additional effort to compute delta-WSIM and std value respectively, but their precisions are relatively high. Static approach (in section 4.2) has several drawbacks like generating many false positives and false negatives, though it shows the improved recall rate especially for some level 1 cases. Thus, appropriate way should be matched to each level by considering these characteristics.

First, the naïve approach is appropriate to level 3. As level 3 is the most robust to noise due to using full conditions of clues, similar pages are clearly distinct from other pages. That makes the WSIM values of level 3 be effective indicators even in the naïve approach. Therefore, naïve approach suffices to level 3 due to the lower cost and its simplicity. Of course, other approaches like std or delta-based approach also shows good results in level 3, however, they require additional cost and complex application. Second, the static approach is suitable for level 1. As it is difficult to discern clues in level 1, other dynamic approaches are not applicable to level 1. When the distribution of similar pairs is already known or the number of similar pairs is fixed beforehand, the static approach generates better results, though it has several drawbacks like many false positives or false negatives. Practically, it is hard to know the distribution of similar pairs in advance. As level 1 utilizes less kinds of clue conditions than levels 2 and 3 do, the WSIM values of retrieved pairs are easily affected by noise. Thus, naïve approach makes it hard to find a threshold of WSIM value. When the number of similar pair is various for each entity, the static approach does not work well. For example, assume that entity $a$ has ten similar entities and $b$ has one similar entity, it is difficult to fix the number of retrieved pairs, in advance. In this case, the number of retrieved pairs should be dynamically varied according to the condition of each entity. Finally, the delta-based and std-based method adopted the dynamic approach. However, when the distribution is varied for each entity or the size of clues is relatively small, the delta-based approach may not work well. It is because the delta values may not be distinct in these cases due to low WSIM values. The std-based approach is applicable to these cases, because the standard deviation is considered to get std values. Though the std-based approach shows good results in levels 2 and 3, naïve approach is also well applicable to level 3 without extra effort like std calculation. In summary, delta-based approach and std-based approach are similar in that they obtain clusters composed of similar pairs ‘dynamically’, however, delta-based approach has additional overhead to find dividing line for clone pages and there are some cases which are not covered by the delta-based approach. Thus, it can be said that std-based approach is more effective than delta-based approach and it is applicable to level 2.

In this paper, the four ways are related to levels, but when the levels are excluded, the four ways can be chosen according to whether the distributions of similar pairs are already known or not. The static approach is effective if the distributions are known in advance. Otherwise, WSIM and std are firstly computed by applying std-based approach. And then, other approaches are selectively applied according to the extent of used clues, and the outliers are identified based on std, when necessary.
5. EXPERIMENTAL RESULTS

Experimental data were generated to construct virtual web applications that have similar elements to real web applications. The data setup process consists of two steps: First, a base web application is created. Second, several test web applications are created based on the base WA. The detailed description of each step is as follows,

**Step 1: Creating a base web application**

A base WA, which is referred as ‘original’ in this paper, is a virtual web application which is a basis of several experimental WAs. To get the experimental data, we’ve implemented a simulation tool which can make the web application evolve according to the experimental configuration parameters such as total number of pages and distribution rates. To create a base WA, following steps are conducted: First, the number of pages and probability that each web element is present per each page have been given manually. Web elements include include, import, useBean, form, frame, link, redirect, read_session, write_session, hyperlink, js_function, javascript, js_source, and read_parameter. When the probability of include is 0.9, a page may have include relationship with 90% probability. The probabilities were empirically given, after observing several real WAs including B2B [14], GIMS [21], and PIM [22]. Second, web elements are inserted based on their probabilities for each page. If the probabilities are simply applied to each page, it is possible for each page to have normalized distribution of the web elements. Additional operations have been manually conducted based on a probabilistic way to avoid the normalized WA. Third, random text is inserted in each page for a variety of file sizes and content.

The fundamental statistics of the base data of the experiments are given in Table 1.

<table>
<thead>
<tr>
<th>Pages</th>
<th>INCLUDE</th>
<th>FRAME</th>
<th>REDIRECT</th>
<th>D_HREF</th>
<th>S_HREF</th>
<th>FORM</th>
<th>READ_P</th>
</tr>
</thead>
<tbody>
<tr>
<td>130</td>
<td>147</td>
<td>66</td>
<td>52</td>
<td>155</td>
<td>301</td>
<td>206</td>
<td>81</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Params</th>
<th>IMPORT</th>
<th>USE_BEAN</th>
<th>JS_FUNC</th>
<th>JS_SRC</th>
<th>READ_SS</th>
<th>WRITE_SS</th>
<th>LINK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1010</td>
<td>439</td>
<td>111</td>
<td>151</td>
<td>191</td>
<td>85</td>
<td>99</td>
<td>65</td>
</tr>
</tbody>
</table>

**Step 2: Creating test web applications using the base web application**

In this step, the experimental data are generated based on the original WA. To create test WAs, following steps are conducted: First, the pages are randomly selected to be cloned; for OUT experiments, fifteen pages are selected and for IN experiments, nine pages are selected. The selected pages are inserted to the WA. Second, each clone page is manually modified following each level type. For example, assume that a WA is required for L1/OUT. The test WA is manually generated by modifying the selected clone pages. In detail, the fifteen clone pages are changed to match the original pages using the clues of level 1. The original pages and the clone pages become completely similar from L1’s perspective. Third, the WAs generated at the prior step are mutated as to simulate real web applications, which is referred as DiffN in this paper. It is because the generated WAs are artificially generated according to strict criteria only for experimental purposes.
DiffN in each example (Fig. 17, Figs. 24-26) denotes that it is artificially mutated to obtain the size of $N$ differences in the clues. To mutate the WAs, the target pages are added/deleted/modified according to predefined rate of mutation.

In conclusion, the experimental data generated in this paper include an original WA (base WA), total six WAs for $L_1$-$L_3$ with IN/OUT, and the mutated WAs denoted by DiffN. The experimental data are referred to as ‘simulated data’, after this. Each similarity between two pages is fixed according to the purpose in the simulated data. For example, three types of simulated WAs are generated, when the purpose is to compare levels; WAs that consist of similar pages from the points of levels 1, 2, and 3, respectively. WAs to compare IN and OUT are generated in the same way. There are fifteen similar pairs in the case of the OUT direction. The IN direction has nine similar pairs, where six pairs are strongly similar and three pairs are weakly similar.

Sections 5.1-5.3 present a comparison between the four usages, comparison between three levels, and comparison between IN and OUT using the simulated data, respectively. However, the real WAs have mixed data compared to the virtual WAs that are composed of customized data suitable to the purpose of experiments. Section 5.4 suggests some usage guidelines about the proposed technique and presents outlines of this work. Finally, section 5.5 includes the results that are applied to open web applications, B2B [14], GIMS [21], and PIM [22]. Tool WANA [11, 13] has been extended to conduct those experiments.

5.1 Comparison between Four Usages

5.1.1 Global dynamic selection (in section 4.1)

Global Dynamic Selection is a naïve approach; the quality of its results is quite low compared to those of other approaches. In Fig. 16, the X-axis is the order of similar pairs by WSIM and the Y-axis is the proportion of the actual order to expected order. That is, the X-axis indicates the expected order predicted by WSIM. It predetermines whether or not two pairs are similar. The Y-axis implies how accurate WSIM measures the similarity between two entities; therefore, a y-value of one – the minimum value it can have – denotes 100% precision. In detail, each pair has predetermined ranking from first to ninth (IN) or from first to fifteenth (OUT); however, the predetermined order is not always exactly the same as the actual order. Although two orders are not the same, the difference between them is expected to be small if WSIM measures the similarity well. Of course,
the most ideal case is that the expected order is the same as the actual order; that is, 100% precision.

As shown in Fig. 16, level 3 shows 100% recall rate and 100% precision; however, pairs that are located only in higher ranks are retrieved in levels 1 and 2. In case of pairs in low ranks in levels 1 and 2, such as ranks 8 and 9 in \( IN \), the precision rate becomes low. That is, it is hard to retrieve those cases whose WSIM value is relatively low, and those cases can be threshold points to enhance precision, sacrificing recall.

In conclusion, this naïve approach is suitable for level 3, because effective results can be drawn with a comparatively low computation cost. It is hard to apply to levels 1 and 2, considering the precision and recall rates.

5.1.2 Clustered static selection (in section 4.2)

This section shows that WSIM for level 1 (\( L_1 \)), WSIM for \( L_2 \), and WSIM for \( L_3 \) are the most applicable to simulated data for level 1, data for level 2, and data for level 3, respectively. Fig. 17 illustrates those results for \( OUT \) direction. Section 5.2 presents the results of \( IN \) direction to show the difference between levels. The numbers in the table represent the rankings of the clone pairs for each level. In the tables in Fig. 17, the color of entry that has high ranking is dark. When the rank is relatively small, the entry color becomes light. Therefore, a table that has more dark orange entries indicates that similar pairs are effectively retrieved by WSIM. For example, the \( L_1-OUT \) with data originating in Fig. 17 (a), the ranks of all pairs are one or two, which means WSIM works well in the \( L_1-OUT \) in the simulated data.

In Fig. 17 (a), the results show that \( L_1 \) methods yield the best performance in all cases. In the \( L_1-Origin \) case, the recall rate reaches 100% at the suggested list size of two. As previously described, \( L_1 \) uses very loose clue-patterns to compare similarities, and it is very sensitive to noise. This means that the results of \( L_1 \) are highly likely to have false-positives. However, in the case of \( L_1 \) data type, the \( L_1 \) method achieves the best performance compared to that of other levels.

<table>
<thead>
<tr>
<th></th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
<th>#6</th>
<th>#7</th>
<th>#8</th>
<th>#9</th>
<th>#10</th>
<th>#11</th>
<th>#12</th>
<th>#13</th>
<th>#14</th>
<th>#15</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_1-OUT )</td>
<td>21</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>( L_2-OUT )</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>3</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>( L_3-OUT )</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(a) \( L_1-OUT \)

<table>
<thead>
<tr>
<th></th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
<th>#6</th>
<th>#7</th>
<th>#8</th>
<th>#9</th>
<th>#10</th>
<th>#11</th>
<th>#12</th>
<th>#13</th>
<th>#14</th>
<th>#15</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_1-OUT )</td>
<td>64</td>
<td>13</td>
<td>24</td>
<td>50</td>
<td>74</td>
<td>115</td>
<td>21</td>
<td>105</td>
<td>67</td>
<td>119</td>
<td>21</td>
<td>98</td>
<td>43</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>( L_2-OUT )</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( L_3-OUT )</td>
<td>6</td>
<td>12</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

(b) \( L_2-OUT \)

Fig. 17. Recall rates (partial) and WSIM rank table.
In Fig. 17 (b), especially, the recall rate reaches 100% from the beginning. However, it also shows decreasing trends, as the size of the difference increases. The recall rate of $L_2$ is 80% at the size of two in $Diff^4$. Finally, each page’s rank needs to be checked, because some clone pairs could lose their similarity due to the wide differences in the case of $Diff^8$. Therefore, the modified actual recall rate in $L_2$-$OUT(Diff^8)$ becomes $77.8\% = \frac{7}{9}$ at the suggested list size of one. $L_3$ achieves much lower performance than the other levels, because it could not find any clues satisfying the $L_3$ type pattern. In summary, the $L_2$ method achieves the best performance for $L_2$ data in all cases of modification.

In Fig. 17 (c), the $L_3$ method is the most robust among all approaches, because it does not have any ambiguities in its clues. In the case of $L_3$, only a small number of shared common clues could imply that the pairs are highly similar due to its scarcity. The result shows that the $L_3$ method is the most profitable means to detect the $L_3$ data clone pages. It also shows that the recall rate reaches 100% at the suggested list size of one in $Diff^6$, and 93.3% at the size of two in $Diff^{12}$. The original experimental data have been slightly modified to account for the higher differences between page pairs. The proportional rank of WSIM should be checked in $L_3$, because even a higher rank could actually be very low in terms of relative rank. In $L_3$, only the top ranks should be considered as clone pairs, because the results of the profit types are more precise and consistent than those of other levels.

### 5.1.3 Delta-WSIM based global dynamic selection (in section 4.3)

Figs. 19 and 20 visualize the $vdSet$ of simulated data, composed of the pairs <$WSIM$, delta-$WSIM$>, for WSIM-comparable pairs. Fig. 19 shows the results of $OUT$ direction, and Fig. 20 shows those of $IN$ direction. These figures show three different methods of levels, $L_1$, $L_2$ and $L_3$ in order. The horizontal and vertical axes represent the average values of delta-$WSIM$ and WSIM, respectively. The volumes of the circles represent the number of pairs of suitable candidates. Thus, a smaller circle has more scarcity value than others do. The small circles located at the upper-right corner, far from the largest
DETECTING SIMILAR WEB PAGES USING THREE LEVELS OF CLUES

<table>
<thead>
<tr>
<th>entity 1</th>
<th>entity 2</th>
<th>WSIM</th>
<th>delta-WSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>x1</td>
<td>9.8</td>
<td>2.1</td>
</tr>
<tr>
<td>a</td>
<td>x2</td>
<td>6.5</td>
<td>3.1</td>
</tr>
<tr>
<td>b</td>
<td>y</td>
<td>5.3</td>
<td>3.0</td>
</tr>
<tr>
<td>c</td>
<td>z</td>
<td>3.1</td>
<td>2.5</td>
</tr>
</tbody>
</table>

Fig. 18. Examples of candidate cutlines in selected pairs in vdSet.

Fig. 19. vdSet visualization for dynamic selection (OUT) [10].

Fig. 20. vdSet visualization for dynamic selection (IN).

circle have high probabilities of being candidates for the clone pairs.

The marked oval area in the figures includes several circles containing possible cutlines that can be the greatest lower bound in each cluster by the WSIM value. The list of pairs in the circles is generated, once the candidate circles are selected. Assume that Fig. 18 is part of the list of pairs. Each pair can be a cutline for each cluster. When two or more candidate cutlines exist in one cluster, such as \(<a, x_1>\) and \(<a, x_2>\) in Fig. 18, the user designates one of them as a cutline according to his/her preference. For example, \(<a, x_1>\) is selected to improve precision due to its high WSIM, and \(<a, x_2>\) may be selected to improve the call rate for the opposite reason of \(<a, x_1>\).

In Fig. 19, the candidates are easily detected in the case of L3. The clustered circles located at the upper-right corner exactly represent fifteen clone pairs that are supposed to be detected. In the case of L2, the count of distinguished circles located at the same posi-
tion is eleven, resulting in an accuracy of 100% and a recall rate of 73.3% (= 11/15). If ten more pairs near the first set are included, the recall rate and accuracy become 93.3% (= 14/15) and 66.7% (= 14/21), respectively. A better approach is to combine the static and dynamic selections using dynamic selection only to decide the suggested list size of each page. Thus, the recall rate could be raised to 100%, whilst sacrificing little accuracy. In the case of $L_1$, most of the major clone pairs were located at the corner; however, the candidates are not clearly separated from the other pairs. Six out of ten pairs were clones. The recall rate reached 80% (= 12/15) at 120 candidates. In $L_1$, the results tend to have many false-positives compared to other levels, because the pairs have high probabilities of having similar clues that are very loose and flexible compared to other levels. Thus, this approach is suitable for finding major clone pages in practice, but not for the case that requires a high recall rate. The approaches should be combined to obtain a high recall rate in $L_1$. First, the dynamic selection is applied, and the static selection is then used for other page pairs that may have been missed in previous selections.

Fig. 20 shows the results of IN direction using the delta-WSIM based method. In case of $L_1$, it is hard to achieve effective results with WSIM and delta-WSIM values. In the simulated data in $L_2$, six pairs are strongly similar and three pairs are weakly similar. When this method is applied, only the strongly similar six pairs are retrieved. This method did not find it, although one of the remaining three pairs is ranked with two in its cluster. The $L_3$ IN is a case that needs not only WSIM but also delta-WSIM, because the clustering of similar pairs is not obvious, compared to that of the OUT case.

User intervention is often needed to apply this delta-WSIM based method: A user should manually mark the oval area, including the possible cutline pairs. When there are more than two candidate cutline pairs, a user should select one suitable pair. The graphs for vdSet visualization do not directly reflect similar pairs, but they are used to determine the possible cutline pairs. Additional tasks are required to select similar pairs. These are the vulnerable points of this delta-based approach; therefore, a more effective approach, std-based method, has been suggested.

5.1.4 std-based global dynamic selection (in section 4.4)

Fig. 21 shows that std gives more effective results than WSIM. The meaning of the graph is the same as that of Fig. 16. The X-axis is the identifier of similar pairs and the Y-axis is the ratio of estimated rank to actual rank. As the y-value approaches zero, the similarity measure is well estimated.

![Fig. 21. L1 and L2 OUT using std.](image)
When only $std$ is used in level 1, all 15 similar pairs are retrieved in 150 candidate pairs and 14 similar pairs in 70 pairs. The pairs are sorted by descending order, according to $std$ values. Seven pairs are actual similar pairs in the upper ten pairs, implying high precision. The results of level 2 also represent that $std$ is more effective than WSIM. The average improvement rate of $L1$ is 450% and that of $L2$ is 155%, which are the comparison results of ranks between $std$ and WSIM. One meaningful case is a pair whose ranking is 120 by WSIM value. The rank is improved to 17 when $std$ is applied. This case signifies that the number of necessary pairs is reduced from 120 to 17 to attain 100% recall rate. In the case of level 3, both WSIM and $std$ result in 100% recall and 100% precision, so their graphs are omitted.

It is hard to obtain good results using only $std$ values, excluding WSIM in real applications, because the similar pairs are not evenly distributed in real data, unlike in the simulated data. When investigating real applications, the WSIM and $std$ of actual similar pairs are not always found to be high, together. That is, they have not only high WSIM and high $std$, but also high WSIM and low $std$, or low WSIM and high $std$. This is due to each cluster having various distribution patterns. To be sure, a pair with high WSIM and high $std$ is a strong candidate of similar pair, and in real data, either a pair with <high WSIM, low $std$> or a pair with <low WSIM, high $std$> can be also a candidate of a similar pair.

The $std$-based approach and delta-WSIM based approach look similar in that two values are used in the approaches, and the distribution of each cluster is considered. The Delta-WSIM based approach is applicable when the WSIM is suddenly reduced once or more in a cluster that is composed of sorted pairs in WSIM descending order. Visualization is necessary for manual cutline checking to determine the cutline in a cluster; however, additional tasks should be conducted after finding the cutline. The additional task is to retrieve the pairs whose WSIM exceed the selected cutline pair’s WSIM value. However, the $std$-based approach does not directly require those additional tasks, because the outliers are similar pairs. The $std$-based approach also achieves better results than those by the delta-WSIM based approach. This is shown below.

Figs. 22 and 23 intuitively compare the $std$-based approach and delta-WSIM based approach on the simulated data. The X-axis is the value of WSIM, and the Y-axis is $std$ value (a) or delta-WSIM (b). The dark (highlighted) points indicate the actual similar pairs. In the delta-WSIM graph, each point corresponds to one pair, unlike Figs. 19 and 20, to directly compare to the $std$-based approach.

![Fig. 22. <WSIM, $std$> vs. <WSIM, delta-WSIM> (L2-OUT).](image-url)
In the *std*-based approach, which is presented at (a) in the two figures, the group of similar pairs explicitly appears. Compared to the latter, the actual similar pairs are relatively obscure in the delta-WSIM based approach. The right graph shows the highlighted points mixed with light points and that the boundary is not clear. Especially, it is not easy to find the actual similar pairs that are located on the left hand side, in the delta-WSIM based approach. The simulated data include fifteen similar pairs for *OUT* data and nine similar pairs for *IN* data. In Fig. 22 (b), the pair < WSIM = 1.501, *d*-WSIM = 0.345> can be manually selected as a lower bound, which means the points whose values are larger than the selected pair can be retrieved as similar pairs. Thus, the number of pairs whose WSIM is larger than 1.501 is 119, and the number of pairs whose *d*-WSIM value is larger than 0.345 is 35. The precision is too low in that case. In the *std*-based approach, a point whose *std* is 5.68 is manually selected as a lower bound. Consequently, 13 out of 15 pairs are retrieved, implying 87% precision. When WSIM is also considered, the recall rate can become 100%. For *IN* direction, it can be shown that the *std*-based approach is clearer than the delta-WSIM based approach.

5.2 Comparison Among Levels

Section 5.1.2 presents the results of *OUT* direction to show the applicability of the clustered static selection method. This section shows the results of *IN* direction experimented with similar ways as the *OUT* direction. The results are similar to those of the *OUT* direction, described in section 5.1.2: WSIM *L*1, WSIM *L*2, and WSIM *L*3 achieve the best results form the simulated data for level 1 (Fig. 24), level 2 (Fig. 25), and level 3 (Fig. 26), respectively. Nine similar pairs are generated for this experiment. The similarity degree of #1-#3 is low, that of #4-#6 is medium, and that of #7-#9 is high.

![Fig. 23. <WSIM, std> vs. <WSIM, delta-WSIM> (*L2*-IN).](image)

![Fig. 24. Recall rates and WSIM rank table for *L1*-IN data.](image)
Fig. 24. (Cont’d) Recall rates and WSIM rank table for $L_1$-IN data.

Fig. 25. Recall rates and WSIM rank table for $L_2$-IN data.

Fig. 26. Recall rates and WSIM rank table for $L_3$-IN data.

Fig. 24 illustrates that WSIM$_{L1}$ $L_1$ is the most suitable for $L_1$-IN data. For example, only one pair is retrieved by WSIM$_{L3}$ $L_3$. As the Diff degree increases, the WSIM ranking tends to sharply decrease in most cases, except the #3 cases. The #3 pair by WSIM$_{IN}$
L1 shows strange results; when Diff1 has been applied, the WSIM ranking of #3 fell to the 22nd place from the 15th. After Diff2 was applied, the ranking moved up to third place. We investigated this case, and the two entities in #3 were found to be a weakly similar pair. That is, two entities share only one similarity clue that is a ‘submit’ relation with seven parameters. Therefore, the WSIM ranking is accidentally increased.

L2-IN shows the best results as expected in Fig. 25 because WSIM\textsubscript{IN} L2 has been applied to the simulated data. The ranking moved up to first place in #2, which is an uncommon result like that in the case of #3 in L1-IN.

As for the results of OUT direction in level 3, WSIM\textsubscript{IN} L3 shows the best performance on the L3-IN data. As WSIM of level 3 utilizes strong similarity clues, Diff\# hardly affects the WSIM ranking.

5.3 IN and OUT Comparison

The meaning and usefulness of directions have been explained in the previous section. This section presents the comparison between IN and OUT. Figs. 27 and 28 show the recall rate for each level using simulated data. The static selection technique is applied, because it gives relatively clear results in the experiment compared to those of other methods. The two figures indicate that in the case of IN data, the results by WSIM\textsubscript{IN} are better than those of WSIM\textsubscript{OUT}, and vice versa.

5.4 Summarization and Guidelines

The experiments in the previous sections were conducted on simulated data that were set up under specific conditions. The results show that six types of WSIM, which are classified by two directions and three levels, suited their purpose. The results also include the four usages described in section 4. The three levels require a different approach to find similar pairs, because the clues used in each level have different similarity
degrees; that is, the clues in level 3 are strong and the clues in level 1 are weak. The distribution of each cluster is also an important factor affecting the quality of the results. Although the \textit{std}-based approach shows relatively improved results, the computational cost is more than that of other approaches.

The naïve approach (4.1), which uses only WSIM value, suffices to show the difference between levels and the necessity of finding a suitable approach for each level. However, the precision and recall rate are too low to be used practically. The clustered static selection method (4.2) improves the recall rates, but the static size to be retrieved may result in many false positives and false negatives: When the number of actual similar pairs is more than the fixed size, false negatives are generated. Conversely, false positives are generated. Accordingly, the clustered static method is suitable to show the difference between levels or directions on simulated data, but it is not suitable for actual web applications, where the distribution is unknown beforehand. The delta-WSIM based method (4.3) has been suggested to complement the static method by using the dynamic size to be retrieved. The cutline is determined in the method with WSIM and delta-WSIM. In each cluster, a pair that has high WSIM and high delta-WSIM can be a candidate cutline. As the cutline is dynamically determined, considering the distribution of each cluster, false positives in a cluster that has few similar pairs and false negatives in a cluster that has many similar pairs are reduced. This method is complicated for use in real applications, because the users must manually select the threshold, although the distribution is visualized, as shown in Fig. 19 or Fig. 20. Otherwise, some metrics may be needed; however, it is hard to define and validate the metrics. The delta-WSIM is also an absolute unit. It may not reflect the relative outlier in some cases. Therefore, the \textit{std}-based approach (4.4) has been suggested, which is more intuitive and incorporates the relative distribution in each cluster better than does the delta-WSIM based method. This approach does not require several steps, as does the delta-WSIM based method. It is only to find the pairs that have high WSIM or high \textit{std} values. The results of experiments conducted on simulated data show that this method is superior to other methods, although additional computation for \textit{std} is needed. Only \textit{std} is sufficient to find similar pairs on simulated data. However, it is better to use both WSIM and \textit{std} on real data, because several clues suitable to different levels are mixed in real data.

<table>
<thead>
<tr>
<th>section</th>
<th>static/dynamic</th>
<th>usage values</th>
<th>selection target</th>
<th>best-fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>dynamic</td>
<td>WSIM</td>
<td>global</td>
<td>\textit{L3}</td>
</tr>
<tr>
<td>4.2</td>
<td>static</td>
<td>WSIM</td>
<td>cluster</td>
<td>\textit{L1}</td>
</tr>
<tr>
<td>4.3</td>
<td>dynamic</td>
<td>WSIM, delta-WSIM</td>
<td>cluster</td>
<td>\textit{L2}</td>
</tr>
<tr>
<td>4.4</td>
<td>dynamic</td>
<td>WSIM, \textit{std}</td>
<td>global</td>
<td>\textit{L2}</td>
</tr>
</tbody>
</table>

Table 2 summarizes some guidelines for the differentiation usage. These guidelines are settled by considering the tradeoffs between cost and the quality of results.

In the case of level 3, the naïve approach achieves sufficiently good results because the similarity degree of clues in level 3 is rather strong. The clues in level 1 are weak compared to those of other levels, so the static clustered approach is suitable for improving the recall rate in level 1. The delta-WSIM based method and \textit{std}-based method are
appropriate for level 2, but the \textit{std}-based method is more efficient than delta-WSIM based method in most cases.

5.5 Open Source Results

WSIM has been applied to open source: WA B2B [14], PIM [22], GIMS [21]. B2B is a web application for shopping malls in JSP. Partial pages in B2B have parallel structures. PIM is a web application for personal information management in JSP. GIMS is implemented with JSP, and is used to manage global internship members. Table 3 shows their statistics.

<table>
<thead>
<tr>
<th>Source</th>
<th>Pages</th>
<th>Relations</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2B</td>
<td>141</td>
<td>816</td>
<td>503</td>
</tr>
<tr>
<td>PIM</td>
<td>109</td>
<td>1939</td>
<td>853</td>
</tr>
<tr>
<td>GIMS</td>
<td>63</td>
<td>156</td>
<td>52</td>
</tr>
</tbody>
</table>

Table 3. Open sources statistics.

Table 4 presents the results of the naïve approach (global dynamic selection) for level 3 – \textit{L3-OUT}. The precision varies from 48% to 79% for the upper 5% and 10% pairs. Two developers judged the similarity for B2B and PIM. In the case of GIMS, we referred to the developer’s document, where the groups included similar pages that were summarized by the developer. The results show that the precision of 5% is higher than that of 10%.

Next, WSIM for level 2 \textit{IN} and \textit{OUT} directions have been applied to the three directions using the \textit{std}-based approach and delta-WSIM based approach. The comparative results between the two approaches show that the \textit{std}-based method is more suitable for level 2.

Fig. 29 illustrates two graphs for B2B: The upper graph is the result of the \textit{std}-based approach and the lower graph is the result of delta-WSIM based approach. Most points in the lower graph are clustered on zero for the X-axis, because the purpose of the \textit{std}-based approach is to find the dynamic cutline using the outlier points. This is due to the \textit{std}-based method finding similar pairs by directly using \textit{std} and WSIM values, but the delta-WSIM based method finds only the cutline pairs using the outlier points.

The highlighted points indicate actually similar pairs, although they have low WSIM values. As shown in Fig. 30, similar pairs definitely appear in the \textit{std}-based ap-
DETECTING SIMILAR WEB PAGES USING THREE LEVELS OF CLUES

1815

Fig. 29. delta-WSIM vs. \( \text{std} \) (B2B, L2-OUT) – original.

Fig. 30. delta-WSIM vs. \( \text{std} \) (B2B, L2-OUT) – highlighted.

proach; however, it is hard to detect them using the delta-based approach. The fifteen points were composed of many ‘list.jsp’ files and ‘form.htm’ files in the parallel structure.

The original figures are omitted and highlighted graphs are shown next due to space limitations. Fig. 31 shows the difference between the two approaches in terms of the process of finding similar pairs. The highlighted point in the upper graph, which incorporates actual similar points, corresponds to the three points in the lower graph. The top point of the three points in the lower graph becomes the cutline. Users should check the pairs above the cutline in each cluster after the cutline is determined. In the \( \text{std} \)-based approach, users only check the outlier points, which can be similar pairs.

Fig. 32 shows the results of \( L2-IN \) direction for GIMS. For the \( OUT \) directions, the results of the two approaches were not distinct, so only the \( IN \) direction result is shown in this paper. The highlighted points \( p \) and \( q \) in Fig. 32 denote actual similar pairs. \( p_{\text{std}} \) and \( p_{\text{dta}} \) are identical, \( q_{\text{std}} \) and \( q_{\text{dta}} \) are also the same. \( p \) and \( q \) show two cases, each other: \( p \) is Low WSIM and high \( \text{std} \), and \( q \) is high WSIM and low \( \text{std} \). It is hard to detect both of them relying only on the delta-WSIM based approach or only the naïve approach. \( p_{\text{std}} \) by the \( \text{std} \)-based approach is outstanding, but \( p_{\text{dta}} \) by the \( \text{std} \)-based approach is blended with
other dissimilar pairs. The real pages in $p$ are BoardNotice.jsp and BoardNoticeAdmin.jsp. One main drawback of the $std$-based approach is that it is too susceptible to the distribution of each cluster. Thus, not only $std$ but also WSIM is used to apply this $std$-based approach to real applications. The $std$ of $q$ is zero, but $q$ is actually a similar pair, which is $<BoardNoticeConfirm.inc.jsp, BoardNoticeConfirmAdmin.inc.jsp>$.

Figs. 33 and 34 are individually the results of $L_2$-OUT and $L_2$-IN for PIM. As shown in Fig. 33, the trend of WSIM value is biased to a negative number, which implies that the rate of possible similar pairs is relatively small. Thus, the focus should be on the pairs whose WSIM values or $std$ values are high. Five pairs highlighted in Fig. 33, which are actual similar pairs, can be detected in the $std$-based approach. Those pairs are barely checked in the delta-WSIM based method.

In the case of the $IN$ direction of PIM of the $std$-approach, header and footer pages are included in six pairs, and two $login.jsp$ pages in parallel structures are included in two pairs, among the upper eight pairs highlighted in Fig. 34.
6. RELATED WORKS

Several studies have been made on the similarity of web pages. Dhyani et al. classified web page similarity into three types: content, link, and usage [18]. The context-based approach uses subsequent matching or word occurrence statistics by analyzing texts. The link-based approach is based on citation analysis, and usage logs are used in the usage-based approach. The common purpose of the latter approaches is to measure the relatedness degree of two or more web pages. That is, the target is the semantic relatedness between individual pages, not web applications.

The link-based approaches that are similar to our methodology have been suggested [1-3, 7]. SimRank [1], frequently referenced, is a similarity metric that can be applied to web pages and general objects. In [1], two objects, which are related to similar objects, are regarded as similar. The similarity value is the average similarity of every in-link pair for the two nodes. However, Lin et al. pointed out that SimRank has two drawbacks. When a page has no in-link, the similarity becomes zero. Two pages related to each other...
also have zero similarity. In fact, those are not correct [2]. PageSim has been defined to improve the drawbacks of SimRank [2]. PageSim is based on page rank propagation. First, the relative importance of each page is determined using PageRank [23]. A PageSim of page \( a \) and \( b \) is defined as the total sum of minimum PageRank values that are propagated to \( a \) and \( b \) by other nodes. [20] extends the previous similarity studies, such as, SimRank [1], PageSim [2], Co-citation and Bibliographic coupling [24] by incorporating two directions and a multi-hop neighborhood structure. Hou and Zhang defined page similarity based on the transivity of hyperlinks and page importance [3]. Those similarity metrics are commonly based on the link structure, and their purpose is to find semantically similar web pages to the ‘query page’. This differs from WSIM’s purpose, in that the target of WSIM is web applications and the purpose of WSIM is to find structurally similar entities in web applications to improve maintainability.

Page similarity metrics, which intend to support maintenance activities in web applications, have been defined [4-6]. They are connected with clustering [6], which is an important activity to improve maintainability and understandability. Lucca et al. suggested two techniques to identify duplicate web pages [4]. In the first, the similarity is obtained by computing levenshtein distance [25] between two pages, where HTML tags are transformed into string and edit distances between the two strings are computed. The computational cost is high in this method. The second technique is based on the frequency-based approach, where the frequency of each tag is counted for the two pages and the Euclidean distance of the two pages is computed. The cost of the second method is lower than that of the first method; however, the edit distance based method produces more false positives than the frequency-based method. Lucca et al. have measured page similarity using the edit distance between two pages after transforming a page into a tree [4]. Lucia et al. defined the page similarity measure by computing levenshtein distance between two pages, where two pages had been transformed into two abstract syntax trees (ASTs), and the ASTs are then changed to strings by depth-first traversal [6]. These approaches commonly measure similarity by comparing two abstract representations, such as, trees and strings, which are the results of analyzing and transforming the pages. The relationship between pages and parameters has been used in [7] to obtain pages similarity. A weighted graph is generated according to the relations, and number of parameters and a weighted vector for each page are then produced. The cosine value of two weighted vectors becomes the similarity between the two pages. This approach in [7] is similar to the proposed approach in this paper, but Lee et al.’s approach differs from WSIM. First, only the number of parameters, not the parameter itself, is used. Second, similar pages in parallel structure cannot be detected in [7], because only direct relations are considered in [7]. Third, only one relation between two pages is assumed in [7]; however, this is an uncommon case. Directions or levels are not considered in [7]. Moreover, when the number of parameters between two pages is zero, the similarity of the two pages becomes zero, which is obviously incorrect.

The final purpose of WSIM is to support the maintenance activity during web application development, which is similar to the above studies. Therefore, there are several distinct points in WSIM. In the edit distance based approach, it is important to transform pages into abstract representation, and most of the studies use HTML tags. However, few modern web applications are constructed based on simple HTML: Many heterogeneous languages, such as HTML, CSS, script codes, dynamic pages like jsp, php, and asp, are
used to produce web applications [26]. Due to their short lifecycle, it is hard to produce web applications under a systematic approach by applying software engineering principles [27]. Accordingly, there are limitations to analyze, parse, and convert into abstract representations. When the HTML tags are mainly used in the approach in [4], the measured similarity focused on the UI level. Practically, web developers are often concerned about errors of linking between pages or interchanging parameters, not UI [12]. In existing studies, it is highly possible that two pages, which are similar in linking structure, like the parallel structure, and their UIs are different, have low similarity. As WSIM uses various relation types and parameters, it enables the retrieval of similar pages in linking structure, regardless of the UI.

7. CONCLUSION

This paper proposed methods to estimate similarity in web applications that are based on several clues, including relation types, parameters, source, and target entities. Directions, in and out, are also considered, unlike most of existing studies that use only one direction. Each direction implies a distinct meaning: OUT is used to detect a case in which two pages have different UIs, but they are constructed on similar linking structure. That is, a page has been constructed based on another page and only the inner content has been modified. The IN direction is applied to find similar pages in their usage, for example, header and footer pages that are frequently included to many pages in a web application. Six classes of WSIM exist according to three levels and two directions. Three levels are about the degree of the number of clues used. For example, in level 3, all clues are used. The six WSIMs uncover the similarity of pages in a web application according to their purposes. In this work, notions and metrics are defined formally for accurate application. Four ways to use WSIM are suggested, and we relate each level to one of the ways considering performance and the quality of results. Experimental results show WSIM is effective on the simulated data and real data.

The contribution of this work is that WSIM detects several similar pairs that can be missed by existing work, because it has six views of similarity classified by directions and clues. WSIM focuses on similarity of linking structure, not UI, which is helpful to web developers, who are mostly concerned with the page relations or interchanging parameters.

We expect that prediction for clone pairs could be enhanced by applying machine learning techniques, such as neural networks. By learning partial sets of clone pairs, other clones can be dynamically predicted. We are also researching a method to use the distribution data of pages to improve the quality of our results. In this paper, parameters are utilized as meaningful factors. In detail, the similarity between two pages increases when the pages have identical parameters. However, identifiers are often renamed, even though a page is constructed based on a source page. That is, when page A has ‘car’ as a parameter and page B has ‘vehicle’ as a parameter, it may be reasonable to consider them as ‘similar’. Therefore, in future work, it will be necessary to extend the notion of a parameter with its synonym. We will also extend a tool WANA [11, 13] that has been implemented to show WSIM and WCOX that are web complexity measures, to provide useful information to web developers in their development and maintenance activities.
REFERENCES


Woosung Jung (鄭旼盛) received his B.S. and Ph.D. degree in Computer Science and Engineering from Seoul National University, Korea, in 2003 and 2011, respectively. He was a researcher in SK UBCare from 1998 to 2002. He is currently a senior research engineer at Software Capability Development Center in LG Electronics. His research interests include software evolution, software architecture, adaptive software system and mining software repositories.

Eunjoo Lee (李銀珠) received her B.S., M.S., and Ph.D. degrees in Computer Science from Seoul National University, Korea in 1997, 1999, and 2005, respectively. She was a research staff member at Samsung Advanced Institute of Technology from Nov. 2005 to Feb. 2006. Currently, she is an assistant professor at the School of Computer Science and Engineering at Kyungpook National University. Her current interests include software re-engineering, software metrics, web engineering, and mining software repositories.
Chisu Wu (禹治水) received his B.E. degree in Applied Mathematics from Seoul National University, Korea in 1972 and his M.S. and Ph.D. degrees in Computer Science from Seoul National University in 1977 and 1982, respectively. He served as a researcher at the Loughborough University, UK in 1978. From 1975 to 1982, he was an Associate Professor of Computer Science at Ulsan University, Korea. Currently, he is a Professor of Computer Science and Engineering at Seoul National University, Korea. His current research interests include software engineering and programming languages.