Comparing Clustering Algorithms for the Identification of Similar Pages in Web Applications

Andrea De Lucia[1], Michele Risi[1], Giuseppe Scanniello[2], and Genoveffa Tortora[1]

[1] Dipartimento di Matematica e Informatica, Università di Salerno, Via Ponte Don Melillo, 84084, Fisciano (SA), Italy
[2] Dipartimento di Matematica e Informatica, Università della Basilicata, Viale Dell'Ateneo, Macchia Romana, 85100, Potenza, Italy
{adelucia,mrisi}@unisa.it, giuseppe.scanniello@unibas.it, tortora@unisa.it

Abstract. In this paper, we analyze some widely employed clustering algorithms to identify duplicated or cloned pages in web applications. Indeed, we consider an agglomerative hierarchical clustering algorithm, a divisive clustering algorithm, k-means partitional clustering algorithm, and a partitional competitive clustering algorithm, namely Winner Takes All (WTA). All the clustering algorithms take as input a matrix of the distances between the structures of the web pages. The distance of two pages is computed applying the Levenshtein edit distance to the strings that encode the sequences of HTML tags of the web pages.

Keywords: Clone detection, clustering algorithms, reverse engineering.

1 Introduction

Code cloning in web applications is one of the factors that make their maintenance more difficult and time consuming [1][11]. A code clone is a code portion in source files that is identical or similar to another. Developers introduce clones for various reasons such as lack of a good design, fuzzy requirements, undisciplined maintenance and evolution, and reusing code by copy-and-paste.

Recently, researchers have extensively studied clone detection for static web pages, written in HTML, or dynamic ones [3][11]. Clone detection approaches for web applications are usually based on similarity measures: two pages will be considered clones if they are characterized by the same values of the defined measure [4][5][11][12]. For example, Di Lucca *et al.* [4] encode the sequences of tags of HTML and ASP pages into strings and identify pairs of cloned pages at structural level by computing the Levenshtein string edit distance [9] between these strings. A pair of web pages is considered a clone if their Levenshtein edit distance is zero. Ricca and Tonella in [11] enhance the approach based on the Levenshtein edit distance proposed in [4]. In particular, they adopt a hierarchical clustering algorithm to identify clusters of duplicated or similar pages to be generalized into a dynamic page. Similarly, in [12] the authors propose a semiautomatic approach to identify and align static HTML pages whose structure is the same and whose content is in different

L. Baresi, P. Fraternali, and G.-J. Houben (Eds.): ICWE 2007, LNCS 4607, pp. 415–420, 2007.

languages. Pages with similar structures are identified by adopting an agglomerative hierarchical clustering algorithm. An approach based on a competitive clustering algorithm to identify similar pages at structural and content level is proposed in [2]. To group pages at structural level, their structures are encoded into strings and then the Levenshtein algorithm is used to achieve the distances between pairs of pages. De Lucia *et al.* [3] also use Levenshtein distance to compute the similarity of two pages at structure, content, and scripting code level. Clones are characterized by a similarity threshold that ranges from 0%, for disjoint code, up to 100%, for identical pages.

In this paper, we compare the results of applying some clustering algorithms belonging to the categories hierarchical and partitional [6] in the identification of cloned page at structural level (i.e., pages with similar sequences of HTML tags). Concerning the hierarchical methods we consider the agglomerative [8] and divisive clustering algorithms [7], while within the partitional category we selected a variant of k-means [10] algorithm and the Winner Takes All (WTA) [6] algorithm. Web pages can be both static and dynamic and can be implemented using any kind of server side scripting code. Similarly to the approaches proposed in [2][4][11], we consider the Levenshtein string edit distance [9] as basic metric to identify similarity between pairs of pages. We have compared the selected clustering algorithms on a case study composed of three web applications developed using JSP technology.

The remainder of the paper is organized as follows. In Section 2 we present the process to identify page similarity at the structural level, while the results obtained by applying the considered clustering algorithms are presented in Section 3. Final remarks conclude the paper.

2 Identifying Cloned Pages at Structural Level

The process we adopted is composed of the following sequential phases: *Preprocessing*, *Computing Distance Matrix*, and *Grouping Similar Pages*. The *Preprocessing* phase aims at turning the page structures implemented by specific sequences of HTML tags into a more suitable string representation. This representation is produced by means of a depth-first traversal of the abstract syntax tree of both static and dynamic page structures. Each node of the syntax tree of a page is decorated with a HTML tag and with a set of attributes, such as text attribute, target source code, image attribute, etc. The string representations of the structure of a web page is obtained encoding the HTML tags into symbols of an alphabet before being concatenated into the string corresponding to its structure.

The *Computing Distance Matrix* phase adopts the Levenshtein edit string distance [9] to obtain similarity measures between pairs of pages. The Levenshtein edit distance is defined as the minimum cost to align two strings. The Levenshtein distance between pairs of pages is then used to build the *Distance Matrix* considering all the possible pairs of pages of a given web application. The Distance Matrix is then provided as input to the phase *Grouping Similar Pages*, which can be instantiated with different clustering algorithms. In particular, we considered an agglomerative [8], a divisive [7], k-means [10], and the WTA (Winner Takes All) [6] clustering algorithms. The agglomerative hierarchical clustering algorithm [8] begins with each page in a distinct cluster, and then hierarchically merges clusters together. This results

in a dendrogram, which represents the nested groups of pages and similarity levels at which groups change. The dendrogram can be cut at different levels to obtain different clusters (i.e., the tuning values of our approach). In the divisive hierarchical clustering algorithm [7], clusters are divided until each page is placed in a distinct cluster. The choice of the divisive step represents the tuning values of our approach. The k-means algorithm [10] is used to classify pages of a given data set through a priori fixed number of disjoint clusters (the tuning value of our approach). The main idea is to define a centroid for each cluster of pages to identify. The algorithm iteratively refines the initial centroids, minimizing the average distance of pages to their closest centroids. Finally, we also considered WTA [6], an Artificial Neural Network (ANN) clustering algorithm. The used ANN is essentially based on a two dimensional grid of neurons, which weights are continuously adapted to the vectors of the distances between the pages of a web application. The number of neurons is provided as input (i.e., the tuning value of our approach) and represents the number of clusters that the algorithm should identify (i.e., the number of expected clusters). The neuron with weight vector most similar to the input vector is called the winner. The weights of the winner and its closer neurons are then adjusted towards the input vectors. The training process is concluded either the neurons do not change their position or a termination threshold for the iteration is reached. Clusters are identified associating the pages to the closer neuron in the ANN.

3 The Case Study

The clustering algorithms have been compared on web applications implemented using JSP technology. In particular, we considered the web sites of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE 02) and the 1st International School of Software Engineering (ISSE 03), and the SRA (Student Route Analysis) web application. Some descriptive statistics of these web applications are reported in Table 1. The second row contains the number of files composing the application, while the second and third rows contain the number of static and dynamic pages, respectively.

Table 1. Descriptive statistics of the analyzed web applications

	SEKE 02	*ISSE 03*	*SRA*
Number of Component Files	1268 files in 63 folders	66 files in 14 folders	380 files in 45 folders
Number of HTML Pages	49	12	22
Number of JSP Pages	108	7	45

3.1 Assessing the Results

To compare the results achieved by applying the different clustering algorithms we adopted the precision and recall metrics. In our case the recall is defined as the ratio between the number of actual pairs of similar pages identified by the tool over the total number of actual pairs of similar pages in the web application. Differently, the precision is the number of actual pairs of similar pages identified by the tool over the total number of identified pairs. The precision and recall values are computed

turning both actual clusters and the clusters automatically identified by the tool into pairs of similar pages. In particular, the clones manually identified are reported in a gold matrix, while those identified by the tool are reported in result matrix.

The evaluation of precision and recall is not straightforward. Thus to better assess the achieved results we also considered the trade-off between the values of precision and recall. In particular, we consider the F-measure that is defined as the harmonic mean of the precision and recall values. The trade-off configuration corresponds to the tuning value that allows the identification of the larger F-measure value.

Table 2.a and 2.b report the trade-off configurations achieved on the SEKE 02 web application. In particular, the first columns contain the adopted clustering algorithms, while the trade-off configurations are reported in the second columns. Concerning the WTA algorithm these tables also report the number of expected clusters (i.e., the first number) and the number of clusters that the algorithm actually identified (i.e., the second number). The precision and recall values are shown in the third and fourth columns, respectively. Finally, the F-measures of the presented configurations are contained in the last columns. The precision and recall values corresponding to the trade-off configurations of the static pages of the SEKE 02 web application (see Table 2.a) revealed that the k-means clustering algorithm produces better results. The trade-off configurations are 38 for the divisive clustering algorithm and the hierarchical clustering algorithms, while is 31 for the WTA clustering algorithm. Note that the WTA clustering algorithm produced worse results (i.e. the precision and recall values were 0.523 and 0.578, respectively) than the other clustering algorithms. Table 2.b shows that the better values of the precision and recall were obtained by using the divisive hierarchical clustering algorithm (0.695 is the F-measure value). Indeed, all the clustering algorithms except the WTA clustering algorithm produced comparable results. We can also observe that the number of clusters identified by the WTA algorithm is lower than the number of clusters identified by the other algorithms. The results achieved by employing the trade-off configurations on the ISSE03 web application are reported in Table 3.a and 3.b, respectively. The results obtained by applying the considered clustering algorithms are very similar. Indeed, on the static pages (see Table 3.a) the WTA clustering algorithm produced more relevant results in terms of recall value (i.e., 0.888). The results obtained by using the trade-off configurations on the agglomerative clustering algorithm and the k-means algorithm were the same in terms of precision and recall values.

On the dynamic pages of the ISSE 03 web application better results were achieved when the WTA and k-means clustering algorithms were adopted (see Table 3.b). For both the clustering algorithms the precision and recall values are 1 and 0.666, respectively. Regarding the WTA clustering algorithm, the difference between expected and identified clusters is 1. Let us also note that the agglomerative and divisive clustering algorithms produced the same results.

Finally, Table 4.a shows the results achieved by using the trade-off configurations on the static pages of the SRA web application. This table shows that the divisive clustering algorithm produced better results (i.e. the precision and recall values were 0.959 and 0.979, respectively). Similar results were achieved by employing the agglomerative clustering algorithm and WTA (the precision and recall values were 0.921 and 0.974, respectively). Differently, for the k-means the precision value was 1, while the recall value was 0.812. Regarding the WTA clustering algorithm the

Table 2. Results obtained by using the trade-off configurations on SEKE 02

	Clusters	Prec.	Recall	F-meas.
Agglom.	38	0.565	0.684	0.619
Divisive	38	0.565	0.684	0.619
k-means	40	0.923	0.631	0.750
WTA	31/31	0.523	0.578	0.549

(a)

	Clusters	Prec	Recall	F-meas.
Agglom.	90	0.877	0.574	0.694
Divisive	87	0.820	0.603	0.695
k-means	86	0.781	0.574	0.662
WTA	57/57	0.552	0.660	0.601

(b)

Table 3. Results obtained by using the trade-off configurations on ISSE 03

	Clusters	Prec.	Recall	F-meas.
Agglom.	6	0.5	0.666	0.571
Divisive	5	0.437	0.777	0.559
k-means	6	0.5	0.666	0.571
WTA	3/3	0.363	0.888	0.515

(a)

	Clusters	Prec.	Recall	F-meas.
Agglom.	4	0.5	0.666	0.571
Divisive	4	0.5	0.666	0.571
k-means	5	1	0.666	0.800
WTA	6/5	1	0.666	0.800

(b)

Table 4. Results obtained by using the trade-off configurations on SRA

	Clusters	Prec.	Recall	F-meas.
Agglom.	6	0.921	0.979	0.949
Divisive	7	0.959	0.979	0.969
k-means	9	1	0.812	0.896
WTA	6/6	0.921	0.979	0.949

(a)

	Clusters	Prec.	Recall	F-meas.
Agglom.	33	0.437	0.466	0.451
Divisive	33	0.437	0.466	0.451
k-means	31	0.208	0.333	0.256
WTA	27/26	0.206	0.4	0.272

(b)

number of expected and identified clusters coincides. The results of the trade-off configurations on the dynamic pages are shown in Table 4.b. Let us note that due to the low number of dynamic pages similar at structural level the clustering algorithms generally produced bad results.

3.2 Remarks

To identify the trade-off value the software engineer has to try all the possible configurations and then the clusters identified at each iteration have to be manually assessed. Hence, methods to automatically filter out surely bad tuning configurations and to get more quickly the trade-off configuration should be devised in the identification of similar pages. Indeed, the web application used as case study provided some directions to better support the software engineer. In particular, we observed that in most cases the break-even configuration (i.e., the larger number of expected clusters such that the WTA identifies non empty clusters) is very similar to the trade-off configuration. In particular, we observed that on the static pages of SEKE 02 the break-even and the trade-off configurations are nearly identical. On the other hand, the break-even and trade-off configurations coincide in case WTA is applied on the structure of the static pages of SRA. We also note that a larger difference between break-even and trade-off configurations is obtained when the web applications are composed of few pages (for example, ISSE 03). A larger difference between the break-even and trade-off configurations was also obtained in case the web application has a low number of pages similar at structural level.

4 Conclusions

In this paper we have presented an initial experiment on the use an agglomerative hierarchical clustering algorithm, a divisive clustering algorithm, a variant of the k-means partitional clustering algorithm, and a widely employed partitional competitive clustering algorithm, namely WTA in the identification of web page similarity at the structural level. These algorithms have been employed to detect similar web pages in dynamic and/or static web sites according to the Levenshtein string edit distance. This distance has been used as basic metric to identify similarity between pairs of pages considering their structures. The selected algorithms have been evaluated on two medium and one small web applications developed using JSP technology. The results of the presented case study revealed that all the selected clustering algorithms generally produce comparable results.

References

[1] Boldyreff, C., Tonella, P.: Web Site Evolution. Special issue of Journal of Software Maintenance 16(1-2), 1–4 (2004)
[2] De Lucia, A., Scanniello, G., Tortora, G.: Using a Competitive Clustering Algorithm to Comprehend Web Applications. In: Proc. of 8th IEEE International Symposium on Web Site Evolution, Philadelphia, Pennsylvania, pp. 33–40. IEEE CS Press, Los Alamitos (2006)
[3] De Lucia, A., Francese, R., Scanniello, G., Tortora, G.: Identifying Cloned Navigational Patterns in Web Applications. International Journal of Web Engineering 5(2), 150–174, Rinton Press (2006)
[4] Di Lucca, G.A., Di Penta, M., Fasolino, A.R.: An Approach to Identify Duplicated Web Pages. In: Proc. of 26th Annual International Computer Software and Application Conference, Oxford, UK, pp. 481–486. IEEE CS Press, Los Alamitos (2002)
[5] Di Lucca, G.A., Fasolino, A.R., De Carlini, U., Pace, F., Tramontana, P.: Comprehending web applications by a clustering based approach. In: Proc. of the 10th International Workshop on Program Comprehension, Paris, France, pp. 261–270. IEEE Computer Society Press, Los Alamitos (2002)
[6] Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley-Interscience Publication, JOHN WILEY & SONS, Inc. New York, pp. 576-581
[7] Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data. An Introduction to Cluster Analysis. Wiley, New York (1990)
[8] King, F.: Step-wise clustering procedures. Journal of the American Statistical Association 62, 86–101 (1967)
[9] Levenshtein, V.L.: Binary codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory 10, 707–710 (1966)
[10] Mcqueen, J.: Some methods for classification and analysis of multivariate observations. In: Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
[11] Ricca, F., Tonella, P.: Using Clustering to Support the Migration from Static to Dynamic Web Pages. In: Proc. of International Workshop on Program Comprehension, Portland, Oregon, USA, pp. 207–216 (2003)
[12] Tonella, P., Ricca, F., Pianta, E., Girardi, C.: Restructuring Multilingual Web Sites. In: Proc. of International Conference on Software Maintenance, Montreal, Canada, pp. 290–299. IEEE CS Press, Los Alamitos (2002)