

Stochastic Voting Algorithms for Web Services Group Testing

W.T. Tsai, Dawei Zhang, Raymond Paul, Yinong Chen
Department of Computer Science and Engineering
Arizona State University Tempe, AZ 85287-8809
E-mail: wtsai@asu.edu

Abstract

This paper proposes a stochastic voting for testing a large number of Web Services (WS) under group testing. In the future, a large number of WS will be available and they need to be tested and evaluated in real time. While numerous test input generation techniques are available to generate test inputs, the oracle or the expected output of these test input is often difficult to obtain. One way to obtain the oracle in this case is to give the same input to multiple WS and to establish the oracle by a majority voting. This is based on the assumption that faulty WS often will not produce consistent results, and thus if a majority can be reached, the oracle can be established statistically. However, even correct WS may still produce slightly different outputs, and thus the majority-voting scheme must be carefully designed to distinguish correct but slightly variant output from truly incorrect output. This paper proposes a hierarchical classification based on simulated annealing and multi-dimensional Chi-square statistical techniques to analyze data to see if a majority can be reached. The algorithm is evaluated by a comprehensive simulated data as well as actual data. The data show that the proposed algorithm is effective even in a difficult situation where clusters of WS produce clusters of output.

Keywords: *Web services testing, voting, clustering, Simulated Annealing.*

1. Introduction

Web Services (WS) are emerging as a key technology in today's software engineering practice. Using WS technology, software developers do not have to develop each component of a system. The alternative is to use the WS features of search, matching, discovery, orchestration or composition, and remote invocation.

For a given functional specification, many WS may exist that meet the specification. The questions are, can we trust the WS found over the Internet? If they can be proved to be trustworthy, how can we efficiently find the WS in terms of dependability, performance, and cost, assuming there are thousand of alternative WS available?

Group testing techniques, initially applied in blood test [6], have been proposed to large number of WS [20][21]. Under group testing, the WS with the same specification are tested by applying the same test scripts. The outputs of the WS are expected to be the same or within acceptable range. A voting mechanism is applied with group testing to vote the outputs of the WS under test.

The voting mechanism addresses the following issues:

- The expected output for a test input may not be available, and majority voting can be used to establish the oracle statistically.
- The output of the WS may not be numerical values, and in case it is not, it is necessary to convert the data into numerical data.
- The expected output may not be unique points, and indeed any data in a given range can be considered correct output for many applications.

In [22], we proposed a heuristic voting algorithm that efficiently votes on the outputs of WS under testing. The algorithm uses the idea of k-mean clustering to handle the multi-dimensional data with deviations. The heuristics is based on local optimization and may fail to find the global optimal results. Furthermore, the algorithm assumes that the allowed deviation is known, which may be hard to determine because the deviation is application dependent.

In this paper, we propose the hierarchical Simulated Annealing (SA) clustering algorithm combined with Chi-square goodness-of-fit test to improve the heuristic algorithm proposed in [22]. The SA clustering algorithm considers that the clustering problem is a

comprehensive optimization problem. The SA algorithm is efficient in finding the optimal or near optimal solution. In this paper, the SA clustering algorithm tries to obtain the global optimal results in the given time frame. The algorithm can handle numerical data, non-numerical data, a list of data and structured data. It can also handle data with variance or without variance.

At each step, the algorithm divides data into two clusters. The Chi-square goodness of fit test is used as the termination examination. A cluster is further divided if it fails to pass the test. The process continues until all clusters pass the test. Upon termination, we apply Chi-Square goodness-of-fit test again in testing adjacent clusters combine them to form a new cluster if possible. After all processes are done, each cluster holds the data with the same nature. The corresponding WS have the same nature. The winning cluster is selected with consideration of the size of cluster and the variance. The winning cluster is also the cluster of the correct WS.

The reason that the simple splitting algorithm works is that we have the Chi-Square goodness of fit test as the termination test. It is application dependent. WS under test are supposed to have been designed to follow the same specification. Standards have been defined to restrict the ways the WS are implemented. Thus, the outputs of all correct WS should be within a given margin of variance. At the same time, there are many WS under test. The output data of correct WS should follow the Normal distribution. Thus, Chi-square goodness of fit test is applicable as a termination test.

The proposed algorithm has several advantages. First, it is not necessary to make the difficult decision on the final number of clusters in advance. Second, it not only has a simple splitting process at each step, but also enables splitting processes in parallel. Furthermore, Chi-Square test not only works as the termination test, but also can merge the adjacent clusters after clustering if they have similar nature. By doing so, the chance of error is reduced.

The paper is organized as follows. Section 2 introduces the clustering techniques and the SA process with its applications in clustering. Section 3 introduces the SA algorithm in detail. Section 4 describes Chi-square goodness fit test and how to extend it to handle multi-dimensional data. Section 5 presents a case study on real web service testing data and synthetic data. Section 6 concludes this paper.

2. Clustering and Simulated Annealing Process

Clustering is a general unsupervised classification technique, which has been applied to many applications, such as [24][27][28][29]. It sorts data or patterns into clusters. It measures the dissimilarity between the clusters and the similarity within each cluster. The general object is to put the most similar patterns into the same cluster and to place the most dissimilar pattern into different clusters.

Kaufman [12] summarized that there are two kinds of clustering algorithms: partitioning and hierarchical. The classic partition algorithms are K-mean algorithms and the medoid-based method. Commonly used medoid-based methods are PAM (Partitioning Around Medoids), CLARA (Clustering LARge Applicatons) and CLARANS (Clustering Large Applications based on RANdimized Search) [18]. The K-mean algorithm [10] has the time complexity $O(n)$. K-medoid algorithm has the time complexity $O(n^2)$. [25] Wei compared the performance among CLARA, CLARANS, GAC-R3 [7], and GAC-RARw [7] on both quality of solution and on execution time. Partition algorithms need to decide the exact number of clusters in advance, which has an effect on the quality of clustering. Sometimes, it is a hard decision problem. In hierarchical clustering, clusters have a tree-like structure [10]. A cluster is either merged into a larger cluster or split into smaller clusters.

The clustering problem is a combinatorial optimization problem. Standard clustering methods such as k-mean may only find local optimization. Existing optimization techniques can avoid of being trapped in the local optimization areas. These techniques can be used to solve the clustering problem, such as ANN (Artificial Neural Network) [15], GA (Genetic Algorithm) [11][26], SA (Simulated Annealing Algorithm) and TS (Tabu Search) [2].

This paper applies SA to solve the clustering problem. SA algorithms stem from the material process, which cools a piece material gradually from an initial high temperature. As the temperature slowly cools down, it gives sufficient time for molecules to align themselves and crystallize. After annealing, the piece of material will show a better physical feature. Simulated annealing algorithm was proposed to simulate this material process. It is an optimization method to avoid being restricted to local optimization while searching for optimal or near-optimal global solutions. Metropolis proposed an algorithm simulating the molecular process in 1953 [17]. SA was first proposed by Kirkpatrick in 1983 [13]. Two major

factors in the SA algorithm are the cooling schedule and the definition of move. The cooling schedule includes initial temperature, stopping temperature, the rule of reducing temperature, and the run length at each temperature. The SA process can be represented by a Markov-chain. In [1][16], it was proved that SA can find the global solution under certain conditions and the rules to define cooling schedule were given.

Several researchers have applied SA in the clustering problem [3][5][14][19]. [5] applied SA to evaluating effects of criteria on clustering. It was shown that SA was a good technique for clustering. [14] studied the performance of SA in clustering. By comparing with standard k-mean algorithms, Klein stated that SA has a more reliable solution than that of K-mean algorithms. However, SA needs more computing time than that of K-mean algorithms. [3] also gave an empirical study of k-mean, GA, SA and TS. They also gave the similar statement as Klein did. TS, GA and SA were all better than K-mean methods. However, K-mean methods normally require less execution time.

All these studies have implemented SA in its original form, a one-level partition-based approach, in which the total number of clusters must be determined in advance. It is different from ours. Due to application-dependent, we introduce Chi-Square goodness-of-fit test as the termination examination. It enables us make our SA algorithm hierarchical. Then, the simple splitting process works. Thus, we don't need to determine how many clusters we should have in advance.

3. Voting Scheme for WS Group Testing

This section first presents data pre-processing that converts non-numerical data into numerical data before voting. Then, it presents the hierarchical SA algorithm for clustering and the Chi-square goodness fit test.

3.1. Data Pre-processing

Outputs of WS can be numerical data, non-numerical data, structure data or a list of data. To do voting, it is necessary to convert those non-numerical data into numerical data, and the mapping needs to be one-to-one. WS may also have a list of data as its output. For example a stockbroker WS may recommend a list of stocks for its clients, the names of stocks as well as their ranking in the list are both essential. But a one-to-one mapping can still be established by using two binary data, and the voting

must be carried in two phases, one for each binary data. WS also can output structure data, such as attributes of one object. Structured data can be encoded as a multi-dimensional data. If the output is complex, such as a structure data with both numerical data and non-numerical data, it is also possible to establish a complex one-to-one mapping into numerical data. It is also possible to assign the weight to each dimension to show the importance of each dimension. Similarly, a list of data and non-numerical data can be mapped to numerical number one-to-one.

Numerical data has another issue. Two WS can be treated as both correct implementations, even though they give different outputs because their results are close enough. The proposed algorithm can handle these situations.

Since it is possible to convert a non-numerical, a list of data and structured data into numerical data the rest of the paper assumes that the WS under test will output numerical data only.

3.2. SA Algorithm

Assume that m WS are under group testing, each of which will output a simple numerical data. Certain variances among the data are allowed. The numerical data is pre-processed by placing it in a certain range, e.g., $[0, 100]$, due to the consideration of precision. The aim of voting is to find the most probable correct output and thus the most probable correct WS by clustering similar outputs of WS into the same cluster and separating different outputs apart in different clusters.

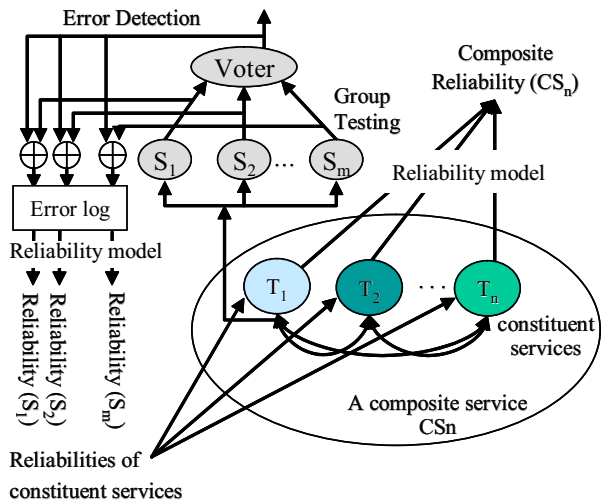


Figure 1. Group testing scheme

Figure 1 illustrates the group testing mechanism we implemented in a trustworthy WS broker. Assume CS_n

is a composite service consisting of n constituent services T_1, T_2, \dots, T_n . While CS_n is performing services, many new services, which are functionally equivalent to a constituent service T_i in CS_n , could have been registered. Testing and ranking the new services are useful for the following reasons:

- If a constituent service in CS_n fails or does not meet certain requirement, the system can immediately find the best replacement;
- If a new composite service needs to be constructed, the ranked candidate components are available.

Assume the newly arrived services S_1, S_2, \dots, S_m are functionally equivalent to the constituent service T_i in CS_n . The system can forward (broadcast) the input to T_i to S_1, S_2, \dots, S_m , as shown in Figure 1. The results from all services are voted by the voting mechanism. The voting mechanism also detects faults by comparing the output of each service with the majority output from voting. A disagreement indicates a fault. Based on the disagreement/failure log generated by group testing, the reliabilities individual services under test can also be evaluated [23].

Figure 2 gives the overview process of the SA algorithm.

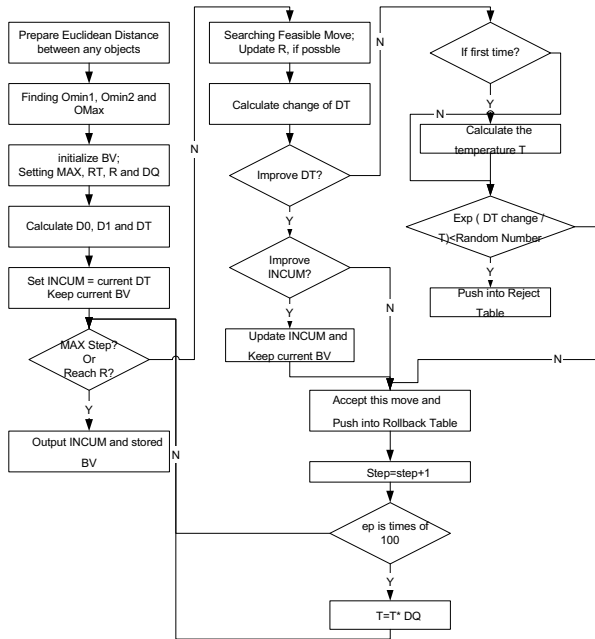


Figure 2. SA algorithm outline

For a given test case, assume the corresponding output set of the m WS under test is $\mathbf{O} = \{O_1, O_2, \dots, O_m\}$, where $O_i, i = 1, 2, \dots, m$, is an array of k elements, which can be interpreted as a point in a k -dimensional space. The distance between two objects can be defined by the Euclidean Distance:

$d(i, j) = \|\mathbf{O}_i - \mathbf{O}_j\|^2 = \sum (O_{ir} - O_{jr})^2$, where O_{ir} and O_{jr} are the r^{th} element in O_i and O_j , respectively.

The SA algorithm as following:

1. Prepare the distance matrix between each pair of objects;
2. Find the two objects with the minimum distance, Namely as $O_{\min-1}$ and $O_{\min-2}$;
3. Find the object O_{Max} that has the maximum distance to one of the two objects $O_{\min-1}$ and $O_{\min-2}$;
4. Define an m -bit binary vector BV such as $\{0, 0, 1, \dots, 1, \dots, 0\}$, where each bit corresponds to an object. The BV is initially set to having two 1s and all other bits are 0. The two 1-valued bits correspond to $O_{\min-1}$ and $O_{\min-2}$, respectively. Bits with the same value mean that corresponding objects are in the same cluster;
5. Calculate within-cluster distance $D0$ and $D1$ as $D0 = \sum d(BV_i, BV_j)$, where BV_i and BV_j are '0'; $D1 = \sum d(BV_i, BV_j)$, where BV_i and BV_j are '1'.
6. Calculate the $Dt = D0 + D1$; and set initial $INCUM = Dt$;
7. Setting the maximum steps MAX , rollback table size RT , rollback times R , set temperature decrease ratio as DQ ;
8. If steps reach MAX or rollback times reach R , then quit, the $INCUM$ is the best sum of within-cluster distance; BV leads to $INCUM$ is the solution we need.
9. Finding the feasible move. The move changes one bit of BV from 1 to 0 or 0 to 1, which indicates move one object from the current cluster to another cluster. Feasible move must not happen to $O_{\min-1}$, $O_{\min-2}$ and O_{Max} . Feasible move is not the moves that appear in the rollback table and reject table. If there is no feasible move, $R = R+1$. And select the first move in RT as current feasible move and empty RT table.

10. Calculating temporal within-cluster distance

- a. Feasible move is to change BV_i from 0 to 1

$$\text{tempZero} = D0 - \sum d(BV_i, BV_j)$$

$$\text{tempOne} = D1 + \sum d(BV_i, BV_k)$$

- b. Feasible move is to change BV_i from 1 to 0

$$\text{tempZero} = D0 + \sum d(BV_i, BV_j)$$

$$\text{tempOne} = D1 - \sum d(BV_i, BV_k)$$

where, $BV_j = 0$ and $BV_k = 1$;
 $temp = tempZero + tempOne$

11. Calculating the $\Delta Dt = Dt - temp$;
12. IF $\Delta Dt > 0$, it is an improvement move; $Dt = temp$, push this move into RT and pop the first item in RT if RT table is full. If $Dt < INCUM$, $INCUM = Dt$, and keep the current BV as best solution; go to step 15; if $\Delta Dt < 0$, go to step 13;
13. If it is the first time $\Delta Dt < 0$, estimate the proper temperature T in order that the probability of accepting bad movies in MAX step is not very tiny. Set such probability as 0.1, T is equal to $Round \{ \Delta Dt / [DQ^{MAX/100} * \ln(0.1)] \}$, usually we set T as times of 10 or 100.
14. Calculating $Exp(\Delta Dt/T)$, and get a random number RAN. IF $Exp(\Delta Dt/T) > RAN$, We also accept this movie. $Dt = temp$, push this move into RT and pop the first item in RT if RT table is full; if $Exp(\Delta Dt/T) < RAN$, we reject this move and push this move into Reject table;
15. Step = step +1; if step is times of 100, $T = T * DQ$; go to step 8.

The variables are temperature T, maximum steps, roll back times and roll back steps. Maximum steps and roll back times control what is the end of SA. These are set with consideration of running time or search steps. Usually, the more steps of SA are, the more precise the solution is. The more steps are, the longer the running time is. Typically, the maximum step is set to 10,000 steps and roll back times is set to 10.

Our experimentation shows that the algorithm proposed can find the best solution no more than 5,000 steps and 6 rollbacks. Rollback is the idea from Tabu Search [9]. It is used here to better escape from local optimization areas. Temperature T is an important parameter. It has effects on precision of solution. Temperature controls tolerance level to the bad moves. The higher the tolerant level, the higher the chance to avoid of being trapped into the local optimization. However, the larger the temperature, the lower the convergence rate will be. A simple rule is used in the algorithm. It allows bad moves with no tiny chance while close to the maximum steps. The formula is listed in Step 13 of the algorithm. By this simple rule, we can set the feasible temperature to obtain good results.

4. Chi-Square Analysis

Chi-Square goodness of fit test is used as the termination test in our clustering algorithm.

4.1. Chi-Square Goodness of Fit Test

Suppose we have one cluster with m data (outputs of WS). If most WS are implemented correctly, the data should follow the normal distribution with a mean value and a standard deviation. The hypothesis to be tested is: these m data follow normal distribution with certain mean value μ and standard deviation σ . If these data pass the chi-square goodness of fit for a given significant level α , the data are proven to belong to the same cluster. In other words, the data contained in the cluster are from WS with almost identical nature. If the Chi-Square test fails, the data in the current cluster are different enough and the cluster need to be further divided into two clusters.

Suppose we have m data with mean value μ and standard deviation σ . The algorithms sort those data in ascending order into k bins. The number of data in the i^{th} bin is Obs_i .

The expected value of Obs_i is:

$E_i = (NORM(Y_i) - NORM(Y_{i-1})) * m$, where NORM is the cumulative distribution function of normal distribution with mean value μ and standard deviation σ . Y_i is the upper limit of the i^{th} bin, while Y_{i-1} is the lower limit of the i^{th} bin.

Then the Chi-value of the i^{th} bin is

$$X_i = (Obs_i - E_i)^2 / E_i.$$

The algorithm sums chi-value of k bins together to form the final chi-value. Then, the algorithm looks up the Chi-square distribution table with significant level α and the degree of freedom, which is $(m-1)$. If the Chi-value we calculate is less than the value in the table, the Chi-square goodness-of-value test succeeds. Otherwise, it fails.

4.2. Extend Chi-Square Goodness of Fit Test for Multi-Dimensional data

Frequently, the outputs of WS are multi-dimensional. The standard Chi-square goodness of fit test cannot be directly applied. This section extends the standard test to handle the multi-dimensional data.

A multi-dimensional datum represents a point in a multi-dimensional space. The data in the same cluster are adjacent points in the space. Suppose we have a hyper-sphere with a central point. Every other point in its surface has the same distance from the central point. In a 2-dimension space, it is a circle. In a 3-dimension space, it is a sphere. A cluster can be considered to be a hyper-sphere. All data in a cluster are points inside or on the surface of the hyper-sphere.

In our test, we need to find the minimum sphere to enclose n points in an m -dimensional space. Arvo presented an algorithm to find the smallest bounding circle for n points in a 2D space [4]. Glassner designed a near optimal bounding sphere for any set of n points in a 3D space with at most 5% bigger than the ideal minimum-radius sphere [8]. We extend Glassner's algorithm to m -dimensional space.

Assume there are K points in an m -dimensional space. The algorithm consists of two steps.

1. Find the initial sphere

From each of the m dimensions, find a pair of points that one point has the minimum coordinate and the other with the maximum coordinate in the dimension. Among these m pairs, we pick the pair with the maximum point-to-point distance. We create an initial sphere using this pair of points as a diameter. The time complexity of this step is $O(K)$.

2. Updating the initial sphere by including the outside points.

Add one dimension to the initial sphere to include the maximum pair in this dimension found in step 1. This procedure continues until all pairs are included in the hyper-sphere. This step also takes $O(m)$ time.

5. Experiments and Data Analyses

Extensive testing and experiments on the algorithms have been performed. The data used comes from both real WS group testing and synthetic data designed to test the performance of the algorithm.

5.1. Data Collection and Analysis

We have implemented 60 different WS reporting the stock prices based on the same business logic. Some of them were modified to produce slightly different results and some were injected with certain faults. Group testing is applied to the 60 WS and data are collected. Table 1 lists the prices at a particular point in time.

Table 1. Sorted experiment data

Stock Price	Stock Price	Stock Price	Stock Price	Stock Price	Stock Price
2.75	11.49	19.41	20.77	21.75	29.2
6.41	12.13	19.8	20.93	21.99	29.3
8.17	12.24	19.8	20.93	22.1	29.3
8.24	13.31	20.28	21.12	22.34	29.8
8.41	13.31	20.35	21.12	22.56	30.35
9.15	14.03	20.39	21.35	22.75	30.73
9.2	14.67	20.41	21.5	28.17	30.85
9.8	15.22	20.47	21.5	28.41	31.49
10.36	15.22	20.62	21.71	28.85	32.13
11.04	18.58	20.72	21.71	29.15	32.24

Applying the clustering algorithm, we obtain two clusters of data, Cluster 0 and Cluster 1. Cluster 0 includes data from 2.75 to 19.41, while Cluster 1 has the other data.

Table 2 is the Chi-Square test for Cluster 0. The final chi-value for cluster0 is 1.700459. The value of Chi-square distribution with 20 degree of freedom with $\alpha=0.001$ is 5.921. The test passes. However, the final chi-value of cluster 1 is 50.10054. The value of Chi-square distribution with 38 degree of freedom with $\alpha=0.001$ is 16.611. The test fails. Cluster1 is further divided into smaller clusters, which are Cluster 1-1 and 1-2. Cluster 1-1 has the data from 19.8 to 22.75, while Cluster 1-2 has the others.

Table 2. Chi-square GoF test over cluster0

bin	Observer	Expected	Chi-value
<2	0	0.0360	0.0360
[2,5]	1	0.8498	0.0266
[5,8]	1	2.8290	1.1825
[8,11]	7	5.4389	0.4481
[11,14]	6	6.0445	0.0003
[14,17]	4	3.8837	0.0035
>=17	2	1.9181	0.0035

The Chi-value of the Cluster 1-1 is 0.5909. The value of Chi-square distribution with 24 degree of freedom with $\alpha=0.001$ is 8.085. The Chi-value of the Cluster 1-2 is 1.7098. The value of Chi-square distribution with 13 degree of freedom and with $\alpha=0.001$ is 2.617. Both cluster 1-1 and cluster 1-2 pass the test. The clustering process ends.

Figure 3 shows the clustering process. We sort data into 3 clusters. There is only less than 0.1% chance that the results are incorrect.

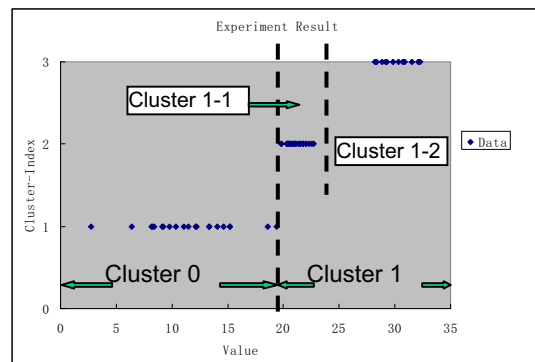


Figure 3. Experiment result

Cluster 0 has 21 data with mean 11.58, standard deviation 3.96. Cluster 1-1 has 25 data with mean 21.15 and standard deviation 0.82. Cluster 1-2 has 14 data with mean 30.00 and standard deviation 1.32. Thus, with either majority criterion or least standard deviation, cluster 1-1 is the best. We recommend that the stock price be expected to be 21.15. The optimal

solution, which we calculate, is 21.09. The relative error rate is 0.2%. The total running time is 2.8 second.

5.2. Simulation on Synthetic Data

In this section, we use synthetic data to test the algorithm. Using controlled random data generator, we created a set of data in different data ranges and thus forming different clusters. Each cluster of data follows the normal distribution. We merge the data into one group and then apply the clustering algorithm. We want to see if the algorithm can successfully cluster the data into the original clusters.

Table 3. Summary of experiments

Index	Data Size	Data Source	SA Cluster	Misplaced Data	Outline Data	SA rounds
1	10	6	7	0	1	6
2	20	7	10	0	3	9
3	30	7	10	0	3	9
4	40	7	10	0	4	10
5	50	7	11	0	6	15
6	60	7	10	0	5	10
7	70	7	9	0	3	12

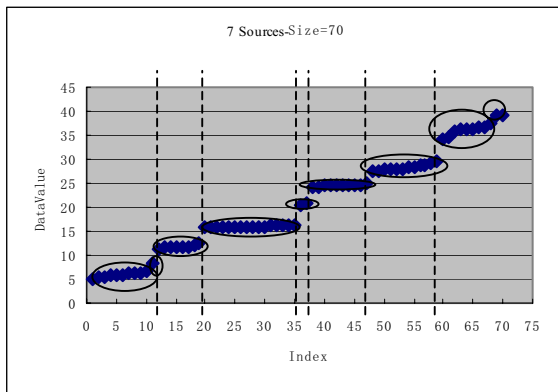


Figure 4. Experiment 7: DataSize=70

Table 3 lists the result related to seven experiments that we performed. Take the experiment 7 as example. Seventy data are generated. They belong to seven different sources or clusters. Our SA clustering algorithms divided them into 9, instead of 7 clusters. “Misplaced data” means that data that are from different sources and mistakenly placed into the same cluster. No data are misplaced. However, the algorithms separated data from same source apart, creating more clusters than that of expected. The reason is that data are randomly generated. Even though data are from the same source, the differences can be significant because of the big size of the source. When we select a few data from the source, the differences may become too significant to place them into the same cluster. Thus, it makes good sense to

separate these data apart. These data are called minority data or outline data. We list number of them in the ‘Outline Data’ column in the table. The number of outline data will reduce when we sample more data from the same source. The last column lists the rounds that we repeat the SA algorithm to obtain the cluster. Figure 4 is the figure of experiment result on data size 70.

6. Summary and Conclusion

The main contributions of the paper are follows: 1) We developed an effective algorithm to meet the need of WS group testing; 2) We applied SA algorithm to find the cluster that represent the correct WS outputs; 3) We introduced the hierarchical clustering process that stops when the Chi-square goodness fit test is met. The process doesn’t need to know the total number of clusters in advance, which is different from the existing SA algorithms. 4) We also extended the standard Chi-square goodness fit test to handle multi-dimensional data. The algorithm has been successfully used with WS group testing and experiment data are collected and evaluated.

Reference

- [1] E. Aarts, J. Korst, *Simulated Annealing and Boltzmann Machines - A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley & Sons, Chichester, England, 1989.
- [2] K.S. Al-Sultan, “A Tabu Search Approach to Clustering Problems”, *Pattern Recognition*, vol 28, 1995, pp. 1443-1451.
- [3] K.S. Al-Sultan, and M.M. Khan, “Computational Experience on Four Algorithms for the Hard Clustering Problem”, *Pattern Recognition letter*, vol 17, n3, March 1996, pp. 295-308.
- [4] J. Arvo, *An Easy Bounding Circle*, Jon Rokne, *Graphics Gems II*, Academic press, Boston, U.S.A, 1991.
- [5] D.E. Brown and L. Huntley, “Pactical Application of simulated Annealing to Clustering” *Pattern Recognition*, vol 25, No4, 1992, pp. 401-412.
- [6] D. Z. Du and F. Hwang, *Combinatorial Group Testing and Its Applications*, World Scientific, 2nd edition, Singapore, 2000.
- [7] V. Estivill-Gastro, A.T. Murray, “Spatial Clustering for Data Mining with Generic Algorithms”, Technical Report FIT-TR-97-10, Queensland University of Technology, Sep 1997.
- [8] A.S. Glassner, *An Efficient Bounding Sphere*, Jack Ritter, *Graphics Gems*, Academic Press, Boston, U.S.A, 1990.

- [9] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research*, vol. 13, no 5, May 1986, pp. 533-549.
- [10] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data Clustering: A Review", *ACM Computing Surveys*, Vol. 31, No. 3, 1999, pp. 255-323.
- [11] D. Jones, and M.A. Beltramo, "Solving Partitioning Problems with Genetic Algorithms", In proceedings of the fourth International Conference on Genetic Algorithms, San Diego, CA, USA, July 1991, pp. 442-449.
- [12] L. Kaufman, P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley & sons, New York, USA, 1990.
- [13] S. Kirkpatrick, C.D. Gelatt Jr. and M.P. Vecchi, "Optimization by Simulated Annealing", *Science*, N.Y. 220, 1983, pp. 671-680.
- [14] R.W. Klein, R.C. Dubes, "Experiments in Projection and Clustering by Simulated Annealing", *Pattern Recognition*, Vol. 22, No. 2, 1989, pp. 213-220.
- [15] T. Kohonen, *Self-Organization and Associative Memory*, 3rd edition, Springer Information Sciences Series, SpringerVerlag, Berlin, 1989.
- [16] P.J.M van Laarhoven and E.H.L. Aarts, *Simulated Annealing: Theory and Applications*, D.Reidel Publishing Company, Norwell, MA, U.S.A, 1987.
- [17] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller, "Equations of state Calculations by Fast Computing Machines", *J.Chem. Phys*, vol 21, 1953, pp. 1087-1092.
- [18] R.T. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining", *Proceedings of 20th International Conference on Very Large Data Bases*, 1994, pp. 144-155.
- [19] S.Z. Selim, K. Alsultan, "A Simulated Annealing Algorithm for the Clustering Problem", *Pattern Recognition*, Vol 24, No10, 1991, pp. 1003-1008.
- [20] W.T. Tsai, Y. Chen, Z. Cao, X. Bai, H. Huang, R. Paul, "Testing Web Services Using Progressive Group Testing", *Advanced Workshop on Content Computing*, Zhenjiang, China, November 2004, pp. 314-322.
- [21] W.T. Tsai, Y. Chen, R. Paul, H. Huang, X. Zhou, X. Wei, "Adaptive Testing, Oracle Generation, and Test Script Ranking for Web Services," to appear in 29th Annual International Computer Software and Applications Conference (COMPSAC), Edinburgh, Scotland, 2005.
- [22] W.T. Tsai, Y. Chen, D. Zhang, H. Huang, "Voting Multi-Dimensional Data with Deviations for Web Services under Group Testing," 4th International Workshop on Assurance in Distributed Systems and Networks (ADSN), Columbus, June 2005, pp. 65-71.
- [23] W.T. Tsai, D. Zhang, Y. Chen, H. Huang, R. Paul, and N. Liao, "A Software Reliability Model for Web Services," submitted to the 8th IASTED International Conference on software engineering and applications, Cambridge, MA, November, 2004, pp. 144-149.
- [24] C.J. Veenman, M.j.t. Reinders, E. Backer, "A Maximum Variance Cluster Algorithm", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 9, Sep 2002, pp. 1273-1280.
- [25] C.P. Wei, Y.H. Lee and C.M. Hsu, "Empirical comparison of fast clustering algorithms for large data sets", *Proceedings of 33rd Hawaii International Conference on System Sciences*, Maui, Hawaii, Jan. 2000, pp. 2013-2022.
- [26] D. Whitley, T. Starkweather, T. and D. Fuquay, "Scheduling Problems and Traveling Salesman: the Genetic Edge Recombination", In *Proceedings of the Third International Conference on Genetic Algorithms* (George Mason university, June 4-7), J.D. Schaffer, Morgan Kaufmann, San Francisco, CA, 1989, pp. 133-140.
- [27] F. Wu and G. Gardarin, "Gradual Clustering Algorithms", *Proceedings Seventh International Conference on Database Systems for Advanced Applications. DASFAA 2001*, pp. 48-55.
- [28] X. Xu, M. Ester, H. Kriegel, J. Sander, "A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases", *Proc. 14th Int. Conf. on Engineering (ICDE '98)*, Orlando, FL, 1998, pp. 324-331.
- [29] Z. Yao and B. Choi, "Bidirectional Hierarchical Clustering for Web Mining", *Web Intelligence 2003*, pp. 620-624.