Unsupervised Linkage Learner Based on Local Optimums

Hamid Parvin and Sajad Parvin

Islamic Azad University Nourabad Mamasani Branch, Nourabad Mamasani, Iran
hamidparvin@mamasaniiau.ac.ir,
s.parvin@iust.ac.ir

Abstract. Genetic Algorithms (GAs) are categorized as search heuristics and have been broadly applied to optimization problems. These algorithms have been used for solving problems in many applications, but it has been shown that simple GA is not able to effectively solve complex real world problems. For proper solving of such problems, knowing the relationships between decision variables which is referred to as linkage learning is necessary. In this paper a linkage learning approach is proposed that utilizes the special features of the decomposable problems to solve them. The proposed approach is called Linkage Learner based on Local Optimums and Clustering (LLLC). The LLLC algorithm is capable of identifying the groups of variables which are related to each other (known as linkage groups), no matter if these groups are overlapped or different in size. The proposed algorithm, unlike other linkage learning techniques, is not done along with optimization algorithm; but it is done in a whole separated phase from optimization search. After finding linkage group information by LLLC, an optimization search can use this information to solve the problem. LLLC is tested on some benchmarked decomposable functions. The results show that the algorithm is an efficient alternative to other linkage learning techniques.

Keywords: Linkage Learning; Optimization Problems, Decomposable Functions.

1 Introduction

Artificial intelligence has found many applications in almost all fields [11-23]. Genetic Algorithms (GAs) as a subfield in artificial intelligence are the most popular algorithms in the category of Evolutionary Algorithms (EAs). These algorithms are widely used to solve real-world problems. However when it comes to solve difficult problems, GA has deficiencies. One of the main problems of simple GAs is their blindness and oblivion about the linkage between the problem variables. It is long time that the importance of the linkage learning is recognized in success of the optimization search. There are a lot of linkage learning techniques. Some are based on perturbation methodology, some are categorized in the class of probabilistic model building approaches and some are the techniques that adapt the linkages along with the evolutionary process by employing special operators or representations.

J.A. Carrasco-Ochoa et al. (Eds.): MCPR 2012, LNCS 7329, pp. 255–264, 2012.

In this paper a new linkage learning approach, which is called LLLC is proposed. The proposed algorithm as its title implies, does not fall in the above mentioned categories, but it is a linkage group identification approach which tries to identify multivariate dependencies of complex problems in acceptable amount of time and with admissible computational complexity.

2 Background

In this section, Deterministic Hill Climbers (DHC) which will be used later in our algorithm and challenging problems which are used to explain and test the proposed algorithm are described. Firstly, some terms should be defined. A partial solution denotes specific bits on a subset of string positions. For example, if we consider 100-bit binary strings, a 1 in the second position and a 0 in the seventh position is a partial solution. A building block is a partial solution which is contained in an optimum and is superior to its competitors. Each additively separable problem is composed of number of partitions each of which is called a "linkage group".

In this study, DHC [10] are used to search for local optimums. In each step, the DHC flips the bit in the string that will produce the maximum improvement in fitness value. This process can be allowed to iterate until no single bit flip produces additional movement. DHC starts with a random string.

2.1 Challenging Problems

Deficiencies of GAs were first demonstrated with simple fitness functions called deceptive functions of order *k*. Deception functions of order *k* are defined as a sum of more elementary deceptive functions of *k* variables. In a deceptive function the global optimum (1, ,1) is isolated, whereas the neighbors of the second best fitness solution (0, ,0) have large fitness values. Because of this special shape of the landscape, GAs are deceived by the fitness distribution and most GAs converge to (0, ,0). This class of functions has a great theoretical and practical importance. An *n*-bit $trap_k$ function has one global optimum in the string where the value of all the bits is 1, and it has $(2^{n/k})$ - *1* local optimums. The local optimums are those individuals that the values of the variables in a linkage group are either 1 or 0 (they are all 1, or they are all 0) [8]. Fitness function of a $trap_k$ problem size n is obtained by equation 1.

$$trap_k(u) = \begin{cases} k & if\,(u = k) \\ k - 1 - u & otherwise \end{cases}$$

$$f_{trap_k}(X) = \sum_{i=1}^{n/k} trap_k(X_{(i-1)*k+1:(i-1)*k+k}) \tag{1}$$

where *u* is the number of ones in the input block of k bits. You can see Fig. 1 to reach a better understanding of $trap_5$ problem. Fig. 1 shows the fitness function for a building block in $trap_5$ problem.

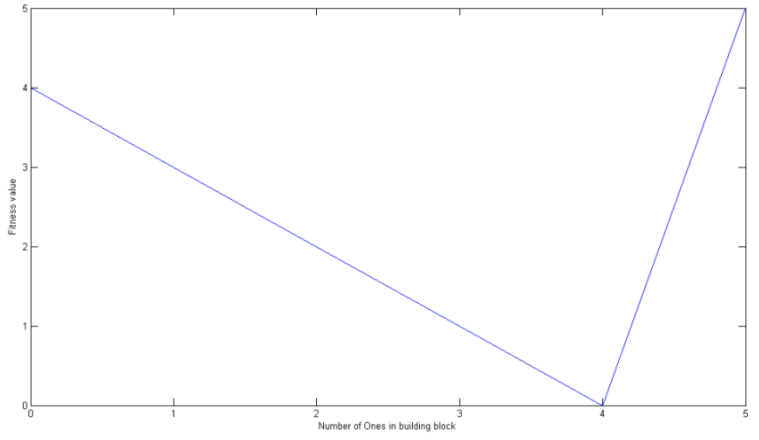


Fig. 1. Fitness-function for a $trap_5$ building block

Also another additively separable function called $deceptive_k$ [8]. An *n*-bit $deceptive_k$ function like an *n*-bit $trap_k$ function has one global optimum in the string where the value of all the bits is 1, and it has $(2^{n/k})$ - *1* local optimums. Fitness function of a $deceptive_k$ problem size n is obtained by equation 2.

$$\text{deceptive}_k(u) = \begin{cases} 1 & if\ (u = k) \\ 0.9 - 0.5 * u / k & otherwise \end{cases} \tag{2}$$

$$f_{\text{deceptive}_k}(X) = \sum_{i=1}^{n/k} \text{deceptive}_k(X_{(i-1)*k+1:(i-1)*k+k})$$

where *u* is the number of ones in the input block of k bits. For yet another more challenging problem we use an additively separable function, one bit overlapping-$trap_k$. An *n*-bit overlapping-$trap_k$ function has one global optimum in the string where the value of all the bits is 1 just similar to $trap_k$ function, and it has $(2^{(n-1)/(k-1)})$ - 1 local optimums. The local optimums are those individuals that the values of the variables in a linkage group are either 1 or 0 (they are all 1, or they are all 0). The main difference in this function with a $trap_k$ function is existence of one shared bit between each consecutive building blocks of a chromosome. In each local optimum the value for a shared bit in two building blocks will be zero if and only if the two building blocks be converged to all 0, otherwise it will be one.

2.2 Linkage Learning

There are lots of approaches in the class of linkage adaptation techniques. Linkage learning GA [2] uses a special probabilistic expression mechanism and a unique combination of the (gene number, allele) coding scheme and an exchange crossover operator to create an evolvable genotypic structure. In [4] punctuation marks are

added to the chromosome representation. These bits indicate if any position on the chromosome is a crossover point or in another words, a linkage group boundary. Linkage evolving genetic operator (LEGO) [5] is another linkage adaptation strategy that in order to achieve the linkages, each gene has associated with it two Boolean flags. These two flags determine whether the gene will link to the genes to its left and right. The two adjacent genes are assumed to be linked if the appropriate flags are both set to true. Therefore building blocks are consecutive linked genes on the chromosome.

Linkage learning is necessary when there are epistatic linkages between variables. Estimation of distribution algorithms (EDAs) are among the most powerful GAs which try to find these epistatic linkages through building probabilistic models that summarize the information of promising solutions in the current population. In another words, by using probabilistic models these algorithms seek to find linkage between variables of the problem. In each generation they find as much information as possible about the variable dependencies from the promising solutions of the population. Knowing this information, the population of the next generation is created. There are numbers of estimation of distribution algorithms which their differences are often in the model building part. Bayesian Networks and marginal product models are examples of the probabilistic models that have been used by Bayesian Optimization Algorithm (BOA) [1] and Extended Compact Genetic Algorithm (ECGA) [3]. Although EDAs scale polynomial in terms of number of fitness evaluations, the probabilistic model building phase is usually computationally expensive. Perturbation-based method, detect linkage group by injecting perturbations in the population of individuals and inspecting the fitness change caused by the perturbation. Gene expression messy genetic algorithm (gemGA) which uses transcription operator for identifying linkage group is classified in this category.

Table 1. Some of the local optimums for Trap3 size 12

1	1	1	0	0	0	0	0	0	1	1	1
0	0	0	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0	1	1	1
1	1	1	1	1	1	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	1	1	1

Dependency Structure Matrix Genetic Algorithm (DSMGA) [8] is another approach which models the relationship among variables using Dependency Structure Matrix (DSM). In DSMGA a clustering method is used for identifying the linkage groups. In spite of these efforts, none of the algorithms have been claimed to be stronger that than Hierarchical Bayesian Optimization Algorithms (HBOA) [6] and [7] which itself in spite of its polynomial scalability in terms of the number of fitness evaluations, is computationally expensive.

An approach called Local Optimums based Linkage Learner (LOLL) which is capable of identifying the groups of variables proposed by Parvin et al. [12]. LOLL algorithm, unlike other linkage learning techniques, is not done along with

optimization algorithm; but it is done in a whole separated phase from optimization search. The only limitation of LOLL is when it faces with an overlapping problem. It will also be at least weak, if it does not crash. In more detail, it will crash, if there are many local optimums in the environment.

3 Local Optimum Based Linkage Learner: LLLC

The main idea in the proposed approach for identifying the multivariate dependencies is using local optimums. But how the local optimums can lead us to identification of the linkage groups?

Local optimums of "Additively Separable problems" have some unique features. As it is obvious, the global optimum of an additively separable problem is the one that all of its building blocks are identified. In another words, in the global optimum, all of the linkage groups of the problem have the highest possible contribution to the overall fitness. But in local optimum solutions, not all of the building blocks are found and those partitions or sub-problems of the problem which their optimum values are not found are occupied by the best competitors of the superior partial solution.

In additively separable problems there are lots of these local optimum solutions. Actually number of such solutions is directly dependant on length of the problem and number of partitions (or sub-problems or linkage groups) of the problem. It can be said that each local solution contains at least one building block and therefore comparison of the optimum solutions can lead us to identification of the linkage groups.

The following example can reveal this concept more clearly. Consider a 12 bit $trap_3$ function. This function has one global optimum 11111111111 and $(2^{12/3}) - 1 = (15)$ local optimums. The strings are local optimum if the bits corresponding to each trap partition are equal, but the value of all the bits in at least one trap partition is 0. Some of local optimums are shown in Table 1: A simple comparison between first local solution and fifth local solution helps us find the second linkage group and comparison between third local solution and fourth local solution helps us find the first linkage group. It means that a DHC halts in a local/global optimum due to optimization of some of its BBs.

Now, the algorithm can be explained and the example is continued later. In an overall view, there are two phases of search and analysis. In search phase some local optimums are found and in analysis phase the linkages between these local solutions are extracted. In the search phase, *K* DHCs are initialized randomly and set to search the landscape (with length (*Xs*) number of variables). When each DHC finds a peak in the landscape and no movements are possible, that solution which is a local optimum will be saved in a set named *HighModals*.

After the search phase, analysis phase starts. In the analysis phase, linkage groups should be identified by clustering the genes. A clustering method is needed for the analysis phase. The clustering method should be able to partition genes into the BBs. But to do the clustering algorithm over the genes, first it is needed to produce a data space where its items are genes and its feature covers the *HighModals* information

latently. To reach this aim, we define a temporary dataset where its features are in type of binary values. A feature represents a local/global optimum in this temporary dataset. It means *i*th feature (local/global optimum) of *j*th gene (item or data point) in the defined dataset represents that the value of *j*th bit in *i*th *HighModals*. Then we extract a co-association matrix out of each feature of the dataset. After that all co-association matrices are aggregated in a consensus co-association matrix. Finally using a hierarchical clustering algorithm, the genes are portioned into BBs. The pseudo code of LLLC algorithm is shown in Fig. 2.

X_s=(1...n);
r=0;
Search Phase:
S.1. Run DHC_s;
S.2. Save local solutions to HighModals set.
S.3. If p--exit--search < (sp) goto Analysis phase.
Else goto S.1.
Analysis Phase:
A.1. Produce new defined dataset named PDS.
A.1. Produce a Number of Co-association Matrices from PDS.
A.1. Aggregate Matrices into a Consensus Co-association Matrix.

Fig. 2. Linkage Learner based on Local Optimums and Clustering Algorithm

Xs is an array with length *n* containing the indexes of the problem variables. *DiscoveredBBs* is an array of arrays, containing the discovered linkage groups. Each linkage group is shown with an array containing the indexes of the variables in the linkage group. *HighModals* is an array containing the local optimums of the problem.

Now, we go back to our simple example: *Xs* is here the array *Xs* = {1, 2, …, 12} *HighModals* set is in Table 1. The ideal *DiscoveredBBs* is {{1,2,3}, {4,5,6}, {7,8,9}, {10,11,12}}.

Table 2. The local optimums of Table 1 are considered as a new defined dataset. Each gene is considered as a data point and each high modal is considered as a feature.

HighModals /Bits	$gene_1$	$gene_2$	$gene_3$	$gene_4$	$gene_5$	$gene_6$	$gene_7$	$gene_8$	$gene_9$	$gene_{10}$	$gene_{11}$	$gene_{12}$
$highmodal_1$	1	1	1	0	0	0	0	0	0	1	1	1
$highmodal_2$	0	0	0	1	1	1	1	1	1	0	0	0
$highmodal_3$	0	0	0	1	1	1	0	0	0	1	1	1
$highmodal_4$	1	1	1	1	1	1	0	0	0	0	0	0
$highmodal_5$	1	1	1	1	1	1	0	0	0	1	1	1

3.1 Finding Local Optimums

After the algorithm passes through the first phase of LLLC, it obtains a lot of local optimums, e.g. look at Table 2. In Table 2 Each gene is considered as a data point and

each high modal is considered as a feature. After the search phase, clustering phase starts. In the clustering phase, we first extract a co-association matrix out of each feature of the dataset. After that all co-association matrices are aggregated in a consensus co-association matrix. Linkage groups finally should be identified by a clustering method which is considered to be a complete hierarchical clustering throughout the paper.

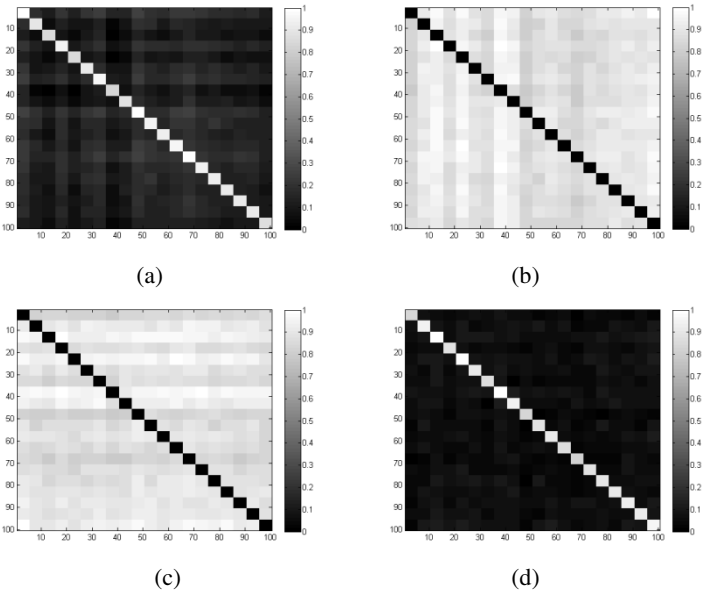
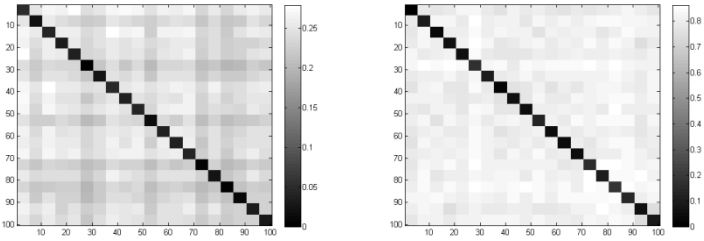


Fig. 3. (a) $Similarity_{i,j}^{0,0}$ for trap_5 size 100. (b) $Similarity_{i,j}^{0,1}$ for trap_5 size 100. (c) $Similarity_{i,j}^{1,0}$ for trap_5 size 100. (d) $Similarity_{i,j}^{1,1}$ for trap_5 size 100.

3.2 Extracting Real Building Blocks Out of Local Optimums

As it is said, if we have two or more optimums in each building block, the LOLL method has some drawbacks in finding the final *DiscoveredBBs*. For handling this drawback we use each of the found *HighModals* as a feature in a defined dataset. Based on the defined dataset we define a co-association matrix denoted by $Sim_{i,j}^{k,q}[\text{p}], \text{where } k, q \in \{0,1\}$, to be one if and only if i-th gene of p-th modal in *HighModals* be k and j-th gene of p-th modal in *HighModals* be q. After that we define matrix $S_{i,j}^{k,q} = \sum_p Sim_{i,j}^{k,q}[p]$. Now we define similarity matrix $Similarity_{i,j}^{k,q} = S_{i,j}^{k,q} / \max(S_{i,j}^{k,q})$ and dissimilarity matrix $Dissimilarity_{i,j}^{k,q} = 1 - Similarity_{i,j}^{k,q}$. For further explanation consider Fig. 3. Fig. 3 depicts the similarity matrix of a trap_5 size 100. Note that each of these four matrices can represent the necessary linkages completely. Fig. 4 also depicts the dissimilarity matrix of a deceptive_5 size 100.



(a) (b)

Fig. 4. (a) $Dissimilarity_{i,j}^{0,0}$ for deceptive$_5$ size 100. ***(b)*** $Dissimilarity_{i,j}^{1,1}$ for deceptive$_5$ size 100.

Then considering the similarity or dissimilarity matrix as a new data space of genes, a clustering algorithm can be employed to partition them. We do this process by a complete linkage hierarchical clustering. By employing a complete linkage hierarchical clustering algorithm over all dissimilarity matrices of Fig. 4-5, we obtain BBs perfectly (in all example the real number of clusters are presumed to be known). This process can also be done by cutting a minimal number of clusters (hyperedges) using an approach like HyperGraph-Partitioning Algorithm (HGPA) [9], [13].

4 Experimental Results and Discussion

Using complete linkage hierarchical clustering (also HGPA) we obtain the full linkages of the problems trap$_5$, one-bit-overlapping trap$_6$, deceptive$_5$, one-bit-overlapping deceptive$_6$, and midlle-trap$_{10}$. It is worthy to mention that in all cases we obtain only 400 local/global optimums.

To discuss why the method works well, this section will deal with a special case of two problems: (a) trap$_5$ and (b) middle-trap$_{10}$. Indeed we try to show theoretically that in these two problems the similarity matrices that there is a special relationship between bits of a linkage. It means that each bit in a linkage has more similarity values with other bits of that linkage rather than bits outside of that linkage. Indeed we want to show that the probability for two arbitrary bits of a linkage to be one-one in the pool of the local/global optimums produced by DHCs are more that the probability for two arbitrary bits, that does not belong to a linkage, to be one-one.

5 Conclusions

With the purpose of learning the linkages in the complex problem a novel approach is proposed. There are other approaches that are claimed to be able to solve those challenging problems in tractable polynomial time. But the proposed approach does not classified into the existence categories. This work has looked at the problem from whole different points of view. Our method is based on some properties of additively decomposable problems in order to identify the linkage groups. The amazing property

of additively decomposable problems that our method is based on is the special form of their local optimums which a bunch of them would give us lots of information about the linkage groups. The proposed algorithm is called LLLC. The algorithm is capable of solving the challenging problems effectively.

LLLC is capable of identifying the linkage groups in a simple and straightforward manner. Moreover we believe that the proposed algorithm (without any major changes) is capable of finding any problem building blocks. Time order of LLLC algorithm is remained for future work. Analyzing the proposed algorithm in the context of optimization problem and along with an optimization search is one of the tasks that can be done as future works. Comparing the results with the other approaches is also left as future work.

References

1. Audebert, P., Hapiot, P.: Effect of powder deposition. J. Electroanal. Chem. 361, 177 (1993)
2. Newman, J.: Electrochemical Systems, 2nd edn. Prentice-Hall, Englewood Cliffs (1991)
3. Hillman, A.R.: Electrochemical Science and Technology of Polymers, vol. 1, ch. 5. Elsevier, Amsterdam (1987)
4. Miller, B.: Geelong, Vic. J. Electroanal. Chem. 168, 19–24 (1984)
5. Jones: personal communication (1992)
6. Pelikan, M., Goldberg, D.E.: Escaping hierarchical traps with competent genetic algorithms. In: Genetic and Evolutionary Computation Conference, GECCO, pp. 511–518 (2001)
7. Pelikan, M., Goldberg, D.E.: A hierarchy machine: Learning to optimize from nature and humans. Complexity 8(5) (2003)
8. Pelikan, M.: Hierarchical Bayesian optimization algorithm: Toward a new generation of evolutionary algorithms. Springer (2005)
9. Strehl, A., Ghosh, J.: Cluster Ensembles — A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research 3, 583–617 (2002)
10. Stuart, R., Peter, N.: Artificial Intelligence: A Modern Approach, 2nd edn., pp. 111–114. Prentice Hall (2003)
11. Parvin, H., Minaei-Bidgoli, B.: Linkage Learning Based on Local Optimums Clustering of Building Blocks. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) ICCCI 2011, Part I. LNCS, vol. 6922, pp. 163–172. Springer, Heidelberg (2011) ISSN: 0302-9743
12. Parvin, H., Minaei-Bidgoli, B., Helmi, B.H.: Linkage Learning Based on Differences in Local Optimums of Building Blocks with One Optima. In: International Conference on Computational Intelligence in Security for Information Systems. LNCS, pp. 286–293. Springer, Heidelberg (2011) ISSN: 0302-9743
13. Minaei-Bidgoli, B., Parvin, H., Alinejad-Rokny, H., Alizadeh, H., Punch, W.F.: Effects of resampling method and adaptation on clustering ensemble efficacy, Online (2011)
14. Parvin, H., Minaei-Bidgoli, B.: Linkage Learning Based on Local Optima. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) ICCCI 2011, Part I. LNCS, vol. 6922, pp. 163–172. Springer, Heidelberg (2011)
15. Parvin, H., Helmi, H., Minaei-Bidgoli, B., Alinejad-Rokny, H., Shirgahi, H.: Linkage Learning Based on Differences in Local Optimums of Building Blocks with One Optima. International Journal of the Physical Sciences 6(14), 3419–3425 (2011)

16. Parvin, H., Minaei-Bidgoli, B., Alizadeh, H.: A New Clustering Algorithm with the Convergence Proof. In: König, A., Dengel, A., Hinkelmann, K., Kise, K., Howlett, R.J., Jain, L.C. (eds.) KES 2011, Part I. LNCS, vol. 6881, pp. 21–31. Springer, Heidelberg (2011)
17. Parvin, H., Minaei, B., Alizadeh, H., Beigi, A.: A Novel Classifier Ensemble Method Based on Class Weightening in Huge Dataset. In: Liu, D., Zhang, H., Polycarpou, M., Alippi, C., He, H. (eds.) ISNN 2011, Part II. LNCS, vol. 6676, pp. 144–150. Springer, Heidelberg (2011)
18. Parvin, H., Minaei-Bidgoli, B., Alizadeh, H.: Detection of Cancer Patients Using an Innovative Method for Learning at Imbalanced Datasets. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (eds.) RSKT 2011. LNCS, vol. 6954, pp. 376–381. Springer, Heidelberg (2011)
19. Parvin, H., Minaei-Bidgoli, B., Ghaffarian, H.: An Innovative Feature Selection Using Fuzzy Entropy. In: Liu, D. (ed.) ISNN 2011, Part III. LNCS, vol. 6677, pp. 576–585. Springer, Heidelberg (2011)
20. Parvin, H., Minaei, B., Parvin, S.: A Metric to Evaluate a Cluster by Eliminating Effect of Complement Cluster. In: Bach, J., Edelkamp, S. (eds.) KI 2011. LNCS, vol. 7006, pp. 246–254. Springer, Heidelberg (2011)
21. Parvin, H., Minaei-Bidgoli, B., Ghatei, S., Alinejad-Rokny, H.: An Innovative Combination of Particle Swarm Optimization, Learning Automaton and Great Deluge Algorithms for Dynamic Environments. International Journal of the Physical Sciences 6(22), 5121–5127 (2011)
22. Parvin, H., Minaei, B., Karshenas, H., Beigi, A.: A New N-gram Feature Extraction-Selection Method for Malicious Code. In: Dobnikar, A., Lotrič, U., Šter, B. (eds.) ICANNGA 2011, Part II. LNCS, vol. 6594, pp. 98–107. Springer, Heidelberg (2011)
23. Qodmanan, H.R., Nasiri, M., Minaei-Bidgoli, B.: Multi objective association rule mining with genetic algorithm without specifying minimum support and minimum confidence. Expert Systems with Applications 38(1), 288–298 (2011)