International Symposium on Advances in Science and Technology
Bandar-Abbas | Iran
7-8 March 2013
7th SASTech
Persian Gulf
Hosted by: Hormozgan University
Organised by: Khavaran Institute of Higher Education

A DNA Approach in Mining and Ranking the Classification Rules

Ramin Maazallahi, Aliakbar Niknafs*
Department of Computer Engineering
Shahid Bahonar University of Kerman, Kerman, Iran
Paper Reference Number: 07-09-5056
Name of the Presenter: Ramin Maazallahi

Abstract

DNA computing is a novel method of computing which enables us to perform complex calculations using DNA molecules and biological operations. In this paper, DNA computing has been used for mining and ranking the classification rules. Since the number of all possible classification rules for a dataset will increase exponentially with respect to the number of attributes, checking all the rules to find proper ones and then ranking them needs an algorithm with exponential time complexity in silicon based computers. Our proposed algorithm is able to solve the mentioned problem in a linear time complexity with the help of DNA and its operators. In this paper, the impact of this method on improving the processing speed is shown.

Key words: Classification rule mining, DNA computing, NP problems, Rule ranking.

1. Introduction

Classification is an important concept of data mining field which aims to build a model for describing a set of samples distributed in predefined classes. Classification Rule Mining is a method for building the model by extracting proper classification rules. When the model is build, it can be used for classifying the future or unknown samples. As a short review on methods of mining the classification rules we can consider these researches: using constraint-based genetic algorithm by Chiu and Hscu (2005), using gene expression programming (GEP) by Zhou and Xiao (2003), applying Ant Colony Optimization (ACO) by Martens et al. (2007), and finally using neural networks by Hongjun et al. (1996). Rule ranking is another step which is done after rule mining to reduce the number of rules and increase the accuracy of classification. As a review on methods of ranking the rules, we indicate these ones: Hybrid-RuleRank model proposed by Najeeb et al. (2011) which uses Simulated Annealing and Genetic Algorithm, using rough set theory by Li and Cercone(2005) and applying genetic network programming by Yang et al. (2008). Also Toloo et al. (2009) proposed a new method for ranking the rules by Data Envelopment Analysis (DEA). Multi-rule ranking and single-rule ranking algorithms and a comparison between them are proposed by Dreiseitl et al. (2010).

* Corresponding Author. Email Address: Niknafs@uk.ac.ir

In this paper we propose a new approach for extracting and ranking the classification rules from a dataset, on the basis of Deoxyribonucleic Acid (DNA) Computing. DNA computing was first established by Adleman (1994). He solved the Hamiltonian path problem which belongs to the set of NP problems by using DNA molecules and biological operations. His work showed that not only the computation in molecular level is possible, but also it is very efficient due to the parallel processing ability of DNA molecules. In 1995 Adleman introduced Restricted Model (Adleman, 1995) having three operators including *Separate*, *Merge* and *detect* and gave a solution for 3-vertex coloring problem. Lipton (1995) used the Restricted Model and solved the Satisfiability (SAT) problem which is also another NP problem. Ouyang et al. (1997) used DNA computing to solve the Maximal Clique Problem in 1997. DNA computing has also been used in Job Shop Scheduling (Yin et al., 2006), Clustering Problem (Rohani Bintiet al., 2008), Knapsack Problem (Darehmiraki and Mishmast Nehi, 2007;Henkel et al.,2007), Bin Packing Problem (Sanches and Soma, 2009) and Integer Linear Programming (Wang and Yang, 2005).

To the best knowledge of the authors of this article, this is the first time DNA computing is used for mining the classification rules. We show how we can encode all possible classification rules and extract proper ones using biological operations proposed by Adleman and Lipton (Adleman, 1994; Lipton, 1995). Our proposed method searches the whole space of classification rules and selects proper ones in a linear time complexity.

The rest of this paper is organized as follows: in Section 2, the fundamental concepts of DNA computing and also a review on rule mining and ranking are given. In Section 3 and 4, the motivation, problem description and the proposed method are represented, respectively. Also we evaluate our method in terms of time complexity in Section 4. In Section 5, we present the result of simulation of our approach on a simple dataset and finally we conclude the paper and give future works in Section 6.

2. Basic Concepts

In this section, the basic concepts of DNA computing and data mining are represented.

2.1. DNA computing

Deoxyribonucleic Acid (DNA) is a topic of biology field which deals with stem cells of living organisms. DNA is composed of polymer chains which are called DNA strands. Polymer chains are made up of nucleotides. The only difference between the types of nucleotides is in their bases, where the four bases are *Adenine*, *Guanine*, *Cytosine* and *Thymine*, abbreviated as A, G, C, and T respectively. So we can consider a DNA strand as a sequence of A, G, C and T letters. A and T are complements of each other. G and C are complements too. Two DNA strands are said to be "Watson Crick complements" if and only if their bases are complements (Adleman, 1994). As an example, DNA strands of AGGCTG and TCCGAC are Watson crick complements. For more explanation, consider a DNA experiment with a number of DNA molecules in a test tube. If a DNA strand in the test tube meets its Watson-Crick complement, then the two strands will anneal; that is twist around each other to form the famous double helix (Adleman, 1998). DNA molecules can be used for modeling the problems by encoding the information and making computations in test tubes. The computations are performed by using the DNA biological operations. So a new computing method named DNA computing is emerged.

DNA computing is a type of computing which uses DNA molecules for storing information and biological operations for acting on the information stored. The main strength of DNA computing is the vast ability of parallel processing offered by DNA molecules which lets us to create and check all potential solutions to a problem in parallel (Ito and Fukusaki, 2004). Higher information density and lower energy consumption are two more benefits of DNA computing (Adleman, 1998).

Several models have been proposed for DNA computing and among them the Adleman-Lipton model (Adleman, 1994; Lipton, 1995) is of particular interest. The basic operations of this model are as follows:

- **Extract (*T, S, T+, T-*):** Given a tube *T* and a string *S*, this operation produces two tubes *T+* and *T-* as follows: All the DNA strands of *T* having *S* in their sequence are extracted and poured into the tube *T+* and the remained DNA strands are poured into the tube *T-*.
- **LengthSeperate(T,L,T_1,T_2):** Given a tube *T* and a number *L*, this operation picks up the strands not longer than *L* and pours them into the tube T_1. The remained strands are poured into the tube T_2.
- **Merge ($T, T_1, T_2, \ldots, T_n$):** Given tubes T_1 to T_n, this operation pours the contents of T_1 to T_n into the tube *T*, so the tube *T* contains all the DNA strands of T_1 to T_n.
- **Copy ($T, T_1, T_2, \ldots, T_n$):** Given a tube *T*, this operation creates tubes T_1 to T_n which are identical copies of *T*. (T_1 to T_n will have the same DNA strands as *T*.)
- **Append (*T, S*):** Given a tube *T* and a string *S*, this operation appends *S* to the end of all the DNA strands of *T*.
- **Detect(*T*):** Given a tube *T*, this operation returns *true* if there is at least one DNA strand in *T*, otherwise it returns *false*.
- **Discard(*T*)**: Given a tube T, this operation simply ignores T.

2.2. Data mining

Data mining is one of research areas of computer science which aims to extract useful knowledge from a dataset. Association rule mining is one of the most known techniques of data mining. By using this technique all the rules which satisfy a minimum support and confidence are extracted from a dataset. Using association rule mining for classification problems is called classification rule mining. The purpose of Classification Rule Mining is to obtain a set of rules for classifying samples. Suppose a dataset consists of samples in the form $X_1X_2\ldots X_nD$. X_1toX_n are called conditional attributes and *D* is called decision attribute. Moreover conditional attributes get discrete values which could be encoded using integer values of *1* to *k*, if there are *k* values for each attribute. If conditional attributes get continuous values, we can convert them to discrete ones by using one of discretization methods (Augasta and Kathirvalavakumar, 2012;Gonzalez-Abrila et al., 2009; Li et al. ,2011). Decision attribute *D* can take an integer value of *1* to *z* if we suppose there are totally *z* possible decisions for classifying each sample. So the purpose of Classification Rule Mining is to extract rules like *r* in this form: *if* $X_1=V_{r,1}$ *and* $X_2=V_{r,2}$ *... and* $X_n=V_{r,n}$ *then* $D=D_r$; $V_{r,i} \in \{0, 1, 2, \ldots, k\}\ \forall\ i=1,2,\ldots,n$; $D_r \in \{1,2,\ldots,z\}$ and value of zeros for an attribute refers to *"don't care value"*. Also the structure of a sample is as follow: $X_1=V_{s,1}$, $X_2=V_{s,2}$,…, $X_n=V_{s,n}$, $D=D_s$; $V_{s,i} \in \{1, 2, \ldots, k\}\ \forall\ i=1,2,\ldots,n$;$D_s \in \{1,2,\ldots,z\}$.

Definition 1. Sample *s* fires rule *r* if we have: $V_{r,i} = 0$ or ($V_{r,i} \neq 0$ and $V_{s,i} = V_{r,i}$) $\forall$ i=1,2,…,n.

Definition 2. Rule *r* classifies sample *s* correctly if sample *s* fires rule *r* and $D_r=D_s$.
Definition 3. Confidence of rule *r*is the number of samples which are correctly classified by rule *r* divided by the number of samplesthat fire the rule *r*.
Definition 4. Support of rule *r* is the number of samples which are correctly classified by rule *r* divided by the total number of samples.

Support and Confidence values measure the quality of a rule. A big confidence value indicates high accuracy and a big support value shows that the rule covers a big number of samples in the dataset. Maximum value for confidence is one and means that the rule correctly classifies all the samples it covers (the samples which fire the rule). After mining the classification rules, Ranking Methods are used to rank the rules and finally some of the rules are chosen for building the classifier.

3. Motivation and Problem Description

In this paper, our goal is applying DNA computing for extracting classification rules with maximum confidence from a dataset and then ranking them according to their support values. To do so, we can generate all possible rules for a dataset and calculate confidence for each rule, then keep only the rules with confidence equal to one. For ranking, we can sort them according to their support in descending order. Running the mentioned process on silicon based computers leads to an algorithm with exponential time complexity. If *n*, *k* and *z* are the number of attributes, the number of values for each attribute and the number of classes, respectively, then the total number of rules that must be checked for support and confidence is $z(k+1)^n$. So we encounter an NP problem that the required time to solve it grows exponentially as the problem size (number of attributes) increases. Whereas using DNA computing and gaining the vast parallel processing ability of DNA molecules, we can solve the same problem in a linear time complexity proportional to the problem size. In fact the main reason of using DNA molecules for computation is the ability of performing an operation on all DNAs in a tube in parallel. If the tube contains trillions of DNA strands then the power of DNA computing is revealed.

4. Proposed Method

In this section our proposed method for extracting and ranking the classification rules based on Adleman-Lipton DNA computing model is demonstrated. We also evaluate our method in terms of time complexity.

The structure of the dataset is presented in Table 1.

Table 1. Structure of the dataset.

X_1	X_2	X_3	…	X_n	D
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	…	$x_{1,n}$	d_1
$x_{2,1}$	$x_{2,1}$	$x_{2,1}$	…	$x_{2,n}$	d_2
⋮	⋮	⋮	⋯	⋮	⋮
$x_{s,1}$	$x_{s,1}$	$x_{s,1}$	…	$x_{s,n}$	d_s

The total number of samples and attributes is given by s and n respectively. $x_{i,j} \in \{1,2,\dots,k\}$ indicates the value of j^{th} attribute for i^{th} sample. $d_i \in \{1,2,\dots,z\}$ indicates the class of i^{th} sample.To solve the problem by DNA computing, we create a separate DNA strand for each value of an attribute. X_i^j is a DNA strand which encodes the j^{th} value of i^{th}

attribute. Moreover, for each class we have a DNA strand too.D^i is a DNA strand encoding the i^{th} class. If these DNA strands are connected together, then new DNA strands are created that encode the rules. For example the DNA strand$X_1^1 X_2^3 D^2$ which is composed of three separate DNA strands of X_1^1, $X_2^3$and D^2encodes this rule: (*if X_1=1 and X_2=3 then class is 2).* To calculate support and confidence of rules, we need other DNA strands which we call them tags. For each class we designate a unique DNA strand as tag. So, $G^i$is the tag associated to the classD^i. Our method is as follows.First we create all possible rules by concatenating the above DNA strands. Then for each sample like *s*, we select all the rules which are fired by it. Next, we append a tag associated with the class of that sample to the end of the selected rules. These tags help us to find the class of samples that each rule covers and will be useful in calculating the confidence and support of the rules. After checking all samples, few tags are appended to the end of each rule. There may be different situations according to the tags added to the rules. Some rules have not received any tags. The support value of these rules is zero, because they are not fired by any sample. Some other rules have received one or more tags not associated with their own class. For these rules, confidence value is less than one, because they are fired by one or more samples having the class not equal to their own class. So these rules classify those samples wrongly. Finally, there are some other rules that all the tags they have received are associated to their class. Confidence value of these rules equals to one, which means they correctly classify all the samples covered by them. We will extract these rules with maximum confidence values and then rank them according to their support. For determining the support of the rules, it is just enough to look at the number of tags they have received. The more tags a rule receives, the more support it has.

4.1 Proposed algorithm

Our proposed algorithm for extracting and ranking the classification rules has five steps using DNA computing:

Step 1: Generate all possible rules which are encoded by single stranded DNAs.
Step 2: Append tags to the rules according to the samples.
Step 3: Keep all the rules which have received at least one tag of their class.
Step 4: Delete all the rules which have received at least one tag not associated to their class. Therefore the remaining rules have confidence = 1.
Step 5: Sort the rules according to their length. So, the rules are ranked according to their support values.

Now we explain each step of our algorithm. The pseudo code for each step is also shown.

Step 1:
//Generate all possible rules which are encoded by single stranded DNAs
for i=1 to n
  Copy(T_0,T_1,T_2,…,T_{k+1})
  for j=1 to k
    Append(T_j,X_i^j)
  end
  Append(T_{k+1},X_i^0)
  Merge(T_0,T_1,T_2,…,T_{k+1})
end
Copy(T_0,T_1,T_2,…,T_z)

for i=1 to z
Append(T_i,D^i)
end
Merge(T_0,T_1,T_2,…,T_z)

The tube T_0 is initially empty. After execution of step 1, T_0 contains all the possible rules. These rules are encoded by DNA strands with this structure: $X_1^{v_1}X_2^{v_2}\dots X_n^{v_n}D^h$; $v_i \in \{0,1,\dots,k\}$ for i=1,2,…,n and $h \in \{1,2,\dots,z\}$. Remind that the value of zero for an attribute refers to "*don't care value"*.

Step 2:
//Append tags to the rules according to the samples
for i=1 to s
Extract(T_0,$X_1^{x_{i,1}}$,T_1, T_0)
Extract(T_0,X_1^0,T_2,T_0)
Merge(T_3,T_1,T_2)
for j=2 to n
Extract(T_3,$X_j^{x_{i,j}}$,T_1,T_3)
Extract(T_3,X_j^0,T_2,T_3)
Merge(T_0,T_3,T_0)
Merge(T_3,T_1,T_2)
end
Append(T_3,G^{d_i}) /d_i is the class of i^{th} sample and G^{d_i} is the tag associated to its class*/*
Merge(T_0,T_3,T_0)
end

After execution of step 2, each rule has received some tags equal to the number of samples it covers.

Step 3:
//Maintain all the rules which have received at least one tag of their class
for i=1 to z
Extract(T_0,D^i,T_i,T_0)
Extract(T_i,G^i,T_i,T_{temp})
Discard(T_{temp})
end

After execution of step 3, T_1 contains all the rules with class 1 that have received at least one tag of G^1, meaning these rules correctly classify at least one sample. This explanation applies to all other tubes T_2 to T_z.

Step 4:
//Delete all rules which have received any tags not associated to their class
//So the remaining rules have confidence values equal to 1
for i=1 to z
for j=1 to z
if i≠j then

```
            Extract(T_i ,G^j ,T_temp,T_i)
            Discard(T_temp)
        end
    end
end
```

After execution of step 4, tube T_1 contains all the rules having class 1 and all the tags they have received is G^1, meaning these rules have confidence value equal to one. The next step is to rank these rules according to their support.

```
Step 5:
//Sort rules according to their length. So, the rules are ranked according to their support values
for i=1:z
Run gel electrophoresis technique on tube T_i
end
```

Until execution of step 4, T_i contains rules that are encoded as DNA strands of the form shown in Fig. 1.

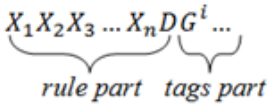

Fig.1. Structure of DNA strands encoding rules.

Each rule has composed of two parts namely the rule part and the tags part. The length of rule part is equal for all rules because each rule has all attributes in its antecedent part. The value of some attributes is "*don't care"*. The role of don't care values is to force the rules with different number of attributes to have the same length. For example consider these two rules: (if X_1=2 then D=3) and (if X_1=1 and X_2=3 then D=2), which are encoded by DNA strands $X_1^2X_2^0X_3^0 \ldots X_n^0D^3$ and $X_1^1X_2^3X_3^0 \ldots X_n^0D^2$respectively. Note that although the number of conditional attributes of these two rules differ, but they have equal length when encoded by DNA strands. So the more tags a rule receives, the more support it has and we just need to use gel electrophoresis technique to sort the DNA strands according to their length. In this way, we have ranked the rules according to their support.

4.2. Evaluation

One measure for evaluation of an algorithm is the time complexity which specifies the execution time of the algorithm as a function of the problem input size. For calculating the time complexity, the number of executions of the main operations is counted. The time complexity for each step of our proposed method is shown in Table 2. Note that in calculation of these time complexities, the number of execution of basic DNA operations like *Append*, *Merge* and *Extract* is counted.

Table 2. Time complexity for each step of the proposed method.

Step of the algorithm	Time complexity
Step 1	O(k×n)

Step 2	O(s×n)
Step 3	O(z)
Step 4	$O(z^2)$
Step 5	O(z)

Remind that *n*, *k* and *z* are the number of attributes, the number of values of each attribute and the number of classes, respectively. The total time complexity of our proposed algorithm is *O(n)*. As it was noted in Section 3, the time complexity of the same algorithm on a silicon based computer is $O(k^n)$. So the efficiency of DNA computing is apparent. We need $k+1+z$ tubes for generating the rules and one tube for combining them (tube T_0 in pseudo codes). We also need three tubes T_1, T_2, T_3 as auxiliary tubes for appending tags to the rules. So the total number of needed tubes is $k+z+5$, independent of the number of attributes.

5. Experimental Results

We have verified the correctness of our proposed approach through extensive simulation results. As a simple example consider the dataset shown in Table 3 which has six samples.

Table 3. A sample dataset for mining and ranking the classification rules using proposed approach.

X_1	X_2	X_3	D
1	1	1	1
1	2	1	1
2	3	2	2
1	3	3	2
1	1	2	1
1	1	3	1

Four of the samples belong to class 1 and two of them to class 2. Due to the lack of space, it is not possible to show the output of each step of our proposed algorithm. Therefore we show the final rules encoded by DNA strands of tubes T_1 and T_2 as shown in Fig. 2.The first DNA strand of T_1 encodes the rule: (if X_2=1 then class=1) and the first DNA strand of T_2 indicates (if X_2=3 then class=2). As can be seen from the dataset of Table 2, these are the rules with confidence=1 and a big support value. Other rules of T_1 and T_2 have confidence=1, but with smaller support values.

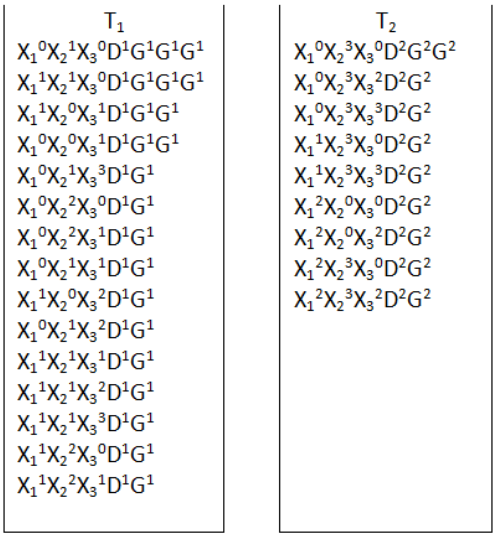

Fig. 2. Final ranked rules for dataset of Table 3.

6. Conclusion and Future Works

The high information density and vast parallel processing power of DNA molecules, makes this possible to use them as a new tool for solving complex problems which need high computational power when using silicon based computers. In this paper we proposed an approach based on DNA computing which is able to extract and rank the classification rules from a dataset. Extracting and ranking the rules is done using the support and confidence measures. The advantage of our method is its low time complexity in spite of searching the whole space of the problem. Our proposed method searches the whole space of the rules and filters out inappropriate ones and finally ranks the remained rules according to their support. The time complexity of our method is *O(n)*.

The rules obtained by our approach can be combined to make a final classifier. This is the key point for future research. We plan to apply the rules obtained by our approach for classifying some benchmark datasets and comparing the results with other classification methods.

References

L.M. Adleman, “Molecular computation of solutions to combinatorial problems,” Science,vol. 266, pp. 1021-1024, 1994.

L.M. Adleman, “On constructing a molecular computer,” 1995.

L.M. Adleman, “Computing with DNA,” Scientific American,pp. 54-61, 1998.

C. Chiu, P. Hscu, “A constraint-based genetic algorithm approach for mining classification rules,” Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 35, pp. 205-220, 2005.

M. Darehmiraki, H. Mishmast Nehi, “A surface-based DNA algorithm for the solving binary knapsack problem,” Applied Mathematics and Computation, vol. 188,pp. 1991-1994, 2007.

S. Dreiseitl, M. Osl, C. Baumgartner, S. Vinterbo, “An evaluation of heuristics for rule ranking,” Artificial Intelligence in Medicine, vol. 50, pp. 175-180, 2010.

L. Gonzalez-Abrila, F.J. Cuberosb, F. Velascoa, J.A. Ortega, "Ameva: An autonomous discretization algorithm," Expert Systems with Applications, vol. 36, pp. 5327-5332, 2009.

C.V. Henkel, T. Bäck, J.N. Kok, G. Rozenberg, H.P. Spaink, "DNA computing of solutions to knapsack problems," Biosystems, vol. 88, pp. 156-162, 2007.

L. Hongjun, R. Setiono, L. Huan, "Effective data mining using neural networks," Knowledge and Data Engineering, proceeding of dexa conference 2012, paris, vol. 8, pp. 957-961, 1996.

Y. Ito, E. Fukusaki, "DNA as a 'Nonmaterial'," Journal of Molecular Catalysis B: Enzymatic in Combinatorial Bioengineering - Development of Molecular Evolution, vol. 28, pp. 155-166, 2004.

J. Li, N. Cercone, "Discovering and ranking important rules," Granular Computing, 2005 IEEE International Conference on, pp. 506-511, 2005.

M. Li, S. Deng, S. Feng, J. Fan, "An effective discretization based on Class-Attribute Coherence Maximization," Pattern Recognition Letters, vol. 32, pp. 1962-1973, 2011.

R. J. Lipton, "DNA solution of hard computational problems," Science, vol. 268, pp. 542-545, 1995.

D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck, B. Baesens, "Classification With Ant Colony Optimization," Evolutionary Computation, vol. 11, pp. 651-665, 2007.

M.M. Najeeb, A.E. Sheikh, M. Nababteh, "A new rule ranking model for Associative Classification using a hybrid Artificial Intelligence technique," Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on, pp. 231-235, 2011.

Q. Ouyang, P.D. Kaplan, S. Liu, A. Libchaber, "DNA solution of the maximal clique problem," Science,vol. 278, pp. 446-449, 1997.

A. Rohani Binti, W. Junzo, P. Witold, "DNA approach to solve clustering problem based on a mutual order," Biosystems, vol. 91, pp. 1-12, 2008.

C.A.A. Sanches, N.Y. Soma, "A polynomial-time DNA computing solution for the Bin-Packing Problem," Applied Mathematics and Computation, vol. 215, pp. 2055-2062, 2009.

M. Toloo, B. Sohrabi, S. Nalchigar, "A new method for ranking discovered rules from data mining by DEA," Expert Systems with Applications, vol. 36, pp. 8503-8508, 2009.

S. Wang, A. Yang, "DNA solution of integer linear programming," Applied Mathematics and Computation,vol. 170, pp. 626-632, 2005.

G. Yang, K. Shimada, S. Mabu, K. Hirasawa, "A personalized association rule ranking method based on semantic similarity and evolutionary computation," IEEE World Congress on Computational Intelligence, pp. 487-494, 2008.

Z. Yin, J. Cui, Y. Yang, Y. Ma, "Job shop scheduling problem based on DNA computing," Journal of Systems Engineering and Electronics, vol. 17, pp. 654-659, 2006.

C. Zhou, W. Xiao, "Evolving accurate and compact classification rules with gene expression programming," Evolutionary Computation, vol. 7, pp. 519-531, 2003.