FMGA: Finding Motifs by Genetic Algorithm

Falcon F.M Liu[1], Jeffrey J.P. Tsai[1,2], R.M Chen[3*], S.N. Chen[4] and S.H. Shih[3]
[1]*Institute of Bioinformatics, Taichung Healthcare and Management University, Taiwan*
[2]*Department of Computer Science, University of Illinois at Chicago, USA*
[3]*Department of Information and Design, Taichung Healthcare and Management University*
[4]*Department of Information Technology, Taichung Healthcare and Management University*
[*]*E-mail: rmchen@thmu.edu.tw*

Abstract

In the era of post-genomics, almost all the genes have been sequenced and enormous amounts of data have been generated. Hence, to mine useful information from these data is a very important topic. In this paper we propose a new approach for finding potential motifs in the regions located from the -2000 bp upstream to +1000 bp downstream of transcription start site (TSS). This new approach is developed based on the genetic algorithm (GA). The mutation in the GA is performed by using position weight matrices to reserve the completely conserved positions. The crossover is implemented with special-designed gap penalties to produce the optimal child pattern. We also present a rearrangement method based on position weight matrices to avoid the presence of a very stable local minimum, which may make it quite difficult for the other operators to generate the optimal pattern. Our approach shows superior results by comparing with Multiple Em for Motif Elicitation (MEME) and Gibbs Sampler, which are two popular algorithms for finding motifs.

1. Introduction

Consensus sequences had been widely used in molecular biology. A consensus sequence is defined as the sequence that reflects the most frequent base or amino acid at some position in a set of aligned DNA, RNA or protein sequences such as binding sites. Consensus sequences often represent conserved functional domains. There is an urgent need to extract potential consensus sequences (motifs) from the upstream and downstream regions of transcription start site (TSS) [1]-[2]. Current researches on finding consensus sequences include: predicting motifs on the promoter, finding consensus sequences in the introns, detection of conserved secondary structures and finding domains on proteins.

During the process of transcription from DNA to mRNA, RNA polymerase II will bind on the promoter regions. How does the RNA polymerase II identify where to bind on? One of the methods is that when TATA-binding protein (TBP) factor finds the TATA box at the promoter, it will bind on the DNA core promoter. Then the other factors will automatically continue to bind on the promoter to start the transcription. In the other case the Sp1 factor of RNA polymerase II will identify GC box to implement the DNA transcription. Besides TBP and Sp1 factors, there are also other factors to aid the DNA transcription. Many of the regulatory elements serving as transcription factor targets are located in the upstream of the core promoter [3]-[5]. The transcription factor binding site is conserved region (consensus sequence) and is called motif. Currently, the well-known transcription factor binding sites are TATA box [TATAAA], BRE [(G/C)(G/C)(G/A)CGCC], Inr [TCA(G/T)T(T/C)] and DPE [(A/G)G(A/T)(C/T)(G/A/C)], etc. However, these transcription factor binding sites do not occur at all promoters. For instance, in Drosophila it was estimated that near 29% have only a TATA box but no DPE; 26% contain only a DPE but no TATA box; 14% possess both TATA and DPE motifs; and 31% do not appear to have either a TATA box or a DPE [6]-[7].

There are many methods proposed to predict motifs in the past. The most popular methods include Multiple Em for Motif Elicitation (MEME) [8], Gibbs sampler [9], Genetic Algorithm (GA) [10], Artificial Neural Network (ANN) and simulated annealing algorithm [11]. Among these methods, Gibbs sampler has the advantage of spending lower computation time. MEME is superior to the other methods by its prediction accuracy, but has the drawback of taking enormous computation time. In this paper, we propose a new approach based on genetic algorithm to predict motifs. The predicted results obtained by using our approach are more accurate than that of Gibbs sampler and spend less computation time than MEME.

This paper is organized as follows. In section 2 we

Proceedings of the Fourth IEEE Symposium on Bioinformatics and Bioengineering (BIBE'04)
0-7695-2173-8/04 $ 20.00 © 2004 IEEE

present the new approach for predicting motifs and a distributed parallel computing framework is proposed to implement the new method. In section 3 we show the prediction results by using the new method and compared with MEME and Gibbs sampler. Finally, we have the concluding remarks in section 4.

2. Method

Our method to predict motifs is to use a total fitness score function and to find the optimal motif using genetic algorithm. We also use the general genetic algorithm framework and operators to serve as our basic architecture as sequence alignment by genetics algorithm (SAGA) did [12]. In the new algorithm, we also include ambiguity codes penalties to charge the scores of consensus sequences so that optimal motifs can be predicted more efficiently.

2.1. Motif pattern

Motif is a conserved pattern that is common in two or more sequences and can be found in DNA, RNA and protein. The general concept of finding motifs is based on the occurrence frequent of a potential region in every sequence. In fact, it is quite difficult for all sequences to have a completely matched pattern, especially for consensus sequences with number of base pairs exceeds 20. Hence, we have to allow mismatched nucleotides to some degree of extent. For example, if we allow a 20% mismatched nucleotides for motif pattern [ACCGGC], it is allowed for 1 nucleotide mismatched (6 x 20% = 1.2).

2.2. Fitness score function

First, we consider the computation of fitness score for one single sequence. Given a motif pattern, there may have several regions in the sequence that match the motif pattern and each has a fitness score according to the fitness score function defined as follows:

$$FS(S_m, P_n) = \max_j \left\{ \sum_{i=1}^{k} match(S_{mji}, P_{ni}) / k \right\}$$

where

$$match(S_{mji}, P_{ni}) = \begin{cases} 1 & if \quad S_{mji} = P_{ni} \\ 0 & if \quad S_{mji} \neq P_{ni} \end{cases}$$

and m is the index of sequences, i is the position within the motif , n is the index of motif patterns, k is the length of motif pattern, j is number of matched regions in the sequence. For example, suppose the motif pattern P_1 and promoter sequence S_1 are as follows:

P_1: ACCGTA
*Promoter S_1: A*TCCGGC*TA*ACCGTA*CTATATTA*
Fitness score: 3/6=0.5 6/6=1
$FS(S_1,P_1)$ = 1

Besides the four nucleotides A, T, C and G, we also include the IUPAC ambiguity codes into motif patterns [13] such that more versatile motif patterns can be considered. The codes V, H, B and D are grouped into N. Then we modify the match function of the fitness score function as follows:

$$match(S_{mji}, P_{ni}) = \begin{cases} 1 & if \quad S_{mji} = P_{ni} \quad for \quad P_{ni} \in \{A,T,C,G\} \\ 0.5 & if \quad S_{mji} = P_{ni} \quad for \quad P_{ni} \in \{M,R,W,S,Y,K\} \\ 0 & if \quad S_{mji} \neq P_{ni} \end{cases}$$

For example, suppose the motif pattern P_1 and promoter sequence S_1 are as follows:

P_1: MCSGTA
*Promoter S_1: T*TCCGTA*TA*ACCGTA* *ACCTTA*
Fitness score: 4.5/6 5/6 4/6
$FS(S_1,P_1)$ = 5/6

2.3. Total fitness score function

The total fitness score function of a motif pattern is the summation of fitness score function for all sequences. It represents the score of a motif pattern in each generation of the genetic algorithm. We define the total fitness score function as follows.

$$TFS(S, P_n) = \sum_{m=1}^{L} FS(S_m, P_n)$$

where L is the total number of sequences. For example, suppose the motif pattern P_1 with 75% nucleotides match (allow 1 base pair unmatched), and promoter sequences $S_1 \sim S_5$ are as follows:

P_1: AGGAGGR
*S_1: GGAGGAAGGA*AGGAAGG*AAGGAAGGAA=5.5/7*
*S_2: ACGAGGGACC*AGGATGG*CACCGCG=5.5/7*
*S_3: GAGCTCA*AGGAGAA*GATCGAGGAGATT=5.5/7*
*S_4: AAGTGCTATT*AGGAGGA*GAAAATCAAG=6.5/7*
*S_5: AGGCTGAGGC*AGGAGGA*TTGCTTAAGG=6.5/7*
$TFS(S,P_1)$=4.21

The TFS will also be charged for ambiguity codes opening and extension penalties. The details are stated in Section 2.5.

2.4. Finding motif by genetic algorithm

We propose the finding motif by genetic algorithm (FMGA) in this section. The architecture is shown in Figure 1. The data is selected in the regions located from the -2000 bp upstream to +1000 bp downstream of transcription start site (TSS). Pseudo codes of the FMGA are stated as follows:

Initialization
Setting total number of iterations: M
Creating candidate motifs randomly: $P_1 \sim P_n$
Import promoter sequences $S_1 \sim S_L$

While(iteration number <= M)
{
While(predicted motif is unchanged for more than K generations)
{
Evaluating TFS for each candidate motif
Keeping the candidate motifs with the highest TFS as the new generations, the remaining candidate motifs are created by weighted wheel selection
Mutation using weight matrix to generate two parent patterns
Crossover with ambiguity codes penalties to select the best child pattern for next generation
}
Rearrangement of candidate motifs
Increasing iteration number by 1
}

Output predicted motifs and corresponding TFS

In this algorithm, initial motif patterns with the same pattern length are created randomly. The users can set the pattern length. All the motif patterns will be different from the initial patterns after *N*-generation evolution. The mutation with weighted matrix can speed up the time to find potential motif patterns. The purpose of crossover is to select the best child pattern after mutation. Charge for ambiguity codes penalties on the IUPAC ambiguity codes is to avoid that the predicted motif patterns contain too many ambiguity bases. Rearrangement of motif patterns is to overcome the problem of local maximum. This algorithm may find multiple potential motif patterns if these patterns have the same scores.

2.5. Operators

The essential operators in FMGA are mutation, crossover and rearrangement that can speed up the FMGA to find potential motif patterns. We introduce these operators as follows.

2.5.1. Mutation

We first create a weight matrix from the matched motif patterns in every sequence. An illustration for creating weight matrix is shown in Table 1 for five matched motif patterns TGACGCA, TGACGCA, AGACGCA, TGACACA and AGACGCA. The score in weight matrix is calculated as the ratio of occurrences of corresponding base and the numbers of matched motif patterns. For example, in column 1 of the table, the numbers of occurrences of A are 2, the numbers of matched motif patterns are 5, the score is equal to 2/5 = 0.4. We keep the bases in columns with score = 1 unchanged, and the bases in other columns are changed randomly (using IUPAC ambiguity codes). Every pattern will generate two parent patterns after mutation.

2.5.2. Crossover

The purpose of crossover is to select a better child pattern from two parent patterns generated by mutation. Two parent patterns are cut at a random position. Combining the left side of parent pattern 1 and the right side of parent pattern 2 produce the first child pattern. Combining the right side of parent pattern 1 and the left side of parent pattern 2 produce the second child pattern. We charge with ambiguity codes penalties for nucleotides that are not A, T, C and G. In our algorithm we consider the IUPAC ambiguity codes as gaps such that the predicted motif patterns will be more coincidental to biology meanings. The code N is charged for the highest score among all ambiguity codes.

We show an example for calculating score of child pattern with ambiguity codes penalties [14] in the following. Each ambiguity code opening with code W, M, R, S, Y or K is charged for a score of 0.5, extension penalties are charged with a score of 0.5 for code N and 0.3 for ambiguity code W, M, R, S, Y or K.

Child pattern: AGWWGAGGNR
*Ambiguity codes penalties: Ambiguity codes opening (0.5*2)+Extension penalties ((0.3*3)+(0.5*1))=2.4*
Score charged with ambiguity codes penalties: 7.5-2.4=5.1

The child pattern with higher score will be selected as the motif pattern in the next generation. The scores to be charged for gap penalties can be set in our system by the users.

2.5.3. Rearrangement

If the predicted motif pattern is unchanged for more than *K* generations (e.g., *K* = 10) in the FMGA, we rearrange the motif pattern by converting the ambiguity codes to A, T, C and G according to the weight matrix. The conversion is to replace the ambiguity codes by the bases with the highest score in the corresponding column of the weight matrix (see Table 2).

2.6 Implementation

The FMGA is implemented in *Java* language and implemented on a JVM framework. The *Java* RMI is used to develop a distributed parallel computing

framework to overcome the problem of enormous computations in GA [15]. A distributed parallel computing framework for FMGA is shown in Figure 2. In our system the computation of TFS is shared to slave PCs. The operation system (OS) of server and slave PCs can be Windows or UNIX due to the cross-platform characteristic of *Java*.

3. Results

The data source is obtained from TSSDB website [16]. The test sequences are located from -2000 bp upstream to +1000 bp downstream of transcription start site. Three groups of genes are used to test our system. Three sets of genes are as follows.

Group1: E2F1, E2F2, E2F3, E2F4, E2F5, E2F6
Group2: TGFB1 (transforming growth factor, beta 1), TGFB1I1, TGFB2, TGFB3, TGFBI, TGFBR1, TGFBR2, TGFBR3, TGFBRAP1
Group3: (Tumor Suppressor Genes) BRCA1, BRCA2, CDKN1A, CDKN2A, CDKN2B, CDKN2C, CDKN2D, E-cadherin (CDH1), E2F1, FABP3, FHIT, M6PR, IGF2R, NF2, NME1, PTEN, TGFB1, TP53

We set the lengths of motif patterns to be 7 and 13 to test the system of FMGA. The predicted motif patterns are shown in Table 3. From the results, it is found that the shorter of the motif length, the easier to find the optimal motif. In this table, TFS(%) is defined as TFS divided by the TFS with completely matched, i.e., if the length of motif is 7 and the number of sequences is 6, the TFS with completely matched is 7 x 6 = 42. Hence, the TFS(%) = 1 if the motif is complete matched in all sequences. In the last column of Table 3, we check the ratio of number of matched sequences when the matched condition is increased from 84% to 90% (more strictly). We can confirm whether the predicted optimal motif is the fittest motif or not?

To evaluate the effectiveness of FMGA, we compare our results with two popular motif-finding algorithms: MEME and Gibbs sampler. We send the three groups of data to Motif Elucidator in Nucleotide Sequence Assembly (MELINA) [17] to predict the motifs. The predicted results are listed in Table 4 to Table 6 for comparisons. In Table 4, we may find that FMGA is superior to the other two algorithms in that FMGA finds two completely matched motifs while the MEME finds only one in the case of 7 bases. In this case, the result obtained by Gibbs sampler is not good. In Tables 5 and 6, it is apparent that FMGA have better prediction results than the others.

From the experiments, it can be found that if the length of motif is longer, it is more difficult to locate completely matched motif in all sequences. Hence, we have to allow higher percentage of mismatch to locate the optimal motif.

As for the computation time, FMGA is faster than MEME and slower than Gibbs sampler. In fact, the GA takes enormous computation time for seeking a better solution. Hence, we use the distributed parallel computing framework to compensate this drawback. It is believed that a better result is more important than a rapid computation but with worse result.

4. Concluding remarks

From the results presented above, the performance of FMGA is better than the two popular algorithms MEME and Gibbs sampler. We had shown that FMGA predicts better motif patterns than Gibbs sampler and spends less computation time than MEME. Moreover, FMGA can predict more potential motifs than the other algorithms because the patterns are generated randomly during the operation processes of GA. This characteristic is used to overcome the possible problems of local minimum.

In the case of motif pattern with length 13, 90% match implies that one nucleotide is allowed for mismatch and 84% match is allowed for 2 nucleotides to be mismatched. Hence, we first predict the motifs under the condition of 84% match and then verify their fitness under a more strict condition of 90% match. For this point, FMGA has also been shown to have more advantages than MEME and Gibbs sampler. From Tables 4 to 6, the discrepancy between the cases of 84% match and 90% match is smaller in FMGA than that in MEME and Gibbs sampler. Furthermore, from several times of experiments with various motif lengths, we see that it is easier to find out motifs with higher fitness for a pattern length of 7 to 10.

Genetic algorithm solves the optimal problem based on the biological characteristics. It uses a simple way to cope with complex problems. In this paper we had proposed a new approach to predict motifs based on the genetic algorithm. The FMGA is implemented under a distributed parallel computing architecture. A lot of biological messages are hidden in promoter, and motif is one of the important messages. The motifs have the possibilities to be the binding sites of transcription factors. If the motifs can be predicted accurately, the biologists can then explore which transcription factors activate genes? In the future, we will extend the FMGA to the applications of finding domains on proteins.

References

[1] G. D. Stormo, "DNA binding sites: representation and discovery," *Bioinformatics*, 16, pp. 16-23, 2000.
[2] Diego di Bernardo1, Thomas Down and Tim

Hubbard, “ddbRNA: detection of conserved secondary structures in multiple alignments,” *J. Bioinformatics*, 19, pp. 1606-1611, 2003.
[3] Stephen T. Smale and James T. Kadonaga, “The RNA Polymerase II Core Promoter,”Annu. Rev. Biochem, 72, pp. 449-479, 2003.
[4] Thomas D. Schneider, “Consensus Sequence Zen,” Applied Bioinformatics, 1(3), pp. 111-119, 2002.
[5] Uwe Ohler, Computational promoter recognition in eukaryotic genomic DNA. Ph.D. Thesis, Technische Fakultat der Unicersitat Erlangen-Nurnderg, 2001.
[6] Jennifer E.F. Butler and James T. Kadonaga, “The RNA polymerase II core promoter: a key component in the regulation of gene expression,” Proc. Genes & Development, 16, pp. 2583-2592, 2002.
[7] Alan K. Kutach and James T. Kadonaga, “The downstream promoter element DPE appears to be as widely used as the TATA box in Drosophila core promoters,” Molecular and Cellular Biology, 20 (13), pp. 4754-4764, 2000.
[8] Timothy L. Bailey and Charles Elkan, “Fitting a mixture model by expectation maximization to discover motifs in biopolymers,”Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology, AAAI Press, Menlo Park, California, pp. 28-36, 1994.
[9] William Thompson, Eric C. Rouchka and Charles E. Lawrence, “Gibbs Recursive Sampler: Finding transcription factor binding sites,” J. Nucleic Acids Research, 31, pp. 3580-3585, 2003.
[10] N. W. Lo, S. W. Changchien, Y. F. Chang and T. C. Lu, “Human promoter prediction based on sorted consensus sequence patterns by genetic algorithms,” Proceedings of the International Congress on Biological and Medical Engineering, D3I-1540: pp 111-112, 2002.
[11] Jonathan M. Keith et al., “A simulated annealing algorithm for finding consensus sequences,” J. Bioinformatics, 18, pp. 1494-1499, 2002.
[12] Cedric Notredame and Desmond G. Higgins, “SAGA: Sequence alignment by genetic algorithm,” J. Nucleic Acids Research, 24, pp. 1515-1524, 1996.
[13] M. Scherf, A. Klingenhoff, T. Werner, “Highly Specific Localization of Promoter Regions in Large Genomic Sequences by PromoterInspector: A Novel Context Analysis Approach,” J. Mol. Biol. 297 (3), pp. 599-606, 2000.
[14] Sadiq M. Sait, Habib Youssef. Iterative computer algorithms with applications in engineering :solving combinatorial optimization problems. Los Alamitos, Calif.: IEEE Computer Society, c1999.
[15] Jim Farley. Java distributed computing. Sebastopol. CA : O'Reilly, 1998.
[16] DBTSS: Database of Transcriptional Start Sites, http://dbtss.hgc.jp/index.html.
[17] Natalia Poluliakh, Toshihisa Takagi and Kenta Nakai, “MELINA: motif extraction from promoter regions of potentially co-regulated genes,” Proc. Bioinformatics, 19, pp. 423-424, 2003.

Table 1. Illustration of score weight matrix

TGACGCA
TGACGCA
AGACGCA
TGACACA
AGACGCA

	1	2	3	4	5	6	7
A	0.4	0	1	0	0.2	0	1
T	0.6	0	0	0	0	0	0
G	0	1	0	0	0.8	0	0
C	0	0	0	1	0	1	0

Table 2. Illustration for rearrangement of motif pattern

TGACGCA
TGACGCA
AGACGCA
TGACACA
AGACGCA
||
WGACRCA

	1	2	3	4	5	6	7
A	0.4	0	1	0	0.2	0	1
T	0.6	0	0	0	0	0	0
G	0	1	0	0	0.8	0	0
C	0	0	0	1	0	1	0

→ TGACGCA

Table 3. Predicted motifs using FMGA

Test Case	Motif Length	Predicted Motif	TFS (84% match)	TFS(%) (84% match)	Matched Seqs# /Total Seqs# (90% match)
Group1	7	AGGAGGA	42	1	6/6
	7	AAAAAAA	42	1	6/6
	13	AAAAAAAAAAAAA	73	0.9358974	4/6
Group2	7	CCCTCCT	63	1	9/9
	13	GGCCGGGCGCGGG	71	0.6068377	5/9
	13	GCGGGGCGCGGGG	71	0.60683763	4/9
Group3	7	AAAAAAA	121	0.8444444	13/18
	7	GCTGAGG	122	0.96825385	14/18
	13	GGGAGGCGGAGGC	152.9	0.6538461	10/18
	13	AAAAAACAAAAAA	151	0.64529914	7/18

Table 4. Comparisons of predicted motifs for MEME, Gibbs sampler and FMGA (Group 1)

Algorithm	Motif Length	Predicted Motifs	TFS (84% match)	TFS(%) (84% match)	Matched Seqs# /Total Seqs# (84% match)	Matched Seqs# /Total Seqs# (90% match)
FMGA	7	AAAAAAA	42	1	6/6	6/6
		AGGAGGA	42	1	6/6	6/6
MEME	7	AAAAAAA	42	1	6/6	6/6
Gibbs Sampler	7	TTTCTTT	38	0.7916667	5/6	3/6
FMGA	13	AAAAAAAAAAAAA	73.0	0.9358974	6/6	4/6
MEME	13	AAAAAAAAAAAAA	73.0	0.9358974	6/6	4/6
Gibbs Sampler	13	AAAAAAAAAAAAA	73.0	0.9358974	6/6	4/6

Table 5. Comparisons of predicted motifs for MEME, Gibbs sampler and FMGA (Group 2)

Algorithm	Motif Length	Predicted Motifs	TFS (84% match)	TFS(%) (84% match)	Matched Seqs# /Total Seqs# (84% match)	Matched Seqs# /Total Seqs# (90% match)
FMGA	7	GGGCTGG	62.0	0.984	9/9	8/9
		CCCTCCT	63.0	1	9/9	9/9
MEME	7	TTTTTTT	58	0.920635	9/9	5/9
Gibbs Sampler	7	GGAGGGA	55.0	0.8730159	8/9	7/9
FMGA	13	GGCCGGGCGCGGG	71.0	0.6068377	6/9	5/9
		GCGGGGCGCGGGG	71.0	0.6068377	6/9	4/9
MEME	13	TTTATTTTATTTT	59.0	0.504273	5/9	2/9
Gibbs Sampler	13	TTTTTTWTTTTTW	48.8	0.417094	5/9	4/9

Table 6. Comparisons of predicted motifs for MEME, Gibbs sampler and FMGA (Group 3)

Algorithm	Motif Length	Predicted Motifs	TFS (84% match)	TFS(%) (84% match)	Matched Seqs# /Total Seqs# (84% match)	Matched Seqs# /Total Seqs# (90% match)
FMGA	7	AAAAAAA	121.0	0.9603174	18/18	13/18
		GCTGAGG	122.0	0.9682539	18/18	14/18
MEME	7	AAAAAAA	121.0	0.9603174	18/18	13/18
Gibbs Sampler	7	TTTTTTT	114.0	0.90476185	17/18	12/18
FMGA	13	GGGAGGCGGAGGC	153.0	0.6538461	13/18	10/18
		AAAAAACAAAAAA	151.0	0.64529914	13/18	7/18
MEME	13	AAAAAAAAAAAAA	151.0	0.6452992	13/18	5/18
Gibbs Sampler	13	GTCTCAAAAAAAA	85.61539	0.36587772	9/18	8/18

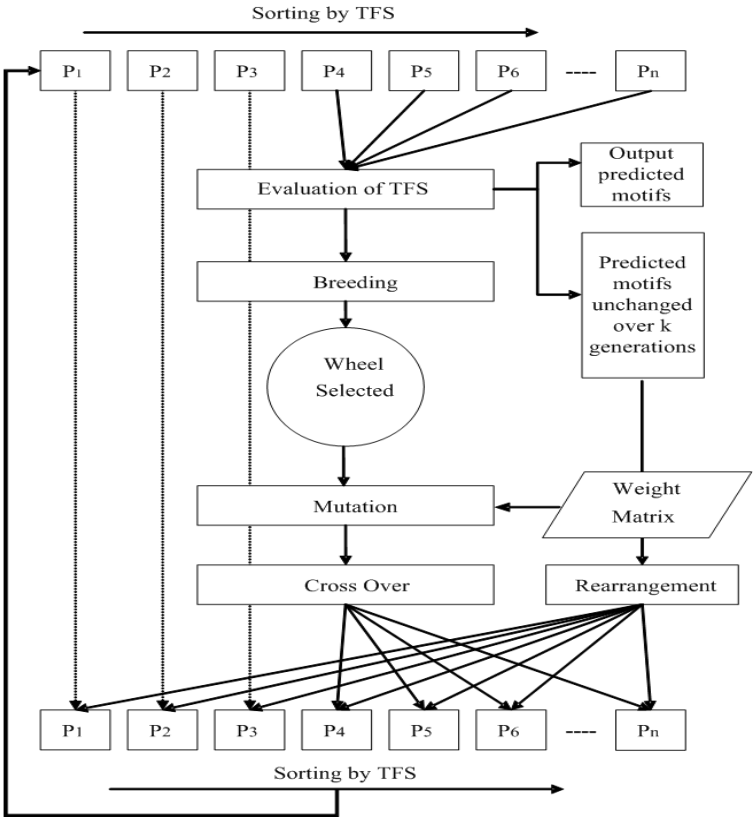

Figure 1. The architecture of finding motifs by genetic algorithm (FMGA)

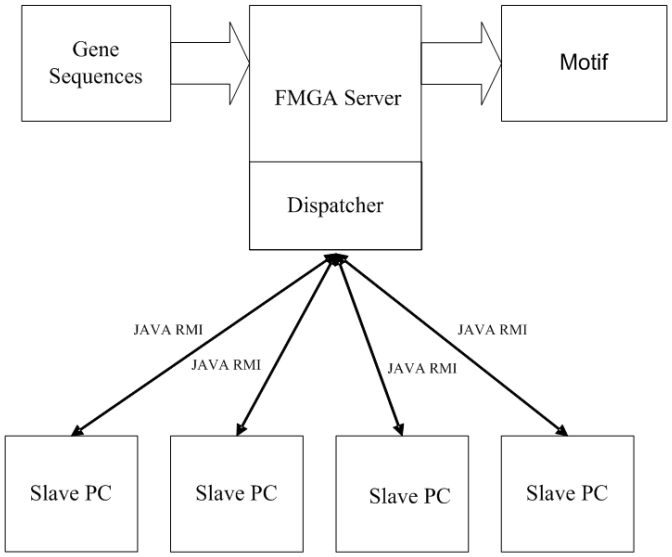

Figure 2. A distributed parallel computing framework for FMGA