TSGA-MSA: Trace Sequence Algorithm for Alignment of MSA

Ruchi Gupta
Department of Computer Science, Research Scholar, Sharda University, Grater Noida, India

Pankaj Agarwal
Department of Computer Science,Faculty of Computing, U.P Technical University Lucknow , India

A.K Soni
Department of Computer Science, Faculty of Computing, Sharda University Grater Noida, India

ABSTRACT
Multiple sequence alignment (MSA) is an NP-complete and important problem in bioinformatics. In this paper, we have proposed iterative alignment method using a Genetic Algorithm for Multiple Sequence Alignment, named TSGA-MSA. The steps in this algorithm are discussed in details and its performances on a set of benchmark datasets from the BAliBase 2.0 are analysed. The experimental results, the effects of the initial generation and genetic operators on the performance of this algorithm, the parameter settings, and a comparison of results with other well-known algorithm are presented and discussed.

Keywords
Genetic algorithm, Multiple Sequence Alignment, DNA etc.

1. INTRODUCTION
We start by introducing some existing methods for solving the problem of MSA using genetic algorithm then the notion of conventional traces as representation of the relation between two sequences.

Then we define scoring schemes based on both, the comparison of two characters of the sequences and the comparison of whole segments of the sequences. The evolutionary genetic model of TSGA-MSA for generating the best scorer alignment is shown in Figure 1 and the process is described below.

2. RELATED STUDY
Genetic algorithm is one of the useful tools determining alignment of multiple sequences. Iterative methods may be implemented through evolutionary approach that use computational heuristics based on natural biological phenomena such as selection, crossover and mutation to evolve a population of candidate solutions based on an objective function because they work similarly to progressive methods but repeatedly realign the initial sequences as well as adding new sequences to the growing MSA [1].

In the following the main characteristics of various existing evolutionary algorithms are presented.

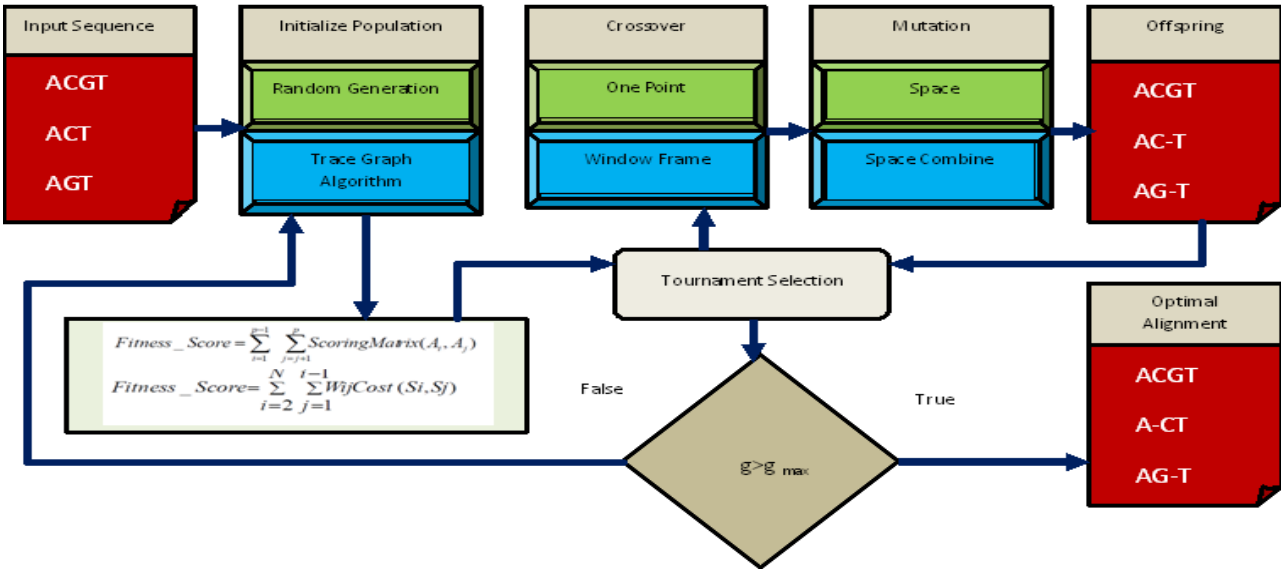

Fig 1: Evolutionary Genetic Model

These algorithms differ in many features like the chromosome representation of the multiple alignment, the used fitness function and the applied operators.The first evolutionary algorithm using an adequate set of problem specific crossover

and mutation operators called *SAGA* [2] proposed by C. Notredame and D.G. Higgins in 1996. SAGA is a straightforward implementation of a genetic algorithm that have been successfully applied to the MSA problems, which tried to optimize a weighted sum-of-pairs function with natural or quasi-natural affine gap penalties. It is used to optimize two different objective functions and shows that they can search large solution space efficiently. But due to repeated use of fitness function it may increase its time complexity.

Based on *SAGA*, two other evolutionary algorithms have been developed: *PGA* [3], which is a parallel implementation of SAGA, and *SAGA-COFFEE* [4], which tries to optimize a consistency based objective function: Let W*i;j* being the percent identity between two aligned sequences, C*i;j* being the number of aligned characters, that are shared between the pairwise alignment and a library containing all pairwise alignments, and L*i;j* being the length of the pairwise alignment. Then the consistency based objective function of SAGA-COFFEE is computed as follows (with *n* being the number of sequences):

$$COFFEE = \left(\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} W_{i,j} \cdot C_{i,j} \right) / \left(\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} W_{i,j} \cdot L_{i,j} \right)$$

Further, GA based algorithms were among the some of the most effective approaches used to solve the MSA problem. In [5], a combination of a GA and DP is used with two different distance matrices. The main drawback of this technique is its limitation in performing crossover and mutation operations. In [6], a GA approach is proposed with a description of the so-called **Center Star Algorithm (CSA).** In addition to this algorithm's convergence problems, forcing the GA to work around the CSA and the initial population creates a major disadvantage for this approach. It leads to the inability of the main search algorithm to explore all parts of the solution space. In [7], a very different GA approach is presented. In this work, five mutation operators are designed to be randomly selected in each cycle of the algorithm. Here, no particular optimization plan is used; therefore, this greedy algorithm just moves toward any potential answer. One of the most appropriate GA approaches to solve the MSA problem is presented in [8]. Although, the authors carefully define their chromosome, crossover and mutation operators, the definition of their scoring function appears to be their main drawback. In [9] a very interesting **divide-and conquer GA based approaches** is presented. Here, the sequences are divided into smaller sequences and then they are aligned by a GA. If these partial alignments generate better results, they would be replaced by the original ones. Although this approach managed to significantly reduce the simulation time, there is no guarantee that the aggregation of these partially optimal strings ends up with the global minimum and/or a reasonable alignment.

In [10], the authors present a very simple implementation of the GA. In this work, the GA's convergence speed is significantly reduced by the simplicity of the algorithm. The fact that this GA approach discards many offspring is the main reason for its slow convergence. In [11], the convergence speed of a GA is increased by combining it with a Simulated Annealing algorithm.

The GA in [12] uses quantum mechanics concepts by employing a binary matrix to represent only four chromosomes that are used to solve the problem. In each GA cycle, the best three solutions are directly copied to the next generation and only one of them (the worst one) is selected for the GA operations. The proposed GA is significantly biased toward good answers, which strongly prevents it from exploring other parts of the solution space. Authors of the research in [13] present a GA based approach to find the optimal cut-off-points to divide the large sequences to several smaller ones. Each of these smaller sequences is solved by an Ant-Colony approach.

The limited use of the GA just to find the cut-offpoints is quite time consuming in this approach.

After that effective GARS approach by Yang Chen, Jinglu Hu [14] based on Genetic Algorithm with Reverse Selection was proposed. But it suffers from premature convergence in which solution reaches locally at an optimal stage. Furthermore a new approach AlineaGA [15] was proposed which used a Genetic Algorithm with local search optimization embedded on its mutation operators for performing multiple sequence alignment. But its mutation probability leads to better solutions in fewer generations and that the mutation operators had a dramatic effect in this particular domain. Recently Amouda Nizam and Jeyakodi Ravi developed new Cyclic Genetic Approach (CGA) [16] with the complete knowledge of the problem and its parameters. In CGA, the values of various parameters are decided based on the problem and fitness value obtained in each generation. But the column score value varies for each execution may not give relatively better alignment. Zahra Narimani [17] proposed a new algorithm that uses a new way of population initialization and simple mutation and recombination operators. The strength of the proposed GA is using simple mutation operators and also a special recombination operator that does not have problems of similar recombination operators in other GAs. The experimental results show that the proposed algorithm is capable of finding good MSAs in contrast to existing methods, while it uses simple operators with low computational complexity. Another multiobjective algorithm [18], based on the non-dominated sorting genetic algorithm, aims to jointly optimize three objectives: STRIKE score, non-gaps percentage and totally conserved columns. It was significantly assessed on the BAliBASE benchmark according to the Kruskal–Wallis test ($P < 0.01$).

As a result optimal sequence alignment with local search space still remains an open challenge. It is generally observed that many algorithms do not consider a sufficient number of benchmark dataset for experimentation and validation. To meet this challenge **various research are are** now being applied.

3. TRACE SEQUENCE ALGORITHM FOR MSA (TSGA-MSA)

The proposed Trace sequence algorithm for MSA (TSGA-MSA) is a modification of the basic GA. Its steps are: generation of initial population, inserting the gaps in a sequence by tracing he graph, generation of the child population by applying genetic operators, formation of a new population for the next generation, and then further application of vertical division and combination and, finally, establishment of the stopping criteria.

A multiple sequence alignment (MSA) of A is obtained by inserting gaps ('-') into the original sequences such that all resulting sequences A*i have equal length L ≥max {ni | i = 1, . . . , r}[1].

3.1 Trace of an alignment

We are taken pairwise alignment A (S1; S2) of two sequences S1 and S2. We can identify the characters of S1 and S2 as the vertices V of the complete bipartite graph G = (V, E) = $K_{n1, n2,}$ i.e., V = S1∪S2 where S_i: = { s_i:| 1≤j≤n_i }.We call G the input alignment graph. The edges in G are called alignment edges and represent possible (mis) matches of the characters in the two sequences. We say that a (mis) match in the alignment A realizes the edge in E that joins the aligned characters. The set of all edges in E that is realized by an alignment is called a trace, a notion first introduced by Sankoff and Kruskal [19]. In Figure 2 an alignment and the corresponding trace are shown. Of course not all edges can be realized where the edge e starts and by end (e) the position of the letter of S_2 where the edge e ends. Start in S_i, end in S_j , and if

start (e) > start(f) and end(e) ≤end(f)

or

start (e) = start(f) and end(e) < end(f):

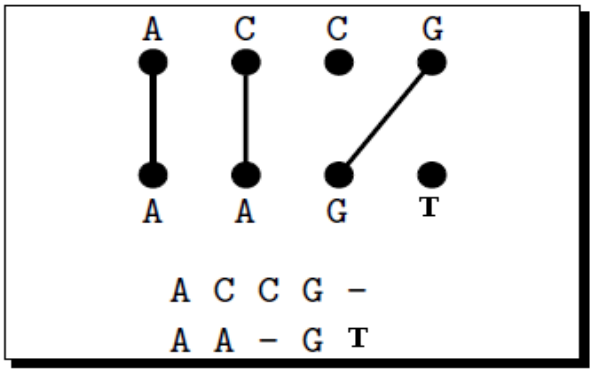

Fig 2: A pairwise trace and the alignment that realize that trace

In the following example depicted in Figure 3, we will consider a novel approach to computing an MSA based on "integer linear programming". Suppose we are given two sequences *a*1 = A G C =T and *a*2 = AGT

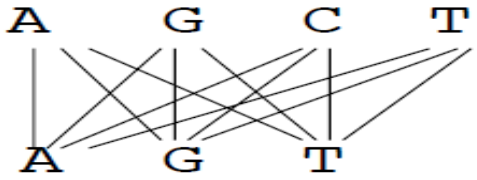

Fig 3: Equal length of sequence matrix

The set of realized edges is called the *trace* of the alignment. An arbitrary subset $T \subseteq E$ of edges is called a *trace*, if there exists some alignment that realizes precisely the edges in *E*. Our goal is to characterize all legal traces. So firstly we have trace all the sequence with same pattern and put these symbols in same column then insert the gap in that place where the same symbol is not matched. Therefore we have to generate the maximum length of sequence by inserting the spaces in a matrix. Suppose we are given three sequences *a*1 = A G C T, *a*2 = A G T and A C T **that are shown in Figure 4.**

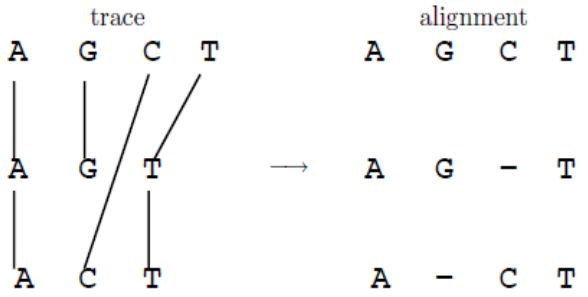

Fig 4: matrix after inserting the space

The encoding process is often the most difficult aspect of solving a problem using genetic algorithms. When applying them to a specific problem it is often hard to find an appropriate representation of the solution that will be easy to use in the crossover process.

3.2 Score functions based on pairs of sequence segments

Traces neglect the order of unaligned residues and therefore only matches and mismatches are scored. Since the (mis)matches correspond to the alignment edges in the input alignment graph, we assign each alignment edge e∈E a weight w_{ij} representing the benefit of realizing that edge. Then a pairwise trace score function can be defined by the formula.

$$Fitness\_Score = \sum_{i=2}^{N} \sum_{j=1}^{i-1} WijCost(Si,Sj)$$

3.3 Selection Procedure

In this case we are applying the tournament selection procedure for finding out the best solution. Two solutions are picked out of the pool of possible solutions, their fitness is compared, and the better is permitted to reproduce.

3.4 Window frame Combine Crossover Operator

In window frame combine operator, both parents are selected, from the middle of 20% of the parent generation. For window frame combine crossovers, each parent is divided into three parts. The different part from these parents are then exchanged and merged together to generate two new individuals. However, the better one will be taken as a child. The crossover is implemented in two steps as described below.

Step 1: To create a 20% window frame from the overall length of both parents and calculate the score scores of the first 20% of columns for both parents. The parent having the better score is divided vertically at that column. The other parent is divided using the same mechanism.

Step 2: We now have cut the 20% window frame of first parent and it combine with second to create new individual and vice versa.

To complete the crossover, the middle part of both parents are to be exchanged and then all three pieces are merged together to generate two new individuals.

3.5 Combine Space Mutation Operator

The purpose of the combine Space operator is to merge two or three spaces together. In the combine space operator we combine space of 20% rows of all the parent chromosomes after crossover.

3.6 New Generation

In this research, the best 50% of the parents and the best 50% of the children were merged together while ensuring that there was no duplication of individuals. Experiments with other splits, such as 40-60 and 60-40 (parent-child) were also performed. This mix (50-50 parent-child) was chosen based on experimental observations to ensure a better balance between exploration and exploitation.

performances of TSGA-MSA were tested for different datasets. When comparing it with different algorithms, such as GAMS, SAGA, MSA-GA, SGA and CLUSTAL, the BAliscores of the solutions were used. The comparative results show that, on average, TSGA-MSA produces better alignments than other algorithm with same number of generations and time. The results are shown that TSGA-MSA successfully found more than 80% accurate solutions than the others in 15 test cases, GAMS in 3, SAGA in 2, MSA-GA in 1, CLUSTALW in 1 and SGA in 0 .Therefore, for the Balibase 2.0 test datasets, TSGA-MSA successfully found better MSAs in 80.01% of the test cases. The overall performances based on the average scores in Table 2 and Figure 5 in which it can be seen that TSGA-MSA achieved a higher average accuracy than all the other methods considered in this section.

Table 1: Experimental results for reference 2 dataset

Dataset Name	No of Gen.	TSGA-MSA	GAMS	SAGA	MSA-GA	SGA	CLUSTAL W
RV11_BB11015	50	0.921	0.843	**0.946**	0.756	0.018	0.592
RV11_BB11025	50	**0.656**	0.317	0.278	0.457	0.013	0.344
RV11_BBS11022	50	0.731	**0.845**	0.643	0.653	0.837	0.718
RV12_BB12006	50	0.845	0.960	0.895	0.908	0.895	**1.000**
RV12_BBS12021	50	**0.989**	0.974	0.811	0.671	0.452	0.961
RV12_BBS12034	50	0.800	**0.983**	0.847	0.748	0.682	0.927
RV11_BBS11009	50	**0.945**	0.877	0.766	0.768	0.763	0.764
RV11_BB11022	50	**0.577**	0.033	0.038	0.002	0.034	0.034
RV11_BBS11006	50	**0.786**	0.635	0.630	0.711	0.364	0.654
RV11_BBS11008	50	0.675	0.826	**0.985**	0.826	0.945	0.525
RV11_BBS11013	50	**0.989**	0.970	0.985	0.941	0.981	0.950
RV11_BBS11017	50	0.875	**0.925**	0.721	0.58	0.758	0.619
RV11_BB11001	50	**0.888**	0.707	0.743	0.703	0.579	0.763
RV12_BBS12010	50	0.678	0.384	0.654	**0.71**	0.364	0.630
RV11_BBS11001	50	**0.945**	0.899	0.75	0.73	0.528	0.732

4. SIMULATIONS AND RESULTS

In the proposed solution, we have also used two specific crossover and mutation operators. In order to determine the best crossover and mutation probabilities; we have carried out three different experiments, using ten randomly selected Balibase 2.0 dataset that were obtained from [20].The

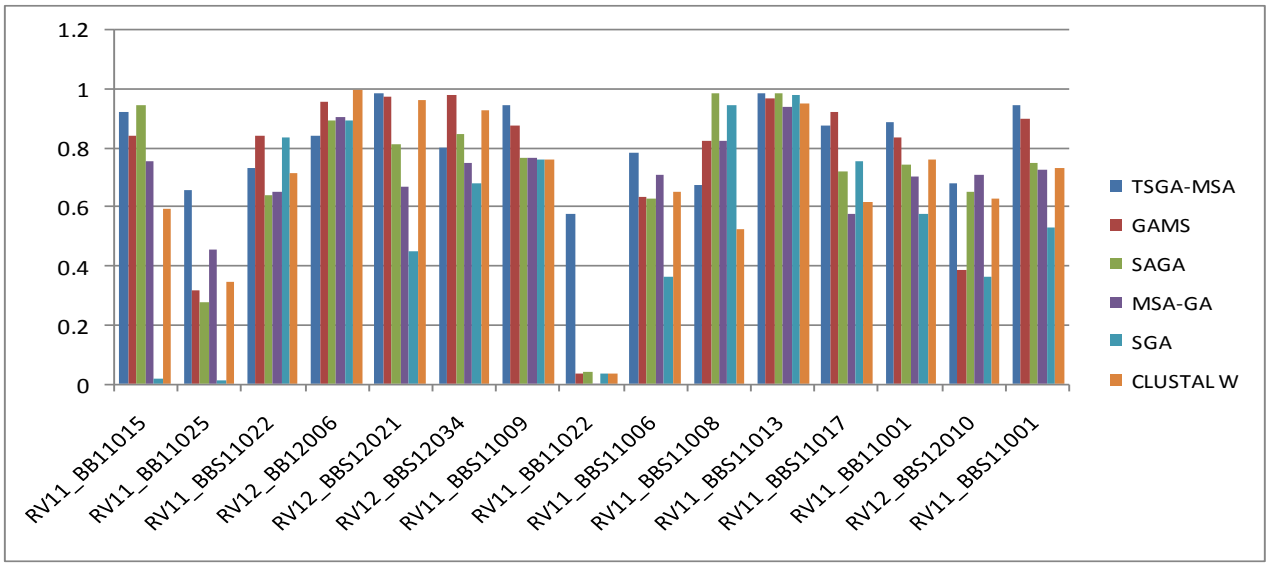

Fig 5: Overall performances of all methods for reference 2 datasets

5. CONCLUSION

The overall performance of the proposed method was also better than other methods considered because of its proposed initial generation, its genetic operators and the operators' settings. The statistical and experimental analyses proved that the proposed method could be considered to solve MSA problems significantly and effectively.

6. REFERENCES

[1] Notredame,C.: Recent progress in multiple sequence alignment: a survey, *Pharmcogemmics*, vol. 3, 2002, pp. 131-144.

[2] Notredame, C. and Higgins, D. G.: SAGA: Sequence alignment by genetic algorithm, *Nucleic Acids Res.*, vol. 24, 1996, pp. 1515–1524.

[3] Anabarasu, L. A.: Multiple sequence alignment using parallel genetic algorithms, *In the Second Asia-Pacific Conference on Simulated Evolution (SEAL-98).* Canberra Australia, 1998, pp.201-210.

[4] Notredame, C., Holm, L. and Higgins, D. G.: COFFEE: an objective function for multiple sequence alignments, *Bioinformatics*, vol. 14(5), 1998, pp. 407-422.

[5] Zhang, C. Wong AKC.: Toward efficient multiple molecular sequence alignment: a system of genetic algorithm and dynamic programming, *IEEE Transactions on Systems, Man and Cybernetics*, 1997, pp. 918 - 932.

[6] Cai, L., Juedes, D. and Liakhovitch, E.: Evolutionary computation techniques for multiple sequence alignment. *Congress on Evolutionary Computation (CEC 2000)*, 2000, pp. 829-835.

[7] Thomsen, R., Fogel GB. and Krink ,T.: Improvement of clustal–derived sequence alignments with evolutionary algorithms, *The 2003 Congress on Evolutionary Computation (CEC '03)*, 2003, pp. 312 - 319.

[8] Nguyen, H. D., Yoshihara, I., Yamamori, K. and Yasunaga, M.: Improved GA-based method for multiple protein sequence alignment, *The 2003 Congress on Evolutionary Computation (CEC '03)*, 2003, pp. 1826 - 1832.

[9] Hsiao, Y. and Chuang, C.: A novel GA–based algorithm approach to fast biosequence alignment, *IEEE Conference on Cybernetics and Intelligent Systems*, 2004, pp. 602 - 607.

[10] Liu, LF. Huo, HW. and Wang, BS. : Aligning multiple sequences by genetic algorithm. *International Conference on Communications, Circuits and Systems (ICCCAS 2004)*, 2004, pp. 994 - 998.

[11] Omar, MF. Salam, RA., Rashid, NA. and Abdullah R. : Multiple sequence alignment using genetic algorithm and simulated annealing, *Proceedings of International Conference on Information and Communication Technologies: From Theory to Applications,*2004, pp. 455 - 456.

[12] Abdesslem, L, Soham , M. and Mohamed , B. : Multiple sequence alignment by quantum genetic algorithm , In *20th International Parallel and Distributed Processing Symposium (IPDPS 2006)*, 2006,pp. 241-250.

[13] Chen, Y., Pan, Y., Chen, J., Liu, W. and Chen, L.: Multiple sequence alignment by ant colony optimization and divide-and-conquer, International *Conference on Computational Science (2)*, vol. 3992 *of Lecture Notes in Computer Science*, 2006, pp. 646-653.

[14] Yang, C. , Jinglu, H. and Songnian, Y.: Multiple Sequence Alignment Based on Genetic Algorithms with Reserve Selection, *ICNSC, 2008*, 2008, pp 1511-1516.

[15] Fernando , J. , Juan , M. , Juan , A. and Miguel, A. : An Evolutionary Approach for Performing Multiple Sequence Alignment, *IEEE*, 2010, pp. 4244-8126

[16] Nizam, A., Jeyakodi , R. and Kuppuswami , S. : Cyclic Genetic Algorithm for Multiple Sequence Alignment, *International Journal of Research and Reviews in Electrical and Computer Engineering (IJRRECE)* Vol. 1, 2010, pp.39-44.

[17] Zahra, N., Hamid, B. and Hassan, A.: A New Genetic Algorithm for multiple sequence alignment, *Int. J. Comp. Intel. Appl*. Vol.11, 2012, pp. 1469-0268.

[18] Francisco, M., Olega, V. and Fernando, R.: Optimizing multiple sequence alignments using a genetic algorithm based on three objectives: structural information, non-gaps percentage and totally conserved columns, *Bioinformatics*, vol.82, 2013, pp. 321-332.

[19] Sanko, D. and Kruskal , J. : Time Warps, String Edits and Macromolecules, *the Theory and Practice of Sequence Comparison*, Addison-Wesley, 1983.

[20] Bahr, A., Thompson, J. D, Thierry, J. C. and Poch, O , BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations, Nucleic Acids Res, vol. 29. , 2001, pp. 323-326.