Parallel Application Systems in Genetic Information Processing

MASATO ISHIKAWA, TOMOYUKI TOYA AND YASUSHI TOTOKI

Institute for New Generation Computer Technology
4-28-21F, Mita 1-chome, Minato-ku, Tokyo 108, Japan
ishikawa@icot.or.jp

Abstract

We have developed parallel application systems for genetic sequence analysis. The systems are three parallel iterative aligners and an alignment workbench for the aligners. The parallel iterative aligners use a best-first search, a hill-climbing search, and a genetic algorithm. Each iterative aligner can produce higher-quality solutions than the conventional tree-based aligner. The genetic algorithm shows better performance than the best-first and hill-climbing searches. The alignment workbench, which features the parallel iterative aligners, realizes alignment which is not only fast and high-quality—it is also constraint-based. When a user has some biological knowledge which indicates that some characters might be aligned in a column, a constraint can be defined for those characters. The constraint set is considered simultaneously in each iteration cycle of parallel alignment. Then appropriate multiple alignment is generated by the aligner and displayed on the workbench's full-color display. The alignment workbench also contains the following characteristic sequence analysis modules: a phylogenetic tree drawer, a motif-database matcher, and a stem region specifier.

1 Introduction

Molecular biology and genetic technology have been advancing at an astonishing rate in recent years. Major activities in these fields are closely related to DNA/RNA and protein. This is because a set of DNA molecules in a cell contain the genetic information for the complete design of the living organism. This information is embodied as protein to build up the body and to keep its mechanisms alive. Each piece of genetic information, represented by a sequence of nucleic acids, is translated into a sequence of amino acids to form protein. As the method to determine DNA or protein sequences has progressed to its current state, the amount of known sequence data has grown rapidly. For example, *Genbank* [1], one of the most widely distributed databases, contains information on more than two hundred million nucleotides. The growing number of genetic sequences in databases inevitably makes the field of genetic information processing one of the most important application areas for computer science.

The fundamental technique for analyzing genetic sequence data by computer is to examine similarities among sequences. This usually requires large amounts of computation to find the similarities, since there are a lot of sequences in the database to be examined. The computational problem can be partly solved with parallel implementation. There have been some experiments with parallel database search [2, 3] and with parallel sequence analysis [4, 5].

We have developed parallel application systems for genetic sequence analysis. The systems has been implemented in a parallel logic programming language and working on parallel computers. The aim of this paper is to show the availability of parallel implementation in the field of genetic information processing. The organization of the rest of this paper is as follows. In Section 2, we briefly explain our application problems. We present our parallel iterative aligners in Section 3. Then, the alignment workbench, which features the parallel iterative aligners is discussed in Section 4. Finally, conclusions are given in Section 5.

2 Genetic sequence analysis

The genetic information, stored in DNA, is partly transcribed as RNA segments. Each segment composed of a chain of four kinds of nucleotides, represented by a sequence of code letters: A, U, G and C. An RNA segment is translated into a shorter chain of twenty kinds of amino acids, represented by a sequence of twenty different code letters. A chain of amino acids folds to become protein in a cell. Because every amino acid has its own proper-

```
(a) problem
CABL :KLGGGQYGEVYEGVWKKYSLTVAVKTLKEDTMEVEEFLKEAAVMKEIKHPNLVQLLGVCTREPPFYIITEFMTYGNLLDY
FER  :LLGKGNFGEVYKGTLKDKTSVAVKTCKEDLPQELKIKFLQEAKILKQYDHPNIVKLIGVCTQRQPVYIIMELVSGGDFLT
TRK  :ELGEGAFGKVFLAECHNLLPEQDKMLVAVKALKEASESARQDFQREAELLTMLQHQHIVRFFGVCTEGRPLLMVFEYMRH
GCPK :SLRGSSYGSLMTAHGKYQIFANTGHFKGNVVAIKHVNKKRIELTRQVLFELKHMRDVQFNHLTRFIGACIDPPNICIVTE
PKC1 :VLGKGNFGKVILSKSKNTDRLCAIKVLKKDNIIQNHDIESARAEKKVFLLATKTKHPFLTNLYCSFQTENRIYFAMEFIG
CGMPK:TLGVGGFGRVELVQLKSEESKTFAMKILKKRHIVDTRQQEHIRSEKQIMQGAHSDFIVRLYRTFKDSKYLYMLMEACLGG
CAMK :ELGKGAFSVVRRCVKVLAGQEYAAKIINTKKLSARDHQKLEREARICRLLKHPNIVRLHDSISEEGHHYLIFDLVTGGEL
FUSED:LVGQGSFGCVYKATRKDDSKVVAIKVISKRGRATKELKNLRRECDIQARLKHPHVIEMIESFESKTDLFVVTEFALMDLH
WEE1 :LLGSGEFSEVFQVEDPVEKTLKYAVKKLKVKFSGPKERNRLLQEVSIQRALKGHDHIVELMDSWEHGGFLYMQVELCENG

(b) iterative alignment [score=2413]
CABL :KLGGGQYGEVYEGVWK---------KYSLTVAVKTLKED----TMEVEEFLKEAAV---MKEIK-HPNLVQLLGVCTREPPFYIITEFMTYGNLLDY
FER  :LLGKGNFGEVYKGTLK----------DKTSVAVKTCKED--LPQELKIKFLQEAKI---LKQYD-HPNIVKLIGVCTQRQPVYIIMELVSGGDFLT-
TRK  :ELGEGAFGKVFLAECH-NLLP---EQDKMLVAVKALKEA---SESARQDFQREAEL---LTMLQ-HQHIVRFFGVCTEGRPLLMVFEYMRH------
GCPK :SLRGSSYGSLMTAHGKYQIFANTGHFKGNVVAIKHVNKK---RIELTRQVLFELKH---MRDVQ-FNHLTRFIGACIDPPNICIVTE----------
PKC1 :VLGKGNFGKVILSKSK---------NTDRLCAIKVLKKDNIIQNHDIESARAEKKVFLLATKTK-HPFLTNLYCSFQTENRIYFAMEFIG-------
CGMPK:TLGVGGFGRVELVQLK--------SEESKTFAMKILKKRHIVDTRQQEHIRSEKQI---MQGAH-SDFIVRLYRTFKDSKYLYMLMEACLGG-----
CAMK :ELGKGAFSVVRRCVKV---------LAGQEYAAKIINTKK-LSARDHQKLEREARI---CRLLK-HPNIVRLHDSISEEGHHYLIFDLVTGGEL---
FUSED:LVGQGSFGCVYKATRK---------DDSKVVAIKVISKRG-RATKELKNLRRECDI---QARLK-HPHVIEMIESFESKTDLFVVTEFALMD-LH--
WEE1 :LLGSGEFSEVFQVEDP--------VEKTLKYAVKKLKVKF-SGPKERNRLLQEVSI---QRALKGHDHIVELMDSWEHGGFLYMQVELCENG-----
       .LG.G.FG.V......         ......A.K.....    ..........E...   ..... H.....................E....

(c) tree-based alignment [score=1940]
CABL :----------KLGGGQYGEVY----EGVWKK---YS-LTVAVKTLKED----TMEVEEFLKEAAVM---KEIK-HP-NLVQLLGVCTREPPFYIITE--FMTYGNLLDY-
FER  :----------LLGKGNFGEVY----KGTLKD----K-TSVAVKTCKED-LPQELKI-KFLQEAKIL---KQYD-HP-NIVKLIGVCTQRQPVYIIME--LVSGGDFLT--
TRK  :----------ELGEGAFGKVFLAECHNLLPEQ--DK-MLVAVKALKEA--SESARQ-DFQREAELL---TMLQ-HQ-HIVRFFGVCTEGRPLLMVFE--YMRH-------
GCPK :SLRGSSYGSLMTAHGKY-QIF--ANTGHFKG------NVVAIKHVNKK--RIELTR-QVLFELKHM---RDVQ-FN-HLTRFIGACIDPPNICIVTE-------------
PKC1 :----------VLGKGNFGKVI----LSKSKN---TD-RLCAIKVLKKDNIIQNHDIESARAEKKVFLLATKTK-HP-FLTNLYCSFQTENRIYFAME--FIG--------
CGMPK:----------TLGVGGFGRVE----LVQLKS---EESKTFAMKILKKRHIVDTRQQEHIRSEKQIM----QGA-HSDFIVRLYRTFKDSKYLYMLMEACLGG--------
CAMK :----------ELGKGAFSVVR--RCVKVLAGQEYAA-KIINTKKLSA-----RDHQ-KLEREARIC---RLLK-HP-NIVRLHDSISEEGHHYLIFD--LVTGGEL----
FUSED:----------LVGQGSFGCVY----KATRKD---DS-KVVAIKVISKR-GRATKELKNLRRECDIQ---ARLK-HP-HVIEMIESFESKTDLFVVTE--F----ALMDLH
WEE1 :----------LLGSGEFSEVF--QVEDPVEK----T-LKYAVKKLKVK-FSGPKERNRLLQEVSIQ---RALKGHD-HIVELMDSWEHGGFLYMQVE--LCENG------
                .LG.G.FG.V.    ......    . ...A.K.....  ...... ....E....   .... H. ....................E  ...
```

Figure 1: Multiple sequence alignment

ties of volume, hydrophobicity, polarity and so on, the order of the amino acids in the protein sequence gives structure and function of the protein.

An important way of discovering new genetic information is analyzing the unknown characters of DNA/RNA and protein from their sequences. We do this by comparing the sequence of nucleic or amino acids. Multiple sequence alignment is one of the most typical methods of sequence similarity analysis. The alignment of dozens of sequences can provide valuable information for researching the function and structure of DNA/RNA or protein, especially if one of the aligned sequence has been well characterized.

Figure 1 shows examples of multiple sequence alignment. Figure 1(a) represents nine parts of different protein sequences. Indexes such as `CABL` are names of the proteins [6]. Each letter in the sequences means an amino acid. For instance, `KLG` stands for a row of Lysine, Leucine and Glycine.

Figure 1(b) is a good multiple sequence alignment obtained by our parallel iterative aligner. Each sequence is shifted by gap insertion—dash characters. Each column of the alignment has the same or similar amino acids. An identical pattern such as `LG.G.FG.V` is considered to be an important site called a *sequence motif*, or simply a *motif* [7], because an important protein sequence site has been conservative along with evolutional cycles between mutation and natural selection. Multiple sequence alignment is useful not only for inferring the structure and function but also for drawing a phylogenetic tree along the evolutional histories of the creatures.

Computers partly solve the problem of multiple sequence alignment automatically, instead of relying on the hands and eyes of experts. The results obtained by computers, however, have not been as satisfactory as those by human experts. That is because multiple sequence alignment is one of the most time and space consuming problems. Dynamic programming (DP) [8, 9], theoretically, provides an optimal solution according to a given evaluation score. This, however, requires memory space for an N-dimensional array (where N is the number of sequences) and calculation time for the N-th power of the sequence length. Though a method was proposed to cut unnecessary computation in DP [10], it still needs too much computation to solve any practical alignment problem.

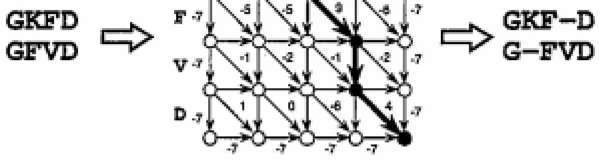


Figure 2: Two-way dynamic programming

Figure 2 shows the algorithm of two-way DP applied to two tiny sequences. The algorithm searches the best path in the figurative network from the top left node to the bottom right node optimizing the total score along the arrows. Each score indicated on an arrow reflects the similarities between the characters being compared. The best path corresponds to the optimal alignment; each arrow in the path corresponds to each column in the alignment. Vertical and horizontal arrows indicate the insertion of gaps.

A number of heuristic algorithms for multiple alignment problems have been devised using two-way DP in order to obtain approximate solutions within a practical time. The most popular one is the tree-based method [11, 12]. In this algorithm, similarity between each pair of sequences is first estimated with its pairwise alignment score obtained by DP. Using a matrix of the similarity scores, a clustering technique constructs a guided tree. Sequences are merged to form a multiple alignment based on the bottom-up branching order of the guided tree. Each node of the tree shows two bunches of sequences to which two-way DP is applied (Figure 3).

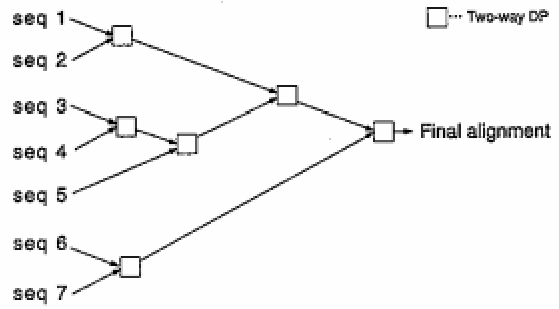


Figure 3: Tree-based alignment method

The tree-based method is fast enough to be applied to practical-scale problems. If sequence similarity is low, however, it often produces low-quality alignment. This happens because once an error occurs in the alignment process, the error cannot be corrected. Figure 1(c), an example of tree-based alignment, is not as good as Figure 1(b), because two sequences are miss-aligned at motif `LG.G.FG.V` and `A.K` in 1(c). Biologists, users of the tree-based algorithm, frequently have to correct such miss-aligned parts by hand.

3 Parallel iterative aligners

In this section, we propose three multiple sequence aligners for genetic sequence analysis. All the aligners, written in parallel logic programming language KL1, are working on parallel computers. They are based on an iterative improvement technique [13, 14], but use different parallel search algorithms: a best-first search, a hill-climbing search, and a multi-group genetic algorithm.

3.1 Parallel best-first search

Figure 4 shows the algorithm of the parallel best-first iterative aligner [15, 16]. First, N sequences which have no gaps are input into an iteration cycle. The sequences are divided into a sequence and $N-1$ alignment sequences. The N different sets produced by the partitioning are then recombined in parallel by two-way DP. The results are compared and the best-score alignment is set at the starting point of the next iteration cycle. In this way, the iteration cycle gradually improves alignment of all the sequences. Iteration terminates when none of the N partitions can be improved further. This parallel routine requires N processing elements (PEs).

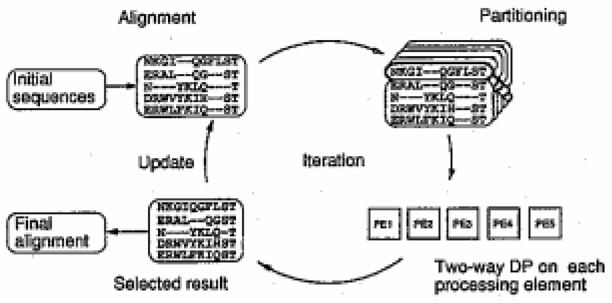


Figure 4: Parallel best-first iterative aligner

Thus, the iterative aligner gradually improves global multiple alignment. Improvement is evaluated by the alignment score defined as follows. The alignment score is a total summation of the similarity scores of every pair of aligned sequences, each of which is derived by summing the similarity values of every character pair in the column. Each similarity value is given by Dayhoff's odds matrix [18], each value of which is a logarithm of the mutation probability of a character pair (zero is the neutral value). A *gap penalty* corresponding to each row of gaps in the two sequences is added to the similarity score. The gap penalty imposed on a row of k gaps is a linear relation: $a+bk$ where a and b are parameters. We set $a=-7$ and $b=-1$ as default values.

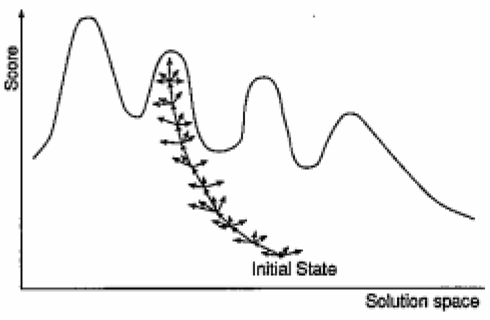


Figure 5: Best-first search

The parallel best-first iterative aligner can produce better resulting alignments than the conventional tree-based algorithm. The resulting alignments rarely need any manual corrections (Figure 1(b)). On average, the alignment score is about twenty percent better than that obtained by the tree-based algorithm.

The best-first search (Figure 5) is rapid, but has the problem of local optima. Since the solution space has hundreds of peaks, the best-first search converges to one of the nearest peaks. The parallel hill-climbing search described below relaxes this problem.

3.2 Parallel hill-climbing search

Figure 6 shows the algorithm of the parallel hill-climbing iterative aligner. Every PE starts iterative improvement from a no-gap alignment. In each iteration cycle of the improvement process, N sequences are randomly divided into a sequence and $N-1$ alignment sequences, and then recombined by two-way DP. The result of parallel hill-climbing is the best-score alignment among PEs. Iteration terminates when none of the PEs can improve their alignment further. This parallel routine doesn't limit the number of PEs.

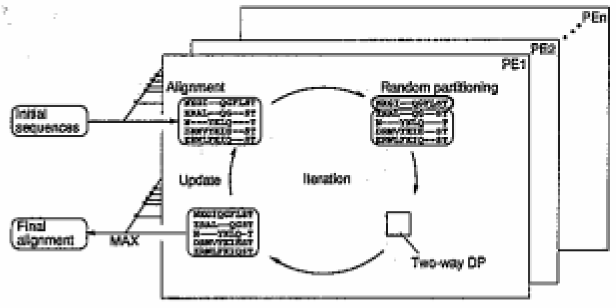


Figure 6: Parallel hill-climbing iterative aligner

When we solved twenty-two sequence alignment problems using twenty-two PEs, the parallel hill-climbing search resulted in the average alignment score of two percent larger than the best-first search. The superiority increased to three percent, when using two hundred and fifty-five PEs. The parallel hill-climbing search, however, showed slower speed of improvement. On average, it took one and a half times as much time to reach the alignment score obtained by the best-first search [17].

Although the parallel hill-climbing search (Figure 7) relaxes the problem of local optima, its convergence is too slow to solve a large alignment problem. The hybrid search by multi-group genetic algorithm described below presents faster convergence keeping the ability of escaping local optima.

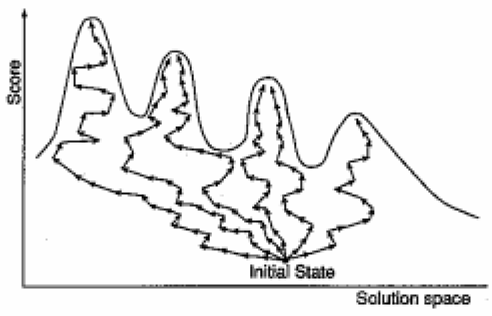


Figure 7: Parallel hill-climbing search

3.3 Multi-group genetic algorithm

We intended to implement the hybrid search (Figure 8) utilizing the merits of both parallel best-first and hill-climbing searches. The framework of the multi-group genetic algorithm could succeed in forming the hybrid search. We describe the mechanism of genetic algorithm (GA) and its multi-group strategy applied to alignment problems [19].

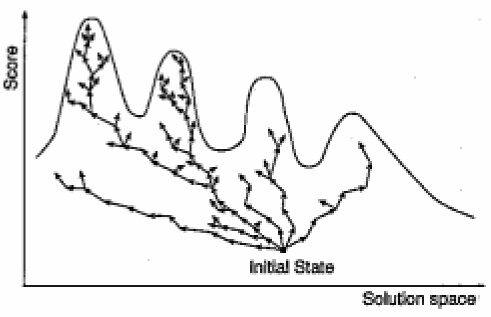


Figure 8: Hybrid search

Mechanism of GA

GAs are stochastic search algorithms based on the biological evolution process [20, 21]. As in Figure 9, GAs simulate the survival of the fittest in a population of solutions which represent points in a search space. In our GA system, each solution, though usually represented by a binary string, corresponds to a possible multiple sequence alignment. A fitness function corresponds to the alignment score. The number of solutions in a population is the same as the number of available PEs, because each PE manages a solution.

The aim of a GA is to find the global optimum of the fitness function by applying genetic operators in each generation. The genetic operators consist of modification, selection, duplication, and crossover.

The modification operator changes certain parts of a solution. Figure 10 shows the modification of an individual alignment. A sequence is randomly chosen from

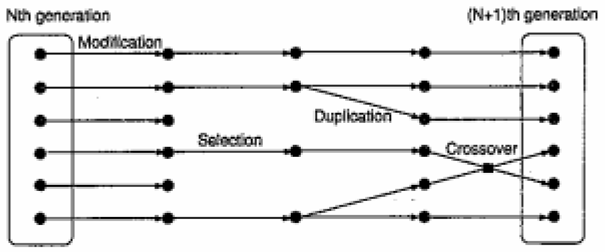


Figure 9: Mechanism of Genetic Algorithm (GA)

among the aligned sequences, and is re-aligned against the other sequences with two-way DP-matching. Although modification means a random perturbation under the orthodox concept of a GA, it is considered an iterative cycle of improvement in our definition.

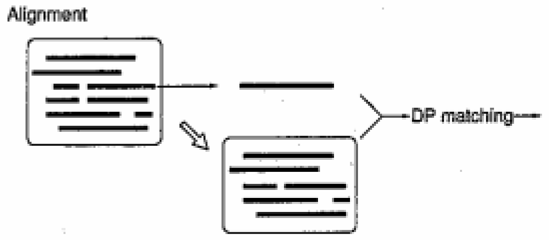


Figure 10: Modification of alignment

The selection operator chooses good solutions from a population according to fitness and the given selection strategy. This operator aims to increase the quality of solutions in the population while maintaining a certain level of diversity. The operator first calculates the relative fitness of all solutions, then discards several of the lesser ones as determined by a parameter value. The same number of solutions are compensated for by duplication operator so that the number of solutions in a population is constant generation after generation.

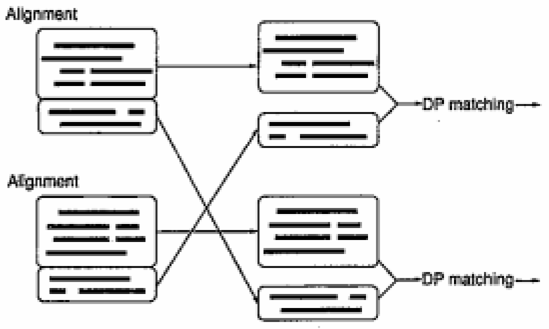


Figure 11: Crossover of Alignments

The crossover operator produces two descendants by exchanging parts of two solutions. This operator aims to improve a solution by replacing a part of a solution with a better part from another solution. Figure 11 shows the crossover of two individual alignments. The sequences to be aligned are randomly divided into two sets: the exchange and unexchange sets. The sequence members in the exchange sets are exchanged between the individual alignments by fixing the current alignment within each set. The exchanged and unexchanged sets are re-aligned with two-way DP-matching. Candidate alignments which should be done with crossover operators are chosen randomly. The number of candidates are determined by a parameter value.

Multi-group GA strategy

We focused on the selection rate of our GA system. By arranging the selection rate, we can implement fairly different search strategies. When the system has the highest selection rate and no crossover (Figure 12), the search becomes narrow and concentrated, which is very similar to the best-first search. On the other hand, without selection and crossover (Figure 13), the search becomes wide and distributed, which is equivalent to the parallel hill-climbing search.

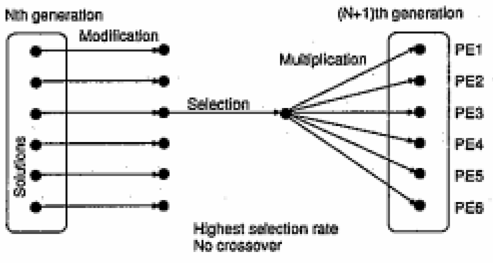


Figure 12: Best-first search in GA

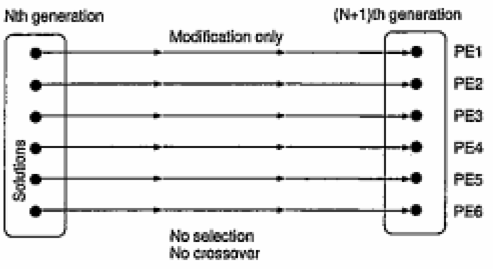


Figure 13: Parallel hill-climbing in GA

The multi-group GA technique can dynamically change search strategies by migration during a GA process [22]. We have devised a multi-group GA as shown in Figure 14 characterized by the following.

- The selection rate varies group by group: 90%, 60%, 30% and 0%.
- Solutions which have higher scores than the maximum score of the next high-selection-rate group migrate to the group.

- Solutions captured in local optima are eliminated; when more than half of solutions in a group have the same score, a solution is randomly selected out of each two same-score solutions and thrown away.
- New solutions are produced in the lowest-selection-rate group by duplication. This keeps the total number of solutions in all groups.

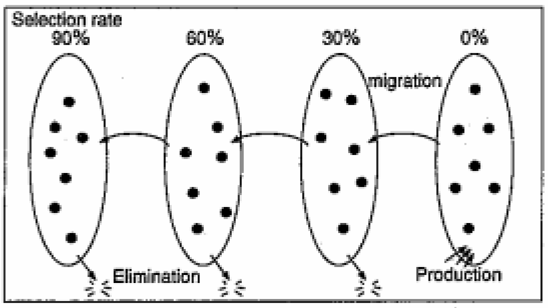


Figure 14: Multi-group GA strategy

In the multi-group GA strategy of our system, a high-score solution migrates to a high-selection-rate group in order to do a concentrated search around its vicinity, and solutions captured in local optima surrender their computational resources to a distributed search in the lowest-selection-rate group. We tested the performance of our GA system using two-hundred and fifty-five PEs. The test alignment problems are the same as those used for the best-first and hill-climbing searches. We found that the GA search reached the alignment score obtained by the best-first search as rapidly as the best-first search, and that the GA search converged to the score level obtained by the hill-climbing search earlier than the hill-climbing search.

4 Alignment workbench

Such iterative aligners improve an alignment with respect to its score. Whatever scoring system we use, the optimal-score alignment is not always the most significant result from a biological perspective. In addition to optimizing alignments under some scoring system, it is also important to refine them using biological knowledge. Alignment workbenchs [23, 24] have been developed for this purpose. We introduce our alignment workbench [25] which features constraint-based parallel iterative alignment.

In this section, firstly we mention a recommended alignment process. We then describe the features of constraint-based aligners. Finally, we discuss refinement tools. The entire workbench is written in the C programming language and works on UNIX workstations with

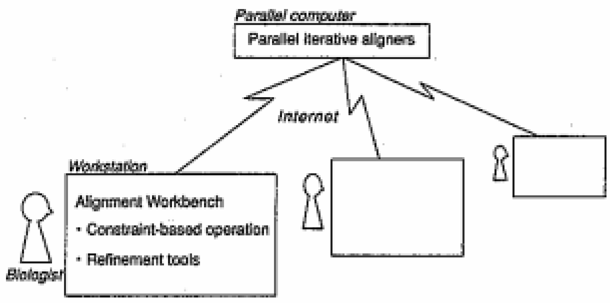


Figure 15: Alignment environment

OSF/Motif. The exception is the parallel aligners, which work on a PIM or a UNIX-based parallel computer such as CM5 and provide the workbench with aligned data via internet. The alignment environment of the workbench is shown in Figure 15.

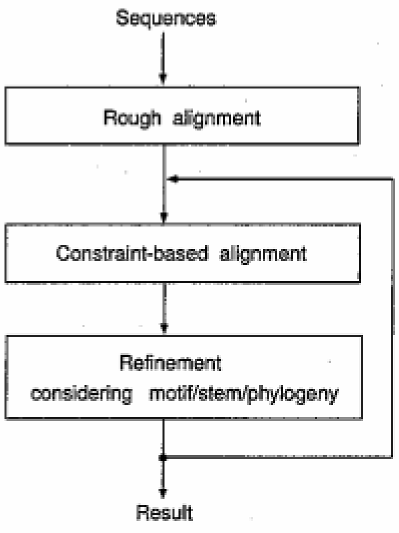


Figure 16: Alignment process

4.1 Alignment process

Figure 16 shows a recommended process for using this alignment workbench. Sequences are initially aligned roughly with the conventional tree-based aligner, which is rapid enough to deal with a lot of long sequences. The rough alignment often reveals some half-aligned patterns, which can be recognized as constraints and forced into alignment. Sequences are elaborately realigned part by part with constraint-based iterative aligners. Being evaluated with respect to sequence motifs, secondary structures, or phylogenetic relationships, the alignment can then be refined manually by mouse-oriented operations and realigned repeatedly based on some newly imposed constraints. Thus, the final alignment can be arranged to meet user specifications.

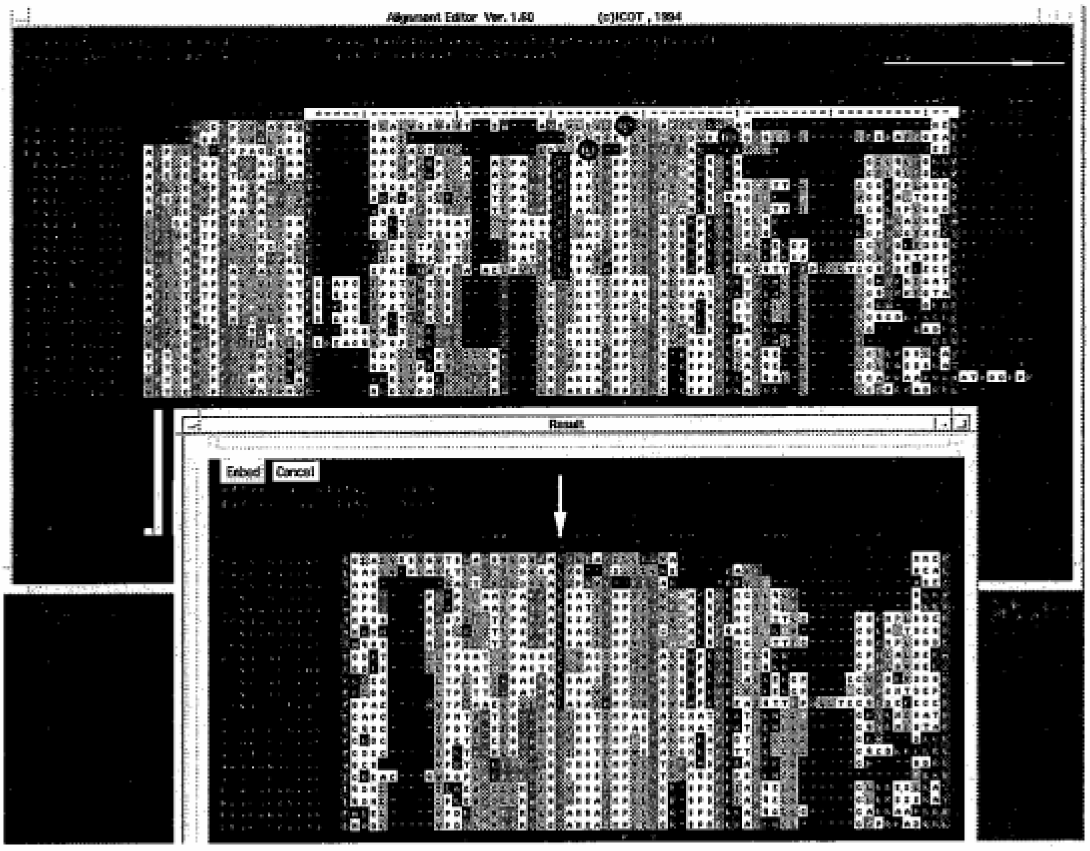


Figure 17: Constraint-based alignment

4.2 Constraint-based alignment

The workbench features constraint-based alignment, in which the tree-based and parallel iterative aligners described in the previous sections are available. The iterative aligners are also useful for keeping a current alignment to some extent, because they are executed starting from the current alignment as the initial state of iteration.

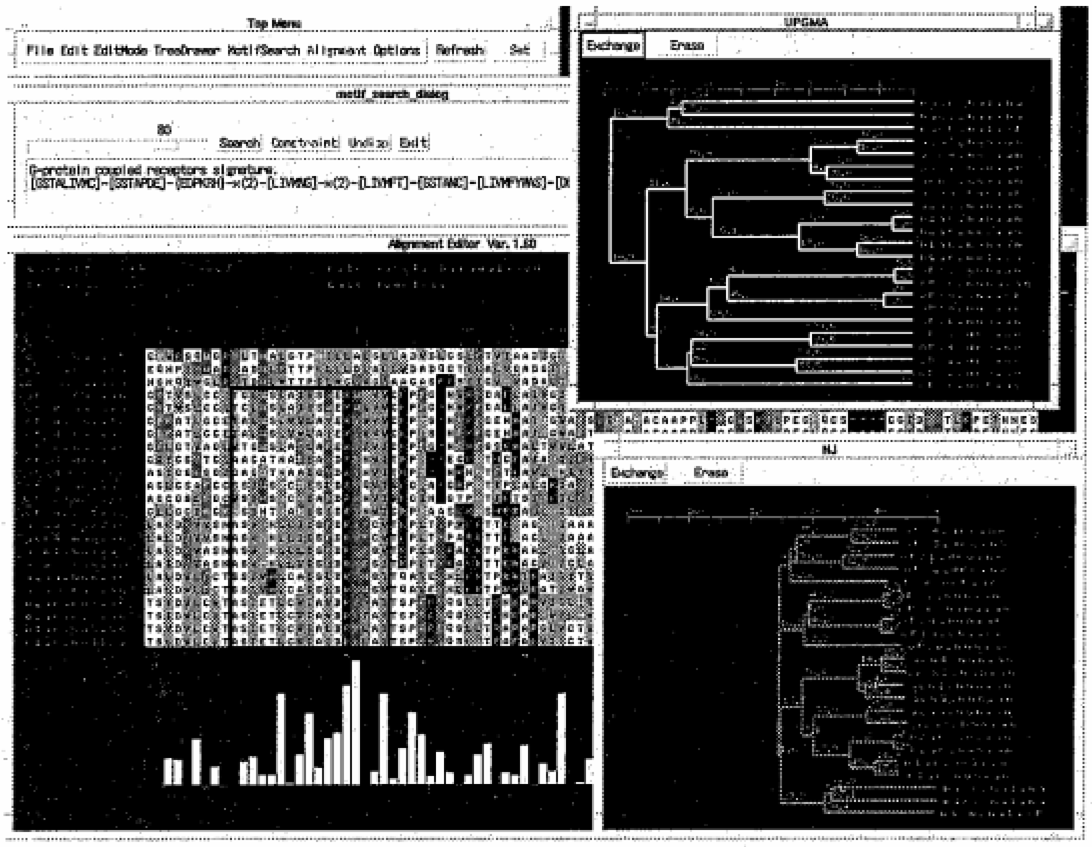
A typical operation of the constraint-based alignment is shown in Figure 17. In the upper window, protein sequences in the rhodopsin super family [26] are roughly aligned. The displayed alignment corresponds to the G helix of the bacteriorhodopsin family, whose structure has already been mapped [27]. The first three sequences, from the bacteriorhodopsin family, are not similar enough to the other sequences to be well-aligned in the rough alignment. It is, however, known that proteins in the bacteriorhodopsin and rhodopsin families contain a retinal whose binding site is the lysine position in the G helix. This knowledge can improve the alignment. A constraint is imposed on the thirteen lysines, which are indicated by the Ks surrounded by black frames. The other sequences, from the signal receptor family, are not constrained, because proteins in the family have no retinals.

Based on the constraint, the parallel iterative aligner realigns partial alignment specified by the user. The small window at the bottom of Figure 17 displays the improved alignment, in which the thirteen lysines are aligned at the column indicated by an arrow. If the user accepts the partial alignment, it will be embedded in the main alignment.

4.3 Refinement tools

The alignment workbench contains three characteristic analysis tools: a motif-database matcher, a phylogenetic tree drawer, and a stem region specifier. Each tool is useful for refining alignment from a biological point of view.

The matcher identifies sequence motifs in a protein sequence alignment, retrieving motif data from the Prosite database [28] (release 9.00). Figure 18 shows the display when a signature motif for G-protein-coupled receptors was found in an alignment of the rhodopsin super family with more than eighty percent consensus. The region is indicated by a black rectangular frame. The motif is represented in Prosite as follows:



G-protein coupled receptors signature.
[GSTALIVMC]-[GSTAPDE]-{EDPKRH}-x(2)-[LIVMNG]-x(2)-[LIVMFT]-[GSTANC]-[LIVMFYWAS]-[D

Figure 18: Sequence motif identification

[GSTALIVMC]-[GSTAPDE]-{EDPKRH}-x(2)-[LIVMNG]-
x(2)-[LIVMFT]-[GSTANC]-[LIVMFYWAS]-[DEN]-R-
[FYWCH]-x(2)-[LIVM].

The drawer constructs an evolutional tree, dependent on the current editing alignment, with UPGMA [29] or NJ [30] method. Trees of aligned sequences are introduced in Figure 18 as well. The upper tree is drawn by UPGMA (unweighted pair-group arithmetic average clustering), and the lower one is done by NJ (neighbor joining). The estimated number of mutation events is represented on each branch. The order of sequences in an alignment display can be changed according to the evolutional tree.

The stem specifier indicates some possible stacking regions in an RNA sequence alignment [31]. Connectable pairs of nucleotide locations are displayed with lines in a circular representation. Each number surrounding the circle corresponds to a location (column) number of the alignment. Figure 19 shows a rough alignment (top) and its refined alignment (bottom) for Leucine-tRNAs, in which the first eight sequences are mitochondrial and the others are nucleic. In the rough alignment, the four main stems specify less than fifty-four percent consensus in whitened parts of the alignment. Four pseudostems are also displayed in the circle. After refinement considering the real stems as constraints, the stem regions aligned with more than seventy percent consensus and no pseudostems.

5 Conclusion

We have developed parallel application systems for genetic sequence analysis: three parallel iterative aligners and their workbench. The parallel iterative aligners use a best-first search, a hill-climbing search, and a multi-group GA. The best-first aligner searches solution space in a concentrated way, whereas the hill-climbing aligner searches in a distributed manner. The multi-group GA forms a hybrid search which features the merits of both best-first and hill-climbing searches.

In the alignment workbench, the iterative aligners work in a constraint-based way which helps users align sequences from various biological perspectives. The workbench also provides a motif-database matcher, a phylogenetic tree drawer, and a stem region specifier to help refine alignments.

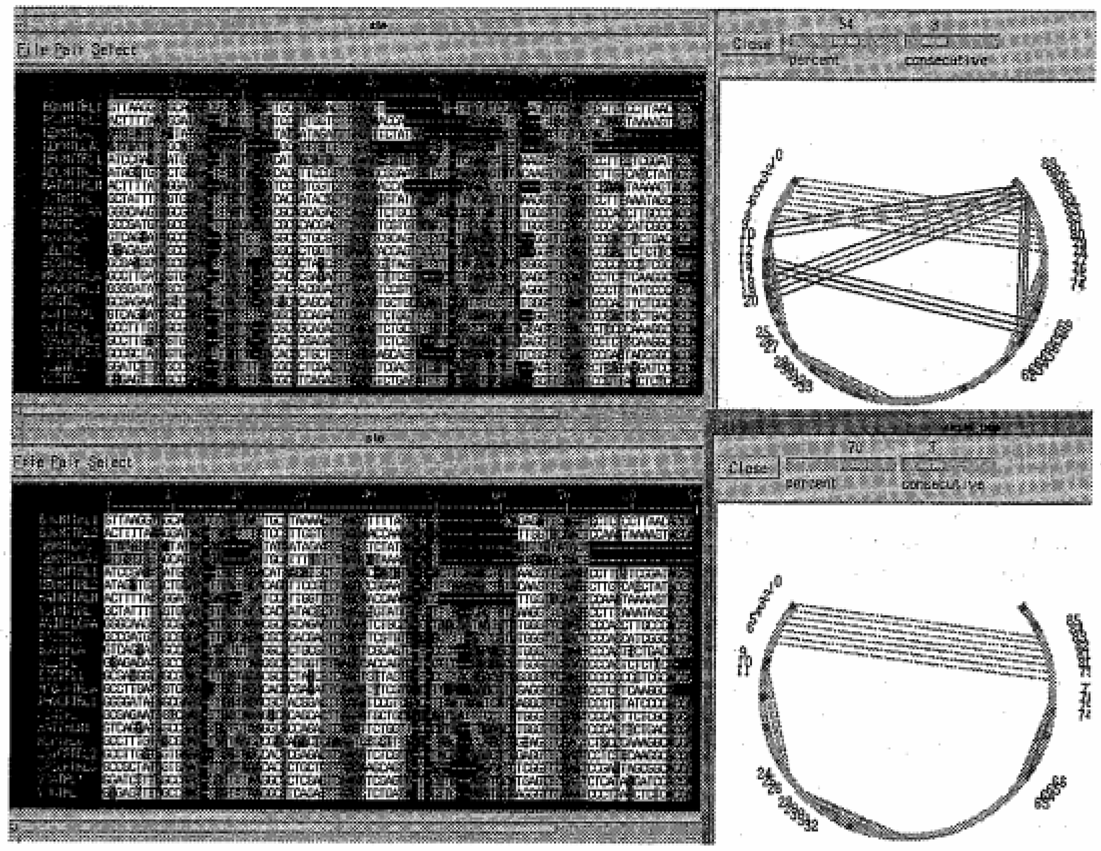


Figure 19: RNA stem specification

Acknowledgements

The authors wish to thank members of the ICOT GIP Project and in the Genome Informatics Research Project for their fruitful discussions.

References

[1] Benson,D.A. Boguski,M. Lipman,D.J. and Ostell,J., "Genbank," *Nucleic Acids Res.,* Vol.22, pp.3441-3444 (1994).

[2] Coulson,A.F.W., Collins,J.F. and Lyall,A., "Protein and nucleic acid sequence database searching: a suitable case for parallel processing," *Comput. J.,* Vol.30, pp.420-424 (1987).

[3] Miller,P.L., Nadkarni,P.M. and Carriero,N.M., "Parallel computation and FASTA: confronting the problem of parallel database search for a fast sequence comparison algorithm," *Comput. Applic. Biosci.,* Vol.7, pp.71-78 (1991).

[4] Iyengar,A.K., "Parallel DNA Sequence Analysis," *MIT/LCS/TR-428,* (1988).

[5] Kawai,M. Kishino,A. and Naito,K., "Rapid Analysis Methodology for Gene Sequences Using a Parallel Processor," *FUJITSU Scientific and Technical Journal,* Vol.27, pp.270-277 (1991).

[6] Hanks,K.S., Quinn,A.M. and Hunter,T., "The Protein Kinase Family," *Science,* Vol.241, pp.42-52 (1988).

[7] Staden,R., "Methods to Define and Locate Patterns of Motifs in Sequences," *Comput. Applic. Biosci.,* Vol.4, pp.53-60 (1988).

[8] Needleman,S.B. and Wunsch,C.D., "A general method applicable to the search for similarities in the amino acid sequences of two proteins," *J. Mol. Biol.,* Vol.48, pp.443-453 (1970).

[9] Smith,T.F. and Waterman,M.F., "Identification of common molecular subsequences," *J. Mol. Biol.,* Vol.147, pp.195-197 (1981).

[10] Carrillo,H. and Lipman,D., "The multiple sequence alignment problem in biology," *SIAM J. Appl. Math.* Vol.48, pp.1073-1082 (1988).

[11] Feng,D.-F. and Doolittle,R.F., "Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees," *J. Mol. Evol.*, Vol.25, pp.351-360 (1987).

[12] Higgins,D.G., Bleasby,A.J. and Fuchs,R., "CLUSTAL V: improved software for multiple sequence alignment," *Comput. Applic. Biosci.*, Vol.8, pp.189-191 (1992).

[13] Berger,M.P. and Munson,P.J., "A novel randomized iterative strategy for aligning multiple protein sequences," *Comput. Applic. Biosci.*, Vol.7, pp.479-484 (1991).

[14] Gotoh,O., "Optimal Alignment between Groups of Sequences and its Application to Multiple Sequence Alignment," *Comput. Applic. Biosci.*, Vol.9, pp.361-370 (1993).

[15] Ishikawa,M., Hoshida,M., Hirosawa,M., Toya,T., Onizuka,K. and Nitta,K., "Protein Sequence Analysis by Parallel Inference Machine," *Proc. Fifth Generation Comput. Syst. '92*, pp.294-299 (1992).

[16] Hirosawa,M., Totoki,Y., Hoshida,M. and Ishikawa,M., "Comprehensive Study on Iterative Algorithms of Multiple Sequence Alignment," *Comput. Applic. Biosci.* (forthcoming).

[17] Ishikawa,M., Totoki,Y., Toya,T., Hoshida,M. and Hirosawa,M., "Protein Sequence Analysis by the Parallel Iterative Improvement Method," *Trans. Inf. Process. Soc. Jap.*, (1994).

[18] Dayhoff,M.O., Schwartz,R.M. and Orcutt,B.C., "A Model of Evolutionary Change in Proteins," *Atlas of Protein Sequence and Structure,* Vol.5, No.3, Nat. Biomed. Res. Found., Washington DC, pp.345-352 (1978).

[19] Ishikawa,M., Toya,T., Totoki,Y. and Konagaya,A., "Parallel Iterative Aligner with Genetic Algorithm," *Proc. AI and Genome Workshop in 13th Int'l. Joint Conf. Artifi. Intelli.*, pp.13-22 (1993).

[20] Goldberg,D.E., "Genetic Algorithms in Search, Optimization, and Machine Learning," Addison-Wesley (1989).

[21] Konagaya,A. and Kondou,H., "Stochastic Motif Extraction using a Genetic Algorithm with the MDL Principle," *Proc. 26th Annu. Hawaii Int'l. Conf. Syst. Sci.*, Vol.1, pp.746-755 (1993).

[22] Muehlenbein,H. and Schlierkamp-Voosen,D., "Predictive Models for the Breeder Genetic Algorithm: Continuous Parameter Optimization," *Evolutionary Computation,* Vol.1, pp.25-49 (1993).

[23] Barber,A.M. and Maizel Jr,J.V., "SequenceEditing-Aligner: A multiple sequence editor and aligner," *Gene Anal. Tech.*, Vol.7, pp.39-45 (1990).

[24] Schuler,G.D., Altschul,S.F. and Lipman,D.J., "A Workbench for Multiple Alignment Construction and Analysis," *PROTEINS,* Vol.9, pp.180-190 (1991).

[25] Ishikawa,M., Totoki,Y., Tanaka,R. and Hirosawa,M., "Multiple Sequence Alignment Editor Featured by Constraint-based Parallel Iterative Aligner," *Proc. 3rd Int'l. Conf. Bioinformatics and Genome Research,* (1994).

[26] Hargrave,P.A., "Seven-helix receptors," *Current Opinion Struc. Biol.,* Vol.1, pp.575-581 (1991).

[27] Blanck,A. and Oesterhelt,D., "The halo-opsin gene. II. Sequence, primary structure of halorhodopsin and comparison with bacteriorhodopsin," *EMBO J.*, Vol.6, pp.265-273 (1987).

[28] Bairoch,A., "PROSITE: a dictionary of sites and patterns in proteins," *Nucleic Acids Res.,* Vol.19, pp.2241-2245 (1992).

[29] Sneath,P.H.A and Sokal,R.R., "Numerical Taxonomy," Freeman and Company (1973).

[30] Saitou,N., and Nei,M., "The neighbor-joining method: A new method for reconstructing phylogenetic trees," *Mol. Biol. Evol.,* Vol.4, pp.406-425 (1987).

[31] Ishikawa,M., Toya,T., Totoki,Y. and Tanaka,R., "Multiple RNA-Sequence Alignment Considering Stem Regions," *Proc. Genome Informatics Workshop V,* Universal Academy Press (1994).