Multiple Sequence Alignment Using Tabu Search

Tariq Riaz, Yi Wang, Kuo-Bin Li

Bioinformatics Institute, 30 Biopolis Street, Singapore 138671

kuobin@bii.a-star.edu.sg

Abstract

Tabu search is a meta-heuristic approach that is found to be useful in solving combinatorial optimization problems. We implement the adaptive memory features of tabu search to align multiple sequences. Adaptive memory helps the search process to avoid local optima and explores the solution space economically and effectively without getting trapped into cycles. The algorithm is further enhanced by introducing extended tabu search features such as intensification and diversification. It intensifies by bringing the search process to poorly aligned regions of an elite solution, and softly diversifies by moving from one poorly aligned region to another. The neighborhoods of a solution are generated stochastically and a consistency-based objective function is employed to measure its quality. The algorithm is tested with the datasets from BAliBASE benchmarking database. We have observed through experiments that for datasets comprising orphan sequences, divergent families and long internal insertions, tabu search generates better alignment as compared to other methods studied in this paper. The source code of our tabu search algorithm is available at http://www.bii.a-star.edu.sg/~tariq/tabu/.

Keywords: tabu search, multiple sequence alignment, combinatorial optimizations

1 Introduction

The multiple sequence alignment is a well known tool in the field of molecular biology. It has wide range of applications that include finding the characteristic motifs among biological sequences, backtracking the evolutionary paths through sequence similarity, identifying the consensus sequence, and predicting the secondary and tertiary structures.

The methods capable of producing optimal alignment are already available but those are feasible only to small number of sequences with short length (Gupta, Kececioglu *et al.* 1995). With greater number of sequences, the problem of multiple sequence alignment is extremely hard to solve because of large number of conformations and enormous amount of computations required.

Numerous techniques have been used to align multiple sequences (Notredame 2002). The most popular approach is the progressive alignment whereby the alignment is generated by aligning the two closely related sequences first and successively adding the remaining sequences one by one (Feng and Doolittle 1987). The progressive alignment method is based on greedy approach and the error introduced once during the alignment cannot be corrected subsequently. This, in the end might generate a local optimal solution. Nonetheless progressive alignment has the advantage of high performance in terms of speed. The other class of algorithms falls into the category of iterative algorithms. The iterative approach starts off with an initial alignment, and based on certain evaluation mechanism iteratively improves the alignment. Simulated annealing (Ishikawa, Toya *et al.* 1993, Kim, Pramanik *et al.* 1994) and genetic algorithm (Notredame and Higgins 1996, Zhang and Wong 1997) are two iterative optimization techniques that have been applied to align the multiple sequences. Other iterative approaches include hidden Markov model-based technique (Karplus and Hu 2001) and dead-end elimination procedure (Lukashin and Rosa 1999).

Tabu search (TS) (Glover, Taillard *et al.* 1993, Golver and Laguna 1997) is an iterative heuristic technique that is found to be capable of solving combinatorial optimization problems. It has been vastly applied to solve the problems in a variety of fields. Some example problems are protein conformation of lattice model (Pardalos and Liu 1997), multiprocessor scheduling (Thesen 1998), redundancy allocation problem (Kulturel-Konak, Smith *et al.* 2003), network design (Crainic and Gendreau 2002), knapsack sharing problem (Hifi 2002), and feature selection (Zhang and Sun 2002). The essential idea underlying TS is its deterministic approach that can escape local optima by using a list of prohibited solutions known as tabu list. A number of tabu criteria are employed to mark the solutions as tabu. Two commonly used tabu criteria are recency and frequency. The recency based tabu list is usually implemented as short-term memory structure, and it keeps track of solutions changed during the recent past. The frequency based tabu list, on the other hand is manifested as long-term memory structure and it tracks the frequency of each visited solution. Irrespective of short-term or long-term memory structure for tabu list, TS normally avoids revisiting the solutions in the tabu list. This helps TS to escape entrapment into cycles and thus saves computational time. The tabu criteria can however be overridden in special cases specified by aspiration criteria. The aspiration criteria allow backtracking to already visited solutions if they can lead to a new path toward better solutions. Besides the basic notions of tabu list and aspiration criteria, TS has two other relatively sophisticated features, i.e. intensification and diversification. Intensification works by recording elite solutions along the search process and examining their neighborhood more thoroughly. Diversification on the other hand encourages exploration of unvisited regions that might have not been reached otherwise (Golver and Laguna 1997). Diversification is performed by redirecting the search path to a solution entirely different from the ones that are already visited.

We have designed and implemented an algorithm based on the tabu search to find the global alignment of multiple sequences. In this paper, we present the methods, the algorithm and the test results. The algorithm is used in conjunction with COFFEE objective function (Consistency based Objective Function For alignmEnt Evaluation) (Notredame, Holm *et al.* 1998), which measures the quality of the alignments produced during the search process. The design of our software system is flexible and can work with any other objective function. We performed the tests on the sequence datasets taken from BAliBASE (Benchmark Alignment dataBASE) (Thompson, Plewniak *et al.* 1999). In order to show the robustness of tabu search we executed the tests in a variety of arrangements such as tabu search vs. simple iterative search and aligned vs. unaligned initial solution. Tabu search vs. simple iterative search set of tests is meant to observe the impact of the tabu characteristics on the alignment quality whereas aligned vs. unaligned initial solution tests are aimed at studying the effect of initial solution on quality of the final solution. Our premise is that the algorithm should be robust enough to reach a near optimal alignment quite irrespective of the kind of initial solution. In later sections we will present to what extent the results agree with our premise.

2 Methods

Tabu search works by starting from an initial (current) solution, and iteratively explores the neighborhood of current solution by generating the moves. We implement the recency based tabu criteria that require the short-term memory structure. In each iteration, the moves are evaluated through the Objective Function (OF) and the best move, provided it is not a tabu move is selected and applied to the current solution. This produces a new current solution for the next iteration. The applied move is added to the tabu list and same move is not allowed for a specified number of iteration called tabu tenure. The tenure is decremented for all the moves in tabu list in every iteration, and a move is removed from the tabu list once its tabu tenure hits zero. We have enforced an aspiration criterion that permits a tabu move to be taken if OF value of the move is the greatest of all those moves observed previously in the search process. We believe that overriding tabu criteria in this way might lead to a new path toward better solutions. Our algorithm runs in two phases. First phase guides the search process iteratively to the point where the solution stabilizes i.e. no improvement in the quality of solution is seen for a specified number of iterations. At this point the search enters into second phase, which carries out intensification and diversification.

Intensification and diversification strategies are incorporated by recording the elite solutions, which are actually the best solutions found in each iteration. While in intensification and diversification mode, the algorithm intelligently explores the neighborhood of elite solutions. It work by identifying the attractive regions of elite solution and searches them thoroughly. The termination condition of the algorithm is based on a set of heuristics. Essentially the algorithm stops when the intensification and diversification strategies fail to identify new attractive regions and the solution stabilizes too. Figure 1 summarizes our tabu search algorithm for multiple sequence alignment.

2.1 Initial solution

The generation of an initial solution is an important step towards getting a final improved alignment. A good initial solution can effectively converge faster and hence cut the computational cost. We tested the algorithm with two types of initial solutions.

2.1.1 Unaligned initial solution

Since the algorithm has no mechanism of introducing new gaps during the iterative search, the initial solution is formed by inserting fixed number of gaps into the sequences at regular intervals. The insertion of gaps this way adds no alignment value to the sequences. Simple heuristic is used to identify the best length of gaps and the intervals. One of the heuristics is to keep the length of gaps proportional to the length of the sequences. The sequences that are shorter in length are padded with extra gaps to make them equal in length.

Figure 2(a) shows an example of the unaligned initial solution where a patch of gaps of length 3 is inserted at the regular intervals of size 10.

```
Initialization:    Generate initial solution $\mathbf{S_i}$.
                   Set the current solution $\mathbf{S_c}$, and the elite solution $\mathbf{S_e}$ to $\mathbf{S_i}$.
                   Initialize tabu list TL to 0.
                   Initialize candidate move list ML to 0.
                   Set the tabu tenure to an arbitrary number.
                   Define aspiration criteria.

Generation:        Generate single sequence and block moves on solution $\mathbf{S_c}$ and add to candidate move list ML
Evaluation:        Evaluate COFFEE objective function for each move in ML.
Selection:         Select the best move from ML.
                   if the selected move is not in TL, apply the move;
                   else if the selected move is in TL and aspiration criteria is met, apply the move;
                   else remove the selected move from ML and go to Selection.

                   Update the tabu list.
                   Establish and update the tabu tenure.
                   Update the long-term memory for elite solution $\mathbf{S_e}$ with the best solution found so far.
                   Repeat the steps Generation, Evaluation and Selection until solution $\mathbf{S_c}$ stabilizes.

Intensification and
Diversification:   Operate on solution $\mathbf{S_e}$ to identify all the poorly aligned regions.
                   Diversify from one poorly aligned region to another and repeat the steps Generation,
                   Evaluation and Selection on each poorly aligned region until the region stabilizes.
                   Repeat Intensification and Diversification until the solution $\mathbf{S_e}$ stabilizes.

End:               $\mathbf{S_e}$ holds the final solution.
```

Figure 1. The tabu search algorithm for multiple sequence alignment.

2.1.2 Aligned initial solution

Aligned initial solution is generated by computing the progressive alignment using an algorithm similar to the one proposed by Feng and Doolittle (Feng and Doolittle 1987). Figure 2(b) shows an example of the aligned initial solution. The steps to construct the aligned initial solution are as follows:

1. Calculate a distance matrix of all N (N-1)/2 pairwise distances for the N input sequences by a global alignment algorithm, for example, the Needleman-Wunsch algorithm (Needleman and Wunsch 1970).
2. Construct a guide tree according to the distance matrix by linking the least distant pairs of sequences followed by successively more distant pairs.
3. Align each node of the guide tree in the order that it is added to the tree until all sequences have been aligned.

```
a)
    ---LKPKILTASR---KIKIKAGFTH---NLEVD--
    ---HVKPYFTKTI---LDMDVVEGSA---ARFDCKV
    ---RVLQVDIVPS---QGEISVGESK---FFLCQV-
    ---AVSKEITNAL---ETWGALGQDI---NLD----
    ---FKIETTPESR---YLAQIGDSVS---L------
b)
    --LKPKILTASRKIKIKAGFTHNLEVD-
    H-VKPYFTKTILDMDVVEGSAARFDCKV
    RVLQVDIVPSQGEISVGESKFFLCQV--
    --AVSKEITNA--LETWGALGQDINLD-
    ---FKIETTPE--SRYLAQIGDSVSL—-
```

Figure 2. **(a).** Unaligned initial solution. **(b)** aligned initial solution.

2.2 Objective function

A global objective function is used to measure the overall alignment quality and to evaluate candidate moves that are to be committed. The objective function used in this paper is COFFEE (Notredame, Holm *et al.* 1998). COFFEE works by first generating the pairwise library of the sequences in the alignment and then it calculates the level of identity between the current multiple alignment and the pairwise library. The global score measuring the quality of the alignment is computed by the following formula.

$$\text{Global Objective Score} = \frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} W_{ij} * Score(A_{ij})}{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} W_{ij} * Len}$$

Where N is the number of sequences; *Len* is the length of the multiple alignment; W_{ij} is the percent identity between the two aligned sequences S_i and S_j; A_{ij} is the pairwise projection of sequences S_i and S_j obtained from the multiple alignment; and $Score(A_{ij})$ is the overall level of identity between A_{ij} and the corresponding pairwise alignment.

2.3 Move strategies

The essential part of our alignment algorithm is the movement of gaps in sequences in such a way that it does not alter the original order of amino acids in the protein sequences. The algorithm performs two kinds of moves: single sequence moves, and block moves.

2.3.1 Single sequence moves

These kinds of moves involve moving a patch of gap(s) of arbitrary length only in one sequence while keeping the remaining sequences in the alignment isolated. The patch of gap(s) as well its new location in the sequence is chosen randomly. Figure 3(a) shows a single sequence move where a patch two gaps in the first sequence is moved to its new random position on the right and the remaining sequences in the alignment stay unaffected.

2.3.2 Block moves

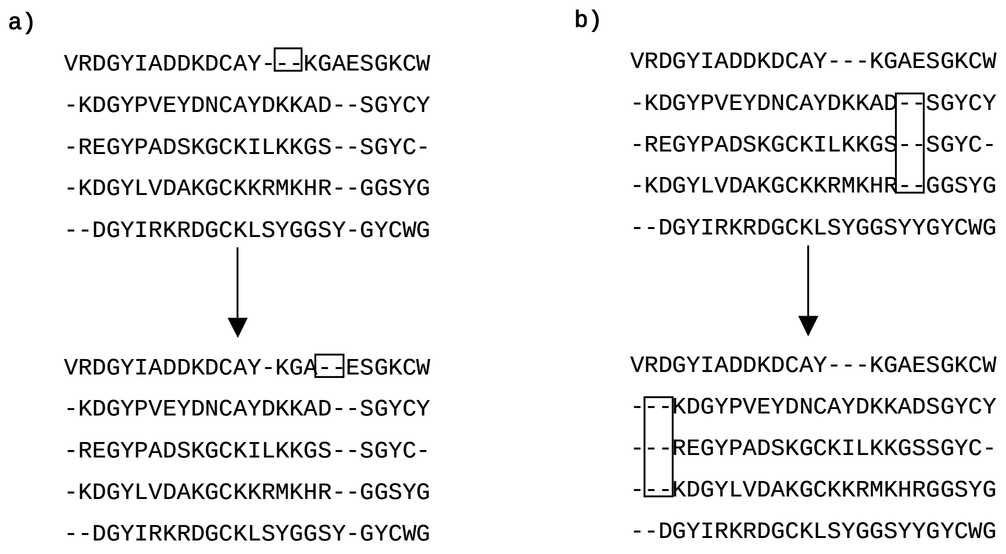
The block moves relocate a rectangle of gaps involving more than one sequence from one position to another in the alignment. The width of the block of gaps can be as low as 1 while the height of the block should be at least 2. A gap block with height 1 is actually a single sequence move. The rationale to maintain single sequence moves separate from block moves is that in most of the cases block moves stand greater chances to be selected as the best move. This is due to the fact that block moves affect larger number of sequences and can improve the alignment greatly. Generation of excessive single sequence moves can unnecessarily waste the CPU time, as each candidate move must be evaluated through the objective function. By distinguishing the single sequence moves from block moves, we can control the generation of each type of moves. The Figure 3(b) shows an example of block move where a block of gaps of width 2 and height 3 is moved to the left to generate a new alignment.

Our implementation of tabu search offers different levels of move generation. Theoretically in each iteration, we can take as many moves as the number of gaps combinations multiplied by the combined length of all sequences. However, the computational resources required for this monumental task are huge and this is almost impractical for large number of sequences in the alignment. We have to resort to such a level of move generation that helps to complete the alignment process in reasonable time. In a typical arrangement, in each iteration the algorithm generates one single sequence move for each sequence and the block moves comprising every possible block of gaps in the alignment.

The moves are generated is a stochastic fashion. In case of single sequence moves, the patch of gap(s) as well as the its new location in the sequence are determined randomly. For block moves, every rectangular block of gaps is moved to some random location, either left or right to its current position

2.4 Intensification and diversification

We propose an improvement in the basic tabu search algorithm by introducing intensification and diversification strategies. Intensification strategies are based on modifying the choice rules that encourage the move combinations and solution features that have been historically found good (Golver and Laguna 1997). Diversification strategies on the other hand encourage the search process to examine unvisited regions and generate solutions that are significantly different from those already visited.



```
a)
VRDGYIADDKDCAY---KGAESGKCW
-KDGYPVEYDNCAYDKKAD--SGYCY
-REGYPADSKGCKILKKGS--SGYC-
-KDGYLVDAKGCKKRMKHR--GGSYG
--DGYIRKRDGCKLSYGGSY-GYCWG

VRDGYIADDKDCAY-KGA--ESGKCW
-KDGYPVEYDNCAYDKKAD--SGYCY
-REGYPADSKGCKILKKGS--SGYC-
-KDGYLVDAKGCKKRMKHR--GGSYG
--DGYIRKRDGCKLSYGGSY-GYCWG

b)
VRDGYIADDKDCAY---KGAESGKCW
-KDGYPVEYDNCAYDKKAD--SGYCY
-REGYPADSKGCKILKKGS--SGYC-
-KDGYLVDAKGCKKRMKHR--GGSYG
--DGYIRKRDGCKLSYGGSYYGYCWG

VRDGYIADDKDCAY---KGAESGKCW
---KDGYPVEYDNCAYDKKADSGYCY
---REGYPADSKGCKILKKGSSGYC-
---KDGYLVDAKGCKKRMKHRGGSYG
--DGYIRKRDGCKLSYGGSYYGYCWG
```

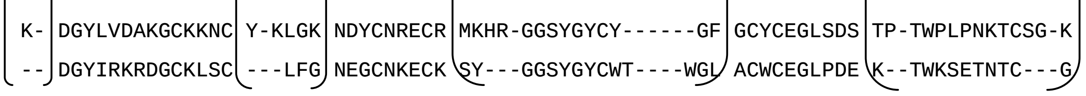
Figure 3. (a) Single sequence move. **(b)** block move.

In this study, we devise a scheme that combines the intensification and diversification strategies. The scheme works by recording elite solutions along the search process. The elite solutions are the best solutions found in each iteration. Intensification strategy starts off once the basic tabu search strategies are unable to improve the quality of alignment any further. The neighborhoods of elite solutions are assumed to be containing attractive regions. Working on those attractive regions could lead to better solutions that are not previously observed. The tabu search algorithm processes elite solutions and seperates the poorly aligned regions from well aligned regions. The concepts of separation is similar to the one used by RASCAL (Thompson, Thierry *et al.* 2003). We implement a local objective function that measures the quality of any local region within the entire multiple alignment. Figure 4 shows an isolation of well aligned regions from poorly aligned regions. The poorly aligned regions are perceived as the attractive regions in the search space that need to be intensified upon. The approach has the dual advantages. It not only saves the computational time by narrowing down the search space to poorly aligned regions but it also helps to keep the well aligned regions intact.

The algorithm implements diversification strategies by softly diversifying from one poorly aligned region to another one. The whole process of intensification and diversification is repeated a number of times until no new poorly aligned regions are identified, and quality of the solution could not be improved any further.

3 Results and discussion

The tabu search algorithm is implemented in SUN Java language and tested on a 1.4GHz Pentium III computer. The source code is available at http://www.bii.a-star.edu.sg/~tariq/tabu/. The platform independent nature of Java language enables the tabu search software to run on any operating system for which the Java virtual machine is available. We analyzed the results of tabu search by comparing the alignments produced by our algorithm with those obtained through other alignment techniques. The sequences samples are taken from BAliBASE (Benchmark Alignment dataBASE) (Bahr, Thompson *et al.* 2001, Thompson, Plewniak *et al.* 1999), which is a database of manually refined multiple sequence alignments categorized by sequence length, similarity and the presence of insertion and N/C-terminal extensions. The database provides the annotation files that mark the core blocks where sequences should be aligned together. The database is divided into five reference sets with a total of 139 datasets. Each dataset consists of protein sequences as few as 4 and as many as 27.



```
VR DGYIADDKDCAYFC ----GR NAYCDEECK K----GAESGKCWYA-GQYGN ACWCYKLPDW VP-IKQKVSGKC---N
-K DGYPVEYDNCAYIC W-N-YD NAYCDKLCK DK---KADSGYCYWV----HI LCYCYGLPDS EP-TKTNGKCKSGK-K
R- EGYPADSKGCKITC F--LTA AGYCNTECT LK---KGSSGYCA------WP ACYCYGLPES VK-IWTSETNKC----
K- DGYLVDAKGCKKNC Y-KLGK NDYCNRECR MKHR-GGSYGYCY------GF GCYCEGLSDS TP-TWPLPNKTCSG-K
-- DGYIRKRDGCKLSC ---LFG NEGCNKECK SY---GGSYGYCWT----WGL ACWCEGLPDE K--TWKSETNTC---G
```

Figure 4: An elite solution isolating well aligned regions from poorly aligned regions. The parts of the sequences inside brackets indicate the attractive regions that are poorly aligned.

The description of each reference set is as follows,

Reference 1: equidistant sequences of similar length

- V1: <25% identity
- V2: 20-40% identity
- V3: >35% identity

Reference 2: family vs. orphans

Reference 3: equidistant divergent families

Reference 4: N/C-terminal extensions

Reference 5: internal insertions

A separate program is used to compute BAliBASE score, which is the percentage of pairs of amino acids that occurs in reference alignments (Thompson, Plewniak *et al.* 1999). The BAliBASE scores computed on alignments generated by tabu search are compared with similar scores obtained from other popular multiple sequence alignment programs including PRRP (Gotoh 1996), ClsustalW (Thompson, Higgins *et al.* 1994), SAGA (Notredame and Higgins 1996), DiAlign (Morgenstern, Dress *et al.* 1996), ML_PIMA (Smith and Smith 1992), MultiAlign (Corpet 1988) and PILEUP8 (GCG Wisconsin Package 10.3 Accelrys Inc. San Diego, CA USA). Except for SAGA and ClustalW, the BAliBASE scores for above mentioned alignment techniques are taken from BAliBASE. The scores for SAGA and ClustalW are calculated by executing their respective software on our systems.

We analyzed the tabu search results from three aspects. The very first set of tests was aimed at to verify the robustness of our algorithm. We ran an extensive set of tests on all the 139 datasets provided by BAliBASE, and computed the scores. The scores using tabu search and other alignment techniques are shown in Table 1. The results are obtained using aligned initial solutions as mentioned previously. Except for those of tabu search, all other scores are taken from BAliBASE.

The average tabu search score for the reference set 1 is slightly below the scores produced by other techniques. Reference set 1 consists of equidistant sequences of similar length. For the rest of the reference sets, tabu search performs relatively better as compared to other methods. In reference set 2 comprising few orphan sequences within a large number of sequences belonging to the same family, and reference set 3 consisting of families of equidistant divergent sequences, tabu search performs better than most of the other techniques. In reference set 4 that contains sequences having N/C-terminal extensions, the performance of tabu search is comparable to the other methods. The reference set 5 having sequences with internal insertions, tabu search outperforms all other methods.

The standard deviation value is one of the lowest for tabu search, which shows its consistent performance across different reference sets. The reference sets ranging from 2 to 5 contain test cases with large number of sequences. Tabu search is able to produce better results than most of the other techniques in these categories, which shows that tabu search can work well with large problem size.

In order to prove objectively that tabu characteristics in the tabu search algorithm is of significance, we performed another set of tests where the results of tabu search are compared with a simple iterative search algorithm. The simple iterative search algorithm works in a similar fashion as tabu search does, except that it does not implement the features like tabu list, tabu tenure, aspiration and intensification/diversification. It searches for the solution

Table 1. BAliBASE score compares tabu search with other methods on BAliBASE benchmark database.

Reference set	Ntcase	PRRP	ClustalW	SAGA	DIALIGN	SB_PIMA	ML_PIMA	MULTALIGN	PILEUP8	Tabu search
V1)	23	0.692	0.615	0.546	0.487	0.536	0.504	0.559	0.571	0.472
V2)	30	0.935	0.933	0.952	0.893	0.910	0.909	0.927	0.911	0.875
V3)	28	0.968	0.974	0.977	0.922	0.961	0.955	0.960	0.962	0.932
Ref1 Average	81	0.865	0.841	0.825	0.767	0.802	0.789	0.815	0.815	0.760
Ref2 Average	23	0.541	0.945	0.954	0.384	0.379	0.371	0.517	0.429	0.889
Ref3 Average	11	0.532	0.723	0.777	0.314	0.267	0.372	0.303	0.323	0.715
Ref4 Average	12	0.323	0.821	0.780	0.853	0.794	0.705	0.292	0.710	0.773
Ref5 Average	12	0.700	0.858	0.868	0.836	0.508	0.572	0.627	0.639	0.905
StDev		0.203	0.080	0.073	0.260	0.242	0.191	0.222	0.203	0.084
W/Average		0.731	0.727	0.708	0.693	0.675	0.673	0.675	0.698	0.803
Average		0.592	0.838	0.841	0.631	0.550	0.562	0.511	0.583	0.808

Ntcase is number of test cases; (V1, V2, V3) are sub-categories inside Ref1. The cells contain the average BAliBASE SP (Sum of pairs) score for each reference set. StDev is standard deviation of the scores; W/Average is the weighted average of the scores. The data for PRRP, DIALIGN, SB_PIMA, ML_PIMA, MULTALIGN and PILEUP8 are taken from the web site of BAliBASE (http://wwwigbmc.u-trasbg.fr/BioInfo/BAliBASE/prog_scores.html). The scores for ClustalW, SAGA and Tabu search are collected by the authors using respective software.

Table 2(a). Tabu search vs. Iterative search for easy to align sequences.

Test case	Tabu search		Simple iterative search		Tabu vs. simple (%)
	Score	*CPU time*	*Score*	*CPU time*	*Score*
1csy_ref1	0.805	20	0.767	37	5
1ycc_ref1	0.927	12	0.879	10	5
1pgtA_ref1	0.948	13	0.590	148	38
1ldg_ref1	0.908	16	0.754	258	17
1mrj_ref1	0.994	51	0.939	90	6
1pii_ref1	0.875	36	0.718	189	18
2cba_ref1	0.843	224	0.715	653	15
3grs_ref2	0.842	8367	0.842	79184	0
1idy_ref3	0.946	1798	0.882	1978	7
1pysA_ref4	0.500	105	0.500	370	0
kinase2_ref5	0.793	14490	0.787	39490	1
1pysA_ref5	0.560	3154	0.522	7420	7
Average	**0.828**	**2357**	**0.741**	**10819**	**11**

Score is the BAliBASE sum of pair score. CPU time is in seconds. Tabu vs. Simple (%) is percentage the tabu search score is better than simple search.

Table 2(b). Tabu search vs. Iterative search for hard to align sequences.

Test case	Tabu search		Simple iterative search		Tabu vs. simple (%)
	Score	*CPU time*	*Score*	*CPU time*	*Score*
1idy_ref1	0.419	5	0.208	4	50
1tgxA_ref1	0.712	10	0.700	18	2
451c_ref1	0.62	11	0.543	14	12
1ton_ref1	0.719	50	0.633	476	12
2pia_ref1	0.648	46	0.333	376	49
1havA_ref1	0.359	79	0.118	373	67
2hsdA_ref1	0.654	30	0.273	182	58
kinase_ref1	0.749	154	0.403	885	46
1gdoA_ref1	0.823	12	0.609	180	26
Average	**0.634**	**44**	**0.424**	**279**	**33**

Score is the BAliBASE sum of pair score. CPU time is in seconds. Tabu vs. Simple (%) is percentage the tabu search score is better than simple search.

iteratively using COFFEE as objective function and terminates when there is no improvement in quality of the alignment for a specified number of iterations.

The results of the comparison between tabu search and simple iterative search are shown in Table 2(a) and 2(b). We divided the datasets into two categories. One category comprises the test cases that are easy to align while the other contains the test cases that are hard to align. The perception of a test case to be easy or hard to align is based on BAliBASE score of the alignment produced by tabu search in the first set of tests. The low BAliBASE score indicates the test case to be hard to align while high BAliBASE score suggests the test case to be easy to align.

It is obvious to notice from Table 2 (a) and 2(b) that tabu search produces alignments of much better quality than that of simple iterative search in both the specified categories. In case of easy to align test cases, tabu search performs on average 11% better than simple iterative search. For hard to align test cases tabu search performs even better i.e. alignments score on average is 33% greater than that of generated by simple iterative search.

Table 3(a). Alignment scores for small datasets using aligned vs. unaligned initial solutions.

Test case	Nseq	Len	Aligned initial solution				Unaligned initial solution			
			Initial score	*Final score*	*Initial vs. final (%)*	*CPU time*	*Initial score*	*Final score*	*Initial vs. final (%)*	*CPU time*
1aboA_ref1	5	63	0.442	0.482	9	3	0.142	0.458	223	15
1csp_ref1	5	67	0.885	0.935	6	227	0.136	0.913	571	299
1wit_ref1	5	96	0.381	0.827	117	7	0.083	0.335	304	52
1tgxA_ref1	4	59	0.692	0.712	3	10	0.306	0.724	137	12
1ad2_ref1	4	207	0.872	0.981	13	13	0.023	0.930	3943	167
1aym3_ref1	4	233	0.875	0.923	5	249	0.119	0.880	639	166
1gdoA_ref1	4	247	0.609	0.823	35	11	0.187	0.614	228	295
1ldg_ref1	4	310	0.707	0.908	28	16	0.062	0.863	1292	258
1amk_ref1	5	248	0.958	0.993	4	498	0.172	0.928	440	513
1led_ref1	4	236	0.846	0.851	1	329	0.110	0.922	738	261
1cpt_ref1	4	401	0.453	0.755	67	49	0.027	0.296	996	442
1dlc_ref1	5	583	0.797	0.928	16	196	0.079	0.817	934	1221
1fieA_ref1	5	670	0.754	0.972	29	1004	0.074	0.676	814	1130
1ad3_ref1	4	436	0.947	0.981	4	446	0.226	0.983	335	345
Average	**4**	**275**	**0.730**	**0.862**	**24**	**218**	**0.125**	**0.739**	**828**	**370**

Nseq is number of sequences in each test case. Len is the average length of the sequences in the test case. Initial and Final Score are the BAliBASE sum of pair scores. Initial vs. Final (%) is the percentage improvement from initial to final alignment. CPU time is in seconds.

Table 3(b). Alignment scores for large datasets using aligned vs. unaligned initial solutions.

Test case	NSeq	Len	Aligned initial solution				Unaligned initial solution			
			Initial score	*Final score*	*Initial vs. final (%)*	*CPU time*	*Initial score*	*Final score*	*Initial vs. final (%)*	*CPU time*
1cpt_ref2	15	383	0.628	0.845	35	20002	0.043	0.071	65	50882
1csy_ref2	19	83	0.752	0.767	2	8732	0.062	0.111	79	5538
1havA_ref2	16	218	0.769	0.898	17	34084	0.069	0.189	174	22857
1idy_ref2	19	57	0.928	0.981	6	3530	0.584	0.694	19	2932
1idy_ref3	27	54	0.534	0.946	77	1798	0.124	0.237	91	4210
1ubi_ref3	22	82	0.234	0.599	156	3636	0.201	0.226	12	11265
1wit_ref3	19	92	0.606	0.803	33	15396	0.113	0.224	98	3923
1lkl_ref4	8	250	0.743	0.750	1	1304	0.002	0.004	100	10859
2cba_ref5	8	258	0.791	0.887	12	2236	0.054	0.104	93	16360
1thm_ref5	11	194	0.794	0.811	2	6447	0.044	0.113	157	15090
Average	**16**	**167**	**0.678**	**0.829**	**34**	**9717**	**0.130**	**0.197**	**89**	**14392**

Nseq is number of sequences in each test case. Len is the average length of the sequences in the test case. Initial and Final Score are the BAliBASE sum of pair scores. Initial vs. Final (%) is the percentage improvement from initial to final alignment. CPU time is in seconds.

Table 2 also shows that tabu search converges to a solution faster as compared to simple iterative search. In both the given categories, time taken by simple iterative search is about five times of the time consumed by tabu search to converge.

The third set of tests is meant to observe the impact of initial solution on quality of the final alignment produced by tabu search. Two types of initial solutions, aligned and unaligned were considered for these tests. As mentioned in the previous sections, unaligned initial solution is formed by simply inserting fixed number of gaps at regular intervals in the initial sequences while aligned initial solution is obtained by a method similar to progressive alignment. The test cases are divided into two categories based on their size. The size is determined by the number and the length of sequences in test cases. The results are shown in Tables 3(a) and 3(b).

For small datasets having on average 4 sequences and the length 275, tabu search is able to achieve alignment of good quality even when the sequences are not pre-aligned. It works as a huge quality improver for small datasets i.e. the improvement on average is more than 800%.

For large datasets that contain on average 16 sequences, tabu search falls short of producing a good alignment when the initial solutions are totally unaligned. The improvement is only around 90%, which is much less than what is required to produce a close to good alignment.

Table 3 also lists the CPU time for various datasets. The CPU time depends not only on the size of the data but also on its complexity, i.e. the number of gaps and the similarity of the sequences. On average the CPU time for test cases with 4 sequences is about 3 minutes whereas for tests with 16 sequences is about 160 minutes. In both categories of small and large datasets, CPU time is larger when working with unaligned initial solution. However the difference is only of the order of around 1.5.

4 Conclusion

The objective of this study is to validate the effectiveness of tabu search and evaluate it as compared to other commonly used techniques for multiple sequence alignment. We have observed through experiments that for test cases comprising orphan sequences, divergent families and N/C-terminal extensions, tabu search performs better that most of the other methods studied in this paper. However, for long internal insertions, the alignments generated by tabu search are the best of all.

We have demonstrated that tabu characteristics of the tabu search help it to find such good solutions that are otherwise hard to be reached. Experiments show that tabu search is not only good at finding solutions for test cases that are easy to align, but it also performs well for the test cases that are relatively hard to align. Its performance with latter is found even more convincing.

The initial solution has proven to be of critical importance when the sequences are aligned with tabu search. For smaller test cases with 4 to 5 sequences of average length 275, tabu search is able to improve the quality of an unaligned initial solution up to the level, which is comparable with the quality of alignment generated from pre-aligned initial solution. However, this phenomenon is not observed for larger test cases of size averaging at around 16. For larger test cases, tabu search is not able to find a good solution when started from a totally unaligned initial solution.

The major cause of lower alignment quality for larger test cases is the huge neighborhood size, and it takes enormously long time to evaluate all the combinations and conformations. In order to complete the search in a reasonable time the algorithm works only on a subset of each neighborhood. Because tabu search works well for smaller test cases with limited neighborhood size, this leads to the conclusion that given the computational time and resources tabu search can find near optimal solutions for larger test cases too.

There is still vast room for improvement in the tabu search software. Objective function (OF) is one of the areas that can be improved upon. It's the OF that leads the search process to a good or bad alignment. If the OF is not a good measure of multiple alignment quality, the alignment might not be the optimal. We use consistency based OF that checks the level of identity of a multiple alignment with corresponding pairwise sequence library. Nonetheless our software is compatible with any user-defined OF, such as sum-of-pairs (Carrillo and Lipman 1988) and NorMD (Thompson, Plewniak *et al.* 2001).

Tabu search comes with a number of parameters that can be experimented with to observe the respective effect on the search process. The parameters like tabu list size, tabu tenure, termination criteria, and neighborhood size can have a direct influence on the quality of the final alignment.

With larger number of sequences in test cases, the problem of multiple alignment turns into a large-scale combinatorial problem. A huge number of solution combinations need to be evaluated that causes the software to take long time to generate the alignment. This problem can be overcome by introducing parallel computing in the algorithm.

5 Acknowledgement

We would like to thank Dr. Espen Enerly and Sheng Zeng for useful discussions and Atif Shahab for his contribution to the design work of tabu search software.

6 References

Bahr, A., Thompson, J. D., Thierry, J. C., and Poch, O. (2001): BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic Acids Res.*, **29**(1): 323-6.

Carrillo, H. and Lipman, D. J. (1988): The multiple sequence alignment problems in biology. *SIAM J. Appl. Math.*, **48**: 1073-1082.

Corpet, F. (1988): Multiple sequence alignment with hierarchical clustering. *Nucleic Acids Res.*, **16**: 10881-90.

Crainic, T. G. and Gendreau, M. (2002): Cooperative Parallel Tabu Search for Capacitated Network Design. *J. Heuristics*, **8**(6): 601-27.

Feng, D. F. and Doolittle, R. F. (1987): Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.*, **25**(4): 351-60.

Glover, F., Taillard, E., and de Werra, D. (1993): A User's Guide to Tabu Search. *Ann. Oper. Res.*, **41**: 3-28.

Golver, F. and Laguna, M. (1997): *Tabu Search*. Boston, Kluwer Academic Publishers.

Gotoh, O. (1996): Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *J. Mol. Biol.*, **264**(4): 823-38.

Gupta, S. K., Kececioglu, J. D., and Schaffer, A. A. (1995): Improving the practical space and time efficiency of the shortest-paths approach to sum-of-pairs multiple sequence alignment. *J. Comput. Biol.*, **2**(3): 459-72.

Hifi, M. (2002): An Efficient Algorithm for the Knapsack Sharing Problem. *Comput. Optim. Appl.*, **23**(1): 27-45.

Ishikawa, M., Toya, T., Hoshida, M., Nitta, K., Ogiwara, A., and Kanehisa, M. (1993): Multiple sequence alignment by parallel simulated annealing. *Comput. Appl. Biosci.*, **9**(3): 267-73.

Karplus, K. and Hu, B. (2001): Evaluation of protein multiple alignments by SAM-T99 using the BAliBASE multiple alignment test set. *Bioinformatics*, **17**(8): 713-20.

Kim, J., Pramanik, S., and Chung, M. J. (1994): Multiple sequence alignment using simulated annealing. *Comput. Appl. Biosci.*, **10**(4): 419-26.

Kulturel-Konak, S., Smith, A. E., and Coit, D. W. (2003): Efficiently solving the redundancy allocation problem using tabu search. *IIE Transactions*, **35**: 515-26.

Lukashin, A. V. and Rosa, J. J. (1999): Local multiple sequence alignment using dead-end elimination. *Bioinformatics*, **15**(11): 947-53.

Morgenstern, B., Dress, A., and Werner, T. (1996): Multiple DNA and protein sequence alignment based on segment-to-segment comparison. *Proc. Natl. Acad. Sci. U S A*, **93**(22): 12098-103.

Needleman, S. B. and Wunsch, C. D. (1970): A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**(3): 443-53.

Notredame, C. (2002): Recent progress in multiple sequence alignment: a survey. *Pharmacogenomics*, **3**(1): 131-44.

Notredame, C. and Higgins, D. G. (1996): SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Res.*, **24**(8): 1515-24.

Notredame, C., Holm, L., and Higgins, D. G. (1998): COFFEE: an objective function for multiple sequence alignments. *Bioinformatics*, **14**(5): 407-22.

Pardalos, P. M. and Liu, X. (1997): Protein Conformation of a Lattice Model Using Tabu Search. *J. Global Optim.*, **11**(1): 55-68.

Smith, R. F. and Smith, T. F. (1992): Pattern-induced multi-sequence alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative protein modelling. *Protein Eng.*, **5**(1): 35-41.

Thesen, A. (1998): Design and Evaluation of Tabu Search Algorithms for Multiprocessor Scheduling. *J. Heuristics*, **4**(2): 141-60.

Thompson, J. D., Higgins, D. G., and Gibson, T. J. (1994): CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, **22**(22): 4673-80.

Thompson, J. D., Plewniak, F., and Poch, O. (1999): BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics*, **15**(1): 87-8.

Thompson, J. D., Plewniak, F., Ripp, R., Thierry, J. C., and Poch, O. (2001): Towards a reliable objective function for multiple sequence alignments. *J Mol Biol*, **314**(4): 937-51.

Thompson, J. D., Thierry, J. C., and Poch, O. (2003): RASCAL: rapid scanning and correction of multiple sequence alignments. *Bioinformatics*, **19**(9): 1155-61.

Zhang, C. and Wong, A. K. (1997): A genetic algorithm for multiple molecular sequence alignment. *Comput. Appl. Biosci.*, **13**(6): 565-81.

Zhang, H. and Sun, G. (2002): Feature Selection using tabu search method. *Patt. Recog.*, **35**(3): 701-11.