A NEW APPROACH FOR ALIGNMENT OF MULTIPLE PROTEINS

XU ZHANG TAMER KAHVECI

Department of Computer and Information Sciences and Engineering, University of Florida, Gainesville, FL, USA, 32611
E-mail: {xuzhang, tamer}@cise.ufl.edu

We introduce a new graph-based multiple sequence alignment method for protein sequences. We name our method HSA (Horizontal Sequence Alignment) for it horizontally slides a window on the protein sequences simultaneously. Current progressive alignment tools build up final alignment by adding sequences one by one to existing alignment. Thus, they have the shortcoming of order-dependent alignment. In contrast, HSA considers all the proteins at once. It obtains final alignment by concatenating cliques of graph. In order to find a biologically relevant alignment, HSA takes secondary structure information as well as amino acid sequences into account. The experimental results show that HSA achieves higher accuracy compared to existing tools on BAliBASE benchmarks. The improvement is more significant for proteins with low similarity.

1. Motivation

Multiple sequence alignment (MSA) of protein sequences is one of the most fundamental problems in computational biology. It is an alignment of three or more protein sequences. MSA is widely used in many applications such as phylogenetic analysis [17] and identification of conserved motifs [22].

The alignment of two sequences with maximum score can be found in $O(L^2)$ time using dynamic programing [14], where L is the length of the sequences. This algorithm can be extended to align N sequences, but requires $O(L^N)$ time [12,18]. A variety of heuristic MSA algorithms have been developed. Most of them are based on progressive application of pairwise alignment. They build up alignments of larger numbers of sequences by adding sequences one by one to existing alignment [5]. We call this a *vertical alignment* since it progressively adds a new sequence (i.e., row) to a consensus alignment. These methods have the shortcoming that the order of sequences to be added to existing alignment significantly affects the quality of the resulting alignment. This problem is more apparent when the percentage of identities among amino acids falls below 25%, called the *twilight zone* [3]. The accuracies of most progressive sequence alignment methods drop considerably for such proteins.

In this paper, we consider the problem of alignment of multiple proteins. We develop a graph-based solution to this problem. We name this algorithm HSA (Horizontal Sequence Alignment) as it *horizontally* aligns

sequences. Here, horizontal alignment means that all proteins are aligned simultaneously, one column at a time. HSA first constructs a directed-graph. In this graph, each amino acid of the input sequences maps to a vertex. An edge is drawn between pairs of vertices that may be aligned together. The graph is then adjusted by inserting gap vertices. Later, this graph is traversed to find high scoring cliques. Final alignment is obtained by concatenating these cliques. The experimental results show that HSA finds better alignments on average than existing popular tools. The quality improvement is much greater for low similarity sequences.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 introduces the algorithm in detail. Section 4 presents experimental results. Section 5 concludes with a brief discussion.

2. Related Work

Finding the multiple sequence alignment that maximizes the SP (Sum-of-Pairs) score is an NP-complete problem [26]. A variety of heuristic algorithms have been developed to overcome this difficulty [21]. These heuristic methods can be classified into four groups: progressive, iterative, anchor-based and probabilistic.

Progressive methods find multiple alignment by iteratively picking two sequences from this set and replacing them with their alignment (i.e., consensus sequence) until all sequences are aligned into a single consensus sequence. Thus, progressive methods guarantee that never more than two sequences are simultaneously aligned. This approach is sufficiently fast to allow alignments of virtually any size. ClustalW [21,23], T-coffee [16], Treealign [7] and POA [11] can be grouped into this class [20].

Iterative methods start with an initial alignment. They then repeatedly refine this alignment through a series of iterations until no more improvements can be made. Depending on the strategy used to improve the alignment, iterative methods can be deterministic or stochastic. Muscle [4] and DIALIGN [13] can be grouped into this class.

Anchor-based methods use local motifs (short common subsequences) as anchors. Later, the unaligned regions between consecutive anchors are aligned using other techniques. MAFFT [10], Align-m [25], L-align [8], Mavid, PRRP [6], DIALIGN [13] belongs to this class.

Probabilistic methods pre-compute the substitution probabilities by analyzing known multiple alignments. They use these probabilities to maximize the substitution probabilities for a given set of sequences. Probcons [3], Hmmt [19], SAGA [15], and Muscle [4] can be grouped into this class.

3. Proposed method

The underlying assumption of HSA is that the residues that have same SSE types have more chance to be aligned compared to the residues that

have different SSE types. This assumption is verified by a number of real experiments and observations [1,2,9,24].

HSA works in five steps: (1) An initial directed graph is constructed by considering residue information such as amino acid and secondary structure type. (2) The vertices are grouped based on the types of residues. The residue vertices in each group are more likely to be aligned together in the following step. (3) Gap vertices are inserted to the graph in order to bring vertices in the same group close to each other in terms topological position in the graph. (4) A window is slid from beginning to end. The clique with highest score is found in each window and an initial alignment is constructed by concatenating these cliques. (5) The final alignment is constructed by adjusting gap vertices of the initial alignment. Next, we describe these five steps in detail.

3.1. *Constructing initial graph*

This step constructs the initial graph which will guide the alignment later. Let s_1, s_2, $\cdots$, s_k be the protein sequences to be aligned. Let $s_i(j)$ denote the jth amino acid of protein s_i. A vertex is built for each amino acid. The vertices corresponding to different proteins are marked with different colors. Thus, the vertices of the graph span k different colors. If available, Secondary Structure Element (SSE) type (α-helix , β-sheet) of each residue is also stored along with the vertex. For simplicity, SSE types include α-helix , β-sheet, and no SSE information, as shown in Figure 1. Two types of edges are defined. First, a directed edge is included from the vertex corresponding to $s_i(j)$ to $s_i(j+1)$ for all consecutive amino acids. Second, an undirected edge is drawn between pairs of vertices of different colors if their substitution score is higher than a threshold. HSA gets the substitution score from BLOSUM62 matrix. A weight is assigned to each undirected edge as the sum of the substitution score and *typeScore* for the amino acid pair that make up that edge. The *typeScore* is computed from the SSE types. If two residues belong to the same SSE type, then their *typeScore* is high. Otherwise, it is low. We discuss this in more detail in Section 3.2. This policy of weight assignment lets residues with same SSE type or similar amino acids have higher change to be aligned in following steps. We will discuss this in Section 3.4. Figure 1 demonstrates this step on three proteins. The amino acid sequences and the SSEs are shown at the top of this figure. The dotted arrows represent the undirected edges between two vertices of different color, the solid arrows only appear between the vertices corresponding to consecutive amino acids of the same protein and they only have one direction, from left to right.

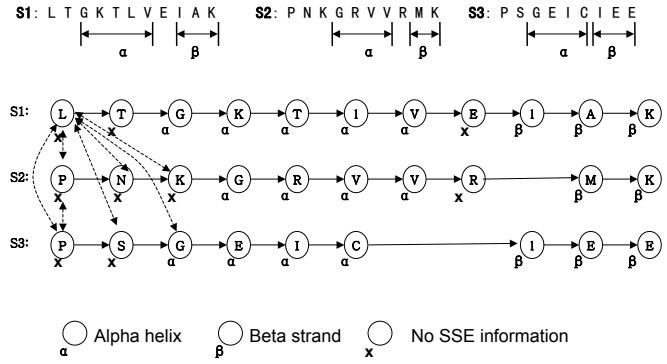

Figure 1. The initial graph constructed for sequence S_1, S_2 and S_3. Each residue maps to a vertex in this graph. The figure shows some edges between the first vertices of the sequences, indicated by dashed arrows. The vertices for different sequences are marked with different colors (colors not shown in figure).

3.2. *Grouping Fragments*

The graph constructed at the first step shows the similarity of pairs of residues. However, multiple alignment involves alignment of groups of amino acids rather than pairs. In this step, we group the *fragments* that are more likely to be aligned together. Here, a fragment is defined by the following four properties: 1) It is composed of consecutive vertices. 2) All the vertices have the same color. 3) All vertices have the same SSE type. 4) There is no other fragment that contains it. For example, in Figure 2, S_1 consists of four fragments: f_1 = LT, f_2 = GKTIV, f_3 = E, and f_4 = IAK. Thus, S_1 can be written as $S_1 = f_1 f_2 f_3 f_4$.

With the knowledge that the fragments with the same SSE type are more likely to be aligned, all sequences are scanned to find fragments with known SSE types. The fragments are then clustered into groups, where each group consists of one fragment from each sequence. To group fragments, we align the fragments first. We use a simplified dynamic programming algorithm by considering each fragment as a residue in the basic algorithm [14]. The score of two fragment pairs is computed from the following formula:

$$totalScore = typeScore - positionPenalty - lengthPenalty$$

The *typeScore* is computed from the SSE types. Fragments with the same SSE type contribute a high score whereas fragments of different SSE types incur penalty. This is because of our assumption that residues with the same SSE type have higher chance to be aligned. Thus *typeScore* is calculated as follows: we check the types of two fragment first and return a number according to the following 5 different conditions. 1) They are the same type of α-helix, we return 4; 2) They are the same type of β-sheet, we return 2; 3) They are the same type of no SSE type, we return 1; 4) They

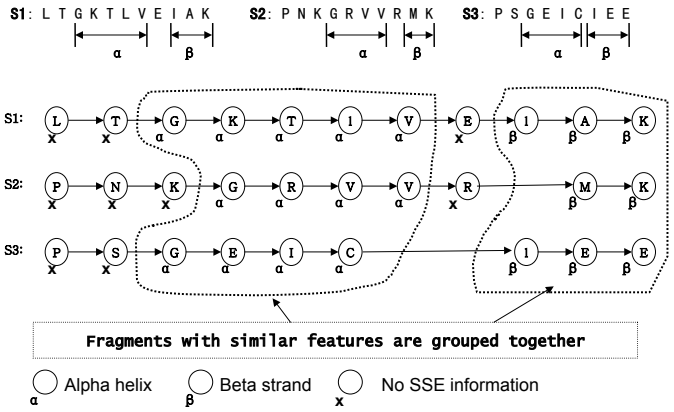

Figure 2. The fragments with similar features, such as SSE types, lengths and positions in original sequences are grouped together.

are α-helix and β-sheet, we return -4; 5) Otherwise, we return 0. The *positionPenalty* is computed as the difference between the positions of two fragments. Here the position of a fragment is the topological position in the original sequence. If two fragments are far away in their sequences, then the pair of them gets a higher penalty. This is because the alignment of such fragments introduce many gaps. The *lengthPenalty* is computed as the difference between the lengths of the two fragments. The length of a fragment is the number of residues it contains. Fragment pairs with similar length will be given smaller penalty. This is because as the lengths the fragment pairs differ more, the number of gap vertices that need to be inserted in the later alignment increases.

Figure 2 demonstrates how HSA groups fragments. Using the example of Figure 1, fragments with same SSE type, similar positions and lengths are clustered into the same group. Two such groups with α-helix and β-sheet are circled in Figure 2.

3.3. *Fragment Position Adjustment*

Once the groups of fragments are determined, we update the graph to bring the fragments in same group close to each other in terms of *vertical* position. Here, *vertical* position corresponds to a position in the topological order of the vertices of the same color. For example, in Figure 3, vertex L in S_1, vertex P in S_2, and vertex P in S_3 are at the same vertical position 1, similarly, vertex T in S_1, vertex N in S_2, and vertex S in S_3 are at the same vertical position 2, etc. As we will discuss later, this process increases the possibility that the vertices in these fragments are aligned.

We update the graph by inserting gap vertices, as shown in Figure 3. First, we compute the number of gap vertices to be inserted based on two

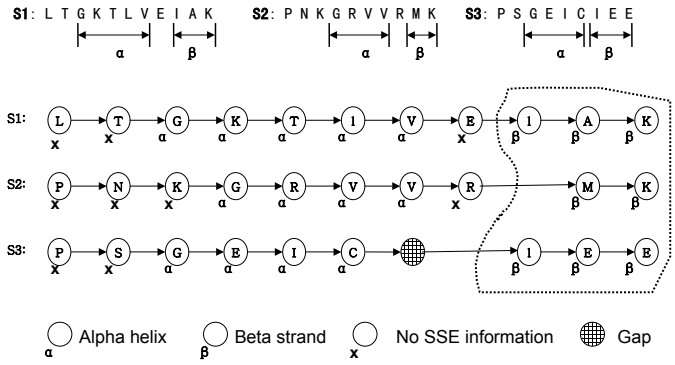

Figure 3. A gap vertex is inserted to let the fragments in same group close to other each other vertically.

factors: 1) The number of residues in fragments. 2) The relative positions of fragments in the same group. Here a good relative position of fragments means that the positions of fragments lead to a high scoring alignment of the vertices in these fragments. We align the vertices in fragments of the same group to compute those positions. Then, we randomly select a position between two consecutive fragment groups. Finally, for each sequence we insert gap vertices at these positions to bring the fragments within the same group together. In Figure 3, a gap vertex is inserted before residue I in S_3 to bring fragments in the group with β-sheet type close to each other.

3.4. *Alignment*

So far, we have prepared the graph for actual alignment by two means. (1) We determined vertex pairs that can be a part of the alignment; (2) We brought sequences to roughly the same size by inserting gap vertices, while keeping similar vertices vertically close. In this step, the sequences are actually aligned by scanning the updated graph in topological order.

As demonstrated in Figure 4, we start by placing a window of width w at the beginning of each sequence. This window defines a subgraph of the graph. Typically, we use $w = 4$ or 6. The example in Figure 4 uses $w = 3$. Next, we greedily choose a clique with the best *expectation score* from this subgraph. We will define the expectation score of a clique later. A clique here is defined as a complete subgraph that consists of one vertex from each color. In other words, if K sequences are to be aligned, a clique corresponds to the alignment of one letter from each of the K sequences. Thus, each clique produces one column of the multiple alignment. For each clique, we align the letters of that clique, and iteratively find the next best clique that 1) does not conflict with this clique, and 2) has at least one letter next to a letter in this clique. This iteration is repeated t times to find t columns.

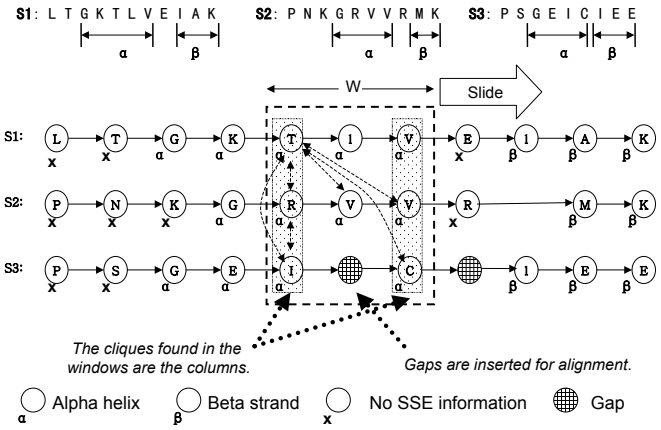

Figure 4. Cliques found in the sliding window (window size = 3) are the columns of the resulting alignment. Gaps are inserted to concatenate these columns.

Typically, $t = 4$. These t cliques define a local alignment of the input sequences. The expectation score of the original clique is defined as the SP score of this local alignment. After finding the highest expectation score clique, we add this clique as a column to existing alignment. We then slide the window to the location which is immediately after the clique found and repeat the same process until it reaches the end of sequences. Each clique defines a column in the multiple alignment. The columns are concatenated and gaps are inserted to align them. Figure 4 illustrates this step, in the window (circled by the dotted rectangle), the highest expectation score clique (the left shadow background marked column) consists of residues T, R, and I in S_1, S_2 and S_3 respectively. Then, the window slides to next location toward the right of the graph (this window is not shown in the Figure 4), and the highest expectation score clique (the right background marked column) in the window consists of residue V, V, and C in S_1, S_2 and S_3 respectively. The two cliques found (marked by shadow background) are two columns in resulting alignment. The resulting alignment is obtained by inserting a gap vertex to S_3.

As mentioned in section 3.1, due to the policy of edge weight assignment, cliques that contain vertices of the same SSE type or similar amino acids have higher score than other possible cliques. Since a clique contains one vertex of each color, finding the best clique does not assure any order for traversal of vertices of different colors. Thus, unlike existing tools, our method is order independent.

3.5. *Gap Adjustment*

After concatenating the cliques in previous step, short gaps may be scattered in the sequence. In this step, the alignment obtained in the

previous step is adjusted by moving the gaps as follows. The sequences are scanned from left to right to find isolated gaps. If a gap is inside a fragment of type α-helix or β-sheet, it is moved outside of that fragment, either before or after. We choose the direction that produces higher alignment score. If a gap is inside a fragment with no SSE type, it is moved next to the neighboring gap only if the movement produces a higher score than the current alignment.

The final alignment is obtained by mapping each vertex in the final graph back to its original residue.

4. Experimental Result

In order to demonstrate the feasibility of our method, we ran it on BAliBASE benchmarks [22] (`http://www-igbmc.u-strasbg.fr/BioInfo/BAliBASE/`). We chose the benchmarks that contain SSE information since our algorithm needs SSE information of sequences. We downloaded ClustalW [21,23], Probcons [3], Muscle [4] and T-Coffee [16] for comparison since they are the most commonly used and the most recent tools. We ran all experiments on a computer with 3 GHz speed, Intel pentium 4 processor, and 1 GB main memory. The operating system is Windows XP.

4.1. *Evaluation of alignment quality*

Alignment of dissimilar proteins is usually harder than the alignment of highly similar proteins. Figures 5, 6 and 7 show the BAliBASE scores of HSA, ClustalW, Probcons, Muscle and T-Coffee on benchmarks with low, medium, and high similarity respectively. From Figure 5, we conclude that for low similarity benchmarks, our method outperforms all other tools. On the average HSA achieves a score of 0.619, which is better than any other tool. HSA finds the best result for 14 out of 21 reference benchmarks. HSA is the second best in 5 of the remaining 7 benchmarks. Figure 6 shows that for sequences with 20-40% identity, HSA is comparable to other tools on average. The average score is not the best one. However, it is only slightly worse than the winner of this group (0.909 versus 0.901). HSA performs best for 2 cases out of 7, including a case for which HSA gets full score. In Figure 7, HSA is higher than other tools on average. HSA performs best on 2 cases out of 7, including a case for which HSA gets full score. High scores of existing methods for sequences with high percentage of identity (Figures 6 and 7) show that there is little room for improvement for such sequences. Proteins at the twilight zone (Figure 5) pose a greater challenge. These results show that our algorithm performs best for such sequences. For medium and high similarity benchmarks, our results are comparable to existing tools.

Figure 8 shows the SP scores of HSA, ClustalW, Probcons, Muscle, T-Coffee and original BAliBASE alignment. On the average, ClustalW,

		ClustalW	Probcons	Muscle	T-Coffee	HSA
Short	1aboA	0.693	0.624	0.616	0.320	0.833
	1idy	0.546	0.679	0.354	0.183	0.700
	1r69	0.655	0.655	0.345	0.234	0.772
	1tvxA	0.223	0.439	0.239	0.235	0.462
	1ubi	0.607	0.464	0.478	0.445	0.648
	1wit	0.630	0.690	0.660	0.707	0.675
	2trx	0.660	0.705	0.712	0.667	0.756
avg		0.573	0.608	0.486	0.398	0.692
Medium	1bbt3	0.512	0.373	0.488	0.440	0.539
	1sbp	0.467	0.585	0.587	0.548	0.590
	1havA	0.222	0.397	0.293	0.256	0.352
	1uky	0.531	0.498	0.535	0.441	0.596
	2hsdA	0.482	0.606	0.748	0.573	0.614
	2pia	0.624	0.700	0.691	0.579	0.608
	3grs	0.377	0.355	0.309	0.383	0.487
avg		0.459	0.502	0.521	0.460	0.541
long	1ajsA	0.388	0.411	0.370	0.379	0.472
	1cpt	0.697	0.719	0.765	0.726	0.810
	1lvl	0.368	0.590	0.451	0.528	0.532
	1pamA	0.405	0.534	0.439	0.461	0.524
	1ped	0.678	0.717	0.746	0.638	0.746
	2myr	0.394	0.568	0.386	0.454	0.630
	4enl	0.664	0.573	0.526	0.582	0.652
avg		0.513	0.587	0.526	0.538	0.624
Avg all		0.515	0.565	0.511	0.465	0.619

Figure 5. The BAliBASE score of HSA and other tools. less than 25 % identity

Muscle, and T-Coffee find the highest SP score for low, medium, and high similarity sequences respectively. However, according to Figures 5 to 7, those methods have relatively low BAliBASE scores. This means that, the alignment with the highest SP score is not necessarily the most meaningful alignment. The SP score of HSA is comparable to other tools on the average. For low similarity sequence benchmarks, the average SP score of HSA is higher than the average SP score of the reference alignment.

4.2. *Performance Evaluation*

The time complexity of our algorithm is $O(W^K N + K^2 M^2)$, where K is the number of sequences, W is the sliding window size, N is the sequence length and M is the number of fragments in a protein sequence. The complexity is computed as follows. The clique, in a window, with the highest expectation score is found in W^K time, and there are N positions for the sliding window. K^2M^2 time is required for aligning fragments. Usually, $M \ll N$. Thus, the total time complexity, in practice, is $O(W^K N)$. Typically W is a small number such as 4. For reasonably small K, $W^K N = O(N)$. Therefore, for small K, the complexity is $O(N)$. As K increases, the complexity increases quickly. However, this complexity is observed

	ClustalW	Probcons	Muscle	T-Coffee	HSA
1fjlA	0.994	0.989	0.971	0.991	1.000
1csy	0.861	0.897	0.799	0.887	0.871
1tgxA	0.833	0.760	0.679	0.817	0.782
1ldg	0.920	0.939	0.954	0.956	0.941
1mrj	0.853	0.925	0.894	0.894	0.925
1pgtA	0.941	0.926	0.912	0.955	0.924
1ton	0.718	0.898	0.865	0.867	0.867
avg	0.874	0.904	0.867	0.909	0.901

Figure 6. The BAliBASE score of HSA and other tools. 20%-40% identity

	ClustalW	Probcons	Muscle	T-Coffee	HSA
1amk	0.978	0.984	0.986	0.988	0.986
1ar5A	0.953	0.956	0.969	0.947	1.000
1led	0.900	0.931	0.950	0.956	0.929
1ppn	0.987	0.983	0.983	0.984	0.981
1thm	0.898	0.900	0.899	0.893	0.910
1zin	0.955	0.975	0.985	0.958	0.978
5ptp	0.948	0.963	0.950	0.961	0.957
avg	0.945	0.956	0.960	0.955	0.963

Figure 7. The BAliBASE score of HSA and other tools. more than 35% identity

only if the subgraphs inside a window is highly connected. It is possible to get rid of the W^K term in the complexity by using longest path methods rather than clique finding methods. The experimental results in Figure 9 coincides with the above conclusion. In general, ClustalW performs best. However, ClustalW achieves this at expense of low accuracy (see Figures 5 to 7). HSA is slower than ClustalW and Muscle. It is, however, faster than Probcons and T-Coffee.

5. Conclusion and Future Work

We developed a new algorithm called HSA for alignment of multiple proteins. HSA is graph-based and differs from existing progressive multiple sequence alignment methods since it builds up the final alignment by considering all sequences at once. HSA first constructs a graph based on the amino acid and SSE types of the residues of the input proteins. It then groups the vertices of this graph with the guide of SSE type information. Next, HSA slides a window from the beginning to the end of the graph and finds cliques in the window. The concatenation of these cliques defines an alignment. HSA obtains the final alignment by adjusting the positions of the gap vertices in the graph.

Experimental results show that HSA achieves high accuracy and still maintains competitive running time. The quality improvement over existing tools is more significant for low similarity sequences. For high or medium similarity sequences, HSA produces comparable accuracy. The running time of HSA is comparable to existing tools.

	REF	ClustalW	Probcons	Muscle	T-Coffee	HSA
Short, <25%	-602	-453	-594	-496	-912	-599
Medium, <25%	-2036	-1466	-2516	-1543	-2461	-1617
Long, <25%	-2989	-1964	-3266	-2291	-2991	-2436
Short, 20%-40%	456	499	508	480	491	493
Medium, 20%-40%	1238	1119	1138	1231	1191	1138
Medium, >35%	3474	3477	3479	3526	3528	3468
AVG overall	-76	202	-208	151	-192	74

Figure 8. The SP score of HSA and other tools.

	ClustalW	Probcons	Muscle	T-Coffee	HSA
Short, <25%	69	238	98	915	194
Medium, <25%	133	638	297	1890	535
Long, <25%	308	1564	584	3240	1191
Short, 20%-40%	62	265	83	1187	421
Medium, 20%-40%	171	695	175	2316	613
Medium, >35%	154	629	136	2502	660
AVG overall	149	672	229	2008	602

Figure 9. The running time of HSA and other tools (measured by milliseconds).

The running time of HSA can be further improved by employing longest path methods rather than cliques. Another future direction is to iteratively refine the alignment by the updating the graph after alignment [6].

References

1. Phil Bradley, Peter S. Kim, and Bonnie Berger. Trilogy: Discovery of sequence-structure patterns across diverse proteins. In *Annual Conference on Research in Computational Molecular Biology*, pages 77 – 88, 2002.
2. Luonan Chen. Multiple Protein Structure Alignment by Deterministic Annealing. In *IEEE Computer Society Bioinformatics Conference (CSB'03)*, volume 00, page 609, 2003.
3. C. Do, M. Brudno, and S. Batzoglou. PROBCONS: Probabilistic Consistency-based Multiple Alignment of Amino Acid Sequences . In *Intelligent Systems for Molecular Biology (ISMB)*, 2004.
4. R.C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, 2004.
5. D.F. Feng and R.F. Doolittle. Progressive Sequence Alignment As A Prerequisite To Correct Phylogenetic Trees. *Journal Of Molecular Evolution*, 25(4):351–360, 1987.
6. O. Gotoh. Significant Improvement in Accuracy of Multiple Protein Sequence Alignments by Iterative Refinement as Assessed by Reference to Structural Alignments. *Journal of Molecular Biology*, 264(4):823–838, 1996.
7. J Hein. A new method that simultaneously aligns and reconstructs ancestral sequences forany number of homologous sequences, when the phylogeny is given. *Molecular Biology and Evolution*, 6(6):649–668, 1989.
8. X. Huang and W. Miller. A time-efficient, linear-space local similarity algorithm. *Advances in Applied Mathematics*, 12:337–357, 1991.
9. Gibrat JF, Madej T, and Bryant SH. Surprising similarities in structure comparison. *Current Opinion in Structural Biology*, 6(3):377–385, 1996.

10. K. Katoh, K. Misawa, K. Kuma, and T. Miyata. MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucleic Acids Research*, 30(14):3059–3066, 2002.
11. C. Lee, C. Grasso, and M.F. Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18(3):452–464, 2002.
12. D.J. Lipman, S.F. Altschul, and J.D. Kececioglu. A Tool for Multiple Sequence Alignment. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 86(12):4412–4415, 1989.
13. B. Morgenstern, K. Frech, A. Dress, and T. Werner. DIALIGN: Finding Local Similarities by Multiple Sequence Alignment. *Bioinformatics*, 14(3):290–294, 1998.
14. S. B. Needleman and C. D. Wunsch. A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. *Journal of Molecular Biology*, 48:443–53, 1970.
15. C Notredame and DG Higgins. SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Research*, 24(8):1515–1524, 1996.
16. C. Notredame, D.G. Higgins, and J. Heringa. T-coffee: a novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, 302(1):205–217, 2000.
17. A. Phillips, D. Janies, and W. Wheeler. Multiple Sequence Alignment in Phylogenetic Analysis. *Molecular Phylogenetics and Evolution*, 16(3):317–330, 2000.
18. Gupta SK, Kececioglu JD, and Schaffer AA. Improving the Practical Space and Time Efficiency of the Shortest-paths Approach to Sum-of-pairs Multiple Sequence Alignment. *Journal of Computational Biology*, 2(3):459, 1995.
19. Eddy SR. Multiple Alignment Using Hidden Markov Models. In *Intelligent Systems for Molecular Biology (ISMB)*, volume 3, pages 114–120, 1995.
20. S.-H. Sze, Y. Lu, and Q. Yang. A polynomial time solvable formulation of multiple sequence alignment. In *International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 204–216, 2005.
21. J.D. Thompson, D.G. Higgins, and T.J. Gibson. CLUSTAL W: Improving the Sensitivity of Progressive Multiple Sequence Alignment through Sequence Weighting, Position-specific Gap Penalties and Weight Matrix Choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.
22. J.D. Thompson, H. Plewniak, and O. Poch. A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Research*, 27(13):2682–2690, 1999.
23. Rene Thomsen, Gary B. Fogel, and Thiemo Krink. Improvement of Clustal-Derived Sequence Alignments with Evolutionary Algorithms. In *Congress on Evolutionary Computation*, volume 1, pages 312–319, 2003.
24. Simossis V.A. and Heringa J. A new method for iterative multiple sequence alignment using secondary structure prediction. In *Intelligent Systems for Molecular Biology (ISMB)*, 2002.
25. I.V. Walle, I. Lasters, and L. Wyns. Align-m–a new algorithm for multiple alignment of highly divergent sequences. *Bioinformatics*, 20(9):1428–1435, 2004.
26. L Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348, 1994.