HAlign 3: Fast Multiple Alignment of Ultra-Large Numbers of Similar DNA/RNA Sequences

Furong Tang,[†,1,2] Jiannan Chao,[†,1,3] Yanming Wei,[4] Fenglong Yang,[5] Yixiao Zhai,[3] Lei Xu,[2] and Quan Zou[*,1,3]

[1]Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, China
[2]School of Electronic and Communication Engineering, Shenzhen Polytechnic, Shenzhen, China
[3]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China
[4]School of Computer Science and Technology, Xidian University, Xi'an, China
[5]School of Medical Technology and Engineering, Fujian Medical University, Fuzhou, China

***Corresponding author:** E-mail: zouquan@nclab.net.
[†]These authors contributed equally to this work.

Associate editor: Keith Crandall

Abstract

HAlign is a cross-platform program that performs multiple sequence alignments based on the center star strategy. Here we present two major updates of HAlign 3, which helped improve the time efficiency and the alignment quality, and made HAlign 3 a specialized program to process ultra-large numbers of similar DNA/RNA sequences, such as closely related viral or prokaryotic genomes. HAlign 3 can be easily installed via the Anaconda and Java release package on macOS, Linux, Windows subsystem for Linux, and Windows systems, and the source code is available on GitHub (https://github.com/malabz/HAlign-3).

Key words: **multiple sequence alignment, suffix tree, center star strategy, common substring, substring selection.**

HAlign, which is implemented with Java (Zou et al. 2015), is a multiple sequence alignment (MSA) tool that uses the center star strategy to speed up the alignment of sequences with high similarity, such as severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) genomes from different strains within the same species (Wu et al. 2020). It first chooses a center sequence from the input, aligns it with the other sequences, and then merges all pairwise alignments into a final MSA (supplementary fig. S1, Supplementary Material online). Furthermore, a suffix tree—a data structure containing all the suffixes of a given string which allows the rapid implementation of many essential string operations (Ukkonen 1995; Baeza-Yates and Gonnet 1996)—is applied to divide and conquer the pairwise alignment. The center star strategy and suffix tree greatly accelerate the alignment procedure, allowing HAlign to process large data sets within a few minutes. In contrast, other state-of-the-art tools, which mainly adopt a progressive method (Feng and Doolittle 1987), take several days to complete the alignment or even fail because of the memory limitations of the hardware or the unacceptable amount of computation time required.

K-ary Tree Modification

In the updated HAlign version, the left-child right-sibling (LCRS) tree was replaced by a k-ary tree to build the suffix tree. The LCRS representation of a suffix tree is space efficient, but it takes several redundant steps when searching for the correct nodes. The suffix tree node structure modified with the k-ary tree considers only four nucleobase types in DNA or RNA sequences. Five pointers in each node were assigned to store the A, C, T/U, G, and N branches, respectively (the fifth pointer represented the unknown base N; fig. 1A). Thus, each visit to a node's particular child would be much more efficient. However, as each node of the k-ary tree stored five pointers regardless of the existence of a branch or not, this tree would incur extra space compared with the LCRS tree (see Supplementary Material online). In the test experiments, the k-ary tree modification almost doubled the searching efficiency while taking up around 12.5% more space across the data sets with different similarities to center sequences (fig. 1B). High-quality SARS-CoV-2 genome sequences all have a 99% similarity to the reference genome. It was established that using 1 million SARS-CoV-2 sequences to search for common substrings through the k-ary tree-modified suffix tree would save nearly 500,000 s (0.5 s for each sequence, as shown in fig. 1B, top panel). Overall, the doubled search efficiency would significantly improve the processing speed when the data sets contain millions of sequences.

Global Substring Selection Algorithm

Searching for common substrings between center and query sequences by suffix tree always results in many redundant substrings. The previous HAlign version was

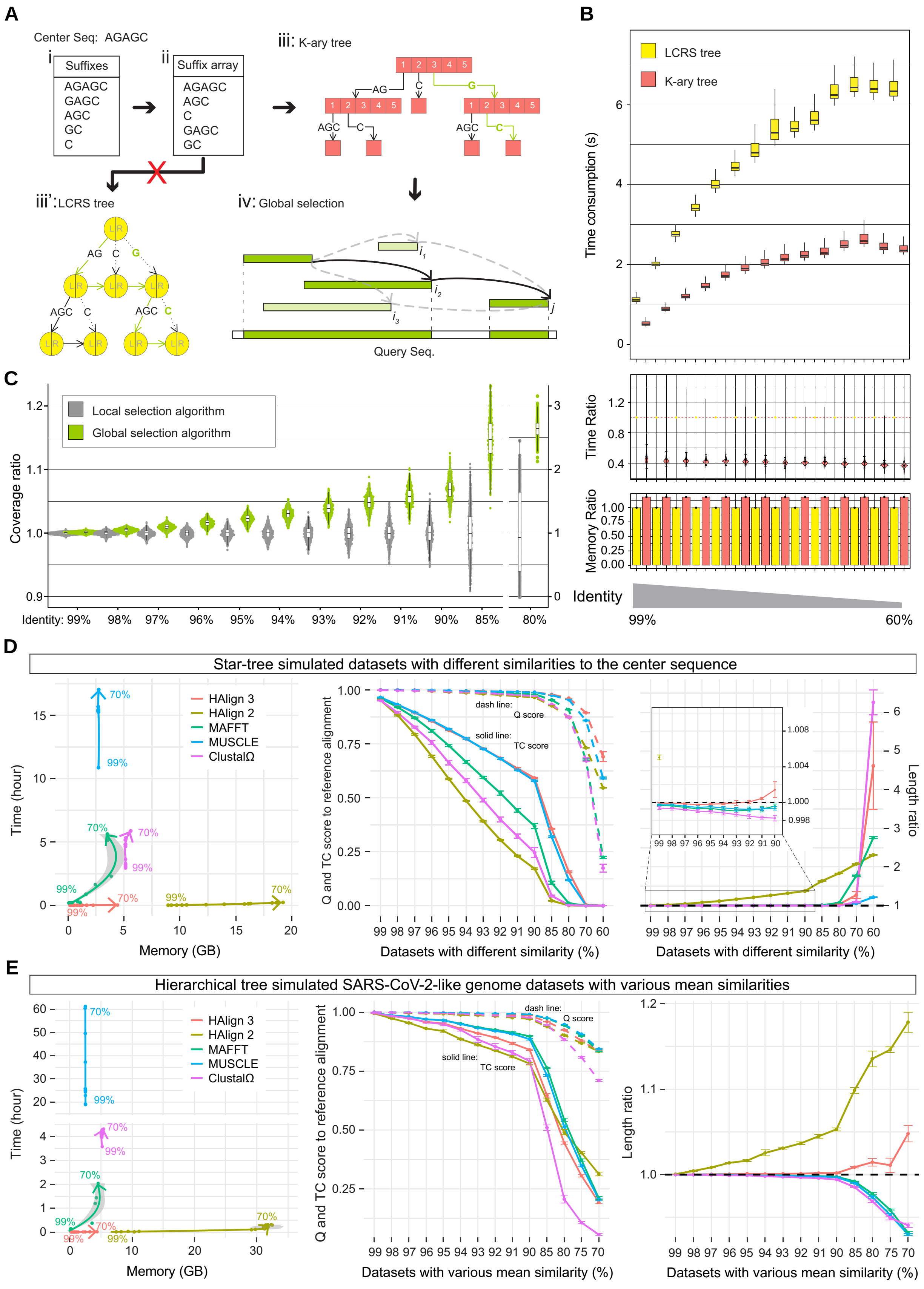

FIG. 1. Illustration of the fast multiple alignment of ultra-large numbers of similar DNA/RNA sequences in HAlign 3. (*A*) Diagram of the k-ary tree, LCRS tree, and global substring selection algorithm. (i) Suffixes of the center sequence (AGAGC). (ii) The suffixes are listed alphabetically to obtain

limited to solving the optimal common substrings by locally selecting the longest substrings and the adjacent substrings with the shortest distances to them. In the updated version, a substring selection algorithm combining the directed acyclic graph (DAG) with dynamic programming (DP) was applied to the global searching for a series of optimal common substrings, which compose the longest path to cover the center/query sequence maximally (fig. 1*A*, Supplementary Material online). This improvement increased the final total length of common substrings between the center and query sequences, especially for data sets with a lower similarity (15% and 150% increase in length for data sets with similarities of 85% and 80%, respectively) compared with the former method (fig. 1*C*). Also, this improvement further decreased the overall consumption as less query sequence lengths needed to be processed by pairwise alignment, which is the most time- and space-consuming procedure. For 80% similarity data sets, around 2,500 bp of pairwise alignment was saved for each sequence (supplementary fig. S2, Supplementary Material online).

Affine Gap Penalty, k-Banded Dynamic Program and Center Sequence Selection

Several other improvements were achieved in this new version of HAlign. The affine gap penalty (Gotoh 1982) was added to the scoring system to make the pairwise alignment more reliable on biological sequences, and the k-band method (Zou et al. 2012) was employed to decrease the time cost of DP. The previous version of HAlign randomly selected a sequence as the center (Su et al. 2017), while the updated version uses the longest input sequence, assuming that this is most likely to retain the information that other sequences might lose by mutation. The new method to select the center sequence was inspired by the greedy strategy used in the CD-HIT (Li et al. 2001) program.

Performance Comparison

Compared with the previous version, in HAlign 3, memory consumption was vastly decreased (fig. 1*D*, left panel) when processing the star-tree simulated data sets with different similarities to the center sequence. When comparing the Q score (the ratio of correctly aligned pairs of letters to aligned pairs in reference) and TC score (the ratio of correctly aligned columns to aligned columns in reference) among five programs, the quality of HAlign 3's alignments was similar to that of MUSCLE's (Edgar 2004) alignments, and these rank as the top two programs (fig. 1*D*, middle panel). The lengths of HAlign 3's alignments of data sets with high similarities (≥92%) were much closer to reference alignment lengths than those of other

the suffix array. The k-ary tree (iii) was used instead of the LCRS tree (iii′) to construct the suffix tree. Coral pink blocks represent the k-ary tree's nodes. Except for the leaf nodes, all the other nodes store five pointers, that is, 1, 2, 3, 4, and 5, which are specifically designed to store the branch of A, C, T, G, and N, respectively. Even if one or more branches do not exist in some cases, they still take up space. The yellow pies represent the nodes of the LCRS tree. Each node stores two pointers: its first child and the next sibling. The solid arrows represent the branches or leaves of trees (iii) and (iii′), while the dashed arrows indicate the parent–child relationship simplified by the representation of the LCRS tree. The green arrows show that the GC substring in green can be found by two steps in the k-ary tree (iii), whereas five steps are necessary for the LCRS tree. (iv) The green bars represent the common substrings between the center and query sequences. Together, arrows (directed edges: only the ones denoting the connectivity from the end of one common substring to the end of another that ends at the right of the former substring are accepted) and bars (nodes) form a DAG. During the DP, the longest path ending at node j was calculated as the maximum of the longest path from the first node to node i_s plus their directed edge from the node i_s to node j. The dark green bars lining the longest path (black arrows) are the final selected common substrings, whose total length (without double counting the overlapping part) is the longest to cover the query sequence. (*B*) Comparison of running time and memory between the k-ary and LCRS methods. Fourteen star-tree simulated data sets (containing 1,000 sequences each) with different similarities were tested. Each center sequence was used to build suffix trees. The running time and memory during the search for the common substrings (relative to the center sequence) of the other 999 query sequences were recorded. Upper panel: the running time consumptions (quartile distribution) for 14 data sets in the order of 99%, 98%, 97%, 96%, 95%, 94%, 93%, 92%, 91%, 90%, 85%, 80%, 70%, and 60%. Middle panel: the running time ratio (mean ± SD) of k-ary to LCRS methods. Both time values were divided by the LCRS consumption time for each sequence. Bottom panel: the running memory ratio (mean ± SD) of k-ary to LCRS methods. The values were normalized by the consumption memory of the LCRS method for each sequence. (*C*) Coverage ratio of the identical bases by selected common substrings of global to local selection algorithms (quartile distribution in boxplot). The total lengths of selected common substrings were first normalized by the known number of identical bases in each group to obtain the percentage of coverage. They were then divided by the mean coverage of the LCRS method. The results of data sets with similarities of 70% and 60% were not shown because there was hardly any common substring between the center and query sequences. (*D*) Performance comparison of HAlign 2 and 3, MAFFT v7.490, MUSCLE v3.8.31, and ClustalΩ v1.2.4 based on star-tree simulated data splits from the data sets used above (nine splits for each data set with similarity of 99%, 98%, 97%, 96%, 95%, 94%, 93%, 92%, 91%, 90%, 85%, 80%, 70%, and 60%). Left panel: comparison of alignment time and memory (mean). The results of data sets with a 60% similarity were not shown because the low similarity sharply increased the running time and memory. Middle panel: comparison of Q and TC scores (mean ± SD). Right panel: comparison of alignment length (mean ± SD). The length of alignments obtained from the five programs was divided by the reference alignment length. The inset shows the zoomed-in details with only one group of HAlign 2 results because the others were out of range. (*E*) Performance comparison of the five programs based on hierarchical tree simulated SARS-CoV-2-like genome data sets with various mean similarities. Fourteen data sets (nine replicates per data set with 100 sequences per replicate) with mean similarities of 99%, 98%, 97%, 96%, 95%, 94%, 93%, 92%, 91%, 90%, 85%, 80%, 75%, and 70% were used. Left panel: comparison of alignment time and memory (mean). Middle panel: comparison of Q and TC scores (mean ± SD). Right panel: comparison of the alignment length (mean ± SD). The length of alignments obtained from the five programs was divided by the reference alignment length. Default parameters running on single-thread were set for all programs in the experiments (*D* and *E*): MAFFT built the guide tree and aligned twice (FFT-NS-2); MUSCLE did the same and then refined the alignment 14 times maximum; ClustalΩ built the guide tree and aligned once; HAlign 2 and 3 built the star-tree and aligned once. The -localMSA mode and suffix tree algorithm were used for HAlign 2, which randomly selected the center sequences (no function to specify the center sequence). HAlign 3 picked the longest simulated sequence as the center sequence.

programs' alignments (fig. 1D, right panel). For the alignment of data sets with long sequence length and high similarity to the center sequence (≥92%), HAlign 3 was the best among the five programs in terms of alignment time, memory, quality, and length.

Three real data sets were also used for further comparisons and it was shown that: (1) 672 human mitochondrial genomes were highly similar to each other (>99%, supplementary fig. S3A, Supplementary Material online), ranging from 16,555 to 16,578 bp (supplementary fig. S3B, Supplementary Material online); (2) 500 SARS-CoV-2 genomes were less similar to each other (>97%, supplementary fig. S3D, Supplementary Material online) ranging from 29,283 to 29,891 bp (supplementary fig. S3E, Supplementary Material online); (3) 647 *Mycobacterium* 23S rRNA sequences were more different from each other (>30%, supplementary fig. S3G, Supplementary Material online) ranging from 1,909 to 3,485 bp (supplementary fig. S3H, Supplementary Material online). The average sum of pairs score (known as aSPscore and corresponding to the sum of pairs score divided by the number of sequence pairs, Supplementary Material online) was used to measure the alignment quality in the absence of reference alignment. The memory consumption and alignment length of HAlign 3 were significantly reduced compared with those of HAlign 2 in all three data sets. The alignment quality of HAlign 3 was comparable with that of MAFFT (Katoh and Standley 2013), MUSCLE, and ClustalΩ (Sievers et al. 2011) in the data sets with an even relatively lower similarity, such as those containing SARS-CoV-2 and 23S rRNA (supplementary fig. S3C, S3F, and S3I, Supplementary Material online). Therefore, in the real data experiment, HAlign 3 can also be comparable with state-of-the-art tools.

To ensure the performance of HAlign 3, SARS-CoV-2-like and mitochondrial-like genomes were further simulated on a hierarchical tree by INDELible v1.03 (Fletcher and Yang 2009; Supplementary Material online). As shown in fig. 1E and supplementary fig. S4B, Supplementary Material online, for the hierarchical tree simulated data sets with various mean similarities (i.e., the mean of the similarities between any two sequences in a given data set), the memory consumption, alignment quality, and length were significantly improved in HAlign 3 compared with the previous version. HAlign 3 could also achieve quite closed alignment quality and length to the top two ranked programs (MAFFT and MUSCLE) for data sets with a higher mean similarity (≥96%) while taking much less time. Similar results were observed in the alignments of simulated genome data sets with preset branch lengths based on real cases but varying scales of tree length (sum of branch lengths) (supplementary fig. S4A, C, and table S1, Supplementary Material online).

Conclusion

Overall, HAlign 3 was much more efficient in processing speed while with minor accuracy loss, which further declined with the increase of sequence similarity. The advantages were more apparent when the number of similar sequences was larger: running on a single-node server, HAlign 3 was the only program that could align 1 million high-quality SARS-CoV-2 genomes successfully in 30 min (supplementary fig. S5, Supplementary Material online). The recently added updates made HAlign a specialized program to deal with ultra-large numbers of similar DNA/RNA sequences, which will be an increasingly common sequence analysis problem as the sequencing technology accelerates the accumulation of intraspecific sequences.

Supplementary Material

Supplementary data are available at *Molecular Biology and Evolution* online.

Acknowledgments

The authors acknowledge the GISAID Community (https://www.gisaid.org) for collecting and sharing the SARS-Cov-2 genome sequences. The work was supported by the National Natural Science Foundation of China (62131004, 62001311, and 61922020), the Special Science Foundation of Yangtze Delta Region Institute (Quzhou) (2020D004), the Research Foundation of Shenzhen Polytechnic (6021310016K, 6021330003K), and the Fellowship of China Postdoctoral Science Foundation (2021M700702).

Data Availability

All test data sets are available on the website (https://github.com/malabz/HAlign-3/tree/main/dataset) and are described in Supplementary material.

References

Baeza-Yates RA, Gonnet GH. 1996. Fast text searching for regular expressions or automaton searching on tries. *J ACM.* **43**: 915–936.

Edgar RC. 2004. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.* **32**:1792–1797.

Feng DF, Doolittle RF. 1987. Progressive sequence alignment as a prerequisitetto correct phylogenetic trees. *J Mol Evol.* **25**: 351–360.

Fletcher W, Yang Z. 2009. INDELible: a flexible simulator of biological sequence evolution. *Mol Biol Evol.* **26**:1879–1888.

Gotoh O. 1982. An improved algorithm for matching biological sequences. *J Mol Biol.* **162**:705–708.

Katoh K, Standley DM. 2013. MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Mol Biol Evol.* **30**:772–780.

Li W, Jaroszewski L, Godzik A. 2001. Clustering of highly homologous sequences to reduce the size of large protein databases. *Bioinformatics* **17**:282–283.

Sievers F, Wilm A, Dineen D, Gibson TJ, Karplus K, Li W, Lopez R, McWilliam H, Remmert M, Söding J, *et al.* 2011. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Mol Syst Biol.* **7**:539.

Su W, Liao X, Lu Y, Zou Q, Peng S. 2017. Multiple sequence alignment based on a suffix tree and center-star strategy: a linear method for multiple nucleotide sequence alignment on spark parallel framework. *J Comput Biol.* **24**:1230–1242.

Ukkonen E. 1995. On-line construction of suffix trees. *Algorithmica* **14**:249–260.

Wu F, Zhao S, Yu B, Chen Y, Wang W, Song Z, Hu Y, Tao Z, Tian J, Pei Y, *et al.* 2020. A new coronavirus associated with human respiratory disease in China. *Nature* **579**: 265–269.

Zou Q, Hu Q, Guo M, Wang G. 2015. HAlign: fast multiple similar DNA/RNA sequence alignment based on the centre star strategy. *Bioinformatics* **31**:2475–2481.

Zou Q, Shan X, Jiang Y. 2012. A novel center star multiple sequence alignment algorithm based on affine gap penalty and K-band. *Phys Proced*. **33**:322–327.