Anchors for Homology-Based Scaffolding

Karl K. Käther[1,†,v], Steffen Lemke[2,vi], and Peter F. Stadler[1,3,vii]

[1] Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, Universität Leipzig, Härtelstrasse 16-18, D-04107 Leipzig, Germany, {karl,studla}@bioinf.uni-leipzig.de
[2] Institut für Biologie, Fachgebiet Zoologie, Universität Hohenheim, Germany
[3] Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany; Institute for Theoretical Chemistry, University of Vienna, Vienna, Austria; Facultad de Ciencias, Universidad Nacional de Colombia, Bogotá, Colombia; Santa Fe Institute, Santa Fe, NM

Abstract. Homology-based scaffolding is the task of ordering contigs based on their co-linearity with the DNA sequence of other, closely related reference genomes. We investigate here how a recent approach to identify synteny anchors can be adapted to this task. The advantage of using anchors is that "sufficiently unique" anchor candidates can be precomputed independently for each genome assembly, including contig-level drafts. This allows an efficient identification of co-linear runs of anchor matches between a contig and a reference genome and, consequently, also order their contigs in relation to the reference. Such data can be improved by aggregating scaffolding results for multiple reference genomes. We present a protoypical pipeline based on the anchor-based synteny detection tool `AncST` and show that it produces scaffolds with accuracies comparable to competing tools at acceptable computational costs. Assuming that anchor candidates have been precomputed for the genomes of interest, the remaining computational efforts are on par with the fastest methods.

Keywords: Genome Assembly · Synteny · Maximum Matching

1 Introduction

Genome assembly is one of the basic tasks in computational biology. While cost-effective high-throughput sequencing methods readily produce high-quality contigs, arranging them in the correct order is a much more difficult problem usually requiring significant coverage with more cost-intensive or less accurate long reads. However, high-quality reference genome assemblies are becoming available at an increasing pace, making homology-based scaffolding an attractive proposition. Several approaches for this task have been proposed [3,2,5].

[v] 0009-0003-3439-5813
[vi] 0000-0001-5807-2865
[vii] 0000-0002-5016-5191
[†] Corresponding author. karl@bioinf.uni-leipzig.de

All homology-based scaffolding starts from alignments of the reference and target genomes and then process the order and orientation information provided by the individual alignments to derive sequences of contiguous scaffolds. For instance, [2] use the `minimap2` sequence aligner [12] and leverage the longest alignments to order the contigs. In `Ragout2` [10] multiple sequence alignments and different levels of synteny blocks are used to create a breakpoint graph. Rearrangment operations are used in [3] to minimize the cost required to turn one genome into the other thereby defining a minimum cost contig order. Even though most of the available tools use sequence alignments explicitly, it has been shown that ordered minimizer sketches, i.e., selections of k-mers, can also be successfully applied for assembly scaffolding [5]. Several tools have been developed to use multiple reference genomes [1,4,10]. The scaffolding process may become more robust to species-specific rearrangements and thus reduces biases from using only one reference. Such approaches can make direct use of multiple sequence alignment blocks [10] or create a consensus from single-reference contig orders [1,4].

Here, we use our recently developed `AncST` approach [9] to produce high-confidence alignments from synteny anchors, which serve as reliable input for a contig scaffolding pipeline. Since `AncST` is developed to assert 1-1 orthology between respective sequence anchors with high confidence, corresponding alignments should serve as a suitable signal for robust homology-based contig scaffolding. Indeed, a simple greedy-like algorithm already recovers correct orders and orientations comparable and partially superior to existing pipelines while promising to be relatively scalable. The new homology-based scaffolding pipeline proposed here exhibits comparably high accuracy. It is a timely development given the massive efforts to close phylogenetic gaps in comprehensive genome sequencing projects, such as the Darwin Tree of Life (DToL),[4], the European Reference Genome Atlas (eERGA),[5] or the Earth Biogenome Project (EBP).[6]

2 Theory

Genome assembly from short-read sequencing data typically stops at the level of contigs, i.e., segments of genomic sequence that are covered by sequencing reads with sufficient overlaps to ensure that they represent a single genomic interval. Scaffolding most generally refers to the task of ordering the set of contigs in their correct chromosomal order. Additional data are required for this task. Examples include the sequencing of mate-pairs, i.e., DNA fragments of approximately known length for which only the sequence at their ends are known, optical mapping, or a closely related reference assembly [13].

Homology-based scaffolding is closely related to synteny analysis, i.e., the identification of co-linear regions between genomes [7]. The main difference is that the detection of syntenic regions concerns the identification of maximal

[4] https://www.darwintreeoflife.org
[5] https://www.erga-biodiversity.eu
[6] https://www.earthbiogenome.org

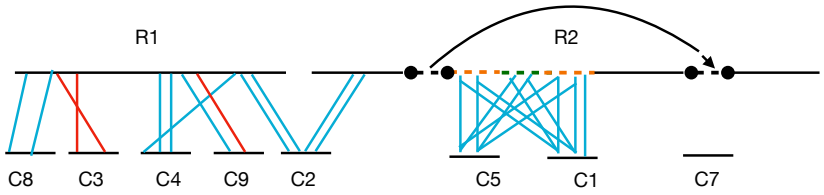

Fig. 1. Homology-based scaffolding. A list of contigs C1 to C8 is ordered along reference chromosomes R1 and R2 based on alignments indicated by colored lines between them. The contigs are initially unordered and placed into the order with respect to their alignments. Here, the `AncST`-based pipeline's strategy is depicted in which a contig gets placed into the order as soon as the majority of its alignments with its best-matching reference chromosome has been seen. Note that not all contigs have to be placed (C6 is missing). Different complications for homology-based scaffolding are shown: Blue lines indicate forward and red ones reverse matches; thus C3 is a contig which should either be turned or marked with one or more inversion events. C2 depicts the case when initial contigs get incorrectly fused in the assembly process which leads to one long scaffold consisting of all contigs aligning to R1 and R2. In R2 the arrow indicates a translocation event and the dashed colored lines denote local duplication events (colors do not matter). Hence, C5 and C1 are in a duplicated region which is difficult to reconcile and C7 is placed incorrectly.

runs of co-linear alignments, while homology-based scaffolding assumes that one or both genomes are given in the form of short intervals, the contigs, that are "almost" always internally co-linear with the other genome. Here we only consider the problem of homology-based scaffolding with a high-quality, chromosome-level reference assembly. Informally, the task is then to order the contigs of the draft assembly linearly along the chromosomes of the reference assembly, and to identify comparably rare rearrangement events between the two genomes, see Fig. 1. Such rearrangement events can be detected if two parts of a draft contig match non-contiguous locations on the same reference chromosome or distinct reference chromosomes. Homology-based scaffolding thus can be seen as an optimization problem that penalizes non-colinearity of the contiguous alignments of a contig with the reference chromosomes. In practice, the problem is complicated by different resolution levels of alignments from nucleotide to synteny block. A simplified version of the problem has been shown to be NP-hard [18]. Available tools thus utilize computationally efficient heuristics to assemble contigs into larger scaffolds successfully. Unambiguous 1-1 matches between a query contig and a reference chromosome thus form valuable information for this task.

Anchors. In recent work [8,9] we investigated the notion of annotion-free anchors as a means to identify syntenic regions efficiently. Two d_0-unique subsequences w and w' in different genomes in G and G' are guaranteed to form a unique match if their edit distance is less than $d_0/2$. In practice, it suffices to use a cutoff of d_0 minus some small tolerance. We call such d_0-unique sequences anchor candidates.

Anchor candidates that satisfy the matching criterion form anchors between the genomes G and G'. Pragmatically, it is preferable to use `blast` bit scores directly. In previous applications we found that considering a pair of anchor candidates x and x' from genomes G and G' as a match yield excellent performance if $s(x, x') > \max(s_0^G(x) + \sqrt{s_0^G(x)}, s_0^{G'}(x') + \sqrt{s_0^{G'}(x')})$, where $s_0^G(x)$ and $s_0^{G'}(x')$ are scores of x and x', resp., against the best non-overlapping matches in the genomes of origin.

Anchors were used in [9] to construct fine-grained synteny maps between not-too-distant genomes. In order to determine anchor candidates efficiently, we use infrequent k-mers with a length close to $\log_4 |G|$, where $|G|$ denotes the genome size, as a indicator for the uniqueness of w. Details of the statistics implemented in `AncST` are described in [9].

Contig sorting. Here we utilize anchors between the contigs of a draft assembly and the chromosomes of a finished assembly to construct scaffolds for the draft assembly. We employ a simple greedy approach. First we determine, for each contig, the reference chromosome with the largest number of matched anchors. In the next step, only anchors between the contig and its assigned reference chromosome are considered. In the second step, we consider each reference chromosome separately. Let $\mathcal{L}$ be an initially empty list. We traverse the anchors between the reference chromosome and its assigned contigs in the order of the reference. For each contig, we keep count of the number of its anchors that have been visited so far. A contig c is appended to $\mathcal{L}$ as soon at least half of its anchors have been visited. The order of their insertion into $\mathcal{L}$ serves as scaffold. For the results presented here, all contigs with alignments are included in a scaffold since this strategy is also employed in the tools we compared with.

Merging of alternative scaffolds can be performed using a simple graph-theoretic approach [4]. Denote by Γ a graph in which the end-points of the contigs serve as vertices. Given a scaffold, the adjacent end-points of consecutive edges are connected by a weighted edge. Multiple scaffolds can be superimposed in this graph with edge weights summed up over all scaffolds in which a given edge appears. In order to extract a consensus scaffold from Γ, we observe that in a scaffold each contig has at most one predecessor and one successor and thus each end-point of a scaffold can be incident to at most one edge. Hence scaffolds that only contain observed adjacencies correspond to matchings in Γ. A maximum weighted matching in Γ thus serves as a best estimate for a consensus scaffolding. We note that a matching will be perfect, i.e., includes all vertices, only if all contigs are placed into circular chromosomes. Otherwise, there will be contig ends that correspond to ends of a scaffold and thus are not incident to any edge. The maximum weighted matching problem can be solved in polynomial time using the *blossom* algorithm [6]. Adding missing edges to Γ with weight 0 transforms the problem into a maximum weight (or minimum cost) perfect matching problem, which can be solved by the highly optimized `Blossom V` implementation. [11]. In the resulting perfect matching, head and tail of the same contigs are connected, resulting in a regular graph Υ of vertex degree 2,

i.e., a vertex disjoint union of cycles. After breaking each cycle by removing its minimum weight edge and removing completely unsupported edges with weight 0, we are left with a disjoint union of paths, each of which specifies (up to the overall orientation) a linear order of contigs as well as the reading direction of each contig, and thus a scaffold.

3 Computational Methods

Synthetic test data. We produced the three simulated draft assemblies of the human genome as follows: For the rearranged version, we first applied rearrangement events (tandem duplications, translocations and inversions). Locations of rearrangements were restricted to respecting annotation items. We distinguish between short-range and long-range rearrangements. The former only affect a single annotation item, while the latter pertain to an interval containing two or more annotation items separated by not more than 1 million bases. In addition to a local inversion of the interval, rearrangements were defined as follows: For translocation we randomly selected a different chromosome, an insertion position, and the orientation of the inserted fragment. Up to four duplication events are allowed, either local or remote. For a local duplication the copy of a selected interval is inserted, possibly in inverted orientation, at a location between 1000 and 100000 bases downstream of the interval in question. Alternatively, duplicates were inserted in a randomly chosen chromosome. If a translocated sequence was chosen to only be translocated but not copied, then the original source interval was deleted in order to avoid bloating the genome size. We created one version in which at least 5% of the human genome are rearranged and another one in which at least 10% are rearranged. The rearranged genomes are documented in the supplementary material.

Contigs were extracted in the same manner from both the rearranged and conserved human genome. In order to mimic the fact that contig assembly usually stops at repetitive sequences, we defined contig ends at stretches of at least 10000 N in the hard-masked version of the human genome. In addition, we introduced random breakpoints within the initial fragments. At this stage we only accepted contigs consisting of at least 20000 bases and 30% unmasked sequence. Each initial fragment was subjected to 30 repetitions of the random fragmentation attempts. In order to simulate assembly errors, we fused 5% of the resulting fragments at random. Moreover, reading direction was determined at random.

Scaffolding. We compared our approach with two widely used homology-based scaffolding approaches:
`CSAR` and `Multi-CSAR` were installed from their sources[7]. Default parameters were used with the option `-nuc` to run on the nucleotide level on the two pairs of genomes as well as both reference genomes for the multi-reference version.

[7] https://github.com/ablab-nthu/CSAR, https://github.com/ablab-nthu/Multi-CSAR

`RagTag` was installed from its source[8] and run on the two pairs of genomes. To obtain a multi-reference version we used its `merge` command on the result of the pairwise results. Default parameters were used.
`AncST` was used to produce pairwise alignments between the respective genomes. The pipeline relies on `GenMap` to produce k-mer statistics. We used the parameters $k = 16$ and $e = 0$. The initial window size for anchor construction was set to 500 bp, with intervals overlapping at their midpoint. Windows with an aggregate k-mer count less than the 7th percentile were used as anchor candiates. The resulting pairwise scaffolds were reconciled using a reimplementation of the ideas behind the merging algorithm in `Multi-CSAR` [4] as described in the Methods section. For each appearance of a contig in a scaffold, the weight produced by `CSAR` is added. It measures the amount of DNA aligned by `MUMmer`. In real-life applications, `AncST` alignments produce a conceptually similar score that can be used instead of the `CSAR` scores.

Evaluation. In order to compare the different scaffolding results we employ a Needleman-Wunsch global alignment algorithm [15] to quantify the edit distance between reconstructed scaffolds and the correct order of the contigs in the reference. A score of 1 is used for matching contigs, while insertions, deletions, and mismatches are scored with 0. The total score of the optimal alignment therefore equals the number of matched contigs for an optimal solution and, after determining the correct orientations of the aligned contigs, can directly be used as a performance measure. More precisely, we compute the *precision* as the percentage of the Needleman-Wunsch alignment score divided the maximally achievable score, which is is given by the total number contigs minus 1/2 times the number of fused contigs. Alternatively, we may normalize the Needleman-Wunsch alignment score by the total number of contigs that are placed into any scaffold, resulting in a measure that is akin to *specificity*. Moreover, we also compute the total length of DNA aligned by the correctly aligned and oriented contigs.

In addition to evaluating the alignments, we count the number of correctly connected contigs among all possible correct connections (*precision*) and all made connections (*specificity*). Note that here the made connections can be higher than the originally correct ones. Finally, we record the number of breaks which are introduced to the original chromosomes when they are covered by more than one newly assembled scaffold.

The genomes assemblies of chimpanzee (Clint_PTRv2/panTro6) and bonobo (Mhudiblu_PPA_v0/panPan) were used as references, while the human genome (GRCh38/hg38) was used as a basis for the artificial test data sets. All sequence data were downloaded from the UCSC genome browser[9].

Availability. The relevant code and data are available in the supplementary material. A generic prototype is availabe at https://github.com/Norsbus/AncST.

[8] https://github.com/malonge/RagTag
[9] https://hgdownload.soe.ucsc.edu/goldenPath/

Results

We employed three different scaffolding pipelines for comparison: `CSAR` [3] and their respective multi-reference version `Multi-CSAR` [4], `RagTag` [1] with their multi-reference `merge` command, and our pipeline based on `AncST` anchors [9], see Methods. Table 1 summarizes the performance results for scaffolding the contigs of the human genome produced by breaking it up at regions of considerable repeat content and length as well as reversing and fusing some of the resulting contigs.

Tables 2 and 3 show the results for the rearranged assemblies. The details of the construction of the artificial target assemblies can be found in the respective Methods section. We decided to note the time consumption as wall-clock time for the different runs we present here for simplicity. We note that `CSAR` and `RagTag` were run as single-core applications while we used 58 cores for `AncST` anchor computation. `CSAR` cannot be run multi-threaded although the consumed `user` time indicates use of multiple cores. We decided not to benchmark `RagTag` with multiple threads as it is so fast that time consumption is negligible already without multithreading.

Table 1. Comparison of scaffolding performance. Data refer to the artificial contig sets obtained by fragmenting the human genome. The genomes of chimpanzee (tro) and bonobo (pan) are used as references. Performance is measured by different indicators: `al. prec.` and `al. spec.` refer to *alignment precision/specificity*, i.e. the fraction of correctly aligned contigs among all contigs and aligned contigs, respectively. `adj`acency `prec`ision and `spec`ificity count the fraction of correctly connected contigs among all and all connected contigs, respectively. `dna cov`erage `al.` is the amount of DNA covered by all correctly aligned contigs and `dna cov`erage `adj.` is the amount of DNA covered by correctly connected contigs. `chr`omosome `breaks` count the number of breaks in the original human chromosomes which emerge when several newly assembled scaffolds cover one chromosome. Numbers are percentages. Execution times are approximate wallclock times on a Linux server with 256 GB RAM and 64 available cores on two Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz processors. The second time column reports the running time of `AncST` without the computation of anchors for each genome, since the latter can be performed as a one-time preprocessing.

Pipeline	Time[min]	al. prec.	al. spec.	adj. prec.	adj. spec.	dna cov. al.	dna cov. adj.	chr. breaks
CSAR pan	3200	62	62	94	90	63	91	69
CSAR tro	3200	61	61	95	91	59	91	66
Multi-CSAR	6500	79	76	43	68	78	70	953
RagTag pan	25	77	82	93	90	80	87	9
RagTag tro	25	80	84	92	89	83	86	3
Multi-RagTag	52	59	91	48	86	78	75	319
AncST pan	750 (14)	76	85	89	92	80	87	6
AncST tro	750 (14)	79	88	90	92	82	88	4
Multi-AncST	1500 (30)	82	95	86	94	82	84	68

Table 2. Scaffolding results for the chopped human genome with at least 5% rearranged sequence. For details we refer to the caption of Table 1

Pipeline	Time[min]	al. prec.	al. spec.	adj. prec.	adj. spec.	dna cov. al.	dna cov. adj.	chr. breaks
CSAR pan	3200	48	52	82	85	48	83	124
CSAR tro	3200	51	55	83	86	54	85	131
Multi-CSAR	6500	70	72	37	63	64	60	1413
RagTag pan	28	66	79	78	86	72	79	18
RagTag tro	26	69	83	78	86	76	79	17
Multi-RagTag	56	42	88	31	71	68	65	445
AncST pan	800 (14)	60	83	73	88	73	78	6
AncST tro	800 (14)	64	88	70	88	75	80	7
Multi-AncST	1600 (30)	67	96	66	90	77	76	129

Table 3. Scaffolding results for the chopped human genome with at least 10% rearranged sequence. For details we refer to the caption of Table 1

Pipeline	Time[min]	al. prec.	al. spec.	adj. prec.	adj. spec.	dna cov. al.	dna cov. adj.	chr. breaks
CSAR pan	3300	45	49	72	75	38	68	121
CSAR tro	3400	49	53	73	76	42	69	130
Multi-CSAR	6800	66	69	33	56	56	49	1335
RagTag pan	29	65	77	76	83	67	71	21
RagTag tro	27	68	80	77	83	69	72	12
Multi-RagTag	58	40	84	29	73	62	57	416
AncST pan	850 (15)	61	83	70	88	69	74	5
AncST tro	850 (15)	64	87	71	88	72	74	4
Multi-AncST	1700 (32)	68	96	67	90	73	70	130

`CSAR` suffers from incorrectly fusing scaffolds that are mostly correct internally into large pseudo-molecules. Thus, it performs the worst while also consuming the most resources. The `AncST` anchors-based pipeline and `RagTag` yield comparable results. However, `RagTag` is orders of magnitude faster if `AncST` anchors are not precomputed. On the other hand, our pipeline has consistently higher specificity and fewer chromosome breaks.

For the multi-reference experiments, the `AncST`-based pipeline clearly outperforms the two other tools. Leveraging of the pairwise scaffolding results into a multi-reference version is only a matter of seconds to few minutes.

A more detailed breakdown of the resource consumption of the `AncST` pipeline shows that about 1/4 of the computer time is used for the collection of the k-mer statistics with `GenMap` as a basis for determining the anchor candidates, and 2/3 of the effort are expended for the computation of d_0-uniqueness of these candidates through a `blast` comparisons of the anchor candidates against their own genomes. All other tasks in the pipeline account for less than 10% of the wallclock time. Hence, as can be seen, the computation time for pairwise anchor candidate comparison and scaffolding alone are lower than those for `RagTag`.

4 Discussion and Conclusions

Homology-based scaffolding is likely to gain increasing importance in non-model-organism genomics in response to the rapidly growing number of high quality genomes that are suitable as references. The phylogenetic distance of available reference genomes to a newly sequenced genome thus shrinks, increasing the accuracy of the reference-based approach and making multi-reference scaffolding more attractive. Hence, the methodology has the potential to provide biologists with large numbers of accessible chromosome-level genome assemblies for functional analyses, comparative genomics, and evolutionary studies at moderate experimental and computational costs. This is of particular relevance for large-scale efforts as mentioned in the introduction.

We have shown here that pre-computed anchor candidates, i.e., sufficiently unique DNA in a genome, can be used as a basis for efficient homology-based contig scaffolding. A pipeline based on our recently released anchor-based synteny detector `AncST` performs acceptably well in comparison with two established tools. `Multi-CSAR` suffers from joining scaffolds covering more than one original chromosome and therefore generally performs the worst while otherwise connecting contigs mostly correctly. `RagTag` is extremely fast and yields comparable results to our pipeline for single-reference runs while suffering from a consistently lower specificity. For the multi-reference runs the `AncST`-based version outperforms the other pipelines significantly.

Since the performance bottlenecks are external programs, in particular `MUMmer` [14], `GenMap` [17] and `minimap2` [12] and `blast`, we did not benchmark memory consumption in detail. The choice of these tools is reflected in the performance statistics: `CSAR` uses `MUMmer` to create a complete genome alignment which theoretically has the best resolution for contig scaffolding identified by the number of correctly connected contigs, particularly for the non-rearranged version. `AncST` uses a k-mer heuristic based on `GenMap` k-mer counts and then uses `blast` as an index-based sequence alignment tool both to infer potentially good anchor candidates per genome and to find hits between the sets of candidates. By largely excluding duplicated regions of the genome, the order of these high-confidence alignments can then effectively be used for contig scaffolding. Our strategy hence consumes less resources than a full genome alignment. Nevertheless, it reproduces most of the useful signal. For `RagTag`, `minimap2` creates local pairwise alignments much faster which still reproduce most of the useful signal for contig scaffolding despite performing worse in terms of the specificity and much worse for the multi-reference version.

The restriction to uniquely matching DNA inherent in `AncST` results in an increased robustness against rearrangements relative to the reference genome(s). This robustness is shown by the specificity of contig assignments in the rearranged and multi-reference runs which stay high compared to the non-rearranged and single-reference ones in our approach while dropping considerably for the other tools.

The current version of the `AncST`-based pipeline is not intended for heavy-duty production but rather serves as a proof-of-concept for the practical utility

of the anchor-based approach. As discussed in [9], it may be possible to replace `GenMap` by alternative statistics of sequence-uniqueness, in particular variants of the Lempel-Ziv complexity, see e.g. [16]. Moreover, it is likely possible to adapt `RagTag`'s minimizer sketches as filters in the process of comparing anchor candidates between genomes. In this manner, both of the present performance bottlenecks in the `AncST` software could be addressed.

A distinctive advantage of the anchor-based approach is that anchor candidates can be precomputed and stored as simple annotation files that represent anchors as coordinate pairs associated with their level of the interval's d_0-uniqueness. Pairwise comparisons are then restricted to these candidates, and by construction, result in unambiguous alignments. In fact, disregarding this preprocessing of the genomes reduces the computation times to figures similar to `RagTag`, while asserting a more robust contig ordering.

Non-colinear anchor alignments therefore yield strong evidence for breakpoints arising from genome rearrangements. The anchor matches themselves, moreover, can be fed directly into the analysis of macrosynteny, which is usually performed as part of genome assembly projects. For instance, we can readily track possible inversion scenarios by enumerating regions for which `AncST` anchors align on the opposing strand in considerable numbers. Translocations are indicated by significant colinear stretches of alignments with a breakpoint in one of the species. The current implementation of the `AncST`-based pipeline identifies contigs with alignments to more than one reference chromosome and those with alignments in opposite reading directions on the same reference chromosome. These are strong indicators for rearrangements relative to the reference genome. In principle, these data could be used to extend scaffolds beyond such breakpoints. Whether this can be done with acceptable reliability, however, is currently subject to investigation.

Moreover, given a case where it is advisable to use multiple references for contig scaffolding, our approach seems to perform well and is scalable since `AncST` itself is scalable due the fact that its resource consumption is dominated by the preprocessing that is performed independently for each genome. In contrast, `MUMmer` and `minimap` are genuine pairwise aligners and thus require full computational efforts for all pairs of genomes under consideration. As we have seen, the near-uniqueness of anchor-based alignments adds to the performance particularly in a multiple reference scenario, without incurring significant extra computational costs.

Acknowledgments. This work was funded by the German Research Foundation as part of SPP 2349 *"Genomic Basis of Evolutionary Innovations (GEvol)"* (grant no. 502862570). PFS acknowledges the financial support by the Federal Ministry of Education and Research of Germany (BMBF) through DAAD project 57616814 (SECAI, School of Embedded Composite AI), and jointly with the Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research *Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig*, project identification number: SCADS24B.

Disclosure of Interests. The authors have no competing interests.

Supplemental Material The supplementary material is available at Zenodo (https://zenodo.org/records/15061775). The archive includes all used and created genomes as well as the scripts used to create the latter. Furthermore, all rearrangements are documented and precomputed `AncST` anchors and performance evaluation scripts are provided. The prototypical pipeline is included in the `AncST` Github repository (https://github.com/Norsbus/AncST).

References

1. Alonge, M., Lebeigle, L., Kirsche, M., Jenike, K., Ou, S., Aganezov, S., Wang, X., Lippman, Z.B., Schatz, M.C., Soyk, S.: Automated assembly scaffolding using ragtag elevates a new tomato system for high-throughput genome editing. Genome biology **23**, 258 (2022). https://doi.org/10.1186/s13059-022-02823-7
2. Alonge, M., Soyk, S., Ramakrishnan, S., Wang, X., Goodwin, S., Sedlazeck, F.J., Lippman, Z.B., Schatz, M.C.: RaGOO: fast and accurate reference-guided scaffolding of draft genomes. Genome Biology **20**, 224 (2019). https://doi.org/10.1186/s13059-019-1829-6
3. Chen, K.T., Liu, C.L., Huang, S.H., Shen, H.T., Shieh, Y.K., Chiu, H.T., Lu, C.L.: CSAR: a contig scaffolding tool using algebraic rearrangements. Bioinformatics **34**, 109–111 (2018). https://doi.org/10.1093/bioinformatics/btx543
4. Chen, K.T., Shen, H.T., Lu, C.L.: Multi-CSAR: a multiple reference-based contig scaffolder using algebraic rearrangements. BMC Systems Biology **12**, 69–75 (2018). https://doi.org/10.1186/s12918-018-0654-y
5. Coombe, L., Nikolić, V., Chu, J., Birol, I., Warren, R.L.: ntJoin: Fast and lightweight assembly-guided scaffolding using minimizer graphs. Bioinformatics **36**(12), 3885–3887 (2020). https://doi.org/10.1093/bioinformatics/btaa253
6. Edmonds, J.: Path, trees, and flowers. Canadian J. Math. **17**, 449–467 (1965). https://doi.org/10.4153/CJM-1965-045-4
7. Ghiurcuta, C.G., Moret, B.M.E.: Evaluating synteny for improved comparative studies. Bioinformatics **30**(12), i9–i18 (2014). https://doi.org/10.1093/bioinformatics/btu259
8. Käther, K., Lemke, S., Stadler, P.F.: Annotation-free identification of potential synteny anchors. In: Rojas, I., Valenzuela, O., Rojas Ruiz, F., Herrera, L.J., Ortuño, F. (eds.) International Work-Conference on Bioinformatics and Biomedical Engineering (IWWBIO'23). Lect. Notes Comp. Sci., vol. 13919, pp. 217–230. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-34953-9_17
9. Käther, K., Remmel, A., Lemke, S., Stadler, P.F.: Unbiased anchors for reliable genome-wide synteny detection. Alg. Mol. Biol. (2025), accepted
10. Kolmogorov, M., Armstrong, J., Raney, B.J., Streeter, I., Dunn, M., Yang, F., Odom, D., Flicek, P., Keane, T.M., Thybert, D., Paten, B., Pham, S.: Chromosome assembly of large and complex genomes using multiple references. Genome Res. **28**(11), 1720–1732 (2018). https://doi.org/10.1101/gr.236273.118
11. Kolmogorov, V.: Blossom V: a new implementation of a minimum cost perfect matching algorithm. Mathematical Programming Computation **1**, 43–67 (2009). https://doi.org/10.1007/s12532-009-0002-8
12. Li, H.: Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics **34**(18), 3094–3100 (2018). https://doi.org/10.1093/bioinformatics/bty191
13. Luo, J., Wei, Y., Lyu, M., Wu, Z., Liu, X., Luo, H., Yan, C.: A comprehensive review of scaffolding methods in genome assembly. Briefings in Bioinformatics **22**, bbab033 (2021). https://doi.org/10.1093/bib/bbab033

14. Marçais, G., Delcher, A.L., Phillippy, A.M., Coston, R., Salzberg, S.L., Zimin, A.: MUMmer4: A fast and versatile genome alignment system. PLoS Comp. Biol. **14**(1), e1005944 (2018). https://doi.org/10.1371/journal.pcbi.1005944
15. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. **48**(3), 443–453 (1970). https://doi.org/10.1016/0022-2836(70)90057-4
16. Pirogov, A., Pfaffelhuber, P., Börsch-Haubold, A., Haubold, B.: High-complexity regions in mammalian genomes are enriched for developmental genes. Bioinformatics **35**(11), 1813–1819 (2019). https://doi.org/10.1093/bioinformatics/bty922
17. Pockrandt, C., Alzamel, M., Iliopoulos, C.S., Reinert, K.: GenMap: ultra-fast computation of genome mappability. Bioinformatics **36**(12), 3687–3692 (2020). https://doi.org/10.1093/bioinformatics/btaa222
18. Schrinner, S., Goel, M., Wulfert, M., Spohr, P., Schneeberger, K., Klau, G.W.: Using the longest run subsequence problem within homology-based scaffolding. Algorithms for Molecular Biology **16**, 11 (2021). https://doi.org/10.1186/s13015-021-00191-8