Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

PREPRINT

Efficient Merging of Genome Profile Alignments

André Hennig [1,*] and Kay Nieselt [1,*]

[1] Center for Bioinformatics (ZBIT), Integrative Transcriptomics, Eberhard Karls University Tübingen, 72076, Germany.

*To whom correspondence should be addressed.

Abstract

Motivation: Whole-genome alignment methods show insufficient scalability towards the generation of large-scale whole-genome alignments (WGAs). Profile alignment-based approaches revolutionized the fields of multiple sequence alignment construction methods by significantly reducing computational complexity and runtime. However, WGAs need to consider genomic rearrangements between genomes, which makes the profile-based extension of several whole-genomes challenging. Currently, none of the available methods offer the possibility to align or extend WGA profiles.
Results: Here, we present `GPA`, an approach that aligns the profiles of WGAs and is capable of producing large-scale WGAs many folds faster than conventional methods. Our concept relies on already available whole-genome aligners, which are used to compute several smaller sets of aligned genomes that are combined to a full WGA with a divide and conquer approach. We make use of the SuperGenome data structure, which features a bidirectional mapping between individual sequence and alignment coordinates. This data structure is used to efficiently to transfer different coordinate system into a common one based on the principles of profiles alignments. The approach allows the computation of a WGA where alignments are subsequently merged along a guide tree. The current implementation uses `progressiveMauve` (Darling *et al.*, 2010) and offers the possibility for parallel computation of independent genome alignments. Our results based on data sets up to 326 genomes show that we can reduce the runtime from months to hours with a quality that is negligibly worse than the WGA computed with the conventional `progressiveMauve` tool.
Availability: `GPA` is freely available at `https://lambda.informatik.uni-tuebingen.de/gitlab/ahennig/SuperGenome`. `GPA` is implemented in Java 8, uses `progressiveMauve` and offers a parallel computation of WGAs.

Contact: andre.hennig@uni-tuebingen.de
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Whole-genome sequencing (WGS) has become increasingly affordable through the continuous developments of next-generation sequencing (NGS) technologies. WGS for example is now routinely conducted in a clinical context to monitor pandemic bacterial outbreaks based on the sequencing of different isolates. Single nucleotide variations (SNV) between isolates and a reference genome can help to understand and reconstruct transmission chains (Bryant *et al.*, 2013; Sabat *et al.*, 2013). The disadvantage using a single reference to compare different individuals is that features missing from the reference cannot be detected. Especially in the absence of a closely related reference genome, such an approach is not appropriate (Abdelbary *et al.*, 2018). To overcome this problem, an increasing number of studies incorporate the pan-genome of the species into the analysis of different isolates. Here, gene content and genomic rearrangements such as insertions, deletions, translocations, and inversions are used to explain the manifestation of phenotypic traits like antibiotic resistance (Medini *et al.*, 2005). One approach to compute a pan-genome is based on whole-genome alignments (WGAs). In comparison to a BLAST-based approach or variants of it which are employed to compute orthologous gene groups, the WGA-based approach has the advantage that for the identification of orthologous genes gene neighbourhood is taken into account. In addition, the pan-genome based on a WGA can be generalized to take also non-genic features into account. Further applications of WGAs encompass the identification of pathogenic genomic islands or reconstruction of phylogenomic trees (Chan and Ragan, 2013).

Runtimes of current state-of-the-art aligners that are capable of modeling genomic rearrangements, such as progressiveMauve (Darling *et al.*, 2010), are at least quadratic in the number of genomes. Thus, computation of WGAs of hundreds or thousands of even closely related bacterial genomes with state-of-the-art tools are prohibitive. Currently only Parsnp (Treangen *et al.*, 2014) is able to compute large-scale alignments with hundreds or even thousands of genomes within hours. However, it computes only a core-genome alignment. While Parsnp is a highly valuable tool for the identification of SNPs from the core pan-genome, it does not allow for the detection of genomic rearrangements and pathogenic islands for example. Recent advances in the field of whole-genome alignment methods were made with the introduction of seq-seq-pan (Jandrasits *et al.*, 2018), which offers the efficient extension of existing WGA by new genomes. Through a pairwise iterative alignment process, new genomic sequences are aligned against the consensus sequence of the alignment, which shows a substantial runtime decrease while achieving comparably high-quality alignments.

seq-seq-pan makes use of the profile of a pairwise alignment (and deduces a consensus sequence) for extension by new sequences. Using a profile as a representation for an alignment is a popular concept used to compute multiple sequence alignments (MSA) and was first introduced by Hogeweg and Hesper (1984) in their progressive alignment heuristic. The idea to progressively compute MSAs along a guide tree, where at each node either a pairwise alignment between sequences, sequence-profile or profiles are conducted, reduces the runtime significantly in contrast to the construction of an optimal alignment that maximizes the sum of pairs score. Especially the profile-profile alignment offers a fast and efficient way to combine two separate alignments. Recent advances in this field have been made by Liu and Warnow (2014) with the idea of a divide and conquer approach to compute smaller subsets of alignments which are merged through a profile-profile alignment. The parallel computation of the subset alignments further reduces the computational runtime, while still achieving highly accurate results.

However, currently such a profile-alignment based approach is missing in state-of-the-art alignment tools such as progressiveMauve. Since WGAs need to consider genomic rearrangements, such as translocations and inversions, between the individual genomes, a profile alignment of two or more WGAs is more difficult than a profile alignment of MSAs. Here, we introduce our concept for the first profile-profile alignment of WGAs. The profile-based merging of WGAs is conducted with the help of our SuperGenome data structure (Herbig *et al.*, 2012), which can be used to transfer different WGA-coordinate system into a common one.

Based on this profile-based alignment approach, we implemented GPA (Genome Profile Alignment), a software that can align hundreds to thousands of genomes in a fast and efficient manner. The goals of our whole-genome alignment construction strategy were to adopt the advances from the field of profile-based MSA tools, and therefore significantly reduce computational time and still achieving highly accurate WGAs. Our intention was not to develop a new genome alignment algorithm. Therefore, GPA relies on other whole-genome aligners but using a divide and conquer strategy to merge subsets of alignments along a guide tree. In our current release, we have combined progressiveMauve with our profile-based approach.

The article is organized as follows: In the method section, we first present the SuperGenome data structure and the algorithmic principles that allows an efficient merging of several alignments. The critical aspect is the transfer of different coordinate systems into a common one, which is supported by the SuperGenome. We explain in detail the extension of a given WGA by other genomes or other WGAs. Based on this we then explain how to compute large-scale WGAs from scratch in a fast and parallelized manner. The section concludes with a description of statistics that we used to compare and evaluate our approach with the original progressiveMauve. The results section presents the evaluation of the WGAs computed for the different data set by progressiveMauve and GPA. This evaluation is focused on the runtime needed and the achieved quality of the WGA by both approaches. Finally, we conclude this article with a critical discussion of our results and propose future improvements of our approach.

2 Methods

SuperGenome data structure and construction

The SuperGenome is a data structure, which makes use of a whole-genome alignment (WGA) and features a bidirectional mapping between the alignment coordinates and the original coordinates of each individual genome in the WGA (Herbig *et al.*, 2012). In comparison to multiple sequence alignment, where the order of nucleotides within each sequence is assumed to be preserved, aligning whole-genomes has to consider the occurrence of rearrangements, such as translocations as well as inversions. Regions shared by two or more genomes that do not contain any rearrangements of homologous sequences are called locally collinear blocks (LCBs), and a WGA is typically then represented by a set of such LCBs. Programs computing WGAs of this form are for example Mauve (Darling *et al.*, 2004), progressiveMauve (Darling *et al.*, 2010), Mugsy (Angiuoli and Salzberg, 2010) and TBA (Blanchette *et al.*, 2004). The main advantage of the SuperGenome data structure is that it provides an unambiguous coordinate system and is independent of any pre-chosen reference genome.

We will first introduce some formal terminology before we then describe the algorithm to compute large-scale WGAs to decrease computational runtime using a profile-based approach together with the SuperGenome data structures derived from the profiles.

Given are n genomes $g_i, i = 1, \dots, n$ and a WGA $\mathbf{A}$ on these n genomes. We define a pair of integer arrays G_i and SG_i, which provides the bidirectional mapping between the positions of g_i and $\mathbf{A}$. Both arrays cover either all genomic or alignment positions and are zero-based numbered. In addition, the arrays have a leading entry, that is used to represent the absence of the sequence and are due to that one position longer than either the length of the genome or the alignment. This simplifies the mapping of the coordinate systems since the j-th index represents the j-th position in the genome or the alignment.

Let us assume that the j-th base of g_i is aligned at the k-th position in $\mathbf{A}$. Thus, G_i, representing the mapping of g_i to $\mathbf{A}$, contains the value k at entry j. In the case that the base was aligned as its reverse complement (i.e., representing an inversion), the value of k is negative:

$$G_i[j] = \begin{cases} k, \text{ if base } j \text{ is aligned in forward orientation} \\ -k, \text{ if base } j \text{ is aligned as reverse complement} \end{cases} \quad (1)$$

The respective SuperGenome array SG_i represents the mapping of the aligned sequence of genome g_i in $\mathbf{A}$ and is analogous to G_i, in addition, it also accounts for gaps. In case at position k in $\mathbf{A}$ no base of g_i was aligned the entry is set to zero.

$$SG_i[k] = \begin{cases} j, \text{ if base } j \text{ is aligned in forward orientation} \\ -j, \text{ if base } j \text{ is aligned as reverse complement} \\ 0, \text{ if no base of } g_i \text{ aligns at that position} \end{cases} \quad (2)$$

In addition to the coordinate mapping, the SuperGenome takes the local collinear block (LCB) structure into account, as for example, computed by progressiveMauve and stored in XMFA-format, by also tracking the start and stop positions of all LCBs.

For a given WGA of n genomes, the generation of the data structure is straightforward. For each genome, the pair of arrays G_i and SG_i is first initialized with the corresponding lengths of g_i and $\mathbf{A}$, respectively, and filled with zeros. Through an iteration over every alignment position, the arrays are filled, resulting in a total of $2 \times n$ arrays with a space requirement of $n\times$ length of the alignment and the sum of all genome lengths. From the SuperGenome data structure, the WGA can be derived from the arrays SG_i, where each aligned sequence of g_i can be reconstructed iteratively back from the genomic positions of the entries. Here, inversions (negative values) and gaps (zeroes) have to be taken into account.

Extending an existing WGA

We now first describe how a new genome is added to an existing WGA on n genomes using the SuperGenome approach. Then we show how to extend this principle to merging two WGAs on n and m genomes, respectively, into a WGA on $n + m$ genomes. This step can then be generalized to several alignments and genomes that are combined into a WGA of all involved genomes. The general idea is that at each extension step only a pairwise alignment is computed.

A common approach to make use of an existing alignment $\mathbf{A}$ is a profile alignment, which preserves all prior aligned positions. The profile of $\mathbf{A}$ is represented by a consensus sequence, which is derived from the SuperGenome data structure through a majority call on all alignment positions of $\mathbf{A}$. For the integration of a new genome g_{n+1} into $\mathbf{A}$, unique regions of g_{n+1} and homologous regions of g_{n+1} and the profile have to be computed in order to extend $\mathbf{A}$ by the new genome. This is achieved by computing a pairwise alignment of the profile consensus sequence and the genomic sequence g_{n+1}. This pairwise alignment serves as a guiding alignment to extend the given alignment $\mathbf{A}$ by the new genome. Again we use the SuperGenome data structure for this step (see Figure 1B-D for an illustration of this procedure). For this, the SuperGenome data structure of the pairwise alignment is computed, which includes the array SG_{cons} with all positions of the profile consensus sequence and SG_{n+1}, the SG array of genome g_{n+1}. The extension of $\mathbf{A}$ comprises the transfer of the coordinates of SG_i and G_i for the n genomes into the common coordinate system of SG_{cons} and SG_{n+1}. The new bidirectional coordinate mappings SG'_i and G'_i for genome g_i are derived by

$$SG'_i[j] = SG_i[SG_{cons}[j]] \quad \text{and} \quad G'_i[SG'_i[j]] = j \qquad (3)$$

where $SG_{cons}[j]$ is the index of the consensus sequence, which is aligned at position j in the pairwise alignment, and $SG_i[SG_{cons}[j]]$ the index of the base of g_i that was aligned in $\mathbf{A}$.

This coordinate transfer restores all columns of alignment $\mathbf{A}$. The arrays SG_{n+1} and G_{n+1} do not have to be updated since they are already consistent with the coordinate system of the guiding alignment. Based on the updated SuperGenome data structure on $n + 1$ genomes, the new alignment can now be easily derived and written into the alignment format as described above.

The procedure for adding one genome to a WGA is easily extended to merge the profiles of two WGAs on $n > 1$ and $m > 1$ genomes, respectively. Again, we compute a pairwise alignment, now on the two respective consensus sequences, derive the respective SuperGenome, which together with the SuperGenome of the input WGAs to update the bidirectional mappings (see equation (1)) the two WGAs. A consequence of this merging procedure of two WGAs is that prior aligned bases appear consistently in the merged WGA.

Finally, it is straightforward to generalize the pairwise approach to combining more than 2 WGAs or WGAs with individual genome sequences. Rather than aligning only two sequences, we compute the multiple genome alignment of all consensus sequences derived from the $k > 2$ individual WGAs as well as possible single genome sequences. The generalized workflow of merging several WGAs into a common WGA can be summarized in the following 6 steps (see also Figure 2):

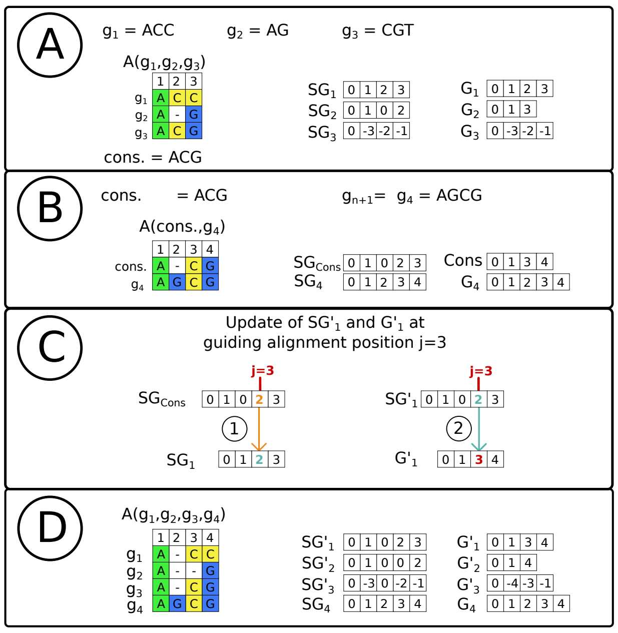

Fig. 1. (A) Based on the three genomic sequence g_1, g_2 and g_3 and their alignment $\mathbf{A}(g_1, g_2, g_3)$, a SuperGenome data structure (union of the SG and G arrays) is computed. In addition, a consensus sequence from the alignment is deducted. (B) A new genomic sequence g_4 is combined with $\mathbf{A}(g_1, g_2, g_3)$. For this a pairwise alignment $\mathbf{A}(cons, g_4)$ is computed and again a SuperGenome data structure is deduced. (C) Update of $\mathbf{A}$ by g_4: for every position j in $\mathbf{A}(cons, g_4)$, (1) array $SG_{cons}[j]$ (orange) contains the index of the aligned consensus sequence positions, which is used to determine the original genomic positions $SG_1[SG_{cons}[j]]$ (example shown in blue). (2) This allows a coordinate transfer $SG'_1[j] = SG_1[SG_{cons}[j]]$ and $G'_1[SG'_1[j]] = j$ (red) into a common coordinate system of $\mathbf{A}(cons, g_4)$. (D) From the updated SuperGenome data structure the new alignment $\mathbf{A}(g_1, g_2, g_3, g_4)$ is easily deduced.

1. Construct SuperGenome data structure for every input alignment.
2. For every SuperGenome compute a consensus sequence (output in FASTA format).
3. Align consensus sequences as well as possible individual genome sequences with whole-genome aligner (e.g., using `progressiveMauve`).
4. Construct SuperGenome data structure for new genome alignment.
5. For every genome i, update SG_i and G_i according to equation (3).
6. Output new alignment derived from updated SG' and G' (output in `XMFA`-format).

s If the merged WGA should account for LCB structures, start and stop positions of all LCBs have to be transferred and added to the once introduced by the guiding alignment. Note that the new LCBs of the merged WGA are defined between every consecutive pair of LCB positions in the SuperGenome.

Genome Profile Alignment - GPA

We have implemented the described approach how to efficiently merge several whole-genome alignments or extend a given WGA by new genomes using our SuperGenome data structure in the tool which we call `GPA`. Our tool is written in Java 8 and can be run on any machine with a Java VM installed. For the computation of the WGAs, we currently make use of `progressiveMauve`, which needs to be installed independently.

`GPA` can be applied in two ways: it can align genome sequences from scratch, or it can be used to extend an existing WGA by new genomes. In the first case, the input data are the genome sequences that need to be provided in FASTA-format. Since the general idea of our approach is to

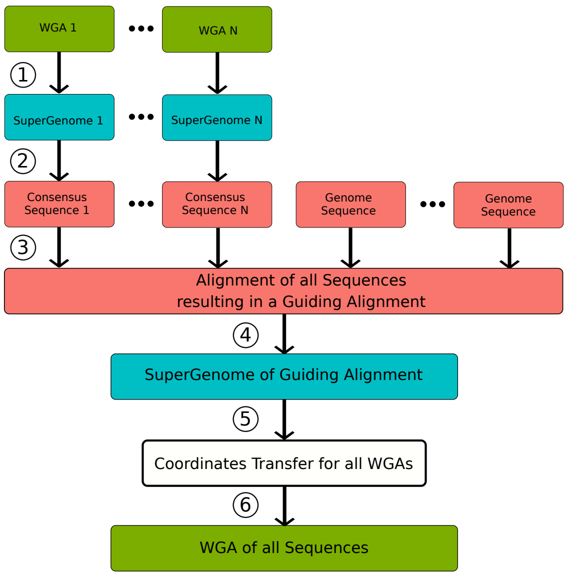

Fig. 2. Workflow of computing a large WGA from several subsets and genomes using the SuperGenome data structure. The workflow consists of 5 steps: 1. Build SuperGenome for every alignment, 2. Compute SuperGenome consensus sequence, 3. Align all consensus and genome sequences, 4. Build SuperGenome of guiding alignment, 5. Merge all alignments and genomes via Coordinates Transfer, 6. Output alignment of all sequences in XMFA-format derived from updated SuperGenome data structure of SG'

combine smaller sets of aligned genomes to a full WGA, these smaller sets first have to be defined. For this, we either make use of a guide tree, which determines the individual merging steps, or all genome sequences are randomly distributed into subsets, where the size of the subsets has to be predefined by the user. The guide tree needs to be provided by the user in Newick tree format and can, for example, be the one as computed by progressiveMauve. As most guide trees are binary and only two sequences are aligned at each node, the provided input tree is further modified, to control the number of sequences/WGAs that is aligned in each step. With a user-defined maximum number of sequences aligned in each step, the nodes of the guide tree (representing the set of sequences which will be aligned) are propagated from the leaves (representing the genomes) towards the root (representing the final WGA) until another propagation to the next node of the tree would exceed the maximum. This modified guide tree is used to compute an internal guide tree structure. In the next step, after the internal guide tree structure has been built, GPA automatically creates a folder structure, which serves to save the WGAs from the intermediate steps in XMFA-format. The computation of the WGAs follows the typical process of progressive alignments, starting at the leaves of the guide tree and the root represents the full WGA. If no guide tree has been provided, GPA merges all WGAs of the individual subsets into a common WGA in one step. To decrease the runtime, GPA provides the possibility to compute the independent subalignments in parallel.

The second case to extend an existing WGA, GPA can be provided with an arbitrary number of WGAs and single genomes. Here, the respective profiles or genomic sequences are aligned in one step. The final WGA then contains all new genomes as well as those which where contained in the input WGAs. Note that in fact the second case is used throughout the computation of a WGA from scratch when using our approach in GPA.

Table 1. The data sets, which were used for the WGA computations as obtained from the NCBI FTP server. All statistics including the median genome length and median GC content have been derived from NCBI.

Organism	# Strains	Median Genome	GC content
		Length (Mb)	
Bacillus cereus	13	5.760	35.1%
Listeria monocytogenes	30	2.975	37.9%
Chlamydia trachomatis	72	1.046	41.3%
Mycobacterium tuberculosis	128	4.385	65.6%
Klebsiella pneumoniae	166	5.590	57.2%
Staphyloccus aureus	176	2.847	32.8%
Bordetella pertussis	326	4.100	67.7%

Experimental setup

Datasets

To explore the performance of GPA, we applied our approach to a large number of complete single chromosome bacterial genomes from the same species which we derived from NCBI (ftp://ftp.ncbi.nlm.nih.gov/genomes). The various data sets reflect different genome lengths and sizes as well as diversities of genomes within a species to explore the performance of GPA. Currently, GPA can only handle single sequences per species. We therefore had to remove all possible plasmids prior to the WGA computation. All data sets used in this work are listed in Table 1 together with the total number of genomes as well as the average genome length and average GC content (Source https://www.ncbi.nlm.nih.gov/).

Evaluation Criteria

Besides runtime assessment, we used three different statistics to compare the whole-genome alignment computed using our guide-tree based SuperGenome approach with the whole-genome alignment computed by applying progressiveMauve to all genomes at once: *pairwise consistency score*, *total column score*, and *F-score*.

The pairwise consistency score reflects how much the WGA agrees with all possible pairwise alignments, which concept was first described by Gotoh (1990) and adapted in T-COFFEE (Notredame *et al.*, 2000). For this, we compute for each pair of genomes in the WGA the percentage of bases that are consistently aligned in the WGA and in the respective pairwise genome alignment. For a given WGA of n genomes, we then report the average pairwise consistency score from all $\binom{n}{2}$ scores.

The total column score, first introduced in BaliBase (Thompson *et al.*, 2005), equals the percentage of identically aligned columns in a given alignment when compared to a so-called reference alignment.

The third statistic, the F-score is the harmonic mean of *precision* and *recall* for identical pairwise aligned bases. The F-score was used in the Alignathon competition (https://compbio.soe.ucsc.edu/alignathon/) for the comparison of two WGAs (Earl *et al.*, 2014), where one of the two WGA served as a reference. In our case, we considered the WGA generated by the original progressiveMauve-approach as the reference.

All three statistics have been widely used to evaluate multiple sequence alignment methods. The total column score is very conservative and prone to small changes in the alignment, and therefore even the best multiple sequence alignment methods may achieve only low TC scores. On the other hand, a comparable PC between a computed and reference alignment, as well as high TC and F-score, are reliable indicators for the similarity of the two alignments. For the calculation of all scores when comparing two WGAs, one needs to take care of possible inversions and translations. Again, the SuperGenome data structure serves extremely useful for this, and therefore we also used it for the calculation of these scores.

Computational Platform

We ran all WGAs on a Linux server with four *Intel(R) 197 Xeon(R) CPU E5-4610 v2 @ 2.30GHz* and 500 GB of memory. We measured the runtime with the `GNU time` command. During all WGA computations, `GPA` used the maximal number of threads needed to ensure that all independent subset alignments could be computed in parallel.

3 Results

With `GPA` we have extended `progressiveMauve` by the possibility to provide an existing sequence alignment in `XMFA`-format and align it to other sequences or alignments. Traditionally, profile alignment is conducted on a pairwise level, however with `GPA` a multiple profile alignment can be computed in one step. With this feature, a progressive alignment strategy that is typically performed along a binary guide tree can now be generalized to non-binary trees with fewer internal nodes. `GPA` provides a parameter k, that controls the maximal degree of multifurcation of every internal node and therefore how many genomes and/or profiles are merged at each step. Our overall intention for this strategy was to multiply align up to hundreds or more bacterial genomes with a significantly reduced runtime and at the same time achieve highly qualitative WGAs.

Runtime Evaluation

In the current implementation, we use `progressiveMauve` as underlying multiple genome alignment method. Therefore, our evaluation focuses on the direct comparison between the WGA produced by applying `progressiveMauve` to all genomes at once and the WGA computed using our iterative merging approach implemented in `GPA`. To compare the runtimes for the WGA construction of both `progressiveMauve` and `GPA`, all WGAs are computed from scratch. In addition, for data sets with less than 100 genomes we chose to split these into randomly distributed groups that were of equal size if possible. For all other data sets we used the guide tree produced by `progressiveMauve`, to determine the individual merging steps.

The results (see Table 2) show a general significant runtime decrease for the WGA construction of `GPA` compared to `progressiveMauve`, independent of the data set. As it can be seen from the results and Figure 3, the runtime of `progressiveMauve` increases at least quadratically with the number of aligned genomes, while for `GPA` the increase shows a linear dependency. This impact can mainly be seen for the WGAs with over 150 strains. None of the WGAs computed by `progressiveMauve` finished after 1650 hours (more than two months) of computing time, where the choice was made to not further wait for the result. Furthermore, the WGA with 80 strains of *K. pneumonia* did not finish after 350 hours, therefore the respective results are also not stated and compared. In contrast to this, `GPA` could compute all WGA computations within a range of hours to maximally days. For example for the WGA of 176 *S. aureus* strains `GPA` needed 92 minutes, while `progressiveMauve` ran more than 1650 hours without reporting a result, thus in this case `GPA` was at least 1000 times faster.

Independent from the used WGA construction method, the runtime comparison between the data sets of the species show differences as well. Here, an essential factor is the average genomic length. The computation time needed for WGAs with the same number of aligned genomes was less for the shorter genomes of *S. aureus* than for *M. tuberculosis* and *B. pertussis*, which show comparable average genome lengths as well as runtimes. The WGAs for *K. pneumonia* that has on average the longest genomes also needed the longest absolute runtime. When comparing the results of *M. tuberculosis* and *B. pertussis*, which have a similar length, the diversity of the genomes within a data set (reflected by a smaller overall pairwise consistency score) does not affect the runtime as much as the genome length.

Table 2. Runtime comparison between the WGA construction using the original `progressiveMauve` (PM) method and our SuperGenome-based iterative profile alignment approach of `GPA`. The results are divided into two distinct groups, whether for `GPA` a guide tree (guide tree) was used or not (random). In addition, the number of LCBs for the respective WGAs is reported.

	Data set		Runtime [min]		# LCBs	
	Organism	# Strains	PM	GPA	PM	GPA
random	*B. cereus* (5.760Mb)	13	225	61	380	214
	L. monocytogenes (2.975Mb)	30	628	28	293	254
	C. trachomatis (1.046Mb)	72	351	14	4	80
guide tree	*M. tuberculosis* (4.385Mb)	10	27	10	3	3
		20	98	22	16	34
		40	475	66	230	118
		80	1665	241	852	313
		128	*	343	-	312
	K. pneumoniae (5.590Mb)	10	55	28	177	200
		20	691	66	784	1341
		40	9401	101	3369	3149
		80	*	170	-	8121
		166	**	1620	-	21149
	S. aureus (2.847Mb)	10	16	5	221	119
		20	64	12	415	665
		40	552	20	1222	1140
		80	5213	57	2492	4650
		176	**	92	-	9239
	B. pertussis (4.100Mb)	10	24	7	64	82
		20	99	25	88	120
		40	503	68	165	324
		80	1683	113	314	644
		326	*	278	-	3838

computation aborted after: * 350 hours, ** 1650 hours

The number of LCBs, which represent genomic architectural differences between the genomes, differ a lot between the organisms analysed here. For example, WGAs of *M. tuberculosis* and *B. pertussis* with similar genome lengths have vastly different number of LCBs. On the other hand, as can be seen from Table 2 with increasing number of aligned genomes from the same organism also the number of LCBs always increases, independent whether the WGA was computed using `progressiveMauve` or `GPA`. When comparing `progressiveMauve` and `GPA` for a given set of genomes, the number of LCBs is largely of similar order of magnitude, though in most cases (but not all) the WGA computed with `GPA` had more LCBs than those for the WGA computed with the original `progressiveMauve`.

Qualitative Comparison of WGAs

In order to compare the WGA computed using `GPA` with the one computed by the original approach of `progressiveMauve`, we applied three evaluation criteria (see Methods section). For each data set, we calculated the average pairwise consistency (PC) score, total column (TC) score and F-score (see Table 3).

For each dataset, independent of the number of aligned strains per species, the average PC score is very similar between the `progressiveMauve`- and `GPA`-derived WGAs. The largest difference

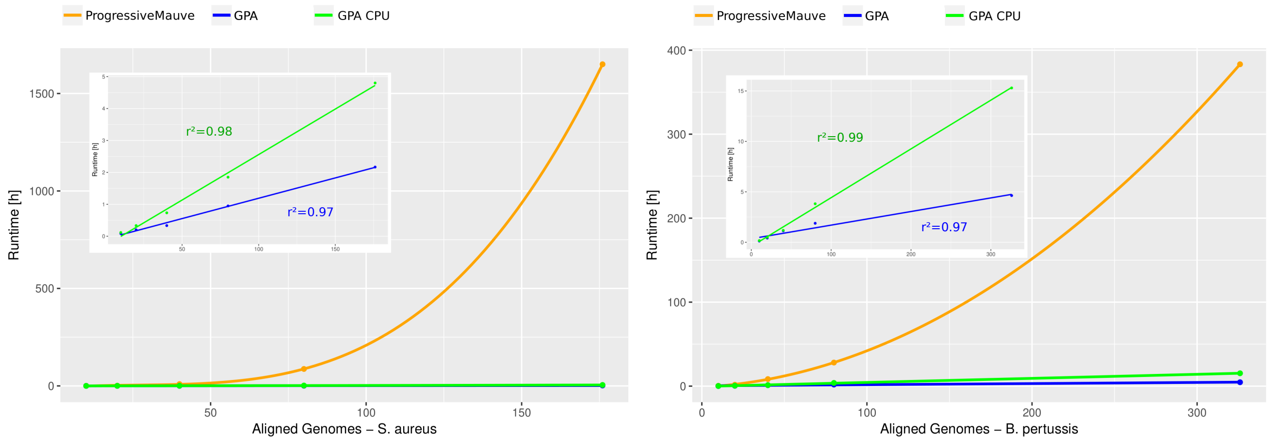

Fig. 3. Comparison of the measured computational runtime needed for the construction of the WGA depending on the number of genomes for the data sets of *S. aureus* (left) and *B. pertussis* (right). In addition to the direct comparison between `progressiveMauve` (orange) and `GPA`, the upper left section only shows the runtime of `GPA` (blue) and `GPA` CPU time (green), together with the r^2 values for the linear regression.

of less than 4% is observed for the alignment of 80 strains of *B. pertussis*. Overall, the pairwise consistency only slightly decreases within a species when adding more genomes to the alignment. The largest difference was observed for the WGAs computed for *B. pertussis*: here the average PC score dropped from 57.9% for 10 genomes down to 37.7% when 326 genomes were aligned. An exception has been observed for the *K. pneumoniae* WGAs. Here the average PC score of the WGA built from 10 genomes was smaller than for the WGAs with 20 and 40 genomes. Generally, the PC score differs most strongly when comparing different bacteria. While the WGAs of *M. tuberculosis* and *C. trachomatis* have average PC scores greater than 90%, the WGAs of *K. pneumoniae* achieve a maximum of 41%.

The TC score, which represents the fraction of identically aligned columns, compares the WGAs computed by `GPA` with the one derived with the original `progressiveMauve`. Overall the TC score is above 60% in most cases, indicating that `GPA` aligns a majority of all columns identically to the original `progressiveMauve` even for larger WGAs with up to 80 genomes. Also, none of the WGAs of our test data sets shows a TC score below 30%. Similar to the PC score the TC score differs between different organisms though here the biggest differences are seen when increasing the number of genomes within a species. Interestingly, the PC and TC scores do not necessarily correlate a lot, i.e., WGAs with similar PC scores do not necessarily have similar TC scores. For example, WGAs with low PC scores (as seen for example in the case of *B. pertussis*) may have higher TC scores than WGAs with high PC scores (e.g. *S. aureus*).

Another indicator of the high similarity of the WGAs computed with `GPA` and those computed with `progressiveMauve` is the F-score. Independent of the organism as well as the number of genomes the value is in most cases above 0.97, and never drops below 0.93. Here, in general the *precision* score is higher than the *recall* for the resulting F-score (data not shown). We observed that WGAs with a high TC score also have a high F-score. On the other hand, increasing the number of genomes for a WGA generally leads to a significant decrease of the TC score, while this behavior is not observed for the F-score. An example is *S. aureus*, where the lowest TC score in all comparisons was achieved, while the F-score is still above 0.95.

Table 3. The WGA generated by `progressiveMauve` (PM) and `GPA` were evaluated with respect to their average pairwise consistency (PC), the total column (TC) score (% of identical aligned columns in PM) and F-score. Both, for the calculation of the TC and F-score, PM is used as the reference. GPA was run with several different parameters k for the merge size, reported for each data set is the one with the highest PC score.

	Data set		PC Score		TC Score	F-Score
	Organism	# Strains	PM	GPA	GPA in PM	
random	*B. cereus* (5.760Mb)	13	60.00%	60.63%	62.40%	0.977
	L. monocytogenes (2.975Mb)	30	73.41%	73.92%	61.50%	0.989
	C. trachomatis (1.046Mb)	72	98.35%	98.36%	86.68%	0.995
guide tree	*M. tuberculosis* (4.385Mb)	10	97.90%	97.81%	93.79%	0.998
		20	96.83%	96.83%	87.74%	0.990
		40	96.00%	95.80%	84.93%	0.990
		80	92.36%	91.35%	72.45%	0.983
		128	-	89.27%	-	-
	K. pneumoniae (5.590Mb)	10	38.77%	38.86%	72.83%	0.989
		20	41.31%	41.07%	65.95%	0.982
		40	39.32%	40.81%	42.49%	0.951
		80	-	40.32%	-	-
		166	-	35.80%	-	-
	S. aureus (2.847Mb)	10	82.45%	81.50%	72.77%	0.981
		20	81.58%	80.00%	69.18%	0.979
		40	76.33%	74.78%	58.17%	0.972
		80	73.02%	71.96%	30.76%	0.959
		176	-	69.12%	-	-
	B. pertussis (4.100Mb)	10	58.77%	57.88%	94.15%	0.989
		20	58.56%	56.94%	82.81%	0.968
		40	57.65%	55.44%	69.45%	0.959
		80	58.69%	54.74%	61.85%	0.934
		326	-	37.65%	-	-

Impact of compressing the guide tree

Next, we analysed the impact of the multifurcation parameter k, which is used to compress the input guide tree. This parameter k reflects the maximal degree of each internal node of the guide tree, which represents the maximal number of profiles or sequences that are merged in a

Table 4. Evaluation of GPA-derived WGAs with respect to the maximal number of sequences merged at a time for the *M. tuberculosis* data set.

#Strains	max. degree k	Runtime [min.]	PC score	TC score	#LCB	F-Score
10	3	6	97.11%	92.64%	17	0.997
10	5	8	97.65%	93.76%	3	0.997
10	7	10	97.81%	93.79%	3	0.998
20	3	9	96.76%	86.61%	47	0.986
20	5	13	96.42%	86.83%	36	0.989
20	7	15	96.67%	86.06%	8	0.986
20	9	22	96.83%	87.74%	34	0.990
40	7	32	94.85%	82.33%	191	0.987
40	9	36	95.30%	83.32%	159	0.989
40	12	66	95.80%	84.93%	118	0.990
80	12	78	91.22%	71.65%	220	0.982
80	17	101	91.13%	72.02%	284	0.982
80	22	241	91.35%	72.45%	313	0.983
128	17	177	88.77%	-	524	-
128	22	343	89.27%	-	312	-

single step during the WGA computation. Clearly, as can be seen from Table 2, the runtime of progressiveMauve increases significantly with increasing genomes. At the same time, the WGAs computed with progressiveMauve have in most cases a higher PC score. Thus, a trade-off between the number of genomes aligned at a time and runtime needs to be considered when choosing k.

For this purpose, for a given initial guide tree on a given set of genomes, we computed WGAs with GPA using different k. For each k we reported the runtime, PC, TC as well as the F-score. As expected larger k result in larger runtimes (see Table 4 for the case of *M. tuberculosis*). On the other hand, the PC and TC score, as well as the F-score, are not greatly affected by the choice of k, while the number of LCBs varies, though a clear pattern cannot be deduced. Similar observations have also been made for the other data sets of *S. aureus*, *K. pneumoniae* and *B. pertussis* (data not shown). Furthermore, the analysis of the WGAs, where no guide tree was provided for GPA, and therefore equally sized groups were merged, shows that at least for smaller sized WGAs highly similar results to progressiveMauve can be achieved as well (see upper parts of Table 2 and 3).

4 Discussion

In this paper we introduced GPA, a tool which extends the whole-genome alignment method progressiveMauve (Darling *et al.*, 2010) by the possibility to align individual genomes or whole-genome alignments to a given profile genome alignment. The challenge in the case of profile-based WGA is that genomic rearrangements like translocations and inversions have to be considered and therefore the profile of genome alignments cannot be as easily derived as in typical multiple sequence alignments. Each individual genome as well as whole-genome alignment represents its own coordinate system, and one challenge when merging a given WGA with individual genomes or another WGA is the transfer of the individual coordinate systems into a common one. With our SuperGenome data structure we have introduced a novel concept that allows the extension of a given WGA by further genomes or other WGAs, through a coordinate transfer along a guiding alignment of their profiles. Currently, we use progressiveMauve together with GPA and guarantee to adhere also to the locally collinear blocks computed by progressiveMauve.

Clearly, progressiveMauve has been shown to be among the best methods for the alignment of bacterial genomes because of its ability to consider rearrangements between the genomes. However, the runtime of progressiveMauve which was shown to be at least quadratic (Darling *et al.*, 2010), as well as lack of parallelization, prevent the computation of WGAs of hundreds or even thousands of genomes. Thus the second goal of this paper was to significantly reduce the runtime of progressiveMauve.

Though by definition the SuperGenome can be derived from whole-genome alignments of arbitrary species, we have restricted our analyses to closely related prokaryotic genomes, i.e., aligning different strains from the same species. For our largest dataset of *B. pertussis* with 326 genomes, we could show that GPA needed less than 7 hours, while a sole progressiveMauve-based alignment was not finished after 350 hours. A linear regression analysis showed a clear linear relationship between runtime and genomes (see Figure 3) and based on the coefficients of the regression the computation of a WGA with 1000 *B. pertussis* genomes would need around 14 hours.

One central feature of progressiveMauve is a binary guide tree and a WGA that is progressively computed along this tree. The key algorithmic ingredient in GPA is the use of a compressed rather than strictly binary guide tree and the adaptation of our introduced profile WGA to the alignment of several profiles in one step. The compressed guide tree leads to a significant reduction of internal nodes and therefore subalignments that are merged at a time. Using our SuperGenome based profile alignment approach together with a compressed guide tree, we could show that GPA is orders of magnitude faster than the original progressiveMauve.

The results of GPA depend on the guide tree and the number of profiles that are merged at each node, our parameter k. As the guide tree determines the order in which the genomes are subsequently added, it is only plausible that the quality of the final WGA decreases if the tree does not reflect their phylogenetic relationship.

The comparison of progressiveMauve and GPA shows that as a trade-off for the reduced runtime of GPA the quality of the WGAs is in most of the cases slightly worse. However, the small differences of the pairwise consistency scores by only a few percents together with the high TC and F-scores show that the WGAs computed with GPA are highly similar to those computed using the original progressiveMauve approach.

We also showed that there is a trade-off between runtime and quality when deciding how many profiles are merged at the same time. The parameter k controls the maximal number of profiles aligned at a time with progressiveMauve. On the other hand, larger k lead to higher PC scores, runtime. Since the results of progressiveMauve show that the computation time needed for WGAs with less than 20 genomes is in most cases below 2 hours, a k between 10 and 20 is currently our default and recommended value. In addition, the generation of WGAs with thousands of genomes by GPA benefits from the usage of large cluster systems, because the independent alignments can now be computed in parallel.

Currently, GPA makes use of progressiveMauve as the underlying whole-genome alignment method, however, the modular implementation allows in principle the support of other whole-genome alignment methods. Since the SuperGenome data structure requires a one-to-one mapping between the nucleotides of the genomes, only methods are feasible that generate alignments with this feature. The disadvantage of these type of methods, which include progressiveMauve, is that duplicated regions cannot be aligned.

In the current version memory consumption is an issue of GPA, since the complete SuperGenome data structure is kept in memory to speed up computation. For example, for the alignment of 326 *B. pertussis* strains, GPA needed more than 300 GB RAM. In future, we will address this issue to enable even larger WGAs with significantly less memory requirements.

The current version of GPA and the SuperGenome data structure uses the consensus sequence derived from the profiles of WGAs, which we used as a simple solution to efficiently align several profiles. Apparently, this is not an optimal solution, and therefore other methods that use a more sophisticated representation of the profiles for the merging step will be considered as possibly improved solutions in a future release.

A possible application of GPA can be seen from comparative genome analyses that rely on a WGA. An example is AureoWiki (Fuchs *et al.*, 2017) (`http://aureowiki.med.uni-greifswald.de/`), a database of 32 different *S. aureus* strains that has been built from the results of a WGA-based pan-genome computation. To incorporate new *S. aureus* strains into the database, would require a WGA with the new genomic sequences. Computing this WGA from scratch could introduce changes that are not consistent with prior results. Here, the profile-based extension of the WGA by GPA preserves the former WGA and allows the addition of new strains without the necessity to completely rebuild the database.

As a conclusion, GPA introduces a time efficient computation of large-scale WGAs through the usage of WGA-profile alignments and adds further utility by the possibility to extend existing WGAs.

References

Abdelbary, M. M., Senn, L., Moulin, E., Prod'hom, G., Croxatto, A., Greub, G., and Blanc, D. S. (2018). Evaluating the use of whole-genome sequencing for outbreak investigations in the lack of closely related reference genome. *Infection, Genetics and Evolution*, **59**, 1–6.

Angiuoli, S. V. and Salzberg, S. L. (2010). Mugsy: fast multiple alignment of closely related whole genomes. *Bioinformatics*, **27**(3), 334–342.

Blanchette, M., Kent, W. J., Riemer, C., Elnitski, L., Smit, A. F., Roskin, K. M., Baertsch, R., Rosenbloom, K., Clawson, H., Green, E. D., *et al.* (2004). Aligning multiple genomic sequences with the threaded blockset aligner. *Genome research*, **14**(4), 708–715.

Bryant, J. M., Grogono, D. M., Greaves, D., Foweraker, J., Roddick, I., Inns, T., Reacher, M., Haworth, C. S., Curran, M. D., Harris, S. R., *et al.* (2013). Whole-genome sequencing to identify transmission of mycobacterium abscessus between patients with cystic fibrosis: a retrospective cohort study. *The Lancet*, **381**(9877), 1551–1560.

Chan, C. X. and Ragan, M. A. (2013). Next-generation phylogenomics. *Biology direct*, **8**(1), 3.

Darling, A., Mau, B., and Perna, N. (2010). progressiveMauve: multiple genome alignment with gene gain, loss and rearrangement. *PloS one*, **5**(6), e11147.

Darling, A. C., Mau, B., Blattner, F. R., and Perna, N. T. (2004). Mauve: multiple alignment of conserved genomic sequence with rearrangements. *Genome research*, **14**(7), 1394–1403.

Earl, D., Nguyen, N., Hickey, G., Harris, R. S., Fitzgerald, S., Beal, K., Seledtsov, I., Molodtsov, V., Raney, B. J., Clawson, H., *et al.* (2014). Alignathon: a competitive assessment of whole-genome alignment methods. *Genome research*, **24**(12), 2077–2089.

Fuchs, S., Mehlan, H., Bernhardt, J., Hennig, A., Michalik, S., Surmann, K., Pané-Farré, J., Giese, A., Weiss, S., Backert, L., *et al.* (2017). Aureowiki Ìµ the repository of the staphylococcus aureus research and annotation community. *International Journal of Medical Microbiology*.

Gotoh, O. (1990). Consistency of optimal sequence alignments. *Bulletin of Mathematical Biology*, **52**(4), 509–525.

Herbig, A., Jäger, G., Battke, F., and Nieselt, K. (2012). GenomeRing: alignment visualization based on SuperGenome coordinates. *Bioinformatics*, **28**(12), i7–i15.

Hogeweg, P. and Hesper, B. (1984). The alignment of sets of sequences and the construction of phyletic trees: an integrated method. *Journal of molecular evolution*, **20**(2), 175–186.

Jandrasits, C., Dabrowski, P. W., Fuchs, S., and Renard, B. Y. (2018). seq-seq-pan: Building a computational pan-genome data structure on whole genome alignment. *BMC genomics*, **19**(1), 47.

Liu, K. and Warnow, T. (2014). Large-scale multiple sequence alignment and tree estimation using sate. In *Multiple Sequence Alignment Methods*, pages 219–244. Springer.

Medini, D., Donati, C., Tettelin, H., Masignani, V., and Rappuoli, R. (2005). The microbial pan-genome. *Current opinion in genetics & development*, **15**(6), 589–594.

Notredame, C., Higgins, D. G., and Heringa, J. (2000). T-coffee: a novel method for fast and accurate multiple sequence alignment1. *Journal of molecular biology*, **302**(1), 205–217.

Sabat, A., Budimir, A., Nashev, D., Sá-Leão, R., Van Dijl, J., Laurent, F., Grundmann, H., Friedrich, A., of Epidemiological Markers (ESGEM, E. S. G., *et al.* (2013). Overview of molecular typing methods for outbreak detection and epidemiological surveillance. *Eurosurveillance*, **18**(4), 20380.

Thompson, J. D., Koehl, P., Ripp, R., and Poch, O. (2005). Balibase 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins: Structure, Function, and Bioinformatics*, **61**(1), 127–136.

Treangen, T. J., Ondov, B. D., Koren, S., and Phillippy, A. M. (2014). The harvest suite for rapid core-genome alignment and visualization of thousands of intraspecific microbial genomes. *Genome biology*, **15**(11), 524.