2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)

Accelerating Clustering using Approximate Spanning Tree and Prime Number based Filter

Dhananjai M. Rao
CSE Department
Miami University
Oxford, OH 45056, USA
raodm@miamiOH.edu

Sutharzan Sreeskandarajan
Department of Biology
Miami University
Oxford, OH 45056, USA
sreesks@miamiOH.edu

Chun Liang
Department of Biology
Miami University
Oxford, OH 45056, USA
liangc@miamiOH.edu

***Abstract*—Motivation: Clustering genomic data, including those generated via high-throughput sequencing, is an important preliminary step for assembly and analysis. However, clustering a large number of sequences is time-consuming. Methods: In this paper, we discuss algorithmic performance improvements to our existing clustering system called PEACE via the following two new approaches: ❶ using Approximate Spanning Tree (AST) that is computed *much faster* than the currently used Minimum Spanning Tree (MST) approach, and ❷ a novel Prime Numbers based Heuristic (PNH) for generating features and comparing them to further reduce comparison overheads. Results: Experiments conducted using a variety of data sets show that the proposed method significantly improves performance for datasets with large clusters with only minimal degradation in clustering quality. We also compare our methods against `wcd-kaboom`, a state-of-the-art clustering software. Our experiments show that with AST and PNH underperform `wcd-kaboom` for datasets that have many small clusters. However, they significantly outperform `wcd-kaboom` for datasets with large clusters by a conspicuous ~550×with comparable clustering quality. The results indicate that the proposed methods hold considerable promise for accelerating clustering of genomic data with large clusters.**

***Index Terms*—Clustering, Minimum Spanning Tree, d2**

I. INTRODUCTION

Clustering nucleotide sequence data has many applications ranging from identification of gene expressions [9], reducing the runtime of sequencing reads to the genomes [18], [8], and inferring phylogenetic relationships of organisms [5], [22], [28]. Clustering approaches are being widely used to group or "bin" the reads belonging to a taxonomic group together in metagenomics analyses [20]. Clustering approaches can be used to reduce sequencing errors by pre-clustering reads in the initial stages of data analysis [13]. Clustering whole large sets of chromosome, genomes and other larger nucleotide sequences using alignment-free methods enable rapid identification of phylogenetic relationships between organisms. An example of such an application would be the clustering of large sets of viral and other microbial genomes [6], [22], [28]. Clustering such microbial data enables understanding the microbial phylogenetic distribution in large datasets and potentially identifying novel microbial taxa [28].

A. Current state-of-the-art

Classical sequence clustering approaches were based on sequence alignments using dynamic programming [15], [21], [25]. The major drawback of alignment-based methods is runtime and memory consumption – *i.e.,* they tend to be very slow and consume a lot of memory [31]. Hence they are not preferred when clustering large datasets. Moreover, even though alignment-based approaches are considered to be highly accurate, the quality of the results could be questionable due to several deficiencies as discussed by Zielezinski *et al* (they specifically discuss five cases) [31].

Alignment-free approaches are more practical alternatives to dynamic programming-based methods. Instead of dynamic programming, alignment-free approaches rely on partial comparisons and pseudo-metrics for clustering. For example, our preliminary clustering software called PEACE used a well established alignment-free, pseudo-metric called `d2` to estimate similarity or "distance" between pairs of reads. PEACE uses the `d2` score (more details in Section II) to build a Minimum Spanning Tree (MST) and then cuts the MST to form clusters. `d2` is also used by `wcd` [8]. Recently, Hazelhurst *et al* [9] further enhanced `wcd` with a suffix array based approach called `kaboom`, to significantly improve the performance of `wcd`. `wcd-kaboom` has shown to outperform (both in runtime and cluster-quality) several mainstream clustering software, including ESTate, xsact, PaCE, CAP3, and TGICL [9]. Consequently, we use `wcd-kaboom` as the reference for performance comparisons.

B. Motivation for this research

The primary advantage of alignment-free methods is that they run *much faster* and often have a small memory footprint but at the cost of some degradation in clustering quality. Consequently, alignment-free methods are widely used for clustering large datasets [31], [27]. However, the ongoing exponential growth in data volumes, continue to pose challenges in accomplishing fast and effective clustering. Even with just 1% pairwise comparisons [9], `wcd-kaboom` takes ~149 minutes (on an Intel Xeon® Gold 6126 CPU @ 2.6 GHz) to cluster ~65K Haemagglutinin (HA) reads. Such a long runtime of ~3.5 hours for a relatively small data set, with a state-of-the-art tool, highlights the ongoing challenges, thereby motivating the need for high-performance clustering solutions.

978-1-5386-5555-9/19/$31.00 ©2019 IEEE
DOI 10.1109/IPDPSW.2019.00037

C. Proposed solution: An overview

This research proposes a high-performance clustering solution to effectively cluster large datasets of both short and long genomic data. The objective is to reduce runtime without significantly degrading clustering-quality. We propose to improve performance using two approaches: ❶ first we focus on improving performance by changing the Minimum Spanning Tree (MST) clustering approach used in PEACE with an Approximate Spanning Tree (AST), and ❷ adding a Prime-Number based Heuristic (PNH) based on the previously proposed prime number based scoring system for inverted repeats detection [23]. We discuss the conceptual underpinnings and algorithmic details of these two approaches in Section V and Section VI.

The paper presents results from experiments conducted using a broad spectrum of data sets in Section VII. We also compare performance of the proposed method with `wcd-kaboom` to highlight the effectiveness of AST and PNH. For example, with AST, the runtime for clustering the HA dataset is just 16 seconds, instead of 8,940 seconds for `wcd-kaboom`, a 550× performance improvement. This is a very substantial performance improvement with clustering quality slightly better than `wcd-kaboom`. This study establishes the potential of the AST approach in providing a high-performance clustering tool well suited for clustering datasets with large clusters. The PNH is a promising approach for further increasing the speed of large sets of viral genomic data.

II. BACKGROUND: PEACE AND D2

PEACE (Parallel Environment for Assembly and Clustering of Gene Expression) is a user-friendly nucleotide sequence clustering tool which is designed to cluster transcript reads obtained by Sanger and NGS sequencing technologies [18], [17]. PEACE clusters nucleotide sequences based on a Minimum Spanning Tree (MST) method as summarized in Figure 1. An MST is generated (via Prim's algorithm) using pairwise sequence comparison between the reads in alignment-free manner using the d2 algorithm – *i.e.,* the MST edges are weighted using d2 pseudo-metric.

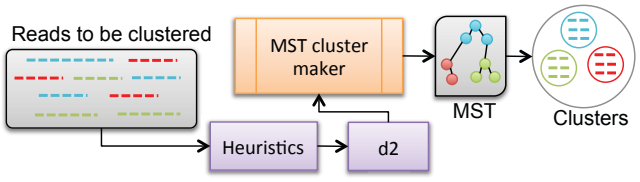


Fig. 1. Overivew of clustering in PEACE

The d2 algorithm works by comparing the frequency of words (strings of a fixed length) appearing in a limited region of each read. PEACE uses a default word size of 6 base pairs. Fragments overlapping by a sufficient length will share neighborhoods of enough similarity to ensure a small distance (close to zero) even in the presence of a moderate number of base errors. Specifically, PEACE uses a sliding window of fixed size r (defaults to 100 in PEACE) to generate d2 score for two sequences x and y ($|x| \geq r$, $|y| \geq r$) via:

$$d2(x,y) = min\{d2(u,v) : u \sqsubseteq x, v \sqsubseteq y, |u| = |v| = r\}$$

where $u \sqsubseteq x$ denotes that u is a substring of x. Defined as such, d2 is, in a mathematical sense, a pseudo-metric – *i.e.,* $d2(x,y) = 0$ does not imply $x = y$. However, as $d2(x,y) \to 0$, x is more similar to y and vice versa. Sufficiently similar reads, based on a user-defined threshold, are clustered together.

Computing d2 distance is expensive requiring a few milliseconds for 1 kilobase sequence. Consequently, to minimize unsuccessful d2 comparisons, PEACE uses *u/v* and *t/v* filtering heuristics that run in microseconds. As indicated in Figure 1, the heuristics minimize the number of d2 comparisons by eliminating comparisons of sufficiently differing sequences – *i.e.,* sequences that would result in a very high d2 distance. A more detailed discussion on the heuristics is available in the literature [18].

Using MST and d2, PEACE has shown to cluster transcript reads with high accuracy and sensitivity. It can be run both sequentially and in parallel, though in this study we are not exploring parallelization. The ability to run PEACE using a Graphical User Interface and a robust set of default/out-of-the-box parameters make it a very easy to use tool for biologists.

PEACE is a highly modularized framework developed in C++ and heavily relies on the language's object-oriented features. The framework consists of loosely coupled core subsystems. The subsystems are further modularized into loosely coupled components. Though its modular design the framework provides the user with a wide choice of filters, heuristics and clustering methods. In this study, we have used these features to implement our proposed Approximate Spanning Tree (AST) and Prime Number-based Filter (PNF).

III. ASSESSMENT METRICS

This study primarily focuses on reducing runtime for clustering. However, the quality of clustering is important [9]. In this study we have used two widely used metrics for assessing the quality of clustering, namely Normalized Mutual Information (NMI) and Purity. NMI is a popular measure of clustering quality based on Information Theory [24]. Values of NMI range from 0 to 1. High NMI values indicate good clustering. In this work, NMI is the primary measure of clustering quality.

In addition to NMI, purity has also been used as a measure to further validate clustering quality. Purity values range from 0 to 1.0. Higher purity values indicate that a cluster does not have reads from other clusters mixed into it. The R packages aricode (https://cran.r-project.org/web/packages/aricode/index.html) and IntNMF [4] have been used to calculate NMI and purity, respectively.

The NMI and purity values have been calculated using the clustering output from PEACE as the reference. Using the output from PEACE is motivated by two reasons. First, PEACE has been tested by multiple authors and has shown

to produce good quality clustering [18], [9]. Second, our objective is to improve the performance of PEACE without impacting clustering quality. Consequently, we have used the output of PEACE as the reference to assess clustering quality.

IV. RELATED WORKS

Many alignment-free sequence clustering tool have been proposed to cluster large sets biological sequences. CD-Hit [14], WCD [8], PEACE [18], and MeShClust [11] are some good examples for alignment-free clustering tools. The popular clustering tool CD-Hit has the ability to cluster large sets of nucleotide sequences, while WCD and PEACE are less fast but more accurate sequence clustering alternatives. CD-Hit clusters sequences using a distance measure based on the minimum number of shared short strings (words) between two sequences. WCD and PEACE use a similar word-based sequence comparison approach but differ from CD-Hit in the distance measure used to compare two sequences. Both WCD and PEACE utilize the d2 distance measure. PEACE differs from WCD by using a novel Minimum Spanning Tree (MST) based clustering approach where the tree branches are weighted by d2 distances. It has been shown that the speed of WCD can be improved by the use of a filter-based preprocessing approach. The proposed suffix array-based filter for WCD named Kaboom was shown to make WCD much faster than PEACE [9]. Essentially, the Kaboom filter significantly reduces the number of d2 comparisons to be performed, to $< 1\%$ in many cases. The filter processing runtime is amortized by reducing the comparatively slower d2 comparison. However, the Kaboom filter requires generation of suffix trees using a separate utility called `mkesa`. MeShClust is a recently proposed tool which claims to cluster nucleotide sequences with high accuracy and considerable speed [11].

The popular tools CD-Hit and WCD are primarily made to cluster expression data such generated by sequencing methods as expressed sequence tags and DNA sequence reads. They are shown to be well suited for clustering next-generation sequencing data and other similar data consisting of short length sequences. The utility of CD-Hit and WCD in clustering whole chromosomes and genomes (which are larger in length than typical sequencing reads) is questionable and has not been extensively tested. Even though MeShClust claims to be efficient in clustering short sequencing reads and relatively larger whole genomes, it was not tested extensively for its whole chromosome/genome and long read clustering ability.

Minimum Spanning Tree (MST) is a widely used method in clustering nucleotide sequences and inferring phylogenetic relationships between the sequences. Examples of such uses of MST include clustering nucleotide sequences to aid in sequence domain searching [7], performing intraspecific phylogenetic analysis [19] and inferring phylogeographic relationships [1]. A major issue associated with the MST is the time complexity of the tree generation. The MST generation has the time complexity of $O(n^2)$ (where n is the number of reads to be clustered) [30]. Several approximate MST models approaches have been proposed to reduce the time complexity using heuristics approaches. An example of an approximate MST approach is minimizing the comparison overhead by building MSTs on K-Means partition of the considered dataset [30]. Another example is the use of a novel centroid-based nearest neighbor rule for the fast approximate MST generation (with the time complexity of $O(n^{\frac{3}{2}} log\, n)$ [12]. Employing such heuristic approaches can aid in the development of efficient MST-based nucleotide clustering and phylogenetic tools [19].

V. APPROXIMATE SPANNING TREE

PEACE uses a Minimum Spanning Tree (MST) based method for clustering [18]. In this method, each read in a dataset is deemed as a node in the MST. The MST is constructed using Prims's algorithm, with d2 scores serving as the edge weights. Recollect that, as discussed in Section II, d2 scores tend to zero for highly similar reads – *i.e.,* similar reads will be adjacent or nearby branches in the MST. Once an MST is constructed, clustering is accomplished by cutting the MST at branches that are significantly distant, based on a given clustering-threshold, currently set to 1.0 in PEACE. Note that this 1.0 is *not* a normalized value but a putative d2 score (with 10.0 corresponding to infinity) that indicates sufficiently different reads.

The most time-consuming operation is the construction of MST. The most dominant issue is that MST construction is an $O(n^2)$ algorithm (where n is the number of reads), which is prohibitive for large datasets [9]. MST construction results in n^2 calls to d2-score computation, which in itself, is expensive and requires few milliseconds. Here, heuristics (than run in microseconds) help by eliminating over 80% of d2 comparisons. The proposed Prime Number based Heuristic (PNH) aims to help in this area and is discussed in Section VI. Nevertheless, the remaining 20% of d2 comparisons take substantial runtime.

The Approximate Spanning Tree (AST) aims to address the $O(n^2)$ time complexity by rapidly adding nodes to the tree, thereby reducing the time constants. Note the AST does not change the asymptotic time complexity of $O(n^2)$ but rather aims to rapidly reduce n. An overview of the AST construction method is summarized in Algorithm 1. The algorithm begins with a list of reads (`estList`) and an arbitrarily chosen `root` read. Next, given a `root`, sufficiently similar reads are obtained via call to the `getReads` method. This method has an $O(n)$ time complexity and uses heuristics to reduce the number of calls to the heavyweight `d2` computation method. Only reads that are below the specified `ASTthresh` threshold are returned.

Unlike MST construction that adds only the most similar read, the AST adds all of the similar reads, thereby rapidly reducing the number of pending reads (*i.e.,* n). Reads are considered similar if their d2 distance is below a given AST-threshold value. Similar to d2 scores, the AST-threshold value is not normalized and this value cannot exceed the clustering-threshold, which is currently set to 1.0. The AST-threshold value is a critical factor that influences the overall performance. The chart in Figure 2 illustrates the impact of

Algorithm 1: Approximate Spanning Tree (AST)

```
1  begin AST(estList, root, ASTthresh)
2      ast.add(root, 0) // d2 distance is 0
3      while ! estList.empty() do
           // Get reads with d2 score < ASTthresh
4          nearList = getReads(estList, root, ASTthresh)
           // Add all similar reads to AST
5          foreach <est, d2score> ∈ nearList do
6              ast.add(root, est, d2score)
7              estList.remove(est)
8          end foreach
9          root = nearList.last().est
10     end while
11     return ast
12 end AST
13 begin getReads(estList, root, ASTthresh)
14     nearList = {}
15     foreach est ∈ estList do
16         if heuristicChain.shouldAnalyze(root, est) then
17             d2score = d2(root, est)
18             if d2score < ASTthresh then
19                 nearList.add(<est, d2score>)
20             end if
21         end if
22     end foreach
23     return nearList
24 end getReads
```

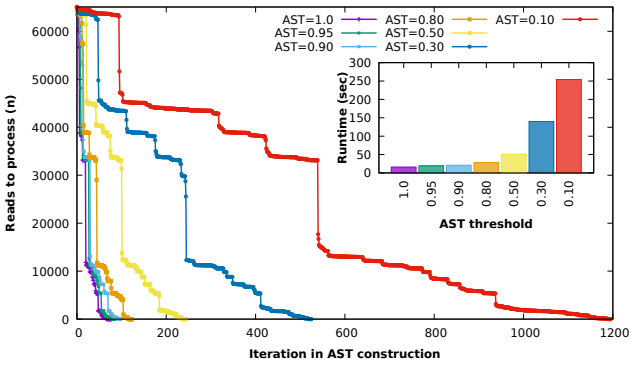
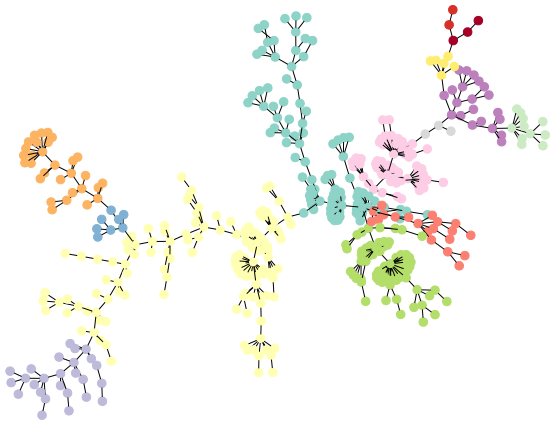


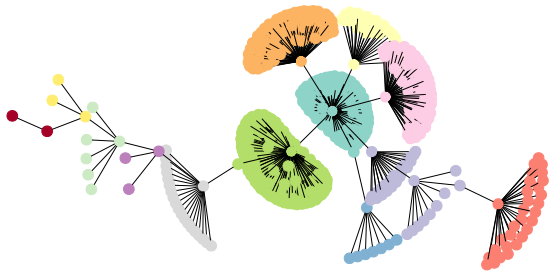
Fig. 2. Impact of AST threshold

changing the threshold on a Haemagglutinin (HA) dataset with ~65K reads. Larger AST-thresholds allow more reads to be added to the AST, thereby rapidly reducing n. This results in fast runtime but the resulting spanning tree drifts further away from the MST. In contrast, smaller threshold values result in a tree closer to an MST – *i.e.,* AST-threshold of 0.0 produces an MST. However, smaller thresholds increase runtime with an $O(n^2)$ time complexity as illustrated by the inset chart in Figure 2.

A comparison of MST and its corresponding AST is shown in Figure 3. The MST has been generated using 500 (for read-ability) randomly chosen reads. The AST has been generated with an AST-threshold of 1.0 (*i.e.,* `AST=1.0` in Figure 2). The nodes in the tree have been color-coded based on the clusters they were assigned. The key feature of interest is the structure of the two trees. The MST has many deeper branches because nodes are added one at a time. On the other hand, the AST has a much shallower structure with some nodes having a very high degree. The high degree nodes are the key in improving performance as they enable many reads to be added to the AST thereby rapidly reducing n as illustrated by the curve for `AST=1.0` in Figure 2.



(a) MST



(b) AST

Fig. 3. Comparison of MST and AST

The impact of varying the AST-threshold on the clustering quality is illustrated by the charts in Figure 4. As illustrated by the NMI curve (see Section III for details on NMI), the quality of clustering degrades as the AST-threshold is increased. NMI degrades as the number of clusters increases with increase in AST-threshold. This is because in the AST potentially shorter links are not captured. This results in additional edges to be cut causing more clusters to be generated. However, as indicated by the chart, the purity of clustering does not degrade – *i.e.,* the AST does not incorrectly cluster disparate reads together. The observation suggests that a cluster-merging operation can

optionally be used to reduce the number of clusters, if needed.

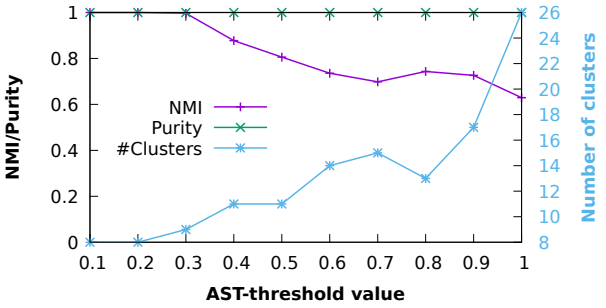


Fig. 4. Impact of AST-threshold

VI. Prime Number-based Heuristic (PNH)

A prime-numbers based heuristic is proposed to further reduce filtering overheads. The PNH is based on the previously proposed prime-numbers based scoring system for perfect inverted repeats detection [23]. In PNH, each nucleotide is represented by a prime number (or its negative value) and a nucleotide sequence is represented as a vector of prime numbers (corresponding to nucleotides). The prime number score or feature for a nucleotide is calculated by summing the corresponding vector of prime numbers. A given number of features, say p, is extracted from a given nucleotide sequence by breaking the sequences into p non-overlapping fragments as shown in Figure 5 (here, p = 3). For each of the p sub-sequences, a prime number score is computed via:

$$S_l = \sum_{i=1}^{L} p_i, \text{where} \quad p_i = \begin{cases} P_A & \text{if } i = \texttt{A} \\ -P_A & \text{if } i = \texttt{T} \\ P_G & \text{if } i = \texttt{G} \\ -P_G & \text{if } i = \texttt{C} \\ 0 & otherwise \end{cases}$$

In the example shown in Figure 5, P_A = 71 and P_G = 113. The prime values have been empirically chosen to provide a balance between sensitivity and specificity. The above approach results in a p dimensional, primes-based feature vector to be generated for a given nucleotide sequence.

Two nucleotide sequences are compared based on the Euclidean distance between the p dimensional primes features generated for each sequence. A given pair of sequences are candidates for pairwise comparison via d2 only if the Euclidean distance is below a given threshold.

A. Choice of primes for PNH

The PNH provides a computationally light approach to extract features from a nucleotide sequence. Each of the p dimensional features embody nucleotide frequencies which is sensitive to biologically meaningful features such as: direct repeats, inverted repeats, and skew patterns [26] but is insensitive to inversions. This is the intuition behind the usage of PNH.

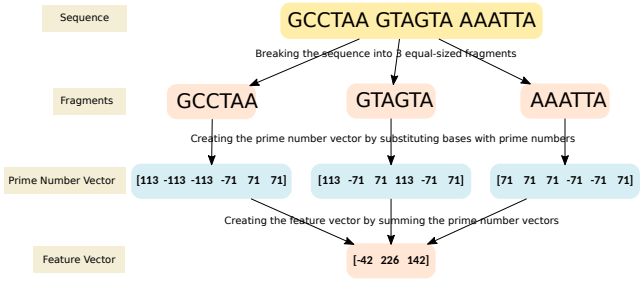


Fig. 5. Example of feature extraction in the PNH

The choice of prime numbers to encode nucleotide is a balance. Small prime numbers (such as 3, 17, 23, etc.) were avoided in an attempt to increase the uniqueness of the feature values [29]. Conversely, very large values are not used to ensure that a few differences do not skew the features. The default values of P_A = 113 and P_G = 71 were identified via preliminary analysis on HIV sequences [6] and were also used as guidance for the prime number choices, but was not highly optimized for any particular organisms' genome. Moreover, smaller values permit the use of faster, primitive datatypes without concerns about numeric overflows (within reason) during summations (see Figure 5).

B. Impact of hyperparameters

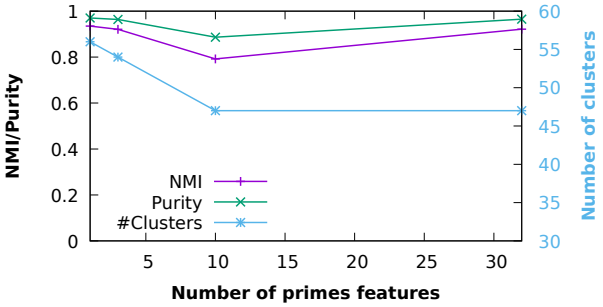
The Prime Numbers-based Heuristic (PNH) involves two key hyperparameters, namely – ❶ the number of features p used for comparing two reads, and ❷ τ_{prime}: the threshold Euclidean distance used to decide if two reads are sufficiently similar. Currently, the value of τ_{prime} for a given reference sequence is computed as the average Euclidean distance from a given reference strain minus the standard deviation.

The number of p features has been set to a default value of 4 based on experimental results summarized in Figure 6. Overall, the number of features p does not have a significant impact on the clustering quality – *i.e.,* NMI and purity do not vary much. However, as the number of features p is increased the number of clusters increase due to too many false-negatives as the reads are broken into too short fragments. Moreover, the memory usage also increases due to the overhead of storing the large features. Consequently, in this study, we have set the value of p to be 4.

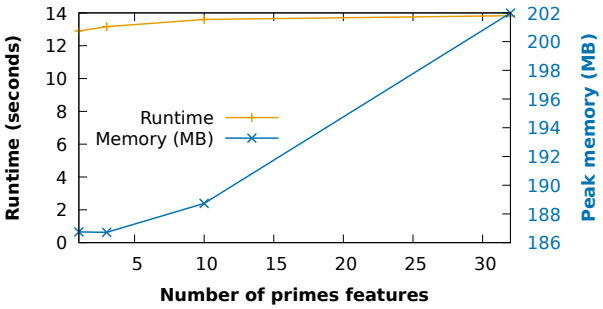
VII. Experiments & Discussions

The effectiveness of the proposed Approximate Spanning Tree (AST) and the Prime Number based Heuristic (PNH) has been empirically assessed using two different types of datasets. The first dataset was Expressed Sequence Tags (ESTs) from different species as summarized in Table I. This dataset consists of characteristic short reads (about 300 – 500 nucleotides) that have been used by several investigators in the past.

`S8` is a synthetic data set from James *et al* [11]. The `C08` is a synthetically generated test dataset generated using ESTsim (see [9] for details). The `A076941` dataset contains a subset



(a) Impact on quality



(b) Impact on runtime

Fig. 6. Impact of varying number of PNH features p

TABLE I
SUMMARY OF EST DATASET USED IN EXPERIMENTS

Name	#Reads	Avg Length (mean± SD)	Size (MB)	Description
S8	150,295	300.9±58	48	Synthetic dataset from [11]
C08	100,585	468.2±74	48	Synthetic ESTs from [9]
A076941	76,941	427.2±128	42	*Arabidopsis* ESTs from [9]

of *Arabidopsis Thaliana* ESTs downloaded from Genbank. The `C08` and `A076941` datasets were primary used because they have been used by Hazelhurst *et al* [9] for assessment of `wcd-kaboom`.

The second dataset was viral genome fragments summarized in Table II. This datasets consists of *much* longer reads (1000s of nucleotides) when compared to the EST dataset. The `HA` dataset consists of Haemagglutinin (HA) segments from multiple Influenza serotypes (H1 – H19) and should ideally result in 20 clusters. This dataset has been downloaded from Influenza Research Database [2]. The `Dengue` dataset consists of the full genome of all 4 serotypes of dengue. These long genomes are analogous to the long reads (~10,000 bases) produced by the RSII platform from Pacific bioscience. It has been downloaded from Virus Variation Resource [3], specifically from the Virus Pathogen Database [16]. Ideally, this dataset should generate 4 clusters but due to high homology between the serotypes, most clustering software yield just 1 cluster.

TABLE II
SUMMARY OF VIRUS DATASET USED IN EXPERIMENTS

Name	#Reads	Avg Length (mean± SD)	Size (MB)	Description
HA	65,069	1698.8±7	117	Influenza Haemmaglutinin (HA)
Dengue	5,286	10,588±181	55	Degune strains (4 serotypes)

A. Experimental Platform

The experiments reported in this paper have been conducted on a contemporary compute cluster running PBS. Each compute node on the cluster consists of two (dual socket) Intel Xeon® CPUs (Gold 6126 @ 2.60 GHz) with hyperthreading disabled. Each CPU has 14 cores and 19 MiB of shared L3 cache. The cluster runs CentOS 7.5 (Linux kernel version 3.10. All of the software compiled using GNU Compiler Collection (GCC) version 6.3.0 at `-O2` optimization level. The timings and memory usage as been recorded using `/usr/bin/time` utility.

B. Design of Experiments

Prior to conducting experiments, the reads in each of the datasets were randomly shuffled to eliminate any potential patterns that may be inherently present when the datasets were downloaded. Each tool configuration was run as an independent PBS job with 1 core and 12 GB of memory reserved for it. The experimental observations for the EST data and Virus data are shown in Table III and Table IV. In these two tables, the row `wcd-kbm` corresponds to `wcd` with `kaboom` filter configuration. The run time data for `wcd-kaboom` includes the time taken to generate suffix arrays as discussed by Hazelhurst *et al* [9]. The row with name `MST` shows results for base case PEACE run using its default MST. The rows labeled `AST=1` and `AST=0.8` show results for AST with AST-thresholds set to 1.0 and 0.8 respectively. The rows with `PNH` prefix show results from using the heuristic along with the default u/v and t/v heuristics in PEACE.

C. Discussions

The results from various experiments conducted using the EST datasets from Table I are summarized in Table III. Similarly, the results for the virus datasets from Table II are summarized in Table IV. The NMI and purity values have been computed using the clustering generated by PEACE-MST as the reference as discussed in Section III.

For the EST datasets, `wcd-kaboom` consistently outperformed PEACE-MST as reported by Hazelhurst *et al* [9]. However, as shown by the results, the proposed AST approach was considerably faster than PEACE-MST without much compromise in the quality in most of the cases. However, `wcd-kaboom` outperformed the AST approach for two out of the three datasets. In these datasets, the performance gains of the AST approach was muted due to the large number of small clusters – *i.e.,* only a few reads were similar and

TABLE III
RESULTS FOR EST DATASETS

Tool info		Time (mm:ss)	RAM (MB)	#Clusters	NMI	Purity
Dataset: S8 (~150K seq)						
wcd-kbm		1:01	787	2224	0.998	1
PEACE	MST	33:05	143	2000	1	1
	AST=1.0	0:46	163	2002	0.999	1.0
	AST=0.8	01:00	163	2002	0.999	1.0
	PNH,AST=1.0	01:14	178	2002	0.999	1.0
	PNH,AST=0.8	01:08	179	2001	0.999	1.0
Dataset: C08 (~100K seq)						
wcd-kbm		2:00	811	5746	0.847	0.999
PEACE	MST	18:10	12587	4552	1.0	1.0
	AST=1.0	2:31	2724	7350	0.837	0.999
	AST=0.8	3:31	5346	6952	0.886	0.999
	PNH,AST=1.0	1:14	2723	39983	0.69	0.999
	PNH,AST=0.8	8:41	5356	39601	0.685	0.999
Dataset: A076941 (~76.9K seq)						
wcd-kbm		0:45	568	18817	0.965	0.998
PEACE	MST	9:57	10576	17422	1.0	1.0
	AST=1.0	1:54	2711	18903	0.949	0.999
	AST=0.8	3:08	2712	18759	0.953	0.999
	PNH,AST=1.0	05:16	2719	39222	0.854	1.0
	PNH,AST=0.8	05:58	2718	39025	0.858	1.0

hence reads could not be rapidly added to the AST (see discussion in Section V). In other words, Kaboom filter was more successful at eliminating redundant calls to heuristics and the heavyweight d2-score generation when compared to AST. In the case of the `A076941` dataset, a dominant portion of the runtime was consumed for running the u/v and t/v heuristics. Analysis of the runtime behaviors suggests that a threshold based on number of successful reads would be beneficial here – *i.e.,* once some s reads below the AST-threshold have been found, stop checking further reads. We plan to pursue this optimization in the near future to assess its efficacy. Some degradation was observed in the `C08` case, with the AST approach generating more clusters. This results in some degradation of NMI with respect to PEACE-MST, but the NMI of 0.837 is comparable to `wcd`'s 0.847.

The most conspicuous performance improvement was observed in the HA dataset, where the AST method finished clustering in ~16 seconds, while `wcd-kaboom` and PEACE-MST took 149 mins and 930 mins respectively. This corresponds to a remarkable $>$ 550× and $>$ 3400× performance improvements! In the case of the Dengue dataset, the performance improvement of AST over `wcd-kaboom` was ~90×. The results suggest that the AST method will yield good performance improvements for datasets with large clusters.

An unforeseen benefit: Our experiments revealed an interesting unforeseen benefit of the AST method for the HA dataset. The dataset has 19 different serotypes and a few unclassified reads. However, the serotypes are highly homologous and hence, the more accurate MST-based approach results in just 8 clusters. However, with the AST approach results in detection of these serotypes as shown in Table IV.

TABLE IV
RESTULS FOR VIRUS DATASETS

Tool info		Time (mm:ss)	RAM (MB)	#Clusters	NMI	Purity
Dataset: HA (~65K seq)						
wcd-kbm		149:55	1820	28	0.566	1.0
PEACE	MST	930:06	4198	8	1.0	1.0
	AST=1	0:16	178	26	0.63	1.0
	AST=0.8	0:27	180	13	0.743	1.0
	PNH,AST=1	0:13	185	53	0.583	1.0
	PNH,AST=0.8	0:17.5	185	45	0.591	1.0
Dataset: Dengue (~5.2K seq)						
wcd-kbm		31:45	945	1	1.0	1.0
PEACE	MST	858:17	407	1	1.0	1.0
	AST=1	0:21	78	2	0	1.0
	AST=0.8	0:27	180	13	0.743	1.0
	PNH,AST=1	0:11	79	44	0	1.0
	PNH,AST=0.8	0:12	79	45	0	1.0

For example, with a threshold of 0.95, the AST method generates 20 clusters separating out the serotypes which could be a desirable result. This behavior will require further investigation.

The Prime Numbers based Heuristic (PNH) was able to further boost runtime performance in the `HA` dataset, but at the cost of reduction in NMI. For this dataset prime heuristics was able to provide another 18% improvement (see Table IV. It also increased performance in the `Dengue` dataset by another 50% with NMI comparable to AST. For the datasets having relatively shorter sequences and a large number of clusters (*i.e.,* `S8`, `C08`, and `A076941`) the PNH was slower. This is attributed to the additional computational overhead of the PNH and this overhead is not effectively amortized. Overall, prime heuristics provided clustering with lower NMIs, except in the case of `S8` dataset. These results suggested that the PNH is effective in clustering viral genomic data possessing highly similar, large nucleotide sequences. This type of clustering has a several biological applications including analysis of microbial genome variations, Influenza A subtype identification, and identifying novel viral strains.

D. Memory consumption

A drawback inherited from PEACE is the increased memory usage for clustering. For example, in the case of the `C08` dataset, the default MST-based clustering in PEACE consumes over 12 GB of RAM while the AST approach consumes about 5 GB. In contrast, `wcd-kaboom` consumes only 26 MB (even when the raw dataset size if 48 MB). The low memory footprint of `wcd-kaboom` is attributed to the fact that it only maintains the suffix array in memory while processing 1 read at a time.

The increased memory footprint of PEACE arises from 2 factors. First, PEACE experiments were conducted by holding all the reads in memory. It does have an option to load reads on-demand, but this option has not been used to minimize runtime. The largest fraction of memory usage arises from the caches maintained by PEACE. The cache has been implemented

using a standard binary heap. It serves as a priority queue to identify the next read to be added to the MST/AST. This cache is pruned, but not aggressively, to reduce runtime. Performing a periodic, deep pruning of this cache will reduce memory footprint, but at the cost of some increase in runtime. In addition, alternative data structures such as 3-tier heap [10] can be utilized to enable efficient pruning. We are planning to explore such enhancements to PEACE in the near future.

VIII. CONCLUSIONS

Clustering is a vital processing step for analysis of genomic data. Several software tools have been proposed to enable fast clustering. Nevertheless, continued advancement in clustering methods is necessary to keep pace with the ongoing exponential growth of genomic data. This study proposed two novel enhancements to an existing clustering software system called PEACE. The first enhancement was the use of Approximate Spanning Tree (AST) which enables *much faster* clustering than the current Minimum Spanning Tree (MST) approach. In addition, a novel Prime Numbers based Heuristic (PNH) is also proposed. The paper discussed these two enhancements in detail and presented an empirical analysis of their effectiveness. The empirical data was collected from experiments conducted using two different types of data sets. Moreover, the paper also presented comparison against `wcd-kaboom`, a *fast*, state-of-the-art clustering software.

The outcomes of this study show that the AST approach effectively increase the performance of PEACE only for datasets with large clusters (viral genomic sequences). In the case of Influenza data set, a dramatic 550× performance improvement was observed. In general, the AST approach did not significantly compromise the quality of the clustering. In most cases, the AST method generates additional clusters, but did not impact purity. These additional clusters can be merged, if necessary.

The outcomes also indicate that the Prime Number based Heuristic (PNH) is a promising approach for further increasing the speed, but only for clustering long genomic sequences. A detailed investigation of the effects of hyperparameter choices for PNH is underway to increase its broader applicability. Furthermore, we are planning enhancements to reduce memory footprint by using multi-tier heaps [10]. Importantly, we plan to explore parallel clustering capabilities of PEACE in conjunction with AST and the PNH.

Rapid clustering of viral and bacterial genomes forming large clusters will provide insight into the genetic diversity. Viral genotypes such as Influenza A subtypes can be identified by clustering unidentified genomic sequences with known genotypes. This type of cluster-based identification approaches has the potential to detect novel viral strains by separating them into new clusters which does not consist any known types. The improved performance of PEACE with AST, with its default parameter settings and without the need for additional external tools (such as `mkesa` for `wcd`), makes it an ideal tool for use by biologists for clustering large datasets.

ACKNOWLEDGMENT

We thank Dr. Jens Muller for his help to run experiments on the Red Hawk compute cluster. We thank Dr. Scott Hazelhurst for his help with using `wcd` for our experiments.

AUTHOR CONTRIBUTIONS

D. M. Rao: A first coauthor, implemented AST and PNH in PEACE, involved in running the experiments, analyzing the data, and writing the paper.

S. Sreeskandarajan: A first coauthor, designed PNH, involved in running experiments, analyzing the data, and writing the paper.

C. Liang: Helped with implementation, testing, and analysis of MST approach in PEACE.

REFERENCES

[1] S. Baric, W. Salzburger, and C. Sturmbauer. Phylogeography and evolution of the tanganyikan cichlid genus tropheus based upon mitochondrial dna sequences. *Journal of Molecular Evolution*, 56(1):54–68, Jan 2003.

[2] P. Bogner, I. Capua, D. J. Lipman, N. J. Cox, et al. A global initiative on sharing avian flu data. *Nature*, 442(5):981, Aug. 2006.

[3] J. R. Brister, Y. Bao, S. A. Zhdanov, Y. Ostapchuck, V. Chetvernin, B. Kiryutin, L. Zaslavsky, M. Kimelman, and T. A. Tatusova. Virus variation resource-recent updates and future directions. *Nucleic Acids Research*, 42(D1):D660–D665, 2014.

[4] P. Chalise and B. L. Fridley. Integrative clustering of multi-level 'omic data based on non-negative matrix factorization algorithm. *PLOS ONE*, 12(5):1–18, 05 2017.

[5] G. B. Cybis, J. S. Sinsheimer, T. Bedford, A. Rambaut, P. Lemeyh, and M. A. Suchard. Bayesian nonparametric clustering inphylogenetics: modeling antigenicevolution in influenza. *Statistics in Medicine*, 37(2):195–206, 2018.

[6] S. Delgado, F. Morán, A. Mora, J. J. Merelo, and C. Briones. A novel representation of genomic sequences for taxonomic clustering and visualization by means of self-organizing maps. *Bioinformatics*, 31(5):736–744, 2015.

[7] X. Guan and L. Du. Domain identification by clustering sequence alignments. *Bioinformatics*, 14(9):783–788, 10 1998.

[8] S. Hazelhurst, W. Hide, Z. Lipták, R. Nogueira, and R. Starfield. An overview of the wcd est clustering tool. *Bioinformatics*, 24(13):1542–1546, 2008.

[9] S. Hazelhurst and Z. Lipták. Kaboom! a new suffix array based algorithm for clustering expression data. *Bioinformatics*, 27(24):3348–3355, 2011.

[10] J. Higiro, M. Gebre, and D. M. Rao. Multi-tier priority queues and 2-tier ladder queue for managing pending events in sequential and optimistic parallel simulations. In *Proceedings of the 2017 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIMPADS '17, pages 3–14, New York, NY, USA, 2017. ACM.

[11] B. T. James, B. B. Luczak, and H. Z. Girgis. Meshclust: an intelligent tool for clustering dna sequences. *Nucleic Acids Research*, 46(14):e83, Aug. 2018.

[12] R. Jothi, S. K. Mohanty, and A. Ojha. Fast approximate minimum spanning tree based clustering algorithm. *Neurocomputing*, 272:542 – 557, 2018.

[13] J. J. Kozich, S. L. Westcott, N. T. Baxter, S. K. Highlander, and P. D. Schloss. Development of a dual-index sequencing strategy and curation pipeline for analyzing amplicon sequence data on the miseq illumina sequencing platform. *Applied and Environmental Microbiology*, 79(17):5112–5120, 2013.

[14] W. Li and A. Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, May 2006.

[15] D. A. Morrison. Is sequence alignment an art or a science? *Systematic Botany*, 40:14–26, Feb. 2015.

[16] B. E. Pickett, D. S. Greer, Y. Zhang, L. Stewart, L. Zhou, G. Sun, Z. Gu, S. Kumar, S. Zaremba, C. N. Larsen, W. Jen, E. B. Klem, and R. H. Scheuermann. Virus pathogen database and analysis resource (vipr): A comprehensive bioinformatics database and analysis resource for the coronavirus research community. *Viruses*, 4(11):3209–3226, 2012.

[17] D. M. Rao. Poster: A parallel framework for ab initio transcript-clustering. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*, pages 331–332, May 2018.

[18] D. M. Rao, J. C. Moler, M. Ozden, Y. Zhang, C. Liang, and J. E. Karro. Peace: Parallel environment for assembly and clustering of gene expression. *Nucleic Acids Research*, 38(suppl 2):W737–W742, 2010.

[19] A. Röhl, H. J. Bandelt, and P. Forster. Median-joining networks for inferring intraspecific phylogenies. *Molecular Biology and Evolution*, 16(1):37–48, 01 1999.

[20] K. Sedlar, K. Kupkova, and I. Provaznik. Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics. *Computational and Structural Biotechnology*, 15(1):48–55, Jan. 2017.

[21] F. Sievers and D. G. Higgins. *Clustal Omega, Accurate Alignment of Very Large Numbers of Sequences*, pages 105–116. Humana Press, Totowa, NJ, 2014.

[22] A. Solovyov, G. Palacios, T. Briese, W. I. Lipkin, and R. Rabadan. Cluster analysis of the origins of the new influenza a (h1n1) virus. *EuroSurveillance*, 14(21), May 2009.

[23] S. Sreeskandarajan, M. M. Flowers, J. E. Karro, and C. Liang. A matlab-based tool for accurate detection of perfect overlapping and nested inverted repeats in dna sequences. *Bioinformatics*, 30(6):887–888, 2014.

[24] A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, Mar. 2003.

[25] J. D. Thompson, T. J. Gibson, and D. G. Higgins. Multiple sequence alignment using clustalw and clustalx. *Current Protocols in Bioinformatics*, Chapter 2(1):2.3.1–2.3.22, 2003.

[26] F. van Hemert, M. Jebbink, A. van der Ark, F. Scholer, and B. Berkhout. Euclidean distance analysis enables nucleotide skew analysis in viral genomes. *Computational and Mathematical Methods in Medicine*, 2018, Oct. 2018.

[27] S. Vinga. Editorial: Alignment-free methods in computational biology. *Briefings in Bioinformatics*, 15(3):341–342, 2014.

[28] D. Wei, Q. Jiang, Y. Wei, and S. Wang. A novel hierarchical clustering algorithm for gene sequences. *BMC Bioinformatics*, 13(1):174, Jul 2012.

[29] C. Ye, G. Ji, L. Li, and C. Liang. detectir: A novel program for detecting perfect and imperfect inverted repeats using complex numbers and vector calculation. *PLOS ONE*, 9(11):1–11, 11 2014.

[30] C. Zhong, M. Malinen, D. Miao, and P. Fränti. Fast approximate minimum spanning tree algorithm based on k-means. In R. Wilson, E. Hancock, A. Bors, and W. Smith, editors, *Computer Analysis of Images and Patterns*, pages 262–269, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[31] A. Zielezinski, S. Vinga, J. Almeida, and W. M. Karlowski. Alignment-free sequence comparison: benefits, applications, and tools. *Genome Biology*, 18(1):186, Oct 2017.