Better Identification of Repeats in Metagenomic Scaffolding

Jay Ghurye and Mihai Pop(✉)

Department of Computer Science and Center for Bioinformatics and Computational Biology, University of Maryland, College Park, USA
jayg@cs.umd.edu, mpop@umiacs.umd.edu

Abstract. Genomic repeats are the most important challenge in genomic assembly. While for single genomes the effect of repeats is largely addressed by modern long-read sequencing technologies, in metagenomic data intra-genome and, more importantly, inter-genome repeats continue to be a significant impediment to effective genome reconstruction. Detecting repeats in metagenomic samples is complicated by characteristic features of these data, primarily uneven depths of coverage and the presence of genomic polymorphisms. The scaffolder Bambus 2 introduced a new strategy for repeat detection based on the betweenness centrality measure – a concept originally used in social network analysis. The exact computation of the betweenness centrality measure is, however, computationally intensive and impractical in large metagenomic datasets. Here we explore the effectiveness of approximate algorithms for network centrality to accurately detect genomic repeats within metagenomic samples. We show that an approximate measure of centrality achieves much higher computational efficiencies with a minimal loss in the accuracy of detecting repeats in metagenomic data. We also show that the combination of multiple features of the scaffold graph provides a more effective strategy for identifying metagenomic repeats, significantly outperforming all other commonly used approaches.

Keywords: Metagenomics · Random forest · Betweenness centrality · Scaffolding · Algorithms · Graph

1 Introduction

Genomic repeats are the most important challenge in genomic assembly even for isolate genomes. When reads are shorter than the repeats (a common situation until the recent development of long read sequencing technologies) it can be shown that the number of genome reconstructions consistent with the read data grows exponentially with the number of repeats [10]. The use of additional information to constrain the one genome reconstruction representing the actual genome being assembled leads to computationally intractable problems. In other words, when reads are shorter than repeats the correct and complete reconstruction of a genome is impossible. In the case of isolate genomes, long read technologies have largely addressed this challenge, at least for bacteria where the

M. Frith and C.N.S. Pedersen (Eds.): WABI 2016, LNBI 9838, pp. 174–184, 2016.
DOI: 10.1007/978-3-319-43681-4_14

majority of genomic repeats fall within the range of achievable read lengths [11]. In metagenomics, however, the problem is compounded by the fact that microbial mixtures often include multiple closely-related genomes differing in just a few locations. The genomic segments shared by closely related organisms – inter-genomic repeats – are substantially larger than intra-genomic repeats and cannot be fully resolved even if long read data were available. Instead, the best hope is to identify and flag these repeats in order to avoid mis-assemblies that incorrectly span across genomes.

To date, most approaches for repeat detection have been based on the basic observation that repetitive segments have unusual coverage depth, fact which is usually ascertained through simple statistical tests. These approaches, however, fail in the context of metagenomic data as well as in other settings (e.g., single cell genomics) that violate the assumption of uniform depth of coverage within the genome, assumption that is critical for the correctness of statistical tests. Furthermore, the challenges posed by repeats to assembly algorithms are not directly related to the depth of sequencing coverage within contigs, rather they result from the fact that repeats "tangle" the assembly graph. More specifically, the correct genomic sequence (whether of a single genome or mixture of genomes) can be represented as one or more linear sub-paths of the graph. Repeats induce links within the graph that are inconsistent with this linear structure, making it difficult for algorithms to reconstruct the true genomic structure. We, therefore

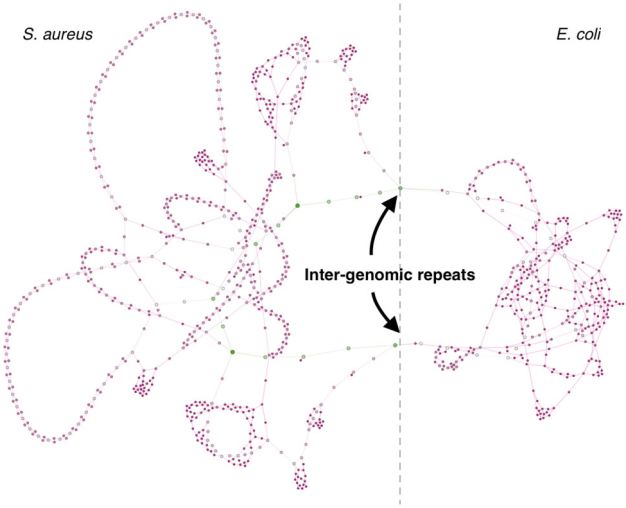

Fig. 1. Assembly graph of a simulated community consisting of 200 Kbp subsets of *Escherichia coli* str. K-12 MG1655 and *Staphylococcus aureus*. Nodes are colored and sized based on their relative betweenness centrality with larger, green nodes indicating a higher centrality. The highlighted nodes are inter-genomic repeats whose deletion would separate the graph. Note that the betweenness centrality measure correctly identifies these nodes.

propose an operational definition of genomic repeats as those nodes in the graph that induce inconsistencies. This definition is orthogonal to depth of coverage considerations - high coverage contigs that do not "tangle" the graph do not impact assembly algorithms, while contigs that confuse the assembly need to be removed whether or not they can be conclusively labeled as "high coverage".

We have previously proposed an operational definition of repeats in terms of betweenness centrality. This approach was implemented in the Bambus 2 [12] scaffolder and is a key component of the MetAMOS metagenomic assembly pipeline [24]. An example of the effectiveness of this approach in a simple community composed of two genomes is shown in Fig. 1. The full implementation of betweenness centrality, however, requires an all-pairs shortest path computation which is computationally too intensive for typical metagenomic datasets. In Bambus 2, for example, repeat finding in a typical stool sample requires days of computation. To overcome this limitation, we demonstrate here that substantial speed-ups can be obtained through the use of approximate betweenness centrality algorithms without sacrificing accuracy. We further extend this operational definition of repeats by integrating a larger set of graph properties to construct an efficient and accurate repeat detection strategy.

2 Related Work

Repeat Detection in Scaffolding

Scaffolding involves using the connectivity information from mate pairs to orient and order pre-assembled contigs obtained from an assembler to reconstruct a genome. This problem of orienting and ordering contigs was shown to be NP-Hard [9]. Various scaffolding methods have been designed based on different heuristics to obtain approximate solutions to the problem. However, all of these methods face difficulties when dealing with contigs originating from repetitive regions in the genome. A common strategy for handling repeats is to identify and remove them from the graph prior to the scaffolding process, then re-introduce them after the contigs have been properly ordered and oriented. Most of the existing scaffolders use depth of coverage information to classify a contig as a repeat. For example, Opera [4] and SOPRA [2] filter out as repetitive contigs with coverage 1.5 and 2.5 times more than average coverage, respectively. The MIP scaffolder [22] uses high coverage (greater than 2.5 times average) as well as high degree (≥ 50) of nodes within scaffold graph to determine repeats. Bambus 2 [12] – a scaffolder specifically designed for metagenomic data – uses a notion of betweenness centrality [1] along with global coverage information to find out repeats.

Betweenness Centrality

In network analysis, metrics of centrality are used to identify the most important nodes within a graph. Several metrics to measure centrality have been proposed,

but in this work, we use betweenness centrality. The betweenness centrality of a particular node is equal to the number of shortest paths from all nodes to all others that pass through that node. Intuitively, a node that is frequently found on paths connecting other nodes is a potential repeat, as along a simple path all nodes should have roughly the same centrality value. The algorithm for computing exact centrality [1] takes $\Theta(mn)$ time on a graph with m nodes and n edges. Several solutions were proposed to overcome this computational cost of computing network centrality, including and exact massively parallel implementation [16], and an approximate solution based on sampling a subset of the nodes [6]. Recently, a better parallel approximation algorithm was proposed by Riondato and Kornaropoulos [21] which uses a strategy for sampling from among the shortest paths in the graph to compute betweenness centrality. The size of chosen sample of paths can provide provable bounds on the accuracy of the centrality value given by the algorithm. The sample size is determined as a function of an approximation factor ϵ and the diameter of the graph.

3 Methods

Construction of Scaffold Graph

A scaffold graph is defined as a graph $G(V, E)$, where V is set of all the contigs. The edges represent links between the contigs inferred from read pairing information – if the opposite ends of a read pair map to different contigs we can infer the possible adjacency of these contigs within the genome. Since most genome assemblers do not report the location of reads within contigs, we infer this information by mapping using bowtie2 [13]. Experimental library size estimates are often incorrect, and we re-estimate here the distance between the paired reads from pairs of reads mapped to a same contig. We record the average insert size l and standard deviation $\sigma(l)$ within a library. For each pair of contigs we retain the maximal set of links that are consistent in terms of the implied distance between the contigs for each implied relative placement of the contigs. Since contigs can be oriented in forward or reverse direction depending on the orientation implied by mapped mate pairs, there exist 4 possible orientations of adjacent contigs (forward-forward, forward-reverse, reverse-forward and reverse-reverse). For each of the possible relative orientation, we need to find a maximal set of consistent links implying that orientation. This set can be identified in $O(nlogn)$ time using an algorithm to find maximal clique in an interval graph [20]. The distance between the contigs implied by the resulting "bundle" of links has mean $l(e) = \frac{\sum \frac{l}{\sigma(l)}}{\sum \frac{1}{\sigma(l)^2}}$ and standard deviation $\sigma(l) = \frac{1}{\frac{1}{\sigma(l)^2}}$, as suggested by Huson et al. [7].

Orienting the Bidirected Scaffold Graph

The scaffold graph derived from the process outlined above is birected [17]. It can be converted into a directed graph by assigning an orientation to each node,

reflecting the strand of the DNA molecule that is represented by the corresponding contig. In computational terms, we need to embed a bipartite graph (the two sets corresponding to the two strands of DNA being reconstructed) within the scaffold graph. In the general case, such an embedding is not possible without removing edges in order to break all odd-length cycles in the original graph. Finding such a minimum set of edges is NP-Hard [5]. We use here a greedy heuristic proposed by Kececioglu and Myers [9] which achieves a 2−approximation and runs in $O(V + E)$ time.

Repeat Detection Through Betweenness Centrality

We start by calculating centrality values for all the nodes in the graph using either an exact or approximate centrality algorithm as outlined in the introduction. Let μ be the mean and σ be the standard deviation of the resulting centrality values. A contig is marked as repeat if its centrality value is greater than $\mu + 3 * \sigma$. This cutoff criterion is the same as the one used in Bambus 2. We have also experimented with other definitions of outliers (such as interquartile range), however the original definition used in Bambus 2 performed better than the interquartile range cutoff (data not shown).

Repeat Detection with an Expanded Feature Set

Centrality is just one of the possible signatures that a node in the graph "tangles" the graph structure, making it harder to identify a correct genomic reconstruction. At a high level, one can view centrality to relate to difficulties in ordering genomic contigs along a chromosome. The orientation procedure outlined above provides potential insights into contigs that may prevent the correct orientation of contigs – contigs adjacent to a large number of edges invalidated by the orientation procedure are possible repeats. Other potential signatures we consider include the degree of graph nodes (highly connected nodes are potential repeats) as well as abrupt changes in coverage between adjacent nodes. The latter information is defined as follows. For each contig we capture the distribution of read coverage values. We then use a Kolmogorov-Smirnov test [15] to identify pairs of contigs that have statistically different distributions of coverage values. We flag all edges that exceed a pre-defined p-value cutoff (in the results presented here we simply use 0.05). We combine these different measures (contig length, centrality, node degree, fraction of number of edges invalidated by the orientation routine that are adjacent to a node, fraction of number of edges with abrupt changes in coverage, and ratio of contig coverage to average coverage) within a Random Forest classifier [14].

To generate training information for the classifier we aligned the contigs to an appropriate set of reference genomes using MUMmer [3] dependent on the data being assembled, and flagged as repetitive all contigs that had more than one match with greater than 95 % identity over 90 % of the length within the reference collection.

4 Results

Dataset and Assembly

To test our methods, we used a synthetic metagenomic community dataset (S1) by Shakya et al. [23] that was derived from a mixture of cells from 83 organisms with known genomes. Reads in the datasets were cleaned and trimmed using Sickle [8]. Assembly was performed using IDBA-UD [19] with default parameters. The assembly of S1 yielded 47,767 contigs.

Extended Feature Set Improves Repeat Detection

We trained a Random Forest classifier that takes into account the various measures outlined above as follows. We simulated a low coverage (10x) dataset using a read simulator provided with the IDBA assembler from the set of 40 genomes downloaded from NCBI[1]. We constructed contigs from the simulated reads and mapped them to reference sequences to identify which contigs are repetitive (have ambiguous placement in the reference set). We used this information to train the classifier, then used the resulting classifier to predict repeats within the synthetic community S1 described above. As can be seen in Fig. 2 the accuracy of the classifier based on multiple graph properties is higher than that of approaches that rely on just coverage as a criterion to classify a contig as a repeat. Classification of repeats using approximate centrality provides higher specificity compared to the coverage approach at the cost of slightly lower sensitivity. The Random Forest approach leverages the advantage of high sensitivity from the coverage approach and high specificity from the centrality approach along with some additional features to provide better overall classification.

Important Parameters in Determining Repeats

We further explored the features of the data that contribute to the better performance of the classifier. In Fig. 3 we show the contribution of each feature to the classifier. The length of contigs, factor not usually taken into account when detecting repeats, appears to have the largest influence. This is perhaps unsurprising as repeats confuse the assembly process as well, fragmenting the assembly. In other words, longer contigs are less likely to represent repetitive sequences. The second most important features is the fraction of edges adjacent to a contig that indicate an abrupt change in coverage. Contigs with unusual coverage in comparison to their neighbors can also be reasonably assumed to be repetitive. Centrality was the third most important factor, as expected. Perhaps surprising, overall depth of coverage or node degree are not as important as features despite these measures being among the most widely used signatures of "repetitiveness" by existing tools.

[1] ftp://ftp.ncbi.nlm.nih.gov/genomes/bacteria/all.fna.tar.gz.

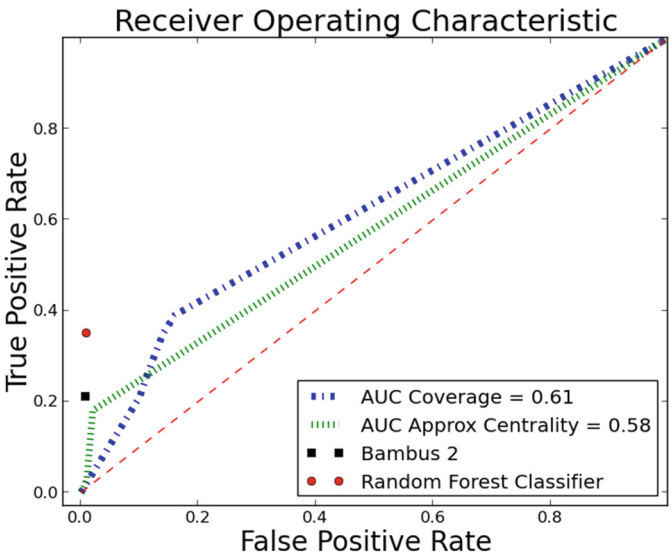

Fig. 2. Plot for comparison of Random Forest classifier with the coverage and centrality approach. The red circle in the plot indicates the sensitivity and specificity obtained by using the Random Forest approach. The black square in the plot indicates the sensitivity and specificity obtained by using Bambus 2.

Comparison of Incorrectly Oriented Pair of Contigs

Beyond testing the simple classification power of different approaches, we also evaluated the different methods in terms of whether the removal of nodes marked as repeats makes the scaffolding process more accurate. Specifically, we explored how different repeat removal strategies affect the contig orientation process. The scaffold graph for the S1 dataset had 21,950 nodes and 31,059 edges. We removed the repeats reported by the different methods from this graph and oriented the resulting graph. We then tracked the accuracy of the results in terms of the number of edges that imply a different relative orientation of the adjacent nodes than the correct one, inferred by mapping the contigs to the reference genomes. Here the relative orientation can either be same if both the contigs on the edge have same orientation (forward-forward and reverse-reverse) and different if the contigs on the edge different orientations (forward-reverse and reverse-forward). The results are shown in Table 1. The centrality based methods and the Random Forest classifier based methods resulted in lower error rates and retained a higher percentage of the edges in the original graph than coverage based methods.

Comparison of Runtime with Bambus 2

The results above show that Bambus 2 has, unsurprisingly, a similar level of accuracy with the approximate centrality approach. We have already mentioned,

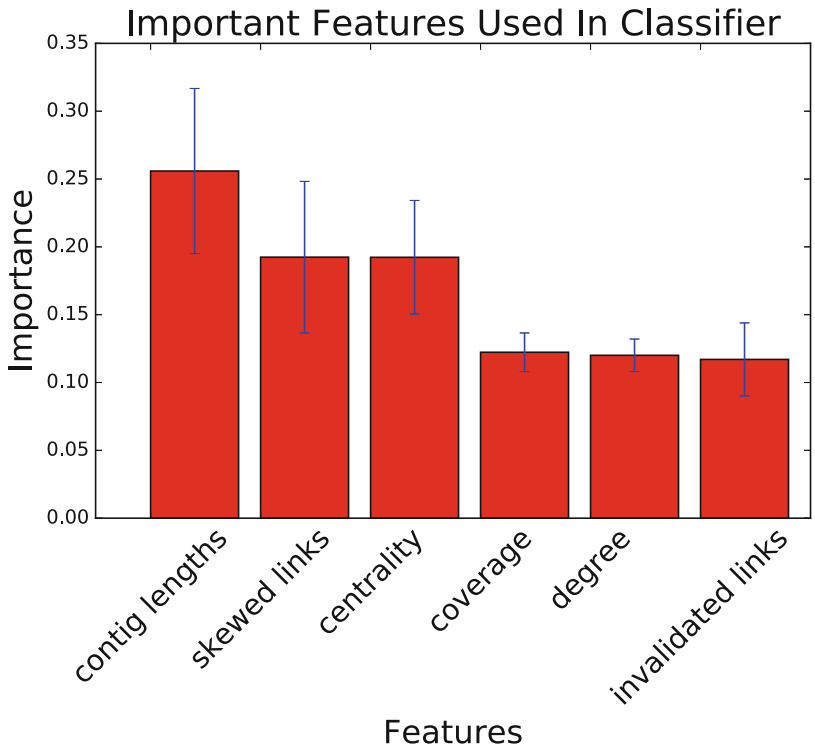

Fig. 3. Importance of features used in Building Random Forest classifier

Table 1. Number of correctly and incorrectly oriented links in scaffold graph using various repeat removal strategies. The % correct column represents the percentage of correctly oriented links as a function of the total number of edges in the original scaffold graph. % wrong column represents the percentage of incorrectly oriented links in the graph obtained by removing repeats.

Method	Correct	Wrong	% correct	% wrong
Bambus 2	12042	867	38.77 %	4.11 %
Approximate betweenness centrality	12336	917	39.71 %	3.94 %
Coverage (MIP, SOPRA)	3840	315	17.49 %	4.72 %
Coverage (Opera)	2007	165	6.46 %	5.62 %
Random forest	12255	807	39.45 %	3.52 %

however, that Bambus 2 is inefficient on large datasets. To explore the efficiency of the approximate centrality approach, we used a real metagenomic dataset (SRX024329 from NCBI) representing sequencing data from the tongue dorsum of a female patient. Assembly of these reads was performed using IDBA yielding 106,380 contigs in total. The scaffold graph constructed from these contigs had 112,502 edges. The 'MarkRepeats' module of Bambus 2 took almost 2 h to detect repeats, whereas the approximate betweenness centrality algorithm found repeats in approximately 5 min, a substantial improvement in speed without a loss of accuracy as shown above. To compare the runtime with training of Random Forest classifier, we trained the classifier on contigs in this dataset. Since we did not have reference sequences for this dataset, we randomly marked a subset of contigs as repeats and performed training. It took about 20 min to calculate features and fit a classifier which was still faster than time taken by Bambus 2.

5 Discussion and Conclusion

Our prior work had introduced the use of network centrality as an approach for detecting repeats in metagenomic assembly, a setting where coverage-based approaches are often ineffective. This approach, implemented in the scaffolder Bambus 2, was, however, inefficient for large datasets, fact that has limited its use. Here we extend our original approach by incorporating multiple features of the scaffold graph (including centrality) that may be signatures of repetitive sequences within a Random Forest classifier. We also show that an approximate calculation of network centrality based on the random sampling of paths obtains similar accuracy as the full centrality computation at a fraction of computational time.

Our results demonstrate that methods that directly capture the effect of repeats on the assembly graph are more effective at detecting repeats than indirect measures such as depth of coverage, particularly in the context of metagenomic assembly. Our new approach improves in both accuracy and efficiency over existing methods for repeat detection, and we plan to incorporate it within the MetAMOS metagenomic assembly pipeline as a replacement for the exisiting code within Bambus 2. We note that the classification accuracy was surprisingly high despite the fact that the classifier was trained on purely simulated data yet applied to real dataset. This underscores the robustness of the feature set we have identified. At the same time the graph features that we have identified as useful in detecting repeats are just a first step towards a better understanding of the features of the data that most influence the ability of assembly algorithms to accurately reconstruct metagenomic sequences. Also classifiers like Random Forest can be implemented in parallel [18] which can provide significant runtime speedups for large metagenomic datasets. We plan in future work to further explore both the feature set and the approaches used to build and train the classifier to increase accuracy and ultimately improve the quality of metagenomic reconstructions.

Acknowledgements. We thank Chris Hill for helping us with generating Fig. 1 and experiments. We also thank Todd Treangen for helping us to improve the manuscript and design experiments.

References

1. Brandes, U.: A faster algorithm for betweenness centrality*. J. Math. Sociol. **25**(2), 163–177 (2001)
2. Dayarian, A., Michael, T.P., Sengupta, A.M.: SOPRA: scaffolding algorithm for paired reads via statistical optimization. BMC Bioinform. **11**(1), 1 (2010)
3. Delcher, A.L., Salzberg, S.L., Phillippy, A.M.: Using MUMmer to identify similar regions in large sequence sets. Curr. Protocols Bioinform. 10.3.1–10.3.18 (2003). Chapter 10:Unit 10.3
4. Gao, S., Sung, W.-K., Nagarajan, N.: Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences. J. Comput. Biol. **18**(11), 1681–1691 (2011)
5. Garey, M., Johnson, D.: Computers and Intractability - A Guide to NP-Completeness. W.H. Freeman & Co., New York (1979)
6. Geisberger, R., Sanders, P., Schultes, D.: Better approximation of betweenness centrality. In: ALENEX, pp. 90–100. SIAM (2008)
7. Huson, D.H., Reinert, K., Myers, E.W.: The greedy path-merging algorithm for contig scaffolding. J. ACM (JACM) **49**(5), 603–615 (2002)
8. Fass, J.N., Joshi, N.A.: Sickle: a sliding-window, adaptive, quality-based trimming tool for FastQ files (version 1.33)
9. Kececioglu, J.D., Myers, E.W.: Combinatorial algorithms for DNA sequence assembly. Algorithmica **13**(1–2), 7–51 (1995)
10. Kingsford, C., Schatz, M.C., Pop, M.: Assembly complexity of prokaryotic genomes using short reads. BMC Bioinform. **11**(1), 21 (2010)
11. Koren, S., Phillippy, A.M.: One chromosome, one contig: complete microbial genomes from long-read sequencing and assembly. Curr. Opin. Microbiol. **23**, 110–120 (2015)
12. Koren, S., Treangen, T.J., Pop, M.: Bambus 2: scaffolding metagenomes. Bioinformatics **27**(21), 2964–2971 (2011)
13. Langmead, B., Salzberg, S.L.: Fast gapped-read alignment with Bowtie 2. Nat. Methods **9**(4), 357–359 (2012)
14. Liaw, A., Wiener, M.: Classification and regression by randomforest. R News **2**(3), 18–22 (2002)
15. Lilliefors, H.W.: On the Kolmogorov-Smirnov test for normality with mean and variance unknown. J. Am. Stat. Assoc. **62**(318), 399–402 (1967)
16. Madduri, K., Ediger, D., Jiang, K., Bader, D.A., Chavarria-Miranda, D.: A faster parallel algorithm and efficient multithreaded implementations for evaluating betweenness centrality on massive datasets. In: 2009 IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2009, pp. 1–8. IEEE (2009)
17. Medvedev, P., Georgiou, K., Myers, G., Brudno, M.: Computability of models for sequence assembly. In: Giancarlo, R., Hannenhalli, S. (eds.) WABI 2007. LNCS (LNBI), vol. 4645, pp. 289–301. Springer, Heidelberg (2007)
18. Mitchell, L., Sloan, T.M., Mewissen, M., Ghazal, P., Forster, T., Piotrowski, M., Trew, A.S.: A parallel random forest classifier for R. In: Proceedings of the Second International Workshop on Emerging Computational Methods for the Life Sciences, pp. 1–6. ACM (2011)

19. Peng, Y., Leung, H.C., Yiu, S.-M., Chin, F.Y.: Meta-IDBA: a de novo assembler for metagenomic data. Bioinformatics **27**(13), i94–i101 (2011)
20. Pop, M., Kosack, D.S., Salzberg, S.L.: Hierarchical scaffolding with bambus. Genome Res. **14**(1), 149–159 (2004)
21. Riondato, M., Kornaropoulos, E.M.: Fast approximation of betweenness centrality through sampling. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, pp. 413–422. ACM (2014)
22. Salmela, L., Mäkinen, V., Välimäki, N., Ylinen, J., Ukkonen, E.: Fast scaffolding with small independent mixed integer programs. Bioinformatics **27**(23), 3259–3265 (2011)
23. Shakya, M., Quince, C., Campbell, J.H., Yang, Z.K., Schadt, C.W., Podar, M.: Comparative metagenomic and RRNA microbial diversity characterization using archaeal and bacterial synthetic communities. Environ. Microbiol. **15**(6), 1882–1899 (2013)
24. Treangen, T.J., Koren, S., Sommer, D.D., Liu, B., Astrovskaya, I., Ondov, B., Darling, A.E., Phillippy, A.M., Pop, M.: MetAMOS: a modular and open source metagenomic assembly and analysis pipeline. Genome Biol. **14**(1), R2 (2013)