Recycler: an algorithm for detecting plasmids from de novo assembly graphs

Roye Rozov[1], Aya Kav Brown[4], David Bogumil[4], Eran Halperin[123*], Itzhak Mizrahi[4], Ron Shamir[1]

1 Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel
2 Molecular Microbiology and Biotechnology Department, Tel-Aviv University, Tel Aviv, Israel
3 International Computer Science Institute, Berkeley, CA, USA
4 The Department of Life Sciences & the National Institute for Biotechnology in the Negev, Ben-Gurion University of the Negev, Beer-Sheva, Israel
* corresponding author, heran@icsi.berkeley.edu

Abstract

Plasmids have garnered a great deal of interest in recent years. They are known to have important roles in antibiotic resistance and in affecting production of metabolites used in industrial and agricultural applications. However, their extraction through deep sequencing remains challenging, in spite of rapid drops in costs for data acquisition and rapid increases in sequencer output. Here, we attempt to ameliorate this situation by introducing a new plasmid-specific assembly algorithm, leveraging assembly graphs and contigs provided by a conventional de novo assembler. We introduce the first plasmid-specific short read assembly tool, called Recycler, and demonstrate its merits in comparison with extant approaches. We show on simulated and real data that Recycler greatly increases the number of true plasmids recovered while remaining highly accurate. On simulated plasmidomes, Recycler showed approximately 40% increase in the proportion of true plasmids recovered over naive assembly in several cases. We validate these results on real data, by comparison against available reference sequences and quantifying annotation of predicted ORFs. Recycler recovered 4 out of 5 known plasmids in assembly of an E. Coli strain, and generated plasmids in high agreement with known plasmid annotation on real plasmidome data. Moreover, 6 out of 8 plasmids previously validated by PCR were completely recovered. Recycler is available at http://github.com/rozovr/Recycler

Introduction

Plasmids are extrachromosomal DNA segments carried by bacterial hosts. They are usually shorter than host chromosomes, circular, and encode nonessential genes. These genes are responsible for either plasmid-specific roles such as self-replication and transfer, or context-specific roles that can be beneficial or harmful to the host depending on its environment. Along with viruses and transposable elements, plasmids are members of the group termed mobile genetic elements [7] as they transmit genes and their selectable functions between cells via horizontal gene transfer. These features distinguish plasmids as a fundamental force in microbial evolution, as they contribute to genome innovation and plasticity.

Much interest has recently arisen for plasmid extraction and characterization, in particular because of their known roles in antibiotic resistance and in increasing metabolic outputs of agricultural or industrial byproducts. For instance, antibacterial resistance genes encoded on plasmids have long been known as a major issue for human health in clinical practice [21], but are also one of today's standard tools in microbiology and genetics when used to select for specific cells [2]. Recent studies aiming to improve plasmid extraction from sequence are based on three different experimental techniques, all involving deep sequencing of short reads. The first technique uses sequencing of cultured isolates, obtaining a mixture of chromosomal and plasmid DNA occurring together in a single strain. This technique can be especially useful for tracking or identifying transfer of mobile elements [9, 6, 16]. De novo assembly for the sake of identifying plasmids can also be augmented by long-read sequencing in this case [6, 11] because sufficient amounts of DNA are available for the strain in question, and the transfer or co-occurrence of shared segments is less of a concern.

Metagenomic approaches target whole microbial ecosystems at once and thus do not depend on culturability of particular strains [8]. This technique allows a much broader view of all taxa present and their plasmids, but unfortunately is limited in that the characterization of each individual strain depends on its coverage in the mixed DNA sample, the frequency of co-occurring repeats shared among different members of the sample, and even the presence of repeats shared between plasmids and their host genomes. All of these confound de novo assembly of metagenomes in general, and as a result, plasmid extraction using metagenome sequencing remains a significant computational challenge. Very deep sequencing [10], new long read technologies [27], and differential co-abundance and binning techniques [12, 22, 5] have all been applied to metagenome assembly for the sake of identifying and characterizing species from highly diverse communities. However assembly of metagenomes remains a highly active area of research: current assembly outputs are lacking and do not represent the true genetic capacity and synteny of genomes present in complex microbial communities.

Most recently, a third technique has emerged that allows recovery of far greater numbers of plasmids. Plasmidome sequencing [3,4,14] allows nearly all sequencing resources to be devoted to circular DNA. Using a protocol described in [3], chromosomal DNA is filtered out and

circular DNA segments are selectively amplified. Based on this protocol, hundreds of new plasmids were identified in the cow rumen [4] and rat cecum [14]. In [14], Jørgensen et al. applied the protocol introduced in [2] combined with bioinformatic validation of circularity. This post-assembly analysis resulted in a 95% PCR validation rate out of 40 randomly selected assembled contigs. This success raises the prospect of in silico refinement of plasmids beyond the initial assembly. Although Jørgensen et al.'s method was shown to have a high validation rate, its output was highly dependent on the contiguity of the underlying assembler's contigs (in their case IDBA-UD [23]): paths composed of multiple contigs connected at their branch points and contigs shorter than 1000 bp were disregarded. Hence, Jørgensen's method was inherently limited to having a high false negative rate. To date, no tools for plasmid assembly from short reads have been introduced to address these limitations.

Here, we wish to use more information in order to improve de novo assembly of sequenced plasmids. Our input is an assembly graph G = (V,E), where the set of nodes V are contigs having associated lengths and coverage levels, and the set of arcs E is composed of directed connections among the nodes. Arcs are the result of branch points in the underlying de Bruijn graph: contig ends are often joined to two (or more) different target contigs based on overlaps, and in many cases the assembler does not have a definite way of choosing which extension is true in order to simplify the branch into a linear path. We aim to cover the graph with a set of cycles and set a coverage level for each cycle, so that agreement with observed node coverage levels is maximized. To do so, we impose an upper bound on variation of coverage over the lengths of cyclic sequences. This assumption is warranted in that our focus is on relatively short (1 kb - 10s of kb) and contiguous sequences. Furthermore, to limit the likelihood of repeats bridging between paths due to different species, we confine cycles to those having the property that there exists an edge (u,v) in the cycle such that its removal leaves a shortest path by weight from v to u.

After defining this problem formally below, we present an algorithm (and its implementation) designed to address it, called Recycler. Recycler leverages assembly graphs output by the SPAdes assembler [1] to specifically enable de novo assembly of plasmids. We show it greatly improves recovery of plasmids over naive assembly and alternative methods, namely Jørgensen's and SPAdes' built-in repeat resolution, introduced in [26]. Instead of using ad-hoc filtering that considers only sufficiently long contigs, Recycler uses all contigs -- often including those shorter than read length -- and the connections between them to generate cycles. We demonstrate Recycler's performance by applying it on both simulated and real data. We find that Recycler greatly increases recall at a slight cost in precision. This is demonstrated via comparisons performed on simulated plasmidomes including 100 - 1600 reference sequences. In these comparisons, Recycler showed approximately 20% increase in F1 score [29], and approximately 40% increase in the proportion of real plasmids recovered in several cases.

We also show that Recycler can be applied for plasmid extraction on real data from bovine plasmidome and from an Escherichia coli isolate. In the isolate case, we assessed Recycler's performance based on available plasmid reference sequences for E. Coli strain JJ1886, finding

recovery of 4 out of 5 known plasmid sequences and prediction of one additional plasmid matching plasmid annotation, and showing a significant difference in coverage from the rest of the genome. For the plasmidome, we found that out of the 75% of candidate cycles matching any annotation, nearly all (97%) matched known plasmid annotations. In this case, Recycler recovered 6 of 8 plasmids previously validated by PCR. Finally, we applied Recycler to bovine metagenome data as well, in this case finding limited success: only about a third of output sequences matched plasmid annotation. However, this result improved on the naive SPAdes result, which produced fewer cycles, out of which only about a quarter matched plasmid annotation. We provide guidance on current limitations preventing successful plasmid recovery in this context.

Methods

Our input is a graph G = (V,E), where each node v in V is a maximal unambiguous contig and is assigned length len(v) and coverage cov(v) [indicative of the average number of times read alignments overlap each position in the contig]. The length len(v) represents the number of k-mers [equal to contig length - k + 1] to avoid double-counting bases common to overlapping segments at their ends. All values are taken from the output of a short read assembly tool.

We seek to find a set of cycles Π and for each $p \in \Pi$, its *latent abundance level* a(p) such that the following criterion holds. Let $\Pi_v \subseteq \Pi$ be the set of cycles that cover vertex v. Then $\forall v \sum_{p \in \Pi_v} a(p) \leq cov(v)$, namely, the total latent coverage values of cycles using vertex v is bounded by its coverage. We seek a solution such that $\sum_{v \in V}[cov(v) - \sum_{p \in \Pi_v} a(p)]$ is minimized. In other words, we want the cycles together to approach the coverage of all vertices.

To address this problem, we adopt a greedy heuristic peeling algorithm (Algorithm 1, supplementary material) that dynamically updates G. We first introduce some definitions needed for the algorithm. We note subsequently cov(v) is considered a value updated during the course of execution (*initialized* to the value output by SPAdes), thus affecting each other expression depending on it whenever its value changes. We assign weight 0 to each edge, and weight $w(v) = \frac{1}{cov(v) * len(v)}$ to each node v. Furthermore, for each path p, we assign each node in it a value f(p,v) representing the relative proportion of length it holds in p: $f(p,v) = \frac{len(v)}{\sum_{v \in p} len(v)}$. This value is motivated by the observation that longer segments tend to be less prone to random fluctuations in coverage, and are thus more reliable coverage indicators. f(p,v) is used to define the mean and standard deviation of weighted coverage of path p as $\mu(p) = \sum_{v \in p} f(p,v) * cov(v)$ and $STD(p) = \sqrt{\sum_{v \in p} f(p,v)(cov(v) - \mu(p))^2}$

respectively, and consequently the coefficient of variation of p, $CV(p) = \frac{STD(p)}{\mu(p)}$. CV(p) is used to allow direct comparison of variation levels between paths independently of the magnitude of coverage of each.

We impose three biologically motivated constraints on the cycles that we consider. To limit the number of cycles considered, and reduce the likelihood of cycles formed from nodes emanating from different species, we demand:

1. $\exists (u,v) \in E$ such that p - (u,v) (the path obtained by removing (u,v) from p) is the shortest path (by sum of weights w(v)) from v to u
2. $CV(p) \leq \tau$
3. $\sum_{v \in p} len(v) \geq L$, where τ and L are defined thresholds.

Recycler processes each strongly connected component separately (Algorithm 1, supplementary material). It repeatedly finds a candidate cycle, assigns it latent coverage and subtracts that coverage from the graph, creating a new *residual coverage* (Figure 1). To avoid exponential growth in the number of cycles considered, we allow at most one path from each node to every one of its predecessors. Recycler finds the set of shortest cycles starting from v and ending at every one of v's predecessors before returning, where path lengths are calculated based on the sum of the weights w(v') assigned to each node v'. In effect, since w is inversely proportional to both coverage and length, and we seek shortest paths, this mechanism favors paths composed of a high proportion of the bases in the observed data over those composed of shorter nodes or those with lower coverage, and favors paths composed of a smaller number of node steps.

Recycler uses low CV values as an indication that nodes in the cycle are likely to belong to the same species. At each step, if the cycle p having the minimum CV value in the current cycle set has length at least L , $CV(p) \leq \tau$, and p (along with its cyclic rotations) has not been seen before, Recycler assigns a coverage level, $\mu(p)$ equal to the mean residual cycle coverage, and reduces that amount from the residual coverage of all cycle vertices. Vertices whose resulting coverage values become non-positive are then removed from the graph, allowing only those with some remaining coverage the opportunity to take part in additional cycles. If $CV(p) \leq \tau$ but p does not have total length at least L, p is not reported, but $\mu(p)$ coverage is subtracted from every node in p anyway. This assures the graph is updated regardless, allowing for additional (possibly sufficiently long) paths to be found.

After each such change, cyclical paths on the component are recalculated the same way using the updated coverage levels. This process continues as long as either new cycles are added that meet chosen thresholds τ and L, or the cardinality of the either V or E changes. This process is further detailed in Figure 1 and Algorithm 1.

Results

We first simulated plasmidomes using known references. We used these data sets to assess Recycler's precision and recall along with those of alternative extant methods by comparing predictions against the ground truth known by the simulation design. We also tested Recycler on real data from an E. Coli isolate, and both a cow rumen metagenome and plasmidome [4]. Since no references are available for metagenome and plasmidome data, we evaluated the accuracy by PCR validation [14] and by measuring the proportion of predicted plasmids having proper annotation as done in [4]. For the bacterial isolates that have a reference, predicted plasmids were compared against the reference sequences directly.

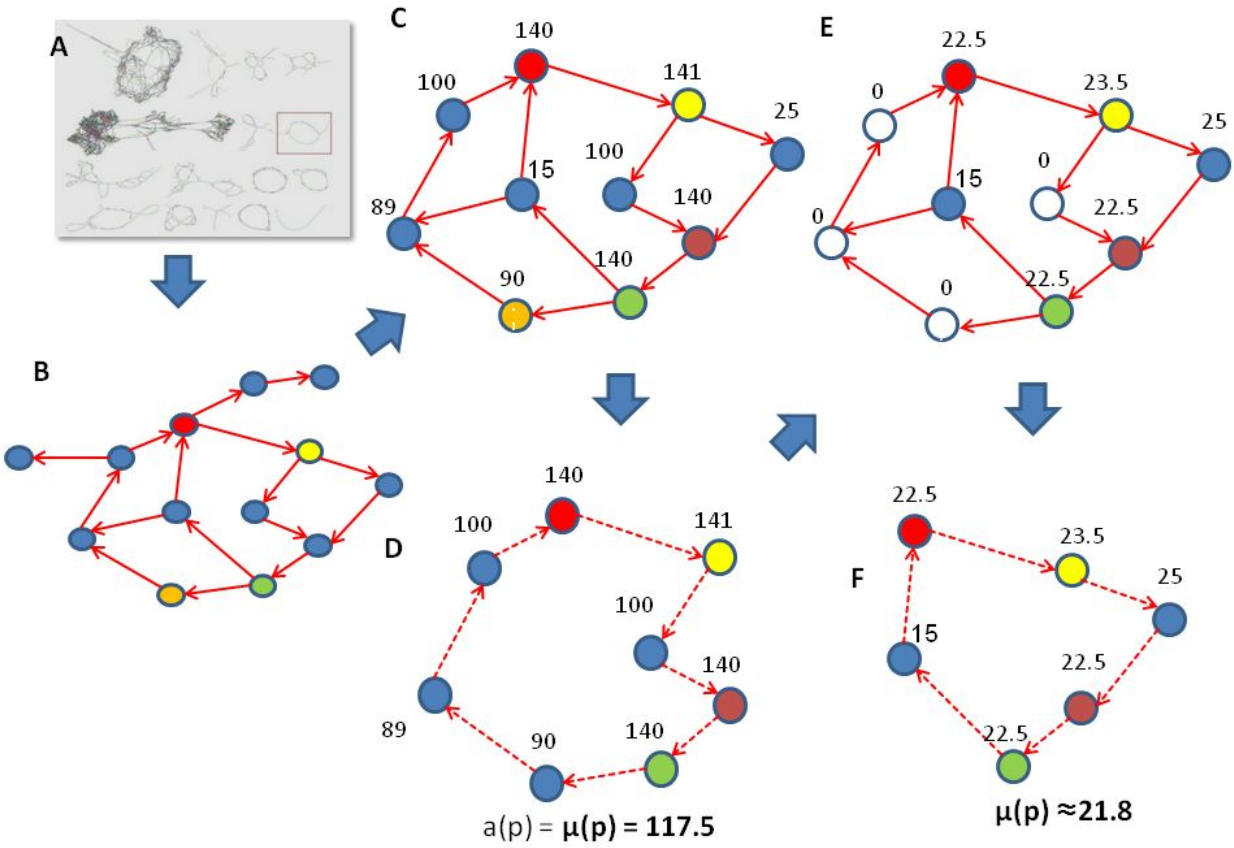

Figure 1. Recycler work-flow. A. The assembly graph. B. A single component is selected from the assembly graph (framed in A) and represented with vertices for contigs and edges for connecting k-mers. C. The reduced component after tip removal. Vertex values are observed contig coverage. For simplicity, all lengths are assumed to be equal and not shown. D. A cycle p from the component along with its vertex coverage values. The variance in coverage within the cycle is relatively low, which fits the scenario that it corresponds to a plasmid. The latent coverage is calculated as the mean of the node coverage values (a(p)=117.5 in this example). E. Coverage values are updated by setting cov(v)= max{cov(v)-a(p), 0} for each vertex v of the cycle p. Empty nodes have coverage zero and are removed by the algorithm (along with their edges, which are kept here for clarity) F. A second cycle q is

considered as a plasmid. As q has low CV it is designated a plasmid with latent coverage equal to its mean coverage (21.83).

Simulated plasmidomes

We simulated error-free paired-end reads from plasmids using a modified version of BEAR [13], a read simulator designed to generate artificial metagenome data. To avoid introducing coverage drops at sequence ends typical of linear sequences, BEAR was modified [https://github.com/rozovr/BEAR] to allow sampling of reads bridging reference sequence ends, as is observed for circular sequences. Plasmid reference sequences were selected from a collection composed of the union of the NCBI plasmids database and sequences reported in [4], filtered to include 2760 sequences with a length range of 1000 to 20,000 bp with a mean of 6337 bp. Five datasets were created, composed of 100 bp mates, with insert sizes $\sim N(500,100)$, varying from 1.25 M pairs sampled on 100 reference sequences doubling successively up to 20 M pairs sampled on 1600 sequences. Abundance levels were assigned using BEAR's low complexity option, which concentrates high abundance to few species using a power law distribution with parameters derived from [25].

Each such dataset was assembled with SPAdes and subsequently its output contigs and assembly graphs were used as inputs to other methods. To test recovery of the ground truth sequences, we used the nucmer alignment tool [15] that is designed for efficiently comparing long nucleotide sequences such as those of whole plasmids or chromosomes. In order to simplify this process, we modified reference sequences to remove non-ACGT characters (before read simulation and alignments). To avoid fragmented alignments caused by differences in start positions, we concatenated each cyclic sequence to itself before mapping; this allowed identification of complete matches at the center of the concatenated contigs when they were present. The mapping results are presented in Table 1 and Figure 2.

We defined true positives (TP) as 100% identity hits covering at least 80% the length of the reference sequence. False positives (FP) were any output cycles not meeting these criteria, and false negatives (FN) were reference sequences not aligned to in the output set using these criteria. Based on these conventions, precision was calculated as TP / (TP + FP) and recall as TP/ (TP + FN). We used the score $F1 = \frac{2 * precision * recall}{precision + recall}$ [29] to combine these measures in a manner that weighs precision and recall equally.

We used SPAdes' outputs before the repeat resolution (RR) stage as inputs to Recycler and a simplified version of Jørgensen's method [described in the supplement], as we found that contigs have greater precision before RR when compared to reference sequences (as shown in Table 1). As expected, recall generally decreased as the number of simulated plasmids increased. This was common to all tested methods. In general, we found that Recycler generated many more predictions than other methods, leading it to have higher recall than alternative approaches but slightly lower precision. The net effect of this tradeoff is shown in

Figure 2, where Recycler is shown to have up to a 20% advantage in some cases and maintains an advantage in all cases. We also found that the relative contribution of true positive plasmids past those provided by SPAdes increased with higher complexity; for the 400 - 1600 plasmid sets Recycler added an average proportion of 40% true positive instances to SPAdes' output.

To further characterize Recycler's performance, we categorized its predictions in terms of mean total path length, number of segments in the path, path coverage, and CV value calculated at the stage the path was removed. For each category values were subdivided into five ranges. In Figure 3 we show the precision values and the relative proportions of counts in the specified ranges.

Table 1

No. Reference Plasmids	SPAdes cycles before RR	SPAdes TPs before RR	SPAdes cycles after RR	SPAdes TPs after RR	Jørgensen cycles	Jørgensen TPs	Recyc. cycs	Recyc. TPs
100	59	59	67	63	64	61	86	72
200	83	82	102	87	88	82	142	109
400	147	146	179	159	170	155	254	206
800	274	270	324	288	327	285	477	368
1600	440	434	525	468	503	454	779	621

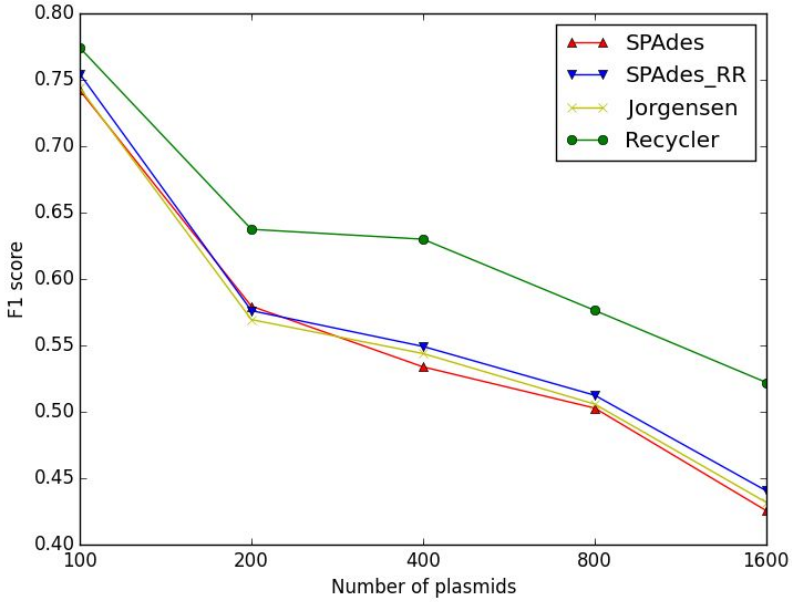

Figure 2. Methods performance on simulated data. SPAdes without repeat resolution (RR) was compared to SPAdes with repeat resolution, the method of Jørgensen et al, and Recycler. The contigs of SPAdes before RR were used as input for the three other methods. Recycler also relied on the graph produced at this stage. F1 score calculation is described in the main text. The number of plasmids is the number of simulated reference sequences in each case.

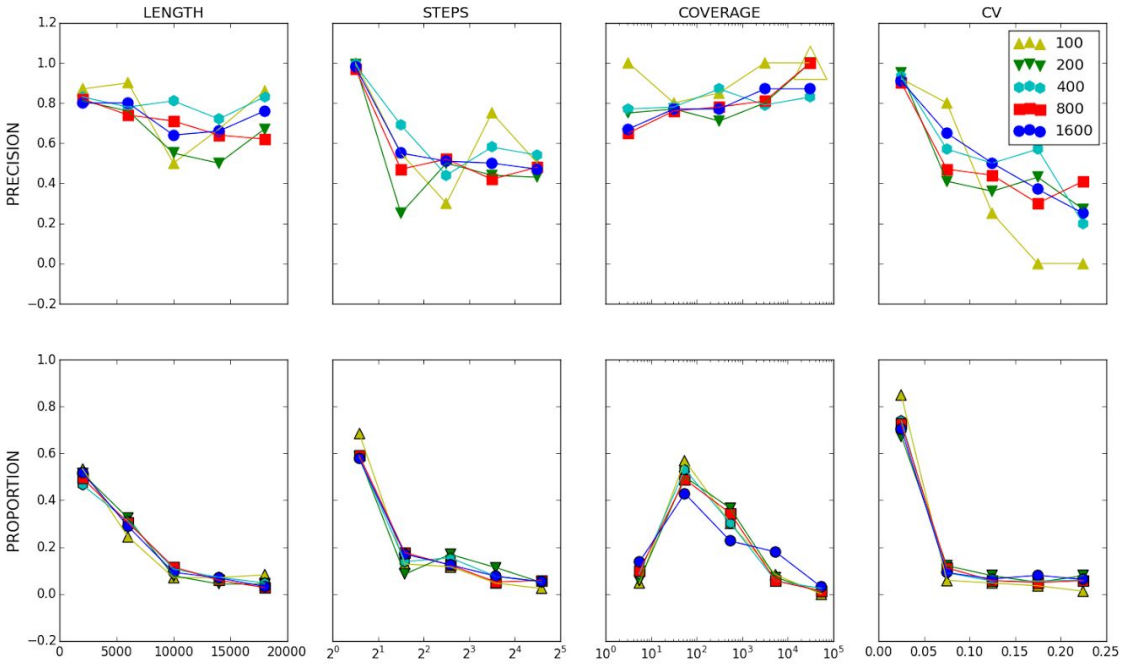

Figure 3. Recycler's precision, stratified by different properties. TOP: For simulated reads, true positive (100% identity over 80% reference length) alignment proportions were tallied inside 5 bins corresponding to value ranges of different properties - total assembly length (LENGTH), number of contigs in the path (STEPS), mean coverage level on the paths (COVERAGE), and path coefficient of variation (CV). Each point represents the precision rate for all simulated plasmids included in that range in the specified reference set. Reference sets are denoted by different colors and marker shapes. Intervals presented are as follows: length - [0,4000) [4000,8000),[8000,12000),[12000,16000),[16000,20000]; steps - [1,2), [2,4), [4,8), [8,16),[16,32]; coverage - [1,10), [10,100), [100,1000), [1000,10000),[10000,100000]; CV - [0,0.05), [0.05, 0.10), [0.10,0.15), [0.15,0.20),[0.20,0.25]. An empty marker is used to indicate the absence of any instances having the given property & bin combination - this occurs at the rightmost yellow coverage marker. BOTTOM: relative proportions of instance counts tested inside each bin, out of all output, taken from each reference set.

Using this stratification, it can be seen that precision greatly depends on CV and number of steps, is lightly aided by coverage, and is least affected by length. We note that in all cases most of the hits were due to single node self-loops, and that these were assigned a CV of 0. Nonetheless, values past these minimal values of one step and 0 CV were nearly monotonically decreasing in precision in terms of CV and number of steps.

The results above emphasize the importance of the default CV parameter. Simulated data was used to assess precision and recall for different values of τ. Values tested were 0.125, 0.1875, 0.25, 0.375, 0.5, and 0.25; the default value was chosen by noting where the maximal mean F1 score was obtained, τ = 0.25. This single value was used subsequently for both real and simulated data.

Real data

All of Recycler's results on real data were subjected to quantification of annotation results as described in [4] and compared against cycles present in the output produced by SPAdes. A summary of these results can be found in Table 2 in the Appendix.

Plasmidomes and PCR validation results

We ran Recycler on a bovine rumen plasmidome sample prepared as described in [4]. This data consisted of 5.1 M paired end 101 bp reads (trimmed to varied sizes for the sake of adapter removal) with an expected insert size of 500 bp [data available upon request]. Recycler output 468 cycles on this data. According to ORF prediction, 352 of the 468 had significant annotation hits. 97% of cycles that were annotated either matched plasmid annotation or aligned with plasmids extracted in [14]. We also tested Recycler's ability to recover 8 plasmids found by an earlier approach (described and compared against in the Appendix) that were PCR validated for circularity as described in [14]. Out of these 8 plasmids, 6 were fully recovered (aligning with 100% identity to 100% of the reference length) in Recycler's output.

Metagenome data

Metagenome data was derived from the rumen of a different cow residing in the same stable as the cow used to derive the plasmidome data. This data consisted of 7.5 M paired end 150 bp reads with expected insert size of 500 bp [data available upon request]. SPAdes was used to

produce the initial assembly graph once again, but in this case failed to run to completion using its default error correction due to a lack of memory, despite being run on a server with 256 GB of RAM. As a result, we performed error correction with BFC [18], which successfully ran to completion, and ran subsequent SPAdes steps as before.

Recycler output 43 cycles on this data. According to ORF prediction, 38 of the 43 had significant annotation hits. 34% of cycles that were annotated either matched plasmid annotation or aligned with plasmids extracted in [14]. The proportion of reported cycles matching known plasmid annotation was higher than for simple cycles output by SPAdes (26%). Still, it reflects the trend seen elsewhere [10] of fragmented assemblies and weak annotation results emerging from metagenome assembly of highly diverse environmental samples.

E. Coli isolate data, comparison against known references

As a final test, we ran Recycler on E. Coli strain JJ1886, downloaded from http://www.ebi.ac.uk/ena/data/view/SRX321704. Annotation for plasmids found in this strain was provided in [16], where 5 plasmids were found having lengths 110 Kb, 55.9 Kb, 5.6Kb, 5.2 Kb, and 1.6 Kb. Of these 5, Recycler output 4 with matching lengths - 55.9 Kb, 5.6Kb, 5.2 Kb, and 1.6 Kb. All except the longest of these had coverage levels differing by more than 2 standard deviations from the mean of the 18 cycles output by Recycler. One other plasmid of length 1.7 Kb plasmid was found that also had distinct coverage.

Of the additional 13 cycles output, we posit most closed cycles due to repeat sequences. All 18 had significant annotation hits following ORF prediction. Of these 67% either matched plasmid annotation or aligned with plasmids extracted in [14].

Discussion

In this article, we describe Recycler, a new algorithm and the first tool available for identification of plasmids from short read-length deep sequencing data. We demonstrate Recycler discovers plasmids that remain fragmented after de novo assembly. We have adapted the approach of choosing among likely enumerated paths using coverage and length properties, (often applied in transcriptome assembly [e.g., 24,28]) for extracting a specific but common inhabitant of metagenomes. We showed that many more real plasmids can be found by only constructing likely cyclical paths on the assembly graph versus alternative methods. We validated this approach on both real and simulated data.

Recycler displays high recall and precision on simulated plasmidomes, and we have suggested a means of separating real plasmids from cycles due to repeats in isolate data. As we have noted, coverage can be very useful for the latter, but the assumption that coverage will always differ significantly between plasmids and their host genome does not hold universally. It is worth noting that as new plasmids are identified and their common sequence motifs are observed, both reference-based identification and a priori trained prediction of plasmid features can be improved and harnessed for supplementing identification based on coverage and length

features alone. We aim to investigate how such knowledge can be leveraged for increased precision without sacrificing recall.

Further investigation will be needed to assess how plasmids can be extracted from environmental samples, in spite of the limitations now hampering metagenome assembly. Currently, a 'Catch-22' situation persists, in that diverse genomes require very high coverage for rare species to be captured, but such high coverage data demand computational resources beyond reach of most investigators in order to obtain high quality assemblies. While new techniques have aimed to address this pain [10, 5], they have yet to see widespread use, and work best when paired with multiple samples to allow for species separation by co-abundance signatures. Along with addressing these concerns, it remains to be seen whether a mixed approach of pre-screening environmental samples for plasmids and computationally filtering them out may benefit metagenome graph simplification.

Acknowledgements

Research partially supported by the Israel Science Foundation grants no. 1425/13 (EH), 317/13 (RS), 1313/13 (IM) and the ISF-NSFC joint program 2015-18 (RS). Additional support was provided by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 640384, IM). RR was supported in part by a fellowship from the Edmond J. Safra Center for Bioinformatics at Tel Aviv University, an IBM PhD fellowship, and by the Center for Absorption in Science, the Israel Ministry of Immigrant Absorption. EH is a Faculty Fellow of the Edmond J. Safra Center for Bioinformatics at Tel Aviv University.

Bibliography

[1] A. Bankevich, S. Nurk, D. Antipov, A. A. Gurevich, M. Dvorkin, A. S. Kulikov, V. M. Lesin, S. I. Nikolenko, S. Pham, A. D. Prjibelski, A. V Pyshkin, A. V Sirotkin, N. Vyahhi, G. Tesler, M. A. Alekseyev, and P. A. Pevzner, "SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing.," *J. Comput. Biol.*, vol. 19, no. 5, pp. 455–77, May 2012.

[2] C. M. Bevan MW, Flavell RB, "A chimaeric antibiotic resistance gene as a selectable marker for plant cell transformation," *Nature*, no. 304, pp. 184–187, 1983.

[3] A. Brown Kav, I. Benhar, and I. Mizrahi, "A method for purifying high quality and high yield plasmid DNA for metagenomic and deep sequencing approaches.," *J. Microbiol. Methods*, vol. 95, no. 2, pp. 272–9, Nov. 2013.

[4] A. Brown Kav, G. Sasson, E. Jami, A. Doron-Faigenboim, I. Benhar, and I. Mizrahi, "Insights into the bovine rumen plasmidome.," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 109, no. 14, pp. 5452–7, Apr. 2012.

[5] B. Cleary, I. L. Brito, K. Huang, D. Gevers, T. Shea, S. Young, and E. J. Alm, “Detection of low-abundance bacterial strains in metagenomic datasets by eigengenome partitioning,” *Nat. Biotechnol.*, vol. 33, no. 10, pp. 1053–1060, Sep. 2015.

[6] S. Conlan, P. J. Thomas, C. Deming, M. Park, A. F. Lau, J. P. Dekker, E. S. Snitkin, T. A. Clark, K. Luong, Y. Song, Y.-C. Tsai, M. Boitano, J. Dayal, S. Y. Brooks, B. Schmidt, A. C. Young, J. W. Thomas, G. G. Bouffard, R. W. Blakesley, J. C. Mullikin, J. Korlach, D. K. Henderson, K. M. Frank, T. N. Palmore, and J. A. Segre, “Single-molecule sequencing to track plasmid diversity of hospital-associated carbapenemase-producing Enterobacteriaceae.,” *Sci. Transl. Med.*, vol. 6, no. 254, p. 254ra126, Sep. 2014.

[7] H. P. Doring and P. Starlinger, “Barbara McClintock’s controlling elements: now at the DNA level,” *Cell*, vol. 39, pp. 253–259, 1984.

[8] J. A. Gilbert and C. L. Dupont, “Microbial metagenomics: beyond the genome.,” *Ann. Rev. Mar. Sci.*, vol. 3, pp. 347–71, Jan. 2011.

[9] K. E. Holt, H. Wertheim, R. N. Zadoks, S. Baker, C. A. Whitehouse, D. Dance, A. Jenney, T. R. Connor, L. Y. Hsu, J. Severin, S. Brisse, H. Cao, J. Wilksch, C. Gorrie, M. B. Schultz, D. J. Edwards, K. V. Nguyen, T. V. Nguyen, T. T. Dao, M. Mensink, V. L. Minh, N. T. K. Nhu, C. Schultsz, K. Kuntaman, P. N. Newton, C. E. Moore, R. A. Strugnell, and N. R. Thomson, “Genomic analysis of diversity, population structure, virulence, and antimicrobial resistance in Klebsiella pneumoniae , an urgent threat to public health,” *Proc Natl Acad Sci*, p. 201501049, 2015.

[10] A. C. Howe, J. K. Jansson, S. A. Malfatti, S. G. Tringe, J. M. Tiedje, and C. T. Brown, “Tackling soil diversity with the assembly of large, complex metagenomes.,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 111, no. 13, pp. 4904–9, Apr. 2014.

[11] M. Hunt, N. De Silva, T. D. Otto, J. Parkhill, J. A. Keane, and S. R. Harris, “Circlator: automated circularization of genome assemblies using long sequencing reads,” Cold Spring Harbor Labs Journals, Jul. 2015.

[12] M. Imelfort, D. Parks, B. J. Woodcroft, P. Dennis, P. Hugenholtz, and G. W. Tyson, “GroopM: an automated tool for the recovery of population genomes from related metagenomes.,” *PeerJ*, vol. 2, p. e603, Jan. 2014.

[13] S. Johnson, B. Trost, J. R. Long, V. Pittet, and A. Kusalik, “A better sequence-read simulator program for metagenomics.,” *BMC Bioinformatics*, vol. 15 Suppl 9, no. Suppl 9, p. S14, Jan. 2014.

[14] T. S. Jørgensen, Z. Xu, M. A. Hansen, S. J. Sørensen, and L. H. Hansen, “Hundreds of circular novel plasmids and DNA elements identified in a rat cecum metamobilome.,” *PLoS One*, vol. 9, no. 2, p. e87924, Jan. 2014.

[15] S. Kurtz, A. Phillippy, A. L. Delcher, M. Smoot, M. Shumway, C. Antonescu, and S. L. Salzberg, “Versatile and open software for comparing large genomes.,” *Genome Biol.*, vol. 5, no. 2, p. R12, Jan. 2004.

[16] V. F. Lanza, M. de Toro, M. P. Garcillán-Barcia, A. Mora, J. Blanco, T. M. Coque, and F. de la Cruz, “Plasmid flux in Escherichia coli ST131 sublineages, analyzed by plasmid constellation network (PLACNET), a new method for plasmid reconstruction from whole genome sequences.,” *PLoS Genet.*, vol. 10, no. 12, p. e1004766, Dec. 2014.

[17] D. Li, C.-M. Liu, R. Luo, K. Sadakane, and T.-W. Lam, “MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph,” *Bioinformatics*, vol. 31, no. 10, pp. 1674–1676, Jan. 2015.

[18] H. Li, “BFC: correcting Illumina sequencing errors,” *Bioinformatics*, vol. 31, no. 17, pp. 2885–7, May 2015.

[19] H. Li and R. Durbin, “Fast and accurate short read alignment with Burrows-Wheeler transform.,” *Bioinformatics*, vol. 25, no. 14, pp. 1754–60, Jul. 2009.

[20] S. S. Minot, N. Krumm, and N. B. Greenfield, “One Codex: A Sensitive and Accurate Data Platform for Genomic Microbial Identification,” Cold Spring Harbor Labs Journals, Sep. 2015.

[21] H. C. Neu, “The Crisis in Antibiotic Resistance,” *Science (80-.).*, vol. 257, no. 5073, pp. 1064–1073, Aug. 1992.

[22] H. B. Nielsen, M. Almeida, A. S. Juncker, S. Rasmussen, J. Li, S. Sunagawa, D. R. Plichta, L. Gautier, A. G. Pedersen, E. Le Chatelier, E. Pelletier, I. Bonde, T. Nielsen, C. Manichanh, M. Arumugam, J.-M. Batto, M. B. Quintanilha Dos Santos, N. Blom, N. Borruel, K. S. Burgdorf, F. Boumezbeur, F. Casellas, J. Doré, P. Dworzynski, F. Guarner, T. Hansen, F. Hildebrand, R. S. Kaas, S. Kennedy, K. Kristiansen, J. R. Kultima, P. Léonard, F. Levenez, O. Lund, B. Moumen, D. Le Paslier, N. Pons, O. Pedersen, E. Prifti, J. Qin, J. Raes, S. Sørensen, J. Tap, S. Tims, D. W. Ussery, T. Yamada, P. Renault, T. Sicheritz-Ponten, P. Bork, J. Wang, S. Brunak, and S. D. Ehrlich, “Identification and assembly of genomes and genetic elements in complex metagenomic samples without using reference genomes.,” *Nat. Biotechnol.*, vol. 32, no. 8, pp. 822–828, Jul. 2014.

[23] Y. Peng, H. C. M. Leung, S. M. Yiu, and F. Y. L. Chin, “IDBA-UD: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth.,” *Bioinformatics*, vol. 28, no. 11, pp. 1420–8, Jun. 2012.

[24] M. Pertea, G. M. Pertea, C. M. Antonescu, T.-C. Chang, J. T. Mendell, and S. L. Salzberg, “StringTie enables improved reconstruction of a transcriptome from RNA-seq reads,” *Nat. Biotechnol.*, vol. 33, no. 3, pp. 290–295, Feb. 2015.

[25] M. Pignatelli and A. Moya, “Evaluating the fidelity of de novo short read metagenomic assembly using simulated data.,” *PLoS One*, vol. 6, no. 5, p. e19984, Jan. 2011.

[26] A. D. Prjibelski, I. Vasilinetc, A. Bankevich, A. Gurevich, T. Krivosheeva, S. Nurk, S. Pham, A. Korobeynikov, A. Lapidus, and P. A. Pevzner, “ExSPAnder: a universal repeat resolver for DNA fragment assembly.,” *Bioinformatics*, vol. 30, no. 12, pp. i293–301, Jun. 2014.

[27] I. Sharon, M. Kertesz, L. A. Hug, D. Pushkarev, T. A. Blauwkamp, C. J. Castelle, M. Amirebrahimi, B. C. Thomas, D. Burstein, S. G. Tringe, K. H. Williams, and J. F. Banfield, “Accurate, multi-kb reads resolve complex populations and detect rare microorganisms.,” *Genome Res.*, vol. 25, no. 4, pp. 534–43, Apr. 2015.

[28] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter, “Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation.,” *Nat. Biotechnol.*, vol. 28, no. 5, pp. 511–5, May 2010.

[29] D. M. W, “EVALUATION: FROM PRECISION, RECALL AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION.”

Appendix

2-step assembly procedure

We tested another assembly strategy, which we call 2-step assembly. That procedure used the ability of SPAdes to leverage libraries of different insert lengths. Reads were first assembled using SPAdes, then aligned the to the assembled contigs using BWA [19], then split into groups (based on alignment properties) which were treated as separate sequenced libraries on which a second round of SPAdes assembly was performed. The splitting of reads was based on BAM file alignment property flags. These flags separated read pairs into 'proper' pairs -- those having correct orientation and expected insert size -- and improper pairs that fail to meet at least one of these criteria. These two read groups were used as separate inputs to a second execution of SPAdes, with the aim of benefitting from improved repeat resolution and formation of more cycles. The last step of this procedure re-examined mapping of read pairs, this time focusing on contig ends. The purpose of this step was the identification of self-loops closed by read pairs where these loops did not close in the assembly graph.

We tested the effect of the 2-step procedure compared on the performance of Recycler using the 1600 reference simulated data set. We assessed the number of cycles generated with and without second steps and application of Recycler. Far fewer cycles were obtained with the 2-step assembly, implying a strong reduction in recall by virtue of the number of candidates alone.

	SPAdes cycles before RR	Recycle cycles
1step	440	779
2step	343	373

Jørgensen's method

We only used the first part of the protocol described Jørgensen's method in order to allow for maximal recall; the second part involved further filtering (and thus reduction) of the first part's results. Circular contigs were identified by finding those having opposite ends that overlap. These were then refined by breaking those that do have such overlaps into halves, and then gluing the far ends by applying the minimus2 assembler, part of the AMOS package.

Table 2

	metagenome	metagenome simple	E. Coli	E. Coli simple	plasmidome	plasmidome simple
no. of seqs annotated as plasmids	13	7	12	3	287	169
no. of other seqs with hits on Jørgensen data	0	0	0	0	54	44
no. of seqs with any nr annotation	38	27	18	3	352	222
total no. of cycs	43	32	18	3	468	317
% of annotated cycs as plasmids	34	26	67	100	82	76
% of annotated cycs as plasmids or Jørgensen	34	26	67	100	97	96
% annotated at all	88	84	100	100	75	70

==========

Algorithm 1:

Inputs:
G = (V,E); V = the set of contigs, E = (k+1)-mers formed by adjacent contigs; for each node v, len(v), cov(v)
max_CV = upper bound for coefficient of variation [default is 0.25]
min_length = lower bound for the minimal length for an accepted plasmid [default is 1000 bp]

func **peel_cycles**(V,E):
- path_count := 0
- initialize cycles and seen_paths to empty sets
- add all sufficiently long self-loops to cycles set
- remove self-loop nodes and edges from component
- set w(v) = 1/((len(v)*cov(v))
- set paths(u,v,W) = {paths from u to v in G where each node v is assigned w(v) and w(u,v) = 0}

```
for comp in strongly_connected_components(G):
        paths <- all shortest paths in comp that are not in seen_paths
        sort paths by CV values

        # iterate as long as you either add paths or remove nodes from comp
        while (path_count != last_path_count OR len(V(comp)) != last_node_count):
```

```
        curr_path = paths.pop() # retrieves lowest CV path

        if get_total__path_mass(curr_path) < 1: # low coverage or very short path
                seen_paths |= curr_path
                for v in V(comp):
                        V(comp).remove_node(v)
                paths = get_shortest_paths(comp, seen_paths)
        if get_path_coverage_CV(curr_path) <= max_CV
         AND curr_path not in seen_paths:
                seen_paths |= curr_path
                update_node_coverage_vals(curr_path, comp)
                if get_total_length(curr_path) >= min_length:
                        cycles |= curr_path
                paths = get_shortest_paths(comp, seen_paths)

return cycles
```