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

Genome analysis

HyAsP, a greedy tool for plasmids identification

Robert Müller[1] and Cedric Chauve[2]

[1]Computational Methods for the Analysis of the Diversity and Dynamics of Genomes, Bielefeld University, 33615 Bielefeld, Germany;
[2]Department of Mathematics, Simon Fraser University, Vancouver BC V5A 1S6, Canada

Abstract

Motivation: Plasmids are ubiquituous in bacterial genomes, and have been shown to be involved in important evolutionary processes, in particular the acquisition of antimicrobial resistance. However separating chromosomal contigs from plasmid contigs and assembling the later is a challenging problem.
Results: We introduce HyAsP, a tool that identifies, bins and assembles plasmid contigs following a hybrid approach based on a database of known plasmids genes and a greedy assembly algorithm. We test HyAsP on a large sample of bacterial data sets and observe that it generally outperforms other tools.
Availability: https://github.com/cchauve/HyAsP
Contact: cedric.chauve@sfu.ca

1 Introduction

Plasmids are extra-chromosomal DNA molecules common in Bacteria. Plasmids differ from chromosomes in various features, such as their length (they tend to be much shorter than chromosomes), copy number (plasmids can be present in multiple copies in a cell) and GC content. They play an important role in horizontal gene transfer and, thus, in the transmission of virulence factors and antibiotic resistance (Dolejska and Papagiannitsis, 2018; Carattoli, 2013). Therefore, the identification of plasmids in bacterial samples is important toward mitigation strategies against the proliferation of drug-resistant bacteria.

Various approaches have been explored for the detection of plasmids, with a recent focus on methods using whole-genome sequencing (WGS) data, and we refer to (Orlek *et al.*, 2017; Arredondo-Alonso *et al.*, 2017; Laczny *et al.*, 2017) for recent reviews on plasmids detection and assembly tools. Among existing tools we can distinguish two kinds of approaches: de-novo methods and reference-based methods. Recycler (Rozov *et al.*, 2017) and plasmidSPAdes (Antipov *et al.*, 2016) belong to the former group of methods. Recycler predicts plasmids by repeatedly peeling off cycles of the assembly graph based on read-depth and length features. plasmidSPAdes assumes that the read depth of plasmids differs from the chromosome; it estimates the chromosomal read depth, removes those likely chromosomal contigs and predicts plasmids from the connected components of the resulting reduced assembly graph. MOB-recon (Robertson and Nash, 2018) is a recent reference-based method; it uses a database of reference plasmids and collections of known replicons and relaxases; contigs are mapped against the reference database and grouped into putative plasmid units, that are further refined by discarding those units without a replicon or relaxase.

In this note, w present HyAsP (Hybrid Assembler for Plasmids), a novel tool for identifying plasmid contigs in WGS assemblies and binning them into individual plasmids, combining in a hybrid approach, ideas from reference-based and de-novo methods, using information on the occurrences of known plasmid genes and considering characteristics such as read depth and GC content. We compare the prediction quality of HyAsP with plasmidSPAdes and MOB-recon on a data set comprising 66 genomes and show that HyAsP generally outperforms both tools.

2 Methods

HyAsP is a greedy algorithm for the detection, binning and assembly of plasmid contigs from an assembly graph using information from known plasmid genes, read depth and GC content. We provide below a high level description of the algorithm (including a pseudocode workflow in Algorithm 1) and experiments, while the full technical details are provided in Supplementary Material.

Identifying seeds. HyAsP starts from an assembly graph – obtained by Unicycler (Wick *et al.*, 2017) by default – and first identifies *seeds*, defined as contigs satisfying criteria related to length, the presence of known plasmid genes, and read depth, defined here as in Unicycler, i.e. as the normalised k-mer coverage of contigs. Seeds are used as starting points to construct chains of contigs potentially defining plasmid fragments.

Greedy extension. Next, seeds are enumerated based on their plasmid genes density and GC content, preferring those with a high gene density and a GC content differing from the average GC content of the assembly graph, and each seed is extended into a contig chain by a greedy algorithm. To extend the current contig chain, its two endpoints are searched for *eligible extensions*, defined as contigs that are adjacent to the endpoint contigs in

Algorithm 1: HyAsP algorithm

```
Data: assembly graph AG, plasmid genes-contigs mapping GCM
Result: collection of predicted plasmids PC

Determine seed contigs and sort them by their eligibility;
Initialise empty PC;
while $\exists$ eligible seed S do
    Initialise new plasmid P with seed S;
    while $\exists$ eligible extension of P do
        Choose extension and add the corresponding contig to P;
    Circularise the contig chain of P (if possible);
    Finalise the read-depth values of the contigs in P;
    foreach contig C in P do
        Decrease the read depth of C in AG by the depth of C in P;
    Add plasmid P to PC;
Split PC into putative plasmids and questionable plasmids;
Bin the plasmids based on their characteristics (optional);
```

the assembly graph ands satisfy criteria aimed at averting likely errors by, e.g., limiting the length of the plasmid (default: 1,750,00 nt) and avoiding overly long gene-free stretches (default: 2,000 nt) as well as fluctuations in GC content and read depth within the current contig chain that are too high (default: 15 %). The eligible extensions are scored using the function below and the extension with the minimum score is chosen.

The score of an extension of the current plasmid (contig chain) P with contig B is defined as

$$
\begin{aligned}
&\mathsf{ext\_score}(\mathsf{P},\mathsf{B})\\
&= \mathsf{weight.depth\_diff} * |1 - \mathsf{depth}(\mathsf{B}) \,/\, \mathsf{average\_depth}(\mathsf{P})|\\
&+ \mathsf{weight.gene\_density} * \mathsf{density}(\mathsf{B})\\
&+ \mathsf{weight.gc\_diff} * |\mathsf{gc\_content}(\mathsf{P}) - \mathsf{gc\_content}(\mathsf{B})|,
\end{aligned}
$$

with depth(.) (resp. density(.), gc_content(.)) being functions returning the read depth (resp. plasmid genes density, GC content), and weight.depth_diff, weight.gene_density, weight.gc_diff are user-definable weights (all 1 by default). The average read depth of plasmid P, average_depth(P), is the average (by default mean) read depth of the nucleotides in the contigs currently underlying P. It is computed using the current read depth of the contigs in the assembly graph and if a contig occurs multiple times in P, its read depth is evenly distributed over the occurrences.

Updating read depth. Once no extension is possible, the resulting contig chain is considered as a potential plasmid fragment. Regardless of its length, it is then circularised if the first and last contig are identical and in the same orientation or do overlap sufficiently. The construction of a plasmid fragment ends by determining a final read-depth value for each contig it contains. An average read depth (by default the mean over all contigs of the current contigs chain) is computed and a contig is assigned the minimum of this average and its read depth in the assembly graph. If a contig has a larger read depth than the value rd it is assigned in the current plasmid, it is kept in the assembly graph, after having decreased its depth by rd, thus allowing contigs (e.g. containing repeats) to be used in later iterations of the seed extension algorithm

Postprocessing. Once all plasmid fragments are constructed, a final quality-filtering step is applied, using several quality requirements derived from a large collection of known plasmids. Plasmids that are long enough (default: 1,500 nt), have sufficient plasmid gene density (default: 0.3) and average read depth (default: 0.4 * median read depth of the initial assembly graph), and are not a subplasmid of another predicted plasmid, are categorised as *putative* and represent the main predictions. The other plasmids, not satisfying at least one of these criteria, are labelled *questionable* and output separately. Optionally, the (putative) non-circular plasmids are then grouped into bins, where plasmid fragments are placed in th same bin if both their read depth and GC content differ by at most a user-specified number of standard deviations of the respective characteristic (calculated from the set of all plasmid fragments).

Integrated pipeline. HyAsP is provided as a component of a pipeline accepting FASTQ files and a database of known plasmid genes as input. The reads are assembled using Unicycler and, subsequently, the plasmid genes are mapped to the assembly contigs using BLAST. Finally, the plasmids are predicted in the assembly graph using HyAsP as described above. HyAsP is a single-threaded program written completely in Python (requiring Biopython, NumPy and pandas) with system calls to BLAST (blastn, makeblastdb). After downloading HyAsP, it can be used directly or installed as a package through pip or a container through singular.

3 Results

Data. We evaluated HyAsP on a collection of real plasmids compiled in (Robertson and Nash, 2018). We compared HyAsP with plasmidSPAdes and MOB-recon, that represent the two approaches (de-novo and reference-based) that are combined in our tool, and performed best in previous benchmarks.

The data set consists of 133 bacterial samples comprising 377 plasmids. To simulate the use of plasmids identification tools on a newly WGS dataset, using only knowledge from previously existing plasmid data, the data set was split in half based on the release date of the read data. The 66 samples released on 19 December 2015 or later formed the *test samples*, spanning 147 plasmids from 8 different species (the *ground truth plasmids*). The other 67 samples were used as the *database samples* and we refer to the corresponding 230 plasmids as the *MOB-database*. Similarly, we created a second set of references to repeat the evaluation with larger database by collecting all plasmids from NCBI databases released before 19 December 2015 (the *NCBI-database*).

Metrics. We assessed the predictions of all tools in terms of their precision and recall by mapping only the putative plasmids (HyAsP), and the plasmid contigs (MOB-recon, plasmidSPAdes), respectively, against the ground truth plasmids using BLAST.

First, we evaluated the ability of the tools to identify plasmid contigs, not considering whether these contigs are correctly grouped together. To this end, the recall of a reference plasmid was computed as the proportion of its sequence matched to the predicted plasmids, where a position of the reference was considered as *matched* when it was included in at least one BLAST hit between the reference and a predicted plasmid. This was extended to an aggregate recall over multiple samples by summing up the number of matched positions for the different reference plasmids and dividing by the total length of all reference plasmids in the considered samples. The precision was computed analogously with the roles of reference plasmids and predicted plasmids reversed. As the scores depend on the 'union' of BLAST hits over several sequences, they are referred to as *union-recall* and *union-precision*.

Second, we assessed the methods as plasmid contigs binning tools and also included the plasmids bins derived by HyAsP from the putative plasmids using default parameters. For the analysis, we redefined the recall to consider only the BLAST hits of the predicted plasmid that covers the largest segment among all hits to the reference plasmid sequences. The definition of the precision was adapted accordingly and as the scores now

Reference	Tool	Precision (union / best)	Recall (union / best)	F1 (union / best)	Transloc. (pred. / ref.)	Unaligned (pred. / ref.)
NCBI	HyAsP	0.8714 / 0.8251	0.8988 / 0.6217	0.8849 / 0.7091	0.0463 / 0.2771	0.1286 / 0.1012
	HyAsP bins	0.8714 / 0.7121	0.8988 / 0.8165	0.8849 / 0.7607	0.1593 / 0.0825	0.1286 / 0.1012
MOB	HyAsP	0.9345 / 0.8879	0.7752 / 0.5565	0.8474 / 0.6842	0.0466 / 0.2187	0.0655 / 0.2248
	HyAsP bins	0.9345 / 0.7982	0.7752 / 0.6952	0.8474 / 0.7431	0.1363 / 0.0800	0.0655 / 0.2248
-	plasmidSPAdes	0.6592 / 0.5608	0.7420 / 0.7366	0.6981 / 0.6368	0.0984 / 0.0054	0.3408 / 0.2580
	MOB-recon	0.7602 / 0.7562	0.5839 / 0.4745	0.6605 / 0.5831	0.0041 / 0.1076	0.2340 / 0.4161

Table 1. Precision, recall and F1 score of HyAsP, HyAsP bins, plasmidSPAdes, and MOB-recon considered as plasmid contigs identification / binning tools, over the 66 test samples using the NCBI-database and MOB-database. For the columns Precision, Recall and F1, in each cell, the values x / y correspond to the values using respectively the union / best evaluation model, i.e. the evaluation as identification / binning tools. For the columns Translocations and Unaligned, in each cell, the values x / y correspond to the values for predicted (resp. reference) plasmids.

only use the BLAST hits of the one sequence with the best match, they are called *best-recall* and *best-precision*. In both analyses, precision and recall were then summarized with the F1 score.

Last we recorded two additional metrics on translocated and unaligned segments. A translocations correspond to a reference (resp. predicted) plasmid being split into several predicted (resp. reference) plasmids, in which case we record the amount of sequence outside of the best match between the predicted and reference plasmids. Unaligned reference (resp. predicted) segments measure the amount of reference (resp. predicted) plasmid sequence that is not present in predicted (resp. reference) plasmids. Unaligned predicted plasmids thus measure the amount of chromosomal contigs present in the predicted plasmids. As previously, we record aggregate statistics over all samples by dividing by the total length of reference and predicted plasmids.

Results. Table 1 presents the results obtained with HyAsP, HyAsP followed by the contigs binning postprocessing step (called "HyAsP bins"), plasmidSPAdes and MOB-recon considered both as plasmid contigs identification tools (union-precision and union-recall), and plasmid binning tools (best-precision and best-recall).

As shown in Table 1, HyAsP outperformed plasmidSPAdes and MOB-recon as a plasmid contig identification tool in both total precision and recall. Consequently, the F1 score of HyAsP was notably higher as well. We also observed a trade-off in precision and recall between the MOB-database and the larger NCBI-database: HyAsP attained a higher precision (0.93) and lower recall (0.78) using the less extensive MOB-database, which led to a similar but slightly lower F1 score (0.85). As expected, the binning of plasmid contigs turned out to be more difficult than their identification (Table 1). All tools showed a notable drop in precision or recall compared to the previous analysis. While HyAsP experienced the largest reduction in recall, it still attained the highest precision. The precision of HyAsP remained relatively stable, indicating that plasmids predicted by HyAsP might not correspond to entire plasmids but rather not confound different plasmids. In addition, the binned plasmids of HyAsP attained a higher F1 score, showing that the binning mechanism improved the predictions overall. The trade-offs between the two databases and between plasmidSPAdes and MOB-recon observed in the previous analysis carried over to the binning of plasmid contigs. Last, we can observe that plasmidSPAdes and MOB-recon perform slightly better in terms of translocations, although at the expense of recruiting a much larger proportion of chromosomal sequence and missing a larger amount of true plasmid sequence.

Subsequently, we analysed the predictions after grouping the samples based on the associated organism at the species level. Using the NCBI-database, HyAsP performed similarly well or better than plasmidSPAdes and MOB-recon for the majority of species. Only for *Klebsiella aerogenes*, did it fall notably behind plasmidSPAdes but still outperformed MOB-recon. While there are species, for which the other tools achieved a higher recall or precision, the trade-off between both is usually smaller for HyAsP leading to a better overall prediction quality (in terms of the F1 score). These results, together with a more extensive set of results, including more refined tables and figures describing the precision, recall and F1 score, and an analysis of the level of misassembly in the predicted plasmids, are available in Supplementary Material.

References

Antipov, D. *et al.* (2016). plasmidSPAdes: assembling plasmids from whole genome sequencing data. *Bioinformatics*, **32**(22), 3380–3387.

Arredondo-Alonso, S. *et al.* (2017). On the (im)possibility of reconstructing plasmids from whole-genome short-read sequencing data. *Microb Genom*, **3**(10). doi: 10.1099/mgen.0.000128.

Carattoli, A. (2013). Plasmids and the spread of resistance. *Int J Med Microbiol*, **303**(6-7), 298–304.

Dolejska, M. and Papagiannitsis, C. C. (2018). Plasmid-mediated resistance is going wild. *Plasmids*. doi: j.plasmid.2018.09.010.

Laczny, C. C. *et al.* (2017). Assessing the heterogeneity of in silico plasmid predictions based on whole-genome-sequenced clinical isolates. *Brief Bioinform*. doi: 10.1093/bib/bbx162.

Orlek, A. *et al.* (2017). Plasmid Classification in an Era of Whole-Genome Sequencing: Application in Studies of Antibiotic Resistance Epidemiology. *Front Microbiol*, **8**, 182.

Robertson, J. and Nash, J. H. E. (2018). MOB-suite: software tools for clustering, reconstruction and typing of plasmids from draft assemblies. *Microb Genom*, **4**(8). doi: 10.1099/mgen.0.000206.

Rozov, R. *et al.* (2017). Recycler: an algorithm for detecting plasmids from de novo assembly graphs. *Bioinformatics*, **33**(4), 475–482.

Wick, R. R. *et al.* (2017). Unicycler: Resolving bacterial genome assemblies from short and long sequencing reads. *PLOS Comput Biol*, **13**(6), 1–22.