GraphPlas: Refined Classification of Plasmid Sequences using Assembly Graphs

Anuradha Wickramarachchi and Yu Lin

Abstract—Plasmids are extra-chromosomal genetic materials with important markers that affect the function and behaviour of the microorganisms supporting its environmental adaptations. Hence the identification and recovery of such plasmidic sequences from assemblies is a crucial task in metagenomics analysis. In the past, machine learning approaches have been developed to separate chromosomes and plasmids. However, there is always a compromise between precision and recall in the existing classification approaches. The similarity of compositions between chromosomes and their plasmids makes it difficult to separate plasmids and chromosomes with high accuracy. However, high confidence classifications are accurate with a significant compromise of recall, and vice versa. Hence, the requirement exists to have more sophisticated approaches to separate plasmids and chromosomes accurately while retaining an acceptable trade-off between precision and recall. We present GraphPlas, a novel approach for plasmid recovery using coverage, composition and assembly graph topology. We evaluated GraphPlas on simulated and real short read assemblies with varying compositions of plasmids and chromosomes. Our experiments show that GraphPlas is able to significantly improve accuracy in detecting plasmidic and chromosomal contigs on top of popular state-of-the-art plasmid detection tools.
The source code is freely available at: https://github.com/anuradhawick/GraphPlas.

Index Terms—plasmid recovery, metagenomics, classification, assembly graph

1 Introduction

The recovery of plasmids remains to be a challenging yet intriguing task in metagenomics analysis due to their participation in environmental adaptations of microorganisms. These sequences often consist of circular deoxyribonucleic acid (DNA) molecules and can replicate independently from the bacterial chromosomes. Plasmids lack genes that are commonly related with main metabolic processes, but rather carry genes that allow the host cell to adapt rapidly to changing environmental conditions and survive under various selective pressures [1], [2], [3]. Moreover, plasmids contribute to horizontal gene transfer among different species of bacteria allowing them to gain genes related to antibiotic resistance and heavy metal resistance [2], [4], [5]. Hence it is biologically important to identify and recover plasmids from environmental samples, and study them to understand their behaviour and functions.

Plasmid studies have benefited immensely from culture-based methods to understand genetic elements from different bacteria [6]. However, the genetic elements of non-cultivable bacteria cannot be studied using these culture-based methods. Alkaline lysis [7] is a widely used method to extract plasmids from bacterial cells, but this method is not suitable for a complex sample containing other eukaryotes [8]. Moreover, analysing plasmids using PCR-based methods such as PCR-based replicon typing (PBRT) scheme [9] are labour-intensive and provide results with limited resolution [10]. Furthermore, culture-independent methods such as exogenous plasmid isolation [11] and Transposon-aided capture (TRACA) [12] may misinterpret the total amount of plasmids present in the sample [8]. Hence, computational methods are preferred for the recovery of plasmids.

Next-generation sequencing (NGS) technologies such as Illumina have enabled us to sequence and analyse bacterial genomes including both their chromosomes and plasmids, directly from their environments [13]. When we sequence environmental samples, we obtain reads from both chromosomes and plasmids of species present in the sample. Hence, certain post-processing steps are required to recover these sequences originating from plasmids. plasmidSPAdes [14], Recycler [15], PLACNET [16] and PLACNETw [17] are some of the tools which have been developed to directly reconstruct plasmid sequences from raw NGS reads.

NGS technologies can produce highly accurate reads with very low error rates. However, due to their limited read lengths (100-300 bp), it becomes challenging to separate reads directly as they may not contain accurate genomic signatures [18]. Hence, a common method carried out to identify plasmids is to first assemble the short NGS reads into much longer sequences called *contigs* and then classify these contigs as originating from chromosomes or plasmids. There are several approaches that rely on coverage and graph topology (e.g. Recycler [15] and SCAPP [19]) for the recovery of plasmids from plasmid assemblies. These tools extract contigs with circular structures by peeling off paths in the assembly graph, observing the coverages in an unsupervised manner. The main limitation of these tools is that the recovered contigs based on circular paths may not be plasmids due to complexities in assembly graphs. Furthermore, circular chromosomes and linear plasmids exist making it further challenging to recover plasmids in an unsupervised manner. Hence, tools have been developed based on supervised machine learning techniques to over-

• *Anuradha Wickramarachchi and Yu Lin were at Research School of Computer Science, College of Engineering and Computer Science, Australian National University.*

E-mail: anuradha.wickramarachchi, yu.lin@anu.edu.au

Manuscript received April XX, XXXX; revised August XX, XXXX.

come these issues.

Tools such as cBar [20], PlasmidFinder [21], mlplasmids [10], PlasFlow [22], PlasClass [23] and Platon [24] utilise the composition profiles of sequences to label contigs as chromosomes or plasmids. However, the assembled contigs are typically shorter and fragmented for plasmid assemblies due to the relatively lower abundance of plasmidic reads. This results in poor composition and gene profiles. Therefore, there is a significant compromise between the precision and recall in plasmid classification (refer to PlasClass and PlasFlow results in Fig. 2 and Fig. 3. These plots are obtained by varying the probability threshold used for identifying plasmids by each tool from 0 to 1, followed by computation of precision and recall for each plasmid and chromosome class.). Hence, improving precision along one class will result in poor precision of the other class with lower recall within the same class, and vice versa. There have been several efforts to address similar issues in metagenomics binning by tools such as GraphBin [25] and GraphBin2 [26]. However, the contigs are clustered into bins only at species level and the plasmid level annotations are not available. Moreover, composition information is not used in GraphBin and GraphBin2.

In this paper, we propose an assembly graph assisted approach of improving plasmid recovery by harnessing information such as composition, coverage and connectivity in the assembly graph. The proposed methodology significantly improves the precision and recall of the plasmid classification on top of popular tools in the field. To the best of our knowledge, this is the first time that composition, coverage and graph topology information of assembled contigs have been utilised together to address the problem of plasmid sequence classification.

2 METHODS

The complete workflow of GraphPlas is demonstrated in Fig. 1. GraphPlas takes assembled contigs and the relevant assembly graph as the input. Then the contigs are classified using an existing plasmid detection program that predicts the plasmid probability of contigs. Then the contigs are labelled as *chromosomes*, *plasmids* or *unclassified* depending on the probability values. Note that the *unclassified* class contains contigs that are either shorter than 1000 bp or ones with probabilities that are in-between the probability boundaries of plasmids and chromosomes.

For the initial prediction, we chose the approaches presented by PlasClass [23] and PlasFlow [22]. This is because PlasClass and PlasFlow outputs probability values on which a confidence level can be applied. Also, the classifications at higher confidence regions are reliable. We varied the probability threshold for PlasClass in order to investigate if the classification can be improved just by parameter tuning. However, from Fig. 2 and Fig. 3 it is clearly evident that means beyond parameter tuning are required for better results. Furthermore, it is evident that at high confidence thresholds the precision of classification is reasonably high for GraphPlas to improve the results.

2.1 Computation of contig similarity metrics

For the steps 2 and 3 in Fig. 1, distances between contigs are required to label the assembly graph in a semi-supervised manner. These distances are derived from summing up negative log values of similarities obtained from three different methods. The computation of each similarity measure is explained in the following sub sections.

2.1.1 Computing similarity of contigs using the topology of assembly graph

Consider the assembly graph G where V is the set of vertices that represents the contigs in the graph. Let L be the set of labelled vertices and U be the set of unlabelled vertices. Topological similarity $S_t(V_i, V_j)$, where $V_i \in U$ and $V_j \in L$, is computed using the random walk probabilities. In GraphPlas we use a variant of label propagation algorithm proposed in [27]. The detailed algorithm is presented in Section 1 of the Supplementary materials.

2.1.2 Computing similarity of contigs using composition

The composition similarity S_k is computed using equation 1 [28].

$$S_k = \frac{\mathcal{N}(D_e(V_i,V_j)|\mu_{intra},\sigma^2_{intra})}{\mathcal{N}(D_e(V_i,V_j)|\mu_{intra},\sigma^2_{intra})+\mathcal{N}(D_e(V_i,V_j)|\mu_{inter},\sigma^2_{inter})} \quad (1)$$

Here $D_e(V_i, V_j)$ represents euclidean distance between 4-mer (tetramer) vectors of vertices V_i and V_j respectively. For this formula we use approach presented by [28]. The mean and standard deviation are computed for each of the tetramer frequency vector distances within and between different microbial species using mean and standard deviation of euclidian tetramer vector distances within species and between species, where μ_{intra} and μ_{inter} represents mean distances and, σ_{intra} and σ_{inter} represents standard deviations of distances within and between species respectively. In order to estimate the μ_{intra}, σ_{intra}, μ_{inter} and σ_{inter} we consider the set of all the reference Chromosomes and Plasmid assemblies from NCBI refseq database. First we seed 50 subsequences of length 10000 bp from each of the reference sequence and compute their normalised tetranucleotide frequency vectors. The resulting histograms are presented in Fig. 4. From these histograms we compute that $\mu_{intra} = 0.02$, $\sigma_{intra} = 0.010/2$, $\mu_{inter} = 0.069$ and $\sigma_{inter} = 0.034$. However, for probability computations in equation 1 we use $\mu_{intra} = 0$ to ensure nearly identical sequences will have high similarity similar to [28].

2.1.3 Computing similarity of contigs using coverage

The similarity of two coverages is computed as $S_c = Poisson(Cov(V_i)|Cov(V_j))$, where $Cov(V_i)$ and $Cov(V_i)$ demonstrate the coverage of vertices V_i and V_j respectively similar to work done by [28].

2.2 Step 1: Initial Classification

GraphPlas is capable of obtaining the seed classifications from either PlasClass or PlasFlow. The composition profiles tend to deviate significantly as the length becomes shorter. Hence, we chose 1000 bp as the cutoff for the initial classification, similar to the default settings of many other binning tools in metagenomics [28], [29], [30]. Therefore, we consider the contigs that are greater than 1000 bp for the initial results of PlasClass and PlasFlow.

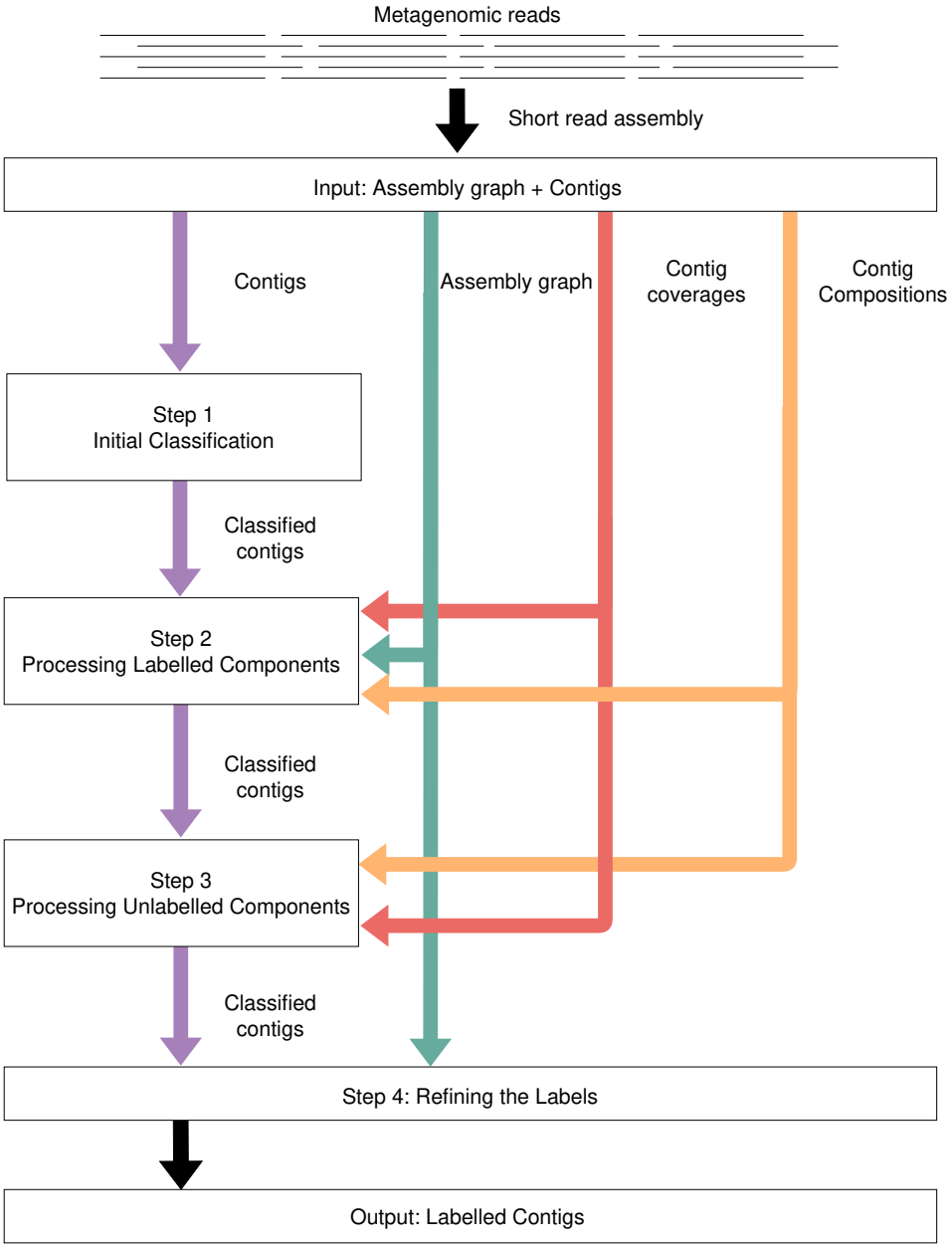


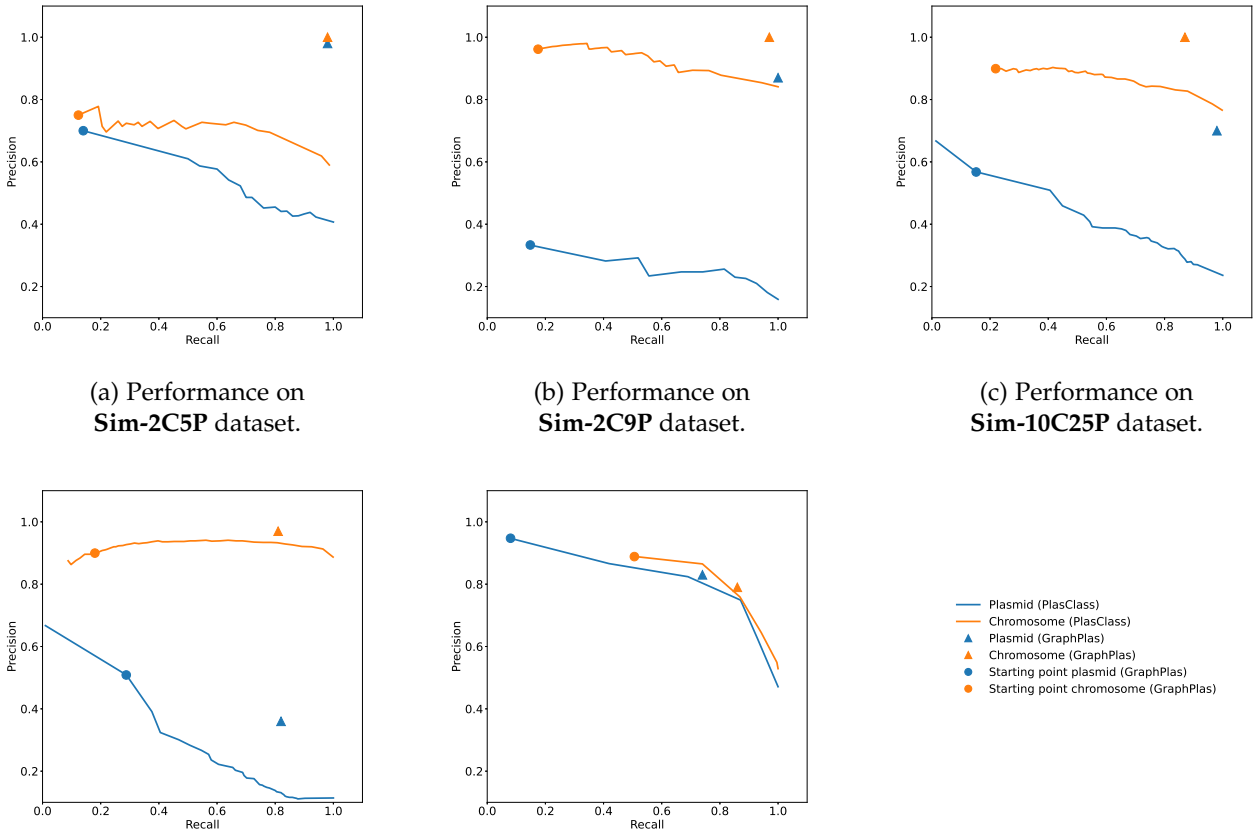
Fig. 1. The workflow of GraphPlas. The inputs for the workflow are the contigs and the assembly graph output from the assembler. The contigs are initially classified and the most confident set of contigs are used as seeds. Contig labels are propagated to other contigs using graph topology, composition and coverage information. Finally the contig labels are refined again and ids are output with the assigned label.

We chose the most confident set of classifications from the classification result provided by the selected tool. First we order the contigs based on the predicted probability in the decreasing order. We label 50% of the contigs below 0.5 probability threshold as **chromosomes** in both tools. We label the top 10% above 0.5 probability threshold as **plasmids** for PlasClass. We chose 20% for PlasFlow as it tends to predict plasmids with a slightly lower tendency. The contigs that are neither classified as plasmids or chromosomes are labelled as *unclassified*.

We introduce *labelled components* and *unlabelled components* to support the next step. A *labelled component* is defined as a component in the assembly graph with atleast one contig with a label other than *unclassified*. Conversely an *unlabelled component* is a component whose all contigs are labelled as *unclassified*. Refer to Fig. 5(a) for the classification result of dataset **Sim-2C9P** using PlasClass. The initial classification result that is chosen by GraphPlas is demonstrated in Fig. 5(b), and unlabelled components are circled in red colour.

2.3 Step 2: Processing labelled components

In this step we consider components of the assembly graph that have at least one labelled contig from step 1. Using the assembly graph and its labelled vertices from the first step, we label the rest of the unclassified contigs. We first consider the contigs that are either 1000 bp or longer for labelling.



(a) Performance on **Sim-2C5P** dataset.

(b) Performance on **Sim-2C9P** dataset.

(c) Performance on **Sim-10C25P** dataset.

(d) Performance on **Sim-14C38P** dataset.

(e) Performance on **Wastewater-Plas** dataset.

Legend.

Fig. 2. Precision-recall curves for PlasClass and GraphPlas. Plots obtained by varing the probability threshold for classification from 0 to 1. For GraphPlas we have a single point for each class as we pick the best starting threshold for each tool.

We propagate the labels from labelled contigs to the other contigs based on the distance computed under all three topological, composition and coverage similarities. The equation 2 is used to compute the combined distance $D(V_i, V_j)$ between an unlabelled and a labelled vertex.

$$D(V_i, V_j) = -log(S_t \times S_k \times S_c) \tag{2}$$

Next we use equation 3 to compute the distance $D_{short}(V_i, V_j)$ for the contigs that are shorter than 1000 bp. Composition of shorter contigs is not considered because composition information can be unreliable for shorter contigs [30]. Hence, the distance computation is limited only to topological and coverage similarities.

$$D_{short}(V_i, V_j) = -log(S_t \times S_c) \tag{3}$$

We use the above equations as the distance metric for a KNN (K-Nearest Neighbours) classifier with up to 5 nearest neighbours. The contigs are then labelled using the majority vote. This classification is done in a step wise fashion for long and short contigs (shorter than 1000 bp) in order. Refer to Fig. 5(c) for the classification result of dataset **Sim-2C9P** after processing labelled components.

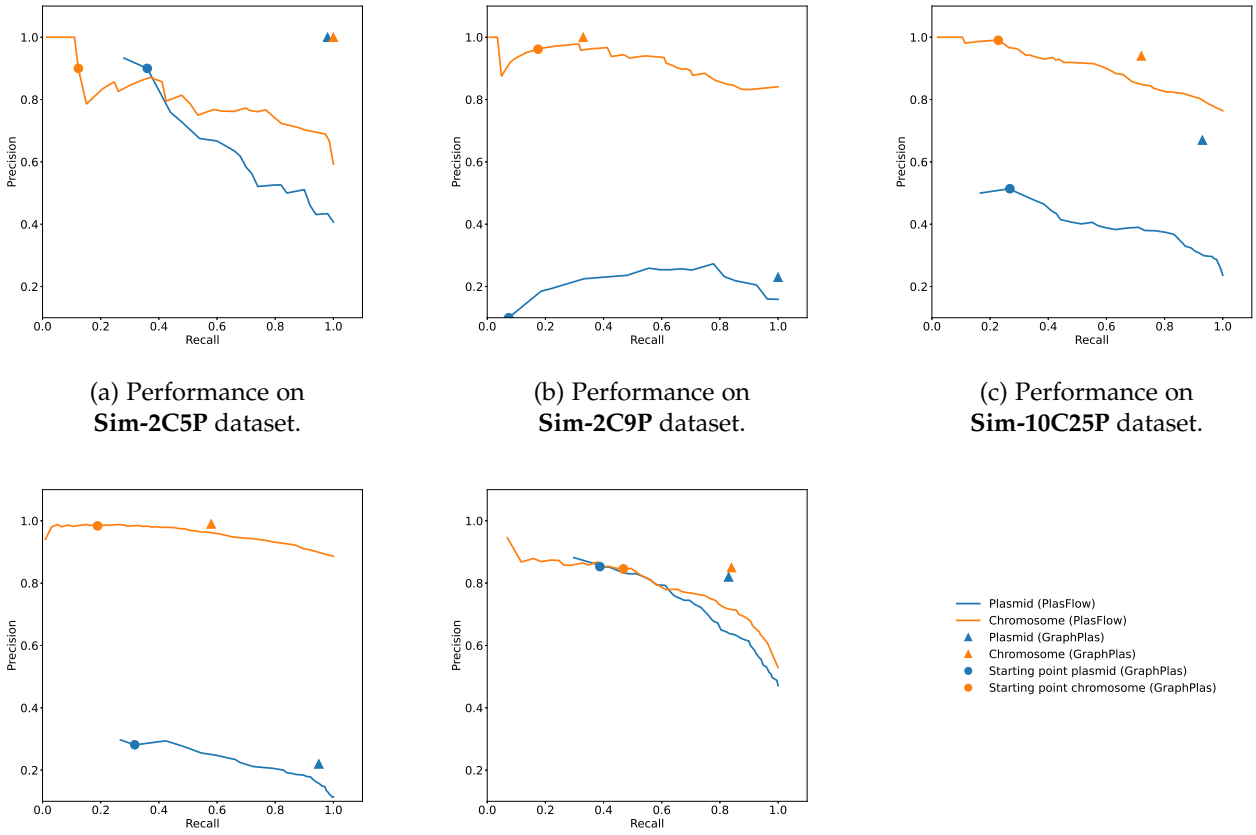
2.4 Step 3: Processing unlabelled components

In this step we label the contigs from assembly graph components that do not contain any labelled contigs. Therefore, we rely on the labelled set of contigs in the entire assembly graph in this step. In order to label such isolated components in the graph, we use a KNN classifier using only the composition and coverage information. We use equations 4 and 5 to compute the distances $D_{isolated}(V_i, V_j)$ and $D_{isolated\_short}(V_i, V_j)$ respectively. $D_{isolated}(V_i, V_j)$ is computed for vertices that are 1000 bp or longer and $D_{isolated\_short}(V_i, V_j)$ is computed for contigs that are shorter than 1000 bp.

$$D_{isolated}(V_i, V_j) = -log(S_k \times S_c) \tag{4}$$

$$D_{isolated\_short}(V_i, V_j) = -log(S_c) \tag{5}$$

Similar to step 2, we use equations 4 and 5 as the distance metric for a KNN classifier to classify the vertices in the isolated components. The KNN classifier is initiated on contigs longer than 1000 bp with only two neighbouring vertices. This is because the longer contigs in the assembly graph tend to have better coverage and composition representations. Furthermore, contigs at the repeat points have multiple edges and elevated coverage values. Hence, such



(a) Performance on **Sim-2C5P** dataset.

(b) Performance on **Sim-2C9P** dataset.

(c) Performance on **Sim-10C25P** dataset.

(d) Performance on **Sim-14C38P** dataset.

(g) Performance on **Wastewater-Plas** dataset.

Legend.

Fig. 3. Precision-recall curves for PlasFlow and GraphPlas. Plots obtained by varing the probability threshold for classification from 0 to 1. For GraphPlas we have a single point for each class as we pick the best starting threshold for each tool.

contigs are avoided by limiting the number of neighbours to a maximum of 2. Majority voting of up to 5 neighbours is used for the labelling process. Refer to Fig. 5(d) for the classification result of dataset **Sim-2C9P** after processing labelled components. Note that there are still unsupported labels from the initial classification, circled in red. There are no neighbouring contigs that support the labels of such contigs.

2.5 Step 4: Refining the labels

We define contigs that are connected to other contigs from a different label without any support of its own label as *ambiguously* labelled contigs. In this step we utilise the assembly graph to correct the labels of such ambiguously labelled contigs. Majority voting is used in order to correct such labels. We start the label correction from non-leaf vertices (*i.e.* vertices with more than 1 neighbour) since they are more informative in terms of neighbours. Finally, the labels of the leaf vertices (*i.e.* vertices with only one neighbour) are corrected to match the neighbouring vertex. Fig. 5(e) demonstrates the assembly graph once GraphPlas refines the final result from previous steps.

3 Experimental Setup

3.1 Datasets

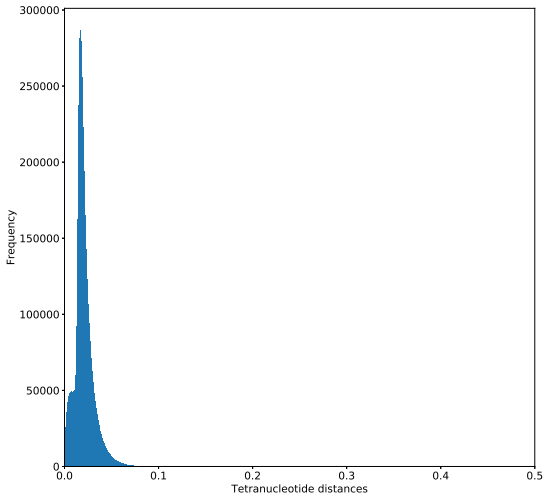
We evaluated GraphPlas using four simulated datasets and one real dataset with varying complexities and plasmid copy numbers. The information on the datasets considered are as follows. Please refer to Section 2 of the Supplementary materials for the detailed information of the simulation.

TABLE 1
Information on the datasets used for the experiments.

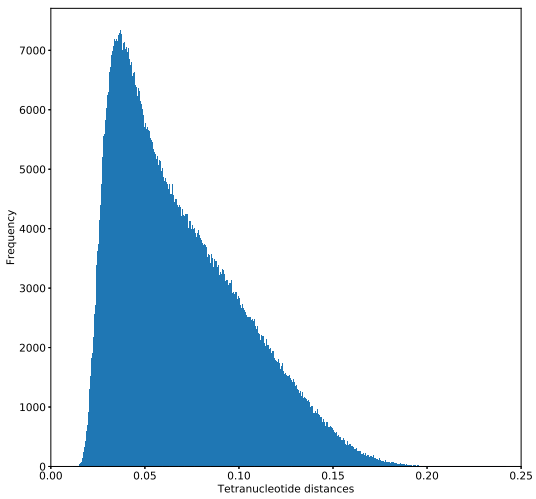
Dataset	Read length (bp)	Number of reads	Number of contigs	Edges in the graph
Sim-2C5P	300	239425	128	471
Sim-2C9P	300	368632	187	921
Sim-10C25P	300	1359961	636	2913
Sim-14C38P	300	3371230	1881	3977
Wastewater-Plas†	125	8757400	32510	3215

†We only considered contigs with length 1000 bp or longer. A unique ground truth was discovered only for 436 contigs. However, the complete graph was utilised in the program.

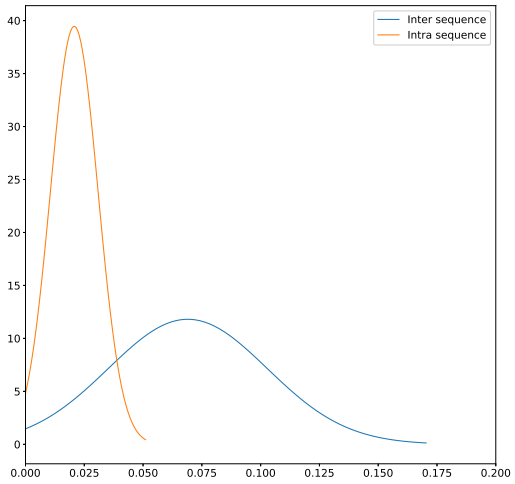
1) We simulated four datasets using InSilicoSeq simulator [31] with MiSeq configuration that produces reads of length 300 bp. Similar to the work



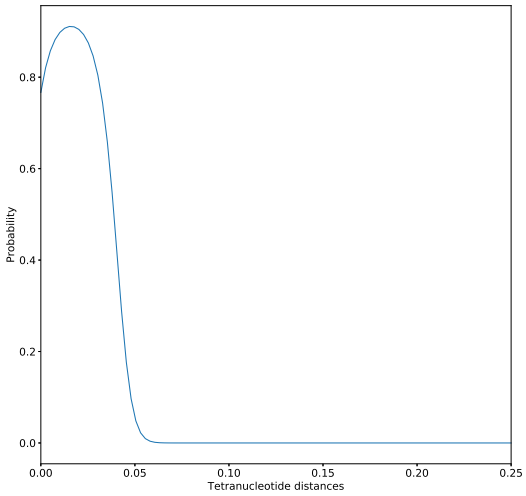
(a) Intra sequence distances.



(b) Inter sequence distances.



(c) Distribution of distances.



(d) Probability vs distance.

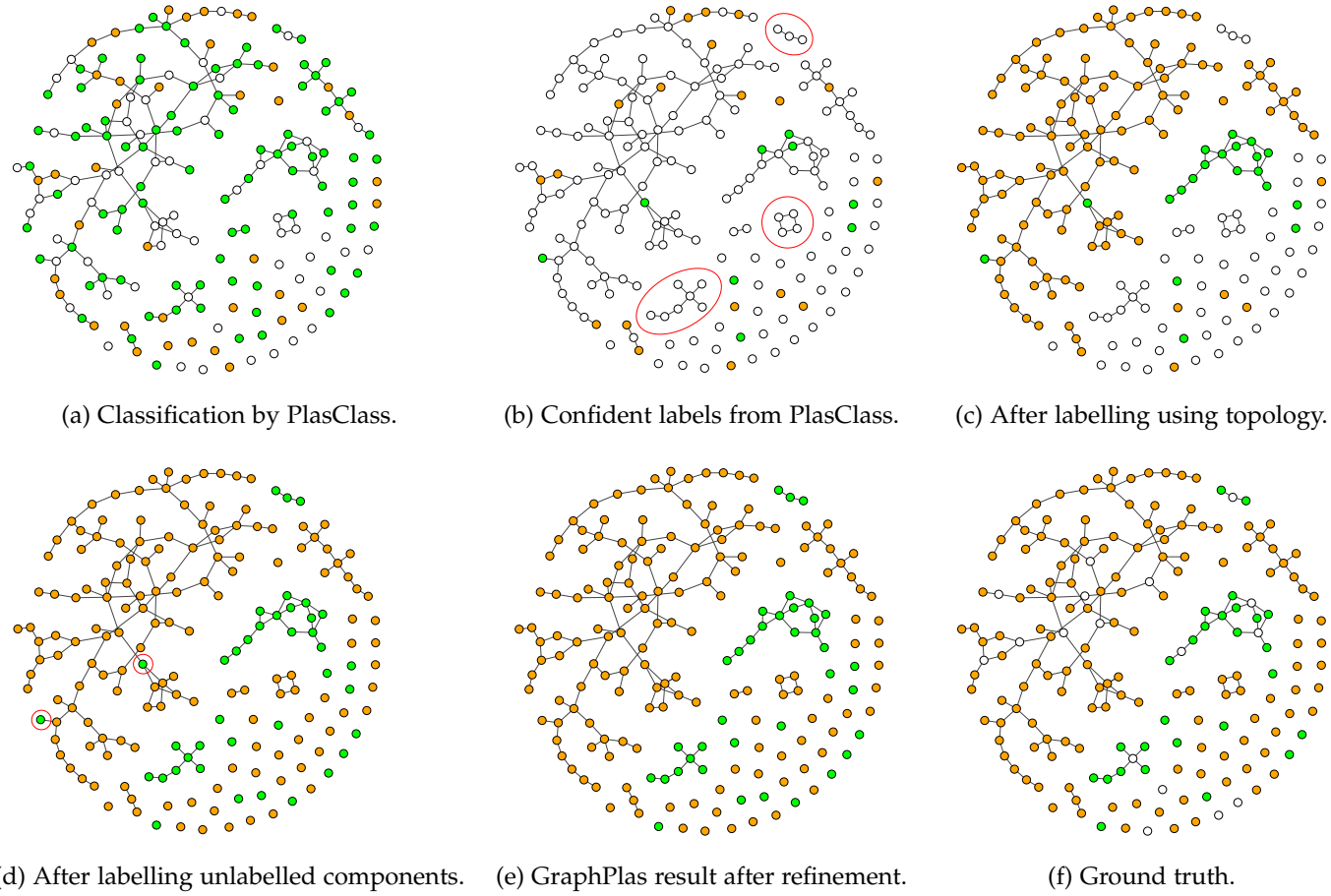
Fig. 4. Distance histograms for intra and inter sequence distances for normalised tetranucleotide distances. (a) and (b) demonstrates the histogram for euclidean distances of tetramer vectors for sequences within a given species and between different species respectively. (c) demonstrates the resulting normal distributions plotted using means and standard deviations obtained from (a) and (b). (d) demonstrates the probability vs distance curve computed using the equation 1.

done by Pellow *et. al* [23] we used the equation $5 \times min(1, log(L)/10)$, where L is the length of the plasmid reference, to compute probability of success for a geometric distribution to obtain the plasmid copy numbers. These copy numbers were used to calculate the simulation coverage of each plasmid. This was performed to amplify the plasmid copy numbers of shorter plasmids.

- **Sim-2C5P**: Contains 1 species with a total of 2 chromosomes and 5 plasmids.
- **Sim-2C9P**: Contains 1 species with a total of 2 chromosomes and 9 plasmids.
- **Sim-10C25P**: Contains 5 species with a total of 10 chromosomes and 25 plasmids.
- **Sim-14C38P**: Contains 7 species with a total of 14 chromosomes and 38 plasmids.

2) We used the wastewater plasmidome sample **ERR1538272** (referred as **Wastewater-Plas**) [32] in order to evaluate the performance of GraphPlas on real datasets. The dataset was assembled using metaSPAdes [33]. The dataset consists of Illumina HiSeq 2500 paired end reads with read length of 125 bp.

Table 1 indicate the number of reads, contigs and the read length of each of the dataset assembled.



(a) Classification by PlasClass. (b) Confident labels from PlasClass. (c) After labelling using topology.

(d) After labelling unlabelled components. (e) GraphPlas result after refinement. (f) Ground truth.

Fig. 5. Assembly graph with contig labels for the dataset **Sim-2C9P** at different stages of GraphPlas. Chromosomes and plasmids are represented in orange and green colours respectively. Unlabelled components are circled in (b). Contigs that need to be refined are circled in (d). The contigs without a unique mapping to a single class are indicated in white.

3.2 Evaluation Criteria

We evaluated GraphPlas using macro-averaged precision, recall and F1-score using the following standard equations where;

- TP_p: the number of actual plasmidic sequences that were classified as plasmidic (true positives for plasmids)
- TP_c: the number of actual chromosomal sequences that were classified as chromosomal (true positives for chromosomes)
- FP_p: the number of non-plasmidic sequences that were classified as plasmidic (false positives for plasmids)
- FP_c: the number of non-chromosomal sequences that were classified as chromosomal (false positives for chromosomes)
- FN_p: the number of plasmidic sequences that were not classified as plasmidic (false negatives for plasmids)
- FN_c: the number of chromosomal sequences that were not classified as chromosomal (false negatives for chromosomes)

$$Precision(\%) = \frac{1}{2} \times \frac{TP_p}{TP_p + FP_p} + \frac{1}{2} \times \frac{TP_c}{TP_c + FP_c} \quad (6)$$

$$Recall(\%) = \frac{1}{2} \times \frac{TP_p}{TP_p + FN_p} + \frac{1}{2} \times \frac{TP_c}{TP_c + FN_c} \quad (7)$$

$$\textit{F1 score}(\%) = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (8)$$

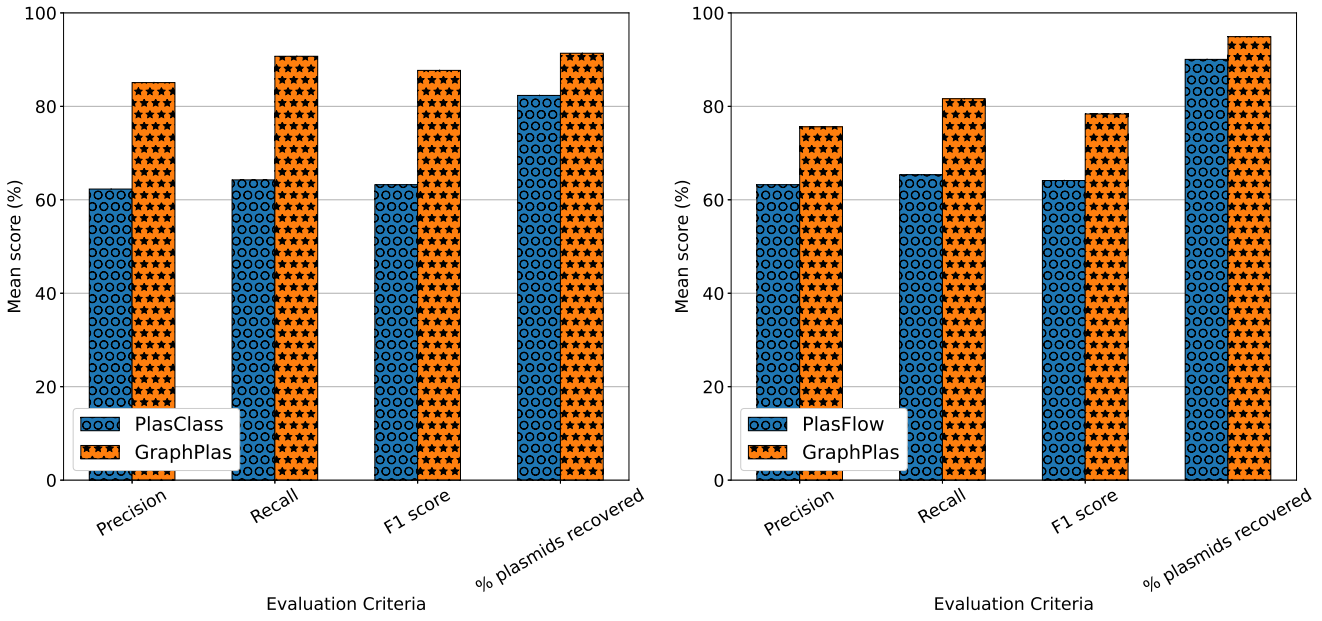
The metrics for each class is averaged in order obtain values with fair representation on each imbalanced class. Similar to previous evaluations in plasmid studies [23], the fraction of plasmidic contigs recovered, $TP_p/(TP_p + FN_p)$, is also considered in our comparison. For the set of simulated datasets, the ground truth label was assigned by mapping the assembled contigs to the respective set of reference genomes. The mapping was performed using Minimap 2.1 [34]. Only the contigs with a unique mapping to either plasmids or chromosomes were considered in the evaluation. Furthermore, the assembled contigs from the real dataset, with unknown ground truth were alined to the NCBI assemblies. The contigs that had a unique mapping to either plasmids or chromosomes were considered in the evaluation. Moreover, we only considered the contigs that are either 1000 bp or longer with an alignment beyong 50% of the query length.

TABLE 2
Comparison of macro-averaged classification results of PlasClass [23] and GraphPlas with PlasClass as the initial classifier.

Dataset	Tool used	Precision (%)	Recall (%)	F1 score (%)	Percentage of plasmids recovered (%)
Sim-2C5P	PlasClass	58.70	56.01	57.33	86.00
	GraphPlas	**99.02**	**99.32**	**99.17**	**100.00**
Sim-2C9P	PlasClass	58.92	64.48	61.58	92.59
	GraphPlas	**93.55**	**98.60**	**96.01**	**100.00**
Sim-10C25P	PlasClass	60.82	64.06	62.40	84.06
	GraphPlas	**85.20**	**93.51**	**89.17**	**100.00**
Sim-14C38P	PlasClass	53.88	58.81	56.23	80.00
	GraphPlas	**66.66**	**81.82**	**73.47**	**82.44**
Wastewater-Plas	PlasClass	79.18	77.97	78.57	69.04
	GraphPlas	**80.98**	**80.35**	**80.67**	**74.39**

TABLE 3
Comparison of macro averaged classification results of PlasFlow [22] and GraphPlas with PlasFlow as the initial classifier.

Dataset	Tool used	Precision (%)	Recall (%)	F1 score (%)	Percentage of plasmids recovered (%)
Sim-2C5P	PlasFlow	65.90	59.33	62.44	94.00
	GraphPlas	**100.00**	**100.00**	**100.00**	**100.00**
Sim-2C9P	PlasFlow	59.05	64.83	61.80	92.59
	GraphPlas	**60.98**	**66.43**	**63.59**	**100.00**
Sim-10C25P	PlasFlow	62.53	65.82	64.13	89.13
	GraphPlas	**73.11**	**81.97**	**77.28**	**96.38**
Sim-14C38P	PlasFlow	57.14	65.88	61.02	94.15
	GraphPlas	**60.66**	**76.37**	**67.61**	**95.12**
Wastewater-Plas	PlasFlow	71.47	70.95	71.21	80.40
	GraphPlas	**83.36**	**83.40**	**83.38**	**83.07**



(a) GraphPlas with PlasClass

(b) GraphPlas with PlasFlow

Fig. 6. Mean binning results of GraphPlas with original PlasFlow [22] and PlasClass [23] results for all the datasets.

4 Results and Discussion

4.1 Recovery of Plasmids from Metagenomics Assemblies

In this section, we present the results of GraphPlas on several assembled datasets including a real dataset. All the datasets were assembled using metaSPAdes [33] assembler. Table 2 and Table 3 show the comparison results with PlasClass and PlasFlow respectively. In Fig. 6 the performance values are summarised for GraphPlas, PlasClass and plasFlow. Furthermore, we show the results graphically in Fig. 2 and Fig. 3. In this diagram we indicate the starting

points for GraphPlas, which are essentially high confidence points on the PlasClass result. Starting from those points GraphPlas pushes towards increasing direction on both recall and precision. In more challenging scenarios with relatively lower precision values, GraphPlas improves the recall while maintaining the same precision.

4.2 Implementation and Performance

According to Table 2 and Table 3 it is evident that the utilisation of assembly graph with composition and coverage information improves the results of plasmid detection over conventional machine learning approaches. Furthermore, the significant compromise on recall to achieve higher precision is also mitigated in GraphPlas leading to better F1-scores.

The demonstrated improvements in GraphPlas prevail due to two main reasons. Firstly, we utilise the most confident classifications provided by the initialisation tools. Hence, at the starting point, the bootstrapping labels are more accurate. Secondly we employ the initial set of labels to train other contigs based on the composition, coverage and topology of the assembly graph. Note that it is highly likely for contigs of the same species to demonstrate a link in the assembly graph since there exists a path in the de Bruijn graph that completes the reference genome [35]. Therefore, GraphPlas is always capable of classifying contigs more confidently and reliably. Moreover, the shorter lengths of contigs do not affect the results significantly because of the connected contigs that are long and confidently classifiable. This is further evident through Fig. 5 where the labelling of contigs in each step are demonstrated. Finally, the actual number of contigs classified into each class are tabulated in Table 4. Note that the improvements over the real dataset is not significant since we have limited the ground truth computations only for contigs longer than 1000 bp. Hence, the classification of contigs shorter than the threshold are not considered.

GraphPlas consists of two main components as the initial classifier and the graph classifier. We have integrated the PlasClass classifier as the initial classifier. The entire program is implemented using python 3.6.7 and tested on a Intel Core i7-7700 CPU @ 3.60GHz $\times$ 8 machine with 16GB of RAM. The host operating system was Ubuntu 18.04.3 LTS. Multithreading capabilities are used for computing the tetramer frequency vectors and in the KNN classifier. The summary of resource utilisation is tabulated under Table 5. GraphPlas only considers the longer contigs for computation of plasmid probabilities. Hence, the memory utilisation for all the experiments were below 300MB. Furthermore, the GraphPlas algorithm completes within 4 minutes for all the datasets considered.

5 Conclusion and Future Work

In conclusion, GraphPlas proposes the ideology of incorporating the assembly graph in plasmid classification. We designed and evaluated GraphPlas which combines conventional machine learning tools with topological information from the assembly graph for the detection of plasmids. We also highlighted the importance of assembly graph and its potential to support in bootstrapping a dataset-specific model to address the problem of plasmid detection. The inclusion of assembly graph information to improve performance of the plasmid classification has room for further improvements. The faulty classifications in the initial seed contigs could mislead the label propagation degrading the overall performance. Furthermore, the coverage of contigs could be inaccurate, leading to misclassifications and hinder the label refinement.

In future, we intend to investigate the viability of using third-generation sequencing (TGS) data for the recovery of plasmid sequences. Furthermore, we intend to extend our approach on assemblies of TGS reads for metagenomics binning with recovery of plasmid sequences.

References

[1] C. M. Thomas and K. M. Nielsen, "Mechanisms of, and barriers to, horizontal gene transfer between bacteria," *Nature Reviews Microbiology*, vol. 3, no. 9, pp. 711–721, 2005. [Online]. Available: https://doi.org/10.1038/nrmicro1234

[2] A. Carattoli, "Plasmids and the spread of resistance," *International Journal of Medical Microbiology*, vol. 303, no. 6, pp. 298 – 304, 2013, special Issue Antibiotic Resistance. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1438422113000167

[3] K. Smalla, S. Jechalke, and E. M. Top, "Plasmid detection, characterization, and ecology," *Microbiology Spectrum*, vol. 3, no. 1, 2015. [Online]. Available: https://doi.org/10.1128/microbiolspec.plas-0038-2014

[4] T. Zhang, X.-X. Zhang, and L. Ye, "Plasmid metagenome reveals high levels of antibiotic resistance genes and mobile genetic elements in activated sludge," *PLOS ONE*, vol. 6, no. 10, pp. 1–7, 10 2011. [Online]. Available: https://doi.org/10.1371/journal.pone.0026041

[5] A.-D. Li, L.-G. Li, and T. Zhang, "Exploring antibiotic resistance genes and metal resistance genes in plasmid metagenomes from wastewater treatment plants," *Frontiers in Microbiology*, vol. 6, p. 1025, 2015. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmicb.2015.01025

[6] A. Bleicher, G. Schöfl, M. del Rosario Rodicio, and H. P. Saluz, "The plasmidome of a salmonella enterica serovar derby isolated from pork meat," *Plasmid*, vol. 69, no. 3, pp. 202 – 210, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0147619X13000024

[7] H. Birnboim and J. Doly, "A rapid alkaline extraction procedure for screening recombinant plasmid DNA," *Nucleic Acids Research*, vol. 7, no. 6, pp. 1513–1523, 11 1979. [Online]. Available: https://doi.org/10.1093/nar/7.6.1513

[8] S. Delaney, R. Murphy, and F. Walsh, "A comparison of methods for the extraction of plasmids capable of conferring antibiotic resistance in a human pathogen from complex broiler cecal samples," *Frontiers in Microbiology*, vol. 9, p. 1731, 2018. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmicb.2018.01731

[9] Alessandra Carattoli and Alessia Bertini and Laura Villa and Vincenzo Falbo and Katie L. Hopkins and E. John Threlfall, "Identification of plasmids by PCR-based replicon typing," *Journal of Microbiological Methods*, vol. 63, no. 3, pp. 219 – 228, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167701205001132

[10] S. Arredondo-Alonso, M. R. C. Rogers, J. C. Braat, T. D. Verschuuren, J. Top, J. Corander, R. J. L. Willems, and A. C. Schürch, "mlplasmids: a user-friendly tool to predict plasmid- and chromosome-derived sequences for single species," *Microbial Genomics*, vol. 4, no. 11, 2018. [Online]. Available: https://doi.org/10.1099/mgen.0.000224

[11] M. J. Bale, M. J. Day, and J. C. Fry, "Novel method for studying plasmid transfer in undisturbed river epilithon." *Applied and Environmental Microbiology*, vol. 54, no. 11, pp. 2756–2758, 1988. [Online]. Available: https://aem.asm.org/content/54/11/2756

[12] B. V. Jones and J. R. Marchesi, "Transposon-aided capture (traca) of plasmids resident in the human gut mobile metagenome," *Nature Methods*, vol. 4, no. 1, pp. 55–61, Jan 2007. [Online]. Available: https://doi.org/10.1038/nmeth964

TABLE 4
Contig classification by GraphPlas with PlasFlow as Initial Classifier.

Classification	Dataset	Total Number of Contigs	Plasmids Classified as Plasmids	Chromosomes Classified as Chromosomes	Plasmids Classified as Chromosomes	Chromosomes Classified as Plasmids
GraphPlas with PlasClass	Sim-2C5P	123	50	72	0	1
	Sim-2C9P	170	27	139	0	4
	Sim-10C25P	585	138	389	0	4
	Sim-14C38P	1800	169	1295	36	300
	Wastewater-Plas	953	334	435	115	69
GraphPlas with PlasFlow	Sim-2C5P	123	50	73	0	0
	Sim-2C9P	170	27	47	0	96
	Sim-10C25P	585	133	302	5	145
	Sim-14C38P	1800	195	919	10	676
	Wastewater-Plas	953	373	422	76	82

TABLE 5
Running time and Memory Consumed by GraphPlas (Using PlasClass as the initial classifier).

Classification	Dataset	Running time (s)	Peak Memory Usage (MB)
GraphPlas with PlasClass	Sim-2C5P	4	112
	Sim-2C9P	6	114
	Sim-10C25P	21	140
	Sim-14C38P	79	158
	Wastewater-Plas	77	196
GraphPlas with PlasFlow	Sim-2C5P	3	112
	Sim-2C9P	4	114
	Sim-10C25P	21	140
	Sim-14C38P	79	158
	Wastewater-Plas	80	200

[13] S. C. Forster, N. Kumar, B. O. Anonye, A. Almeida, E. Viciani, M. D. Stares, M. Dunn, T. T. Mkandawire, A. Zhu, Y. Shao, L. J. Pike, T. Louie, H. P. Browne, A. L. Mitchell, B. A. Neville, R. D. Finn, and T. D. Lawley, "A human gut bacterial genome and culture collection for improved metagenomic analyses," *Nature Biotechnology*, vol. 37, no. 2, pp. 186–192, 2019. [Online]. Available: https://doi.org/10.1038/s41587-018-0009-7

[14] D. Antipov, N. Hartwick, M. Shen, M. Raiko, A. Lapidus, and P. A. Pevzner, "plasmidSPAdes: assembling plasmids from whole genome sequencing data," *Bioinformatics*, vol. 32, no. 22, pp. 3380–3387, 07 2016. [Online]. Available: https://doi.org/10.1093/bioinformatics/btw493

[15] R. Rozov, A. Brown Kav, D. Bogumil, N. Shterzer, E. Halperin, I. Mizrahi, and R. Shamir, "Recycler: an algorithm for detecting plasmids from de novo assembly graphs," *Bioinformatics*, vol. 33, no. 4, pp. 475–482, 11 2016. [Online]. Available: https://doi.org/10.1093/bioinformatics/btw651

[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 Genetics*, vol. 10, no. 12, pp. 1–21, 12 2014. [Online]. Available: https://doi.org/10.1371/journal.pgen.1004766

[17] L. Vielva, M. de Toro, V. F. Lanza, and F. de la Cruz, "PLACNETw: a web-based tool for plasmid reconstruction from bacterial genomes," *Bioinformatics*, vol. 33, no. 23, pp. 3796–3798, 07 2017. [Online]. Available: https://doi.org/10.1093/bioinformatics/btx462

[18] A. Orlek, N. Stoesser, M. F. Anjum, M. Doumith, M. J. Ellington, T. Peto, D. Crook, N. Woodford, A. S. Walker, H. Phan, and A. E. Sheppard, "Plasmid classification in an era of whole-genome sequencing: Application in studies of antibiotic resistance epidemiology," *Frontiers in Microbiology*, vol. 8, p. 182, 2017. [Online]. Available: https://www.frontiersin.org/article/10.3389/fmicb.2017.00182

[19] D. Pellow, M. Probst, O. Furman, A. Zorea, A. Segal, I. Mizrahi, and R. Shamir, "Scapp: An algorithm for improved plasmid assembly in metagenomes," *bioRxiv*, 2020. [Online]. Available: https://www.biorxiv.org/content/early/2020/01/14/2020.01.12.903252

[20] F. Zhou and Y. Xu, "cBar: a computer program to distinguish plasmid-derived from chromosome-derived sequence fragments in metagenomics data," *Bioinformatics*, vol. 26, no. 16, pp. 2051–2052, 08 2010. [Online]. Available: https://doi.org/10.1093/bioinformatics/btq299

[21] A. Carattoli, E. Zankari, A. García-Fernández, M. Voldby Larsen, O. Lund, L. Villa, F. Møller Aarestrup, and H. Hasman, "In silico detection and typing of plasmids using plasmidfinder and plasmid multilocus sequence typing," *Antimicrobial Agents and Chemotherapy*, vol. 58, no. 7, pp. 3895–3903, 2014. [Online]. Available: https://aac.asm.org/content/58/7/3895

[22] P. S. Krawczyk, L. Lipinski, and A. Dziembowski, "PlasFlow: predicting plasmid sequences in metagenomic data using genome signatures," *Nucleic Acids Research*, vol. 46, no. 6, pp. e35–e35, 01 2018. [Online]. Available: https://doi.org/10.1093/nar/gkx1321

[23] D. Pellow, I. Mizrahi, and R. Shamir, "PlasClass improves plasmid sequence classification," *PLOS Computational Biology*, vol. 16, no. 4, pp. 1–9, 04 2020. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1007781

[24] O. Schwengers, P. Barth, L. Falgenhauer, T. Hain, T. Chakraborty, and A. Goesmann, "Platon: identification and characterization of bacterial plasmid contigs in short-read draft assemblies exploiting protein sequence-based replicon distribution scores," 2020. [Online]. Available: https://www.microbiologyresearch.org/content/journal/mgen/10.1099/mgen.0.000398

[25] V. Mallawaarachchi, A. Wickramarachchi, and Y. Lin, "GraphBin: refined binning of metagenomic contigs using assembly graphs," *Bioinformatics*, vol. 36, no. 11, pp. 3307–3313, 03 2020. [Online]. Available: https://doi.org/10.1093/bioinformatics/btaa180

[26] V. G. Mallawaarachchi, A. S. Wickramarachchi, and Y. Lin, "GraphBin2: Refined and Overlapped Binning of Metagenomic Contigs Using Assembly Graphs," in *20th International Workshop on Algorithms in Bioinformatics (WABI 2020)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), C. Kingsford and N. Pisanti, Eds., vol. 172. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, pp. 8:1–8:21. [Online]. Available: https://drops.dagstuhl.de/opus/volltexte/2020/12797

[27] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," School of Computer Science, Carnegie Mellon University, Tech. Rep., 2002.

[28] Y.-W. Wu, B. A. Simmons, and S. W. Singer, "MaxBin 2.0: an automated binning algorithm to recover genomes from multiple metagenomic datasets," *Bioinformatics*, vol. 32, no. 4, p. 605–607, Oct 2015. [Online]. Available: https://doi.org/10.1093/bioinformatics/btv638

[29] Z. Wang, Z. Wang, Y. Y. Lu, F. Sun, and S. Zhu, "SolidBin: improving metagenome binning with semi-supervised normalized cut," *Bioinformatics*, vol. 35, no. 21, pp. 4229–4238, 04 2019. [Online]. Available: https://doi.org/10.1093/bioinformatics/btz253

[30] A. Wickramarachchi, V. Mallawaarachchi, V. Rajan, and Y. Lin, "MetaBCC-LR: metagenomics binning by coverage and composition for long reads," *Bioinformatics*, vol. 36, no. Supplement_1, pp. i3–i11, 07 2020. [Online]. Available: https://doi.org/10.1093/bioinformatics/btaa441

[31] H. Gourlé, O. Karlsson-Lindsjö, J. Hayer, and E. Bongcam-Rudloff, "Simulating Illumina metagenomic data with InSilicoSeq," *Bioinformatics*, vol. 35, no. 3, pp. 521–522, 07 2018. [Online]. Available: https://doi.org/10.1093/bioinformatics/bty630
[32] Y. Shi, H. Zhang, Z. Tian, M. Yang, and Y. Zhang, "Characteristics of arg-carrying plasmidome in the cultivable microbial community from wastewater treatment system under high oxytetracycline concentration," *Applied Microbiology and Biotechnology*, vol. 102, no. 4, pp. 1847–1858, Feb 2018. [Online]. Available: https://doi.org/10.1007/s00253-018-8738-6
[33] S. Nurk, D. Meleshko, A. Korobeynikov, and P. A. Pevzner, "metaspades: a new versatile metagenomic assembler," *Genome Research*, vol. 27, no. 5, pp. 824–834, 2017. [Online]. Available: http://genome.cshlp.org/content/27/5/824.abstract
[34] H. Li, "Minimap2: pairwise alignment for nucleotide sequences," *Bioinformatics*, vol. 34, no. 18, pp. 3094–3100, 05 2018.
[35] P. E. C. Compeau, P. A. Pevzner, and G. Tesler, "How to apply de bruijn graphs to genome assembly," *Nature Biotechnology*, vol. 29, no. 11, pp. 987–991, Nov 2011. [Online]. Available: https://doi.org/10.1038/nbt.2023

Anuradha Wickramarachchi received his B.Sc (Hons) degree in computer science and engineering from University of Moratuwa, Sri Lanka. He is currently a Ph.D. student at the Australian National University, Australia. His research interests include computational biology and metagenomics.

Yu Lin received the Ph.D. degree in computer science from EPFL, Lausanne, Switzerland. He was a postdoctoral scholar at the Department of Computer Science and Engineering, University of California, San Diego. Currently, he is a Lecturer at the Research School of Computer Science, Australian National University. His research interests include algorithm design and computational biology.