UMR IRISA

Institut de Recherche en Informatique et Systèmes Aléatoires

Data Mining and Discrete Optimization For Strains Separation

Tam Truong,

Rumen Andonov (Supervisor), Roland Faure

Keywords: quasi-biclique, matrix completion, K-nearest neighbor imputation, hierarchical clustering, operations research, integer linear programing

Université de Rennes

Inria

GENSCALE

Aalto University

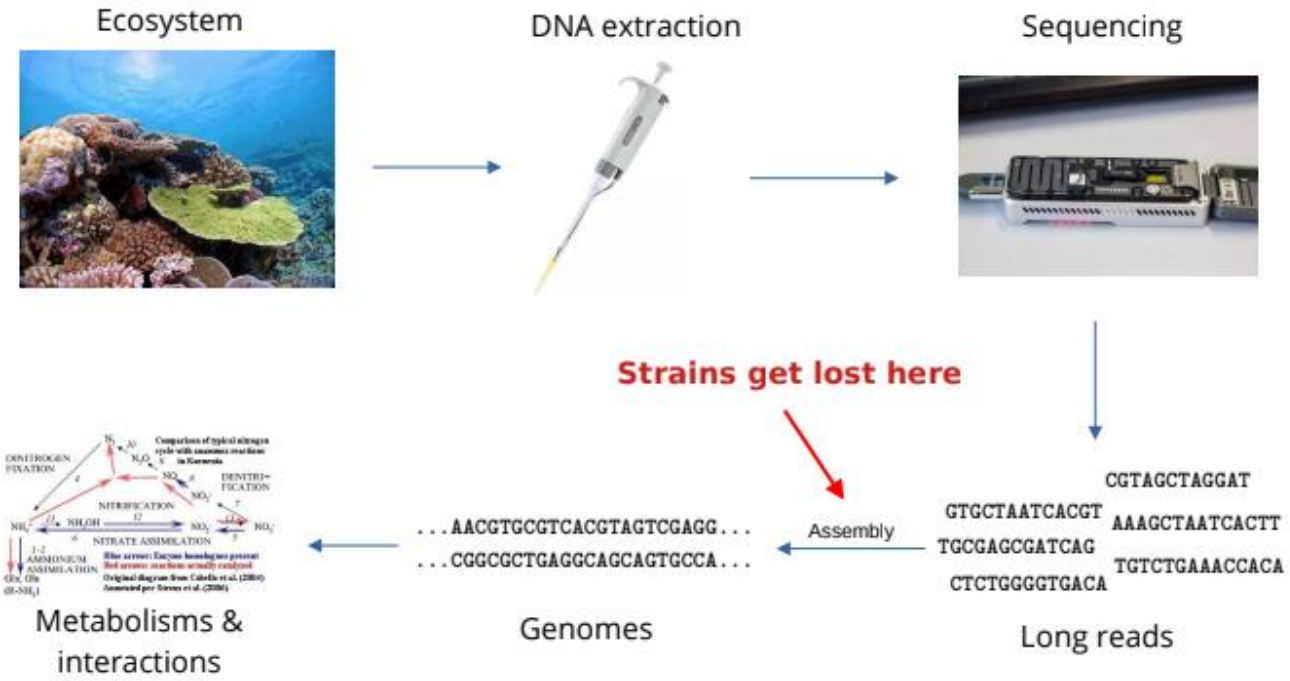
OUTLINE

- Problem and state of the art
- Pipeline:
  - Input data:
    - Binary matrix construction from assembly
    - Dealing with missing values: matrix completion with KNN-imputation
  - Bi-clustering:
    - Definition quasi-biclique and modeling with Integer linear programing
    - Finding a bipartition of reads.
  - Reads splitting:
    - Clustering reads from multiple bipartitions
- Testing and result

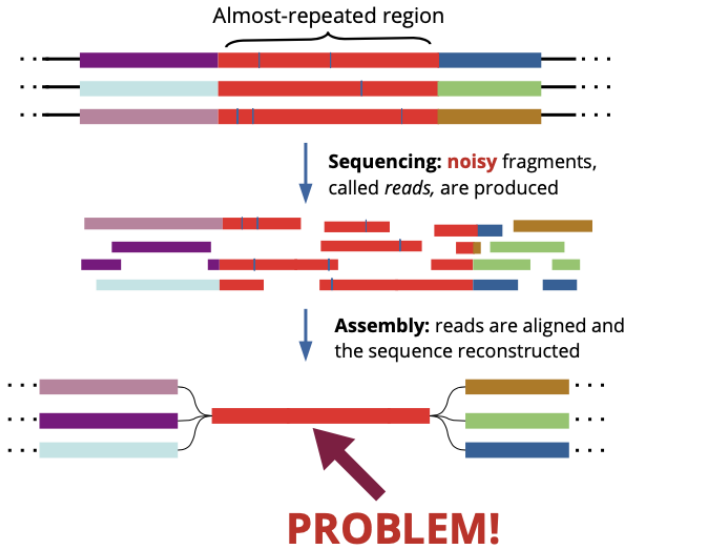
VOCABULARY

- Metagenomics: study of genetic material from the environment.
- Strain: genetic variants or subtypes of a species
- Read: a small section of DNA
- Contig: a set of DNA segments or sequences that overlap.

PROBLEM AND STATE OF THE ART



PROBLEM AND STATE OF THE ART



Similar regions get collapsed in a single sequence

Source: *Hairsplitter*

Multiple strains of the same species each with unique genetic variations.
Assemblers combine strains into one

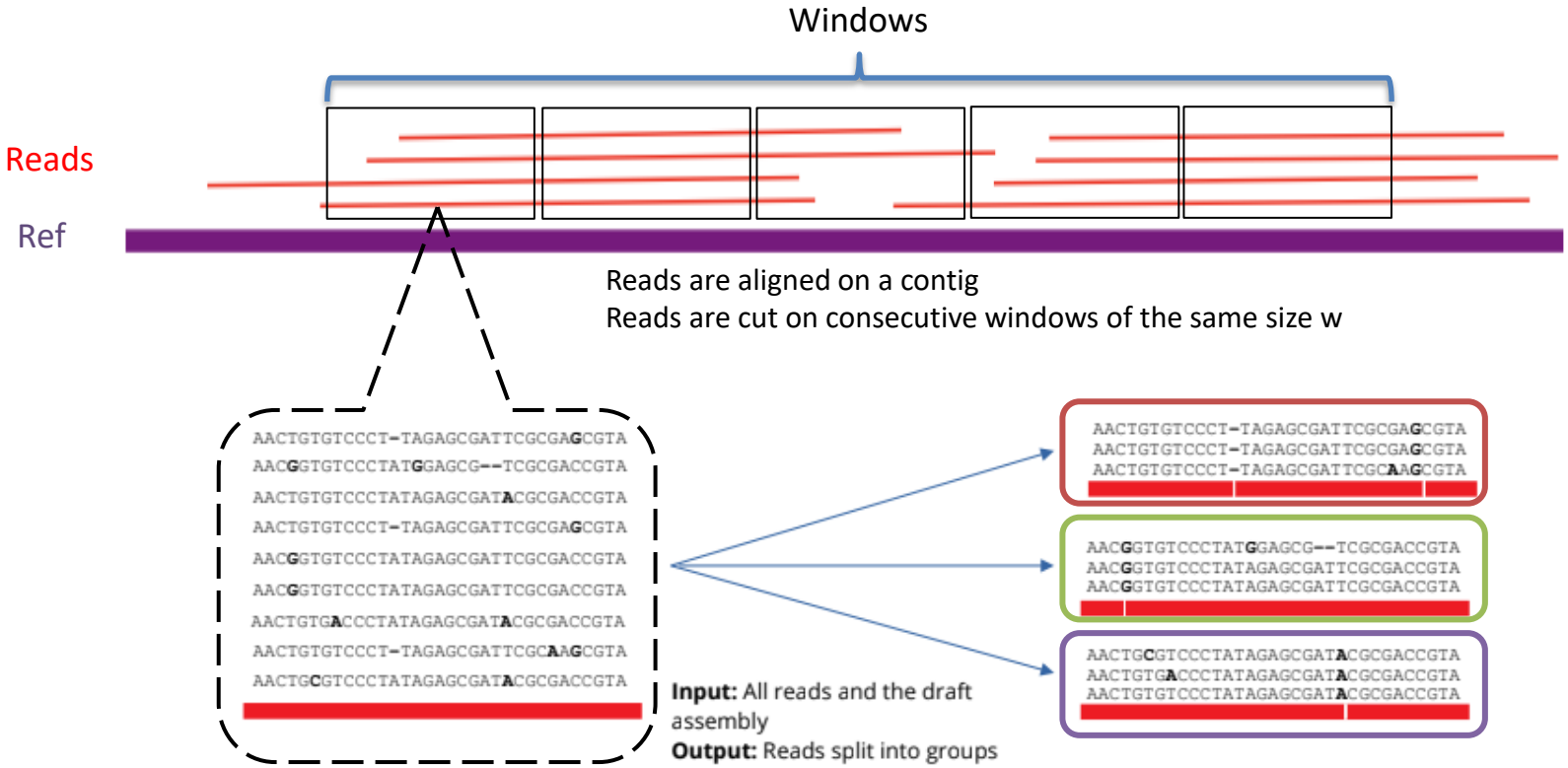
Challenges:

- Unknown and potentially high number of strains
- Unknown depth at which each strain is covered

Existing software: StrainBerry, HairSplitter, Strainy

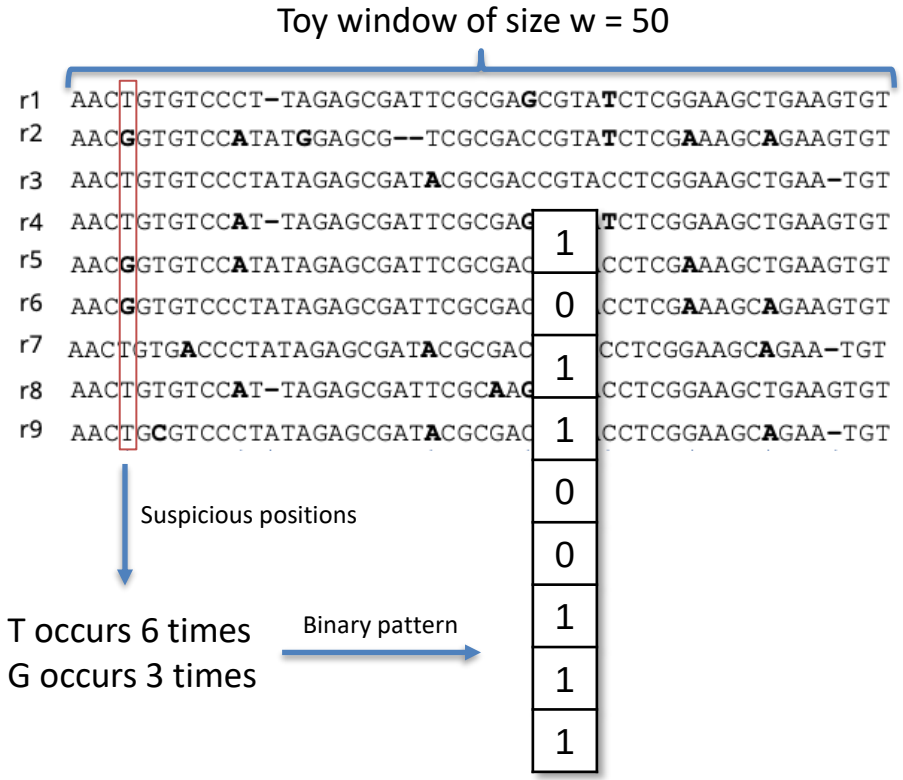
Our goal: Strains separation by combining data mining and discrete optimization techniques.

ASSEMBLY INTO DNA MATRICES



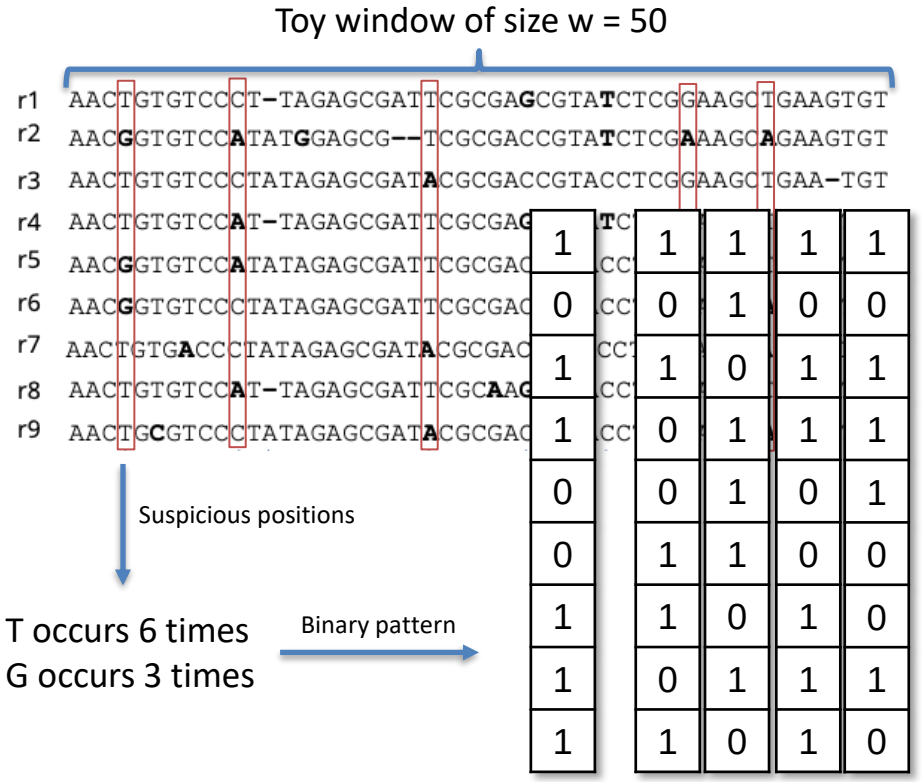
1ST STEP: DNA MATRIX TO BINARY MATRIX

- On a position:
  - Highest frequency base ➔ 1
  - Variants ➔ 0
  - Missing value ➔ NaN
- Create a binary matrix:
  - Positions as columns
  - Reads as rows



1ST STEP: DNA MATRIX TO BINARY MATRIX

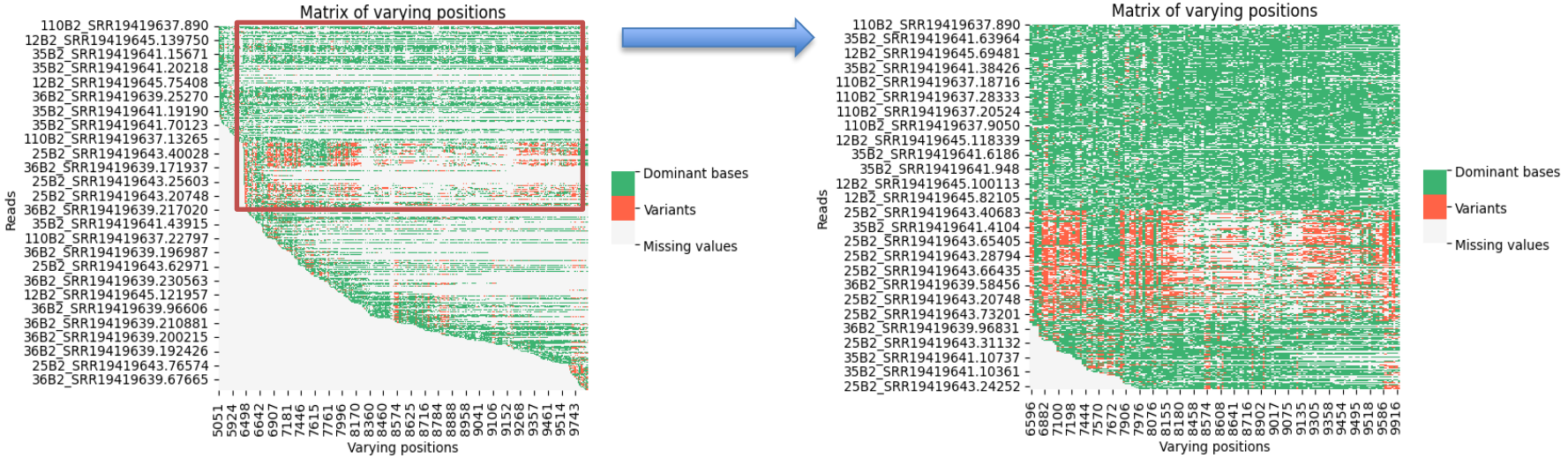
- On a position:
  - Highest frequency base ➔ 1
  - Variants ➔ 0
  - Missing value ➔ NaN
- Create a binary matrix:
  - Positions as columns
  - Reads as rows



1ST STEP: FROM DNA MATRICES TO BINARY MATRICES

Real data: Matrix created from Vagococcus assembly with window of size 5000

Remove shorter reads that span < 60% of the window

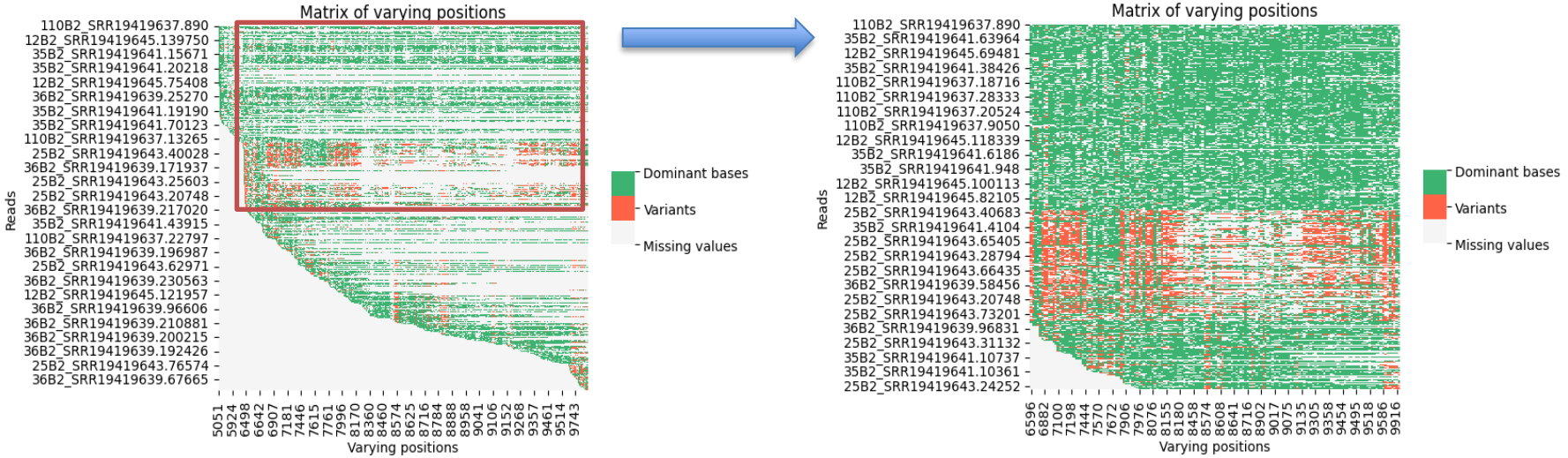


Around 300 suspicious positons left from a window of 5000 bases

1ST STEP: FROM DNA MATRICES TO BINARY MATRICES

Real data: Matrix created from Vagococcus assembly with window of size 5000

Remove shorter reads that span < 60% of the window



Around 300 suspicious positons left from a window of 5000 bases

HIGH RATE OF MISSING VALUES LEADS TO THE LOST OF PATTERN INFORMATION

2ND STEP: MATRIX COMPLETION TO FILL IN MISSING VALUES

Possible causes

Causes:

- Read length variation
- Sequencing errors

But:

✓ Matrix has a structure (reads from the same strains are exactly same)

➔ Missing values can be imputed using information from the matrix

2ND STEP: MATRIX COMPLETION TO FILL IN MISSING VALUES

K-Nearest Neighbors Imputation for Matrix Completion

For each row:

- Find the k-most similar neighbors.
- Calculate the average of the neighbors.
- Fill in the missing values using the average.

r1	1	NA	1	0
r2	1	0	1	0
r3	1	0	1	1
r4	0	1	0	NA
r5	0	1	0	1

2ND STEP: MATRIX COMPLETION TO FILL IN MISSING VALUES

Example

r1	1	NA	1	0
r2	1	0	1	0
r3	1	0	1	1
r4	0	1	0	NA
r5	0	1	0	1

Calculate distance of 2 vectors with missing values

$$d_{xy} = \sqrt{weight * squared\ distance\ from\ present\ coordinates}$$

$$weight = \frac{Total\ number\ of\ coordinates}{Number\ of\ present\ coordinates}$$

- *Let number of neighbors k = 2:*
- *We calculate the distance of r1 to all other reads:*
  - *dist_12 = sqrt((4/3)*(0+0+0)) = 0*
  - *dist_13 = sqrt((4/3)*(0+0+1)) = 4/3*
  - *dist_14 = sqrt((4/2)*(1+1)) = 4*
  - *dist_15 = sqrt((4/3)*(1+1+1)) = 4*
- *Neighbors of r1 = {r2,r3}*
  - *r2 = {1,0,1,0}, r3 = {1,0,1,1}*
  - *Mean of neighbors r2, r3 = {1,0,1,0.5}*
- *=> Fill the missing position in r1 with 0 from the mean of r2 and r3*

2ND STEP: MATRIX COMPLETION TO FILL IN MISSING VALUES

Example

r1	1	0	1	0
r2	1	0	1	0
r3	1	0	1	1
r4	0	1	0	1
r5	0	1	0	1

Calculate distance of 2 vectors with missing values

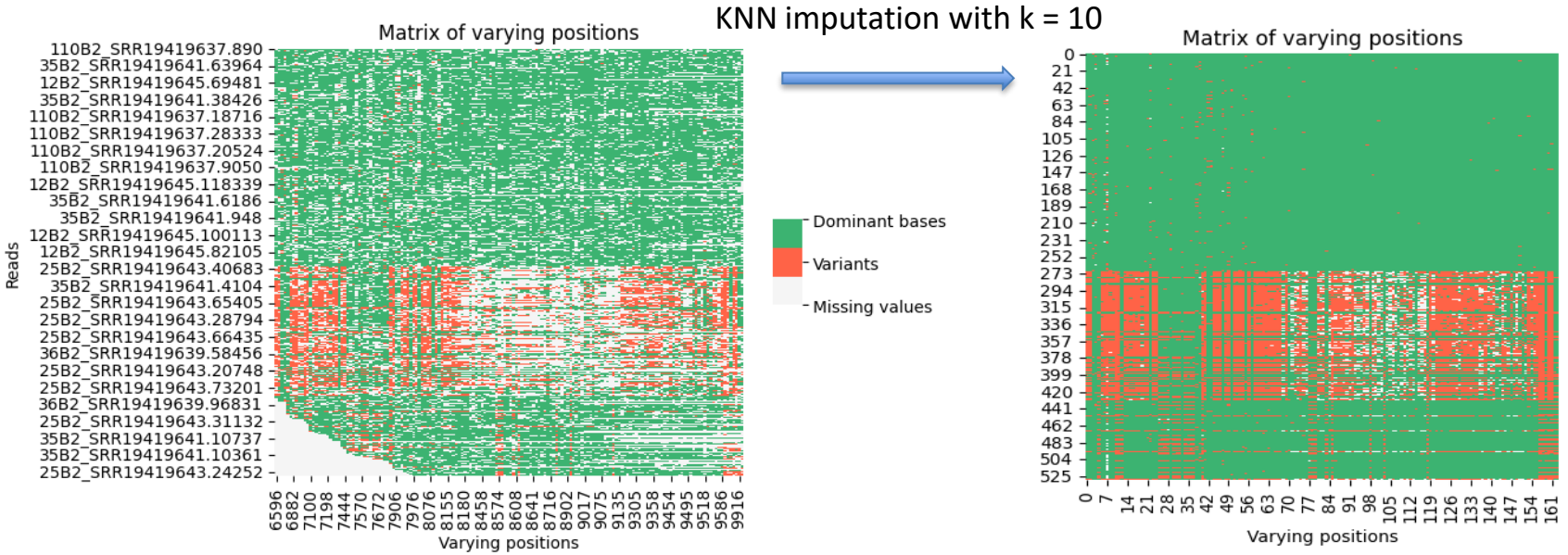
$$d_{xy} = \sqrt{weight * squared\ distance\ from\ present\ coordinates}$$

$$weight = \frac{Total\ number\ of\ coordinates}{Number\ of\ present\ coordinates}$$

- *Let number of neighbors k = 2:*
- *We calculate the distance of r1 to all other reads:*
  - *dist_12 = sqrt((4/3)*(0+0+0)) = 0*
  - *dist_13 = sqrt((4/3)*(0+0+1)) = 4/3*
  - *dist_14 = sqrt((4/2)*(1+1)) = 4*
  - *dist_15 = sqrt((4/3)*(1+1+1)) = 4*
- *Neighbors of r1 = {r2,r3}*
  - *r2 = {1,0,1,0}, r3 = {1,0,1,1}*
  - *Mean of neighbors r2, r3 = {1,0,1,0.5}*
- *=> Fill the missing position in r1 with 0 from the mean of r2 and r3*

2ND STEP: MATRIX COMPLETION TO FILL IN MISSING VALUES

Real data: Vagococcus



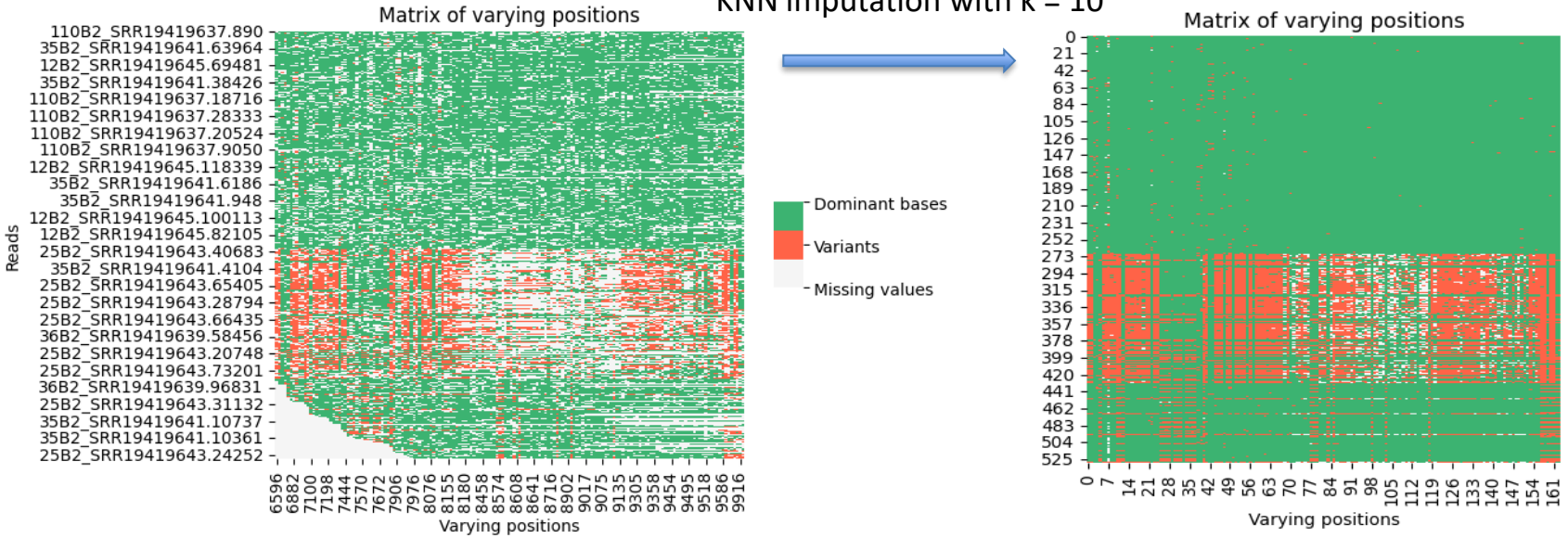
Initial matrix

Matrix after completion

2ND STEP: MATRIX COMPLETION TO FILL IN MISSING VALUES

Real data: Vagococcus

KNN imputation with k = 10



Initial matrix

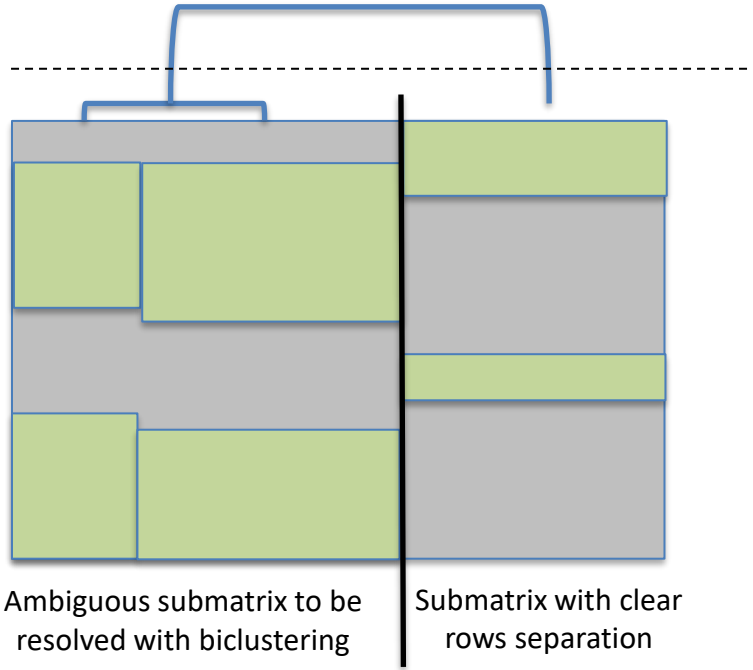
Matrix after completion

We want to separate the reads based on local dissimilarities

Matrix dimension could be substantial

3RD STEP: MATRIX SIZE REDUCTION BY HIERARCHICAL CLUSTERING BY COLUMNS

Divide the original matrix into distinct submatrices

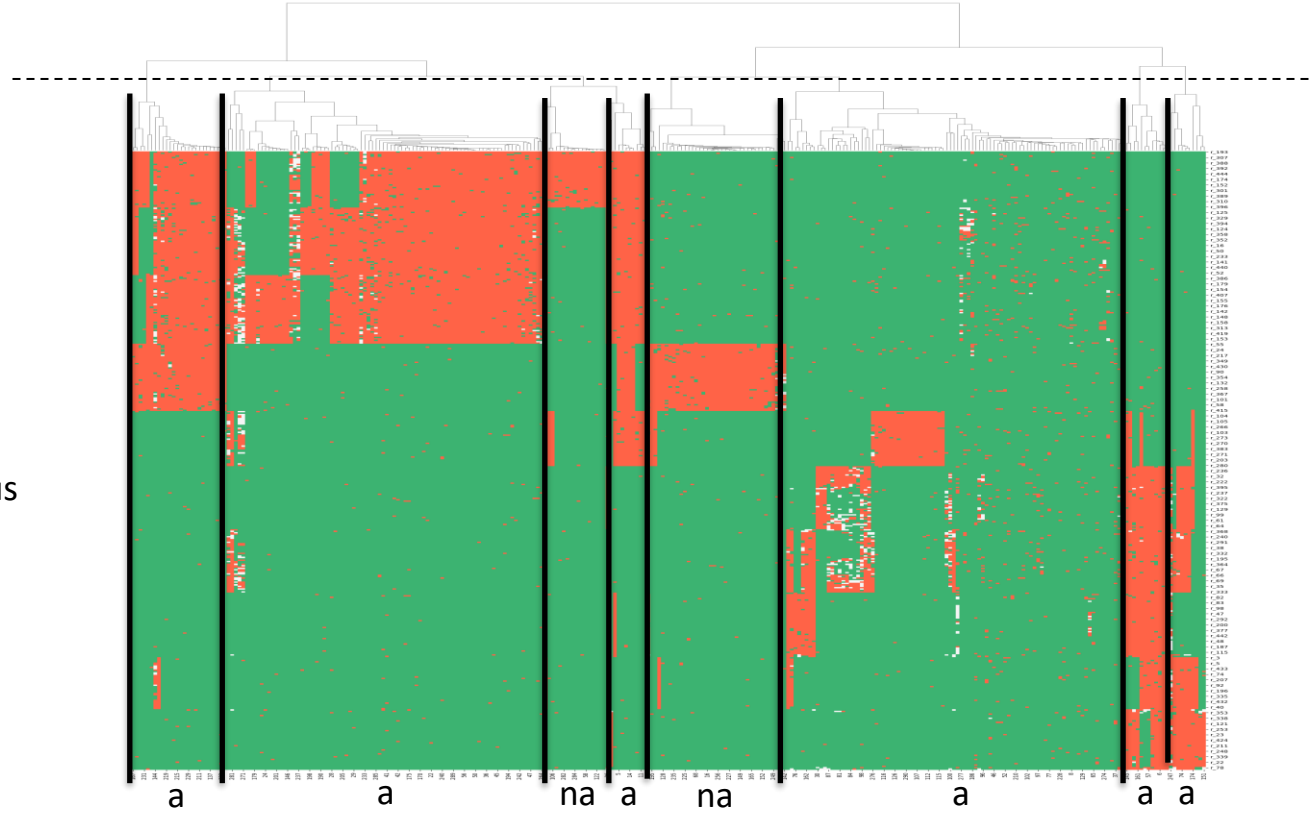


3RD DIVIDE AND CONQUER

Example:

a: ambiguous

na: not ambiguous



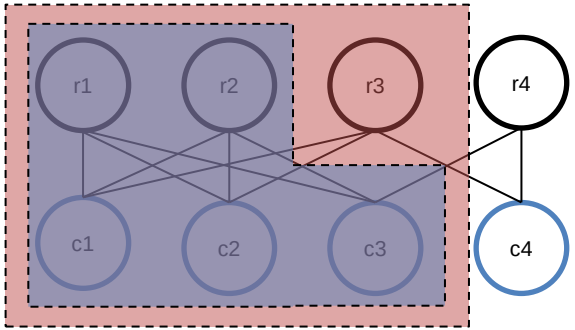
4TH STEP: BI-CLUSTERING TO RESOLVE AMBIGUITY

Matrix representation

	c1	c2	c3	c4
r1	1	1	1	0
r2	1	1	1	0
r3	1	1	0	1
r4	0	0	1	1

Simultaneous clustering of rows and columns.

Graph representation



Biclique: clique in bigraph

Quasi-biclique: almost complete subgraph.
Edge r3-c3 is missing.

➔ In order to tolerate errors, quasi-biclique is used instead of biclique

➔ Finding maximum quasi-biclique is NP-Hard

4TH STEP: BI-CLUSTERING TO RESOLVE AMBIGUITY

Identify Maximum sub-matrix of almost all 1

$$\max \sum_{i \in U} \sum_{j \in V} A_{i,j} x_{ij}, \quad (1)$$

$$1 - v_i \geq x_{ij}, \ \forall i \in U, \forall j \in V \quad (2)$$

$$1 - v_j \geq x_{ij}, \ \forall i \in U, \forall j \in V \quad (3)$$

$$1 - u_i - v_j \leq x_{ij}, \ \forall i \in U, \forall j \in V \quad (4)$$

$$\sum_{i \in U} \sum_{j \in V} (1 - A_{i,j}) x_{ij} \leq \epsilon \times \sum_{i \in U} \sum_{j \in V} x_{ij} \quad (5)$$

$$u_i, v_j \in \{0,1\}, \ x_{ij} \in \{0,1\} \ \forall i \in U, \ \forall j \in V \quad (6)$$

- A_ij = {1,0} ⇔ the coefficient of the cells in the matrix
- x_ij = {1,0} ⇔ whether the cell is selected
- (1) ⇔ find the largest submatrix of almost all 1
- (2),(3),(4) ⇔ the bi-cluster is a rectangle.
- (5) ⇔ an error rate of epsilon is allowed

r1	1	0	1	0
r2	1	0	1	0
r3	1	0	1	1
r4	0	1	0	1
r5	0	1	0	1

4TH STEP: BI-CLUSTERING TO RESOLVE AMBIGUITY

Identify Maximum sub-matrix of almost all 1

$$\max \sum_{i \in U} \sum_{j \in V} A_{i,j} x_{ij}, \quad (1)$$

$$1 - v_i \geq x_{ij}, \ \forall i \in U, \forall j \in V \quad (2)$$

$$1 - v_j \geq x_{ij}, \ \forall i \in U, \forall j \in V \quad (3)$$

$$1 - u_i - v_j \leq x_{ij}, \ \forall i \in U, \forall j \in V \quad (4)$$

$$\sum_{i \in U} \sum_{j \in V} (1 - A_{i,j}) x_{ij} \leq \epsilon \times \sum_{i \in U} \sum_{j \in V} x_{ij} \quad (5)$$

$$u_i, v_j \in \{0,1\}, \ x_{ij} \in \{0,1\} \ \forall i \in U, \ \forall j \in V \quad (6)$$

- A_ij = {1,0} ⇔ the coefficient of the cells in the matrix
- x_ij = {1,0} ⇔ whether the cell is selected
- (1) ⇔ find the largest submatrix of almost all 1
- (2),(3),(4) ⇔ the bi-cluster is a rectangle.
- (5) ⇔ an error rate of epsilon is allowed

r1	1	1	0	0
r2	1	1	0	0
r3	1	1	0	1
r4	0	0	1	1
r5	0	0	1	1

Bi-cluster without errors i.e. Biclique

1st possible bi-cluster computed without errors

4TH STEP: BI-CLUSTERING TO RESOLVE AMBIGUITY

Identify Maximum sub-matrix of almost all 1

$$\max \sum_{i \in U} \sum_{j \in V} A_{i,j} x_{ij}, \quad (1)$$

$$1 - v_i \geq x_{ij}, \ \forall i \in U, \forall j \in V \quad (2)$$

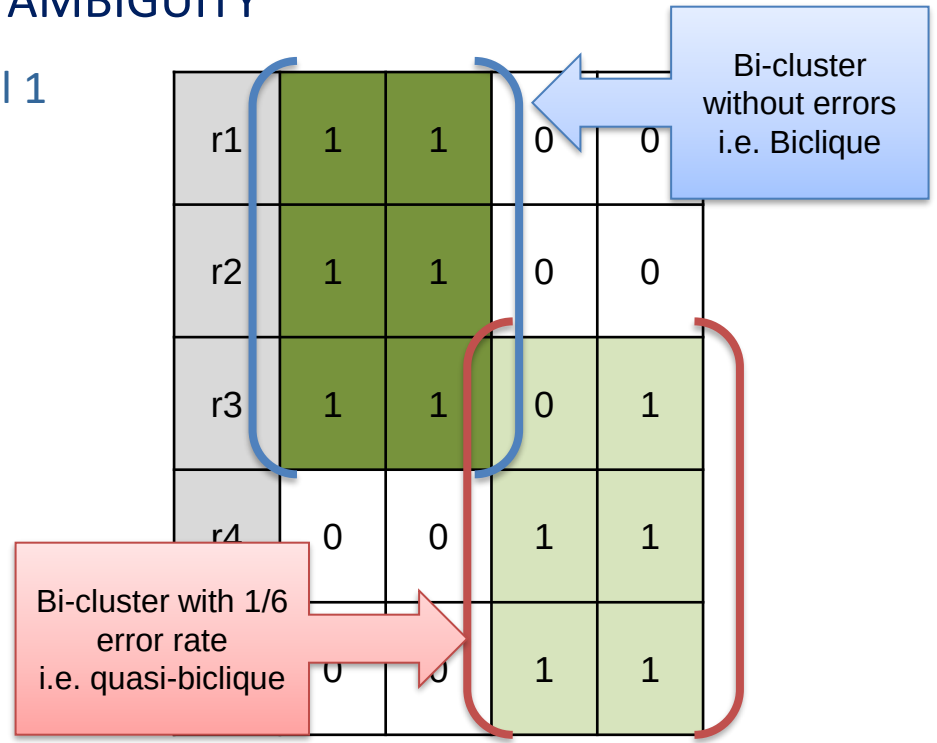
$$1 - v_j \geq x_{ij}, \ \forall i \in U, \forall j \in V \quad (3)$$

$$1 - u_i - v_j \leq x_{ij}, \ \forall i \in U, \forall j \in V \quad (4)$$

$$\sum_{i \in U} \sum_{j \in V} (1 - A_{i,j}) x_{ij} \leq \epsilon \times \sum_{i \in U} \sum_{j \in V} x_{ij} \quad (5)$$

$$u_i, v_j \in \{0,1\}, \ x_{ij} \in \{0,1\} \ \forall i \in U, \ \forall j \in V \quad (6)$$

- A_ij = {1,0} ⇔ the coefficient of the cells in the matrix
- x_ij = {1,0} ⇔ whether the cell is selected
- (1) ⇔ find the largest submatrix of almost all 1
- (2),(3),(4) ⇔ the bi-cluster is a rectangle.
- (5) ⇔ an error rate of epsilon is allowed

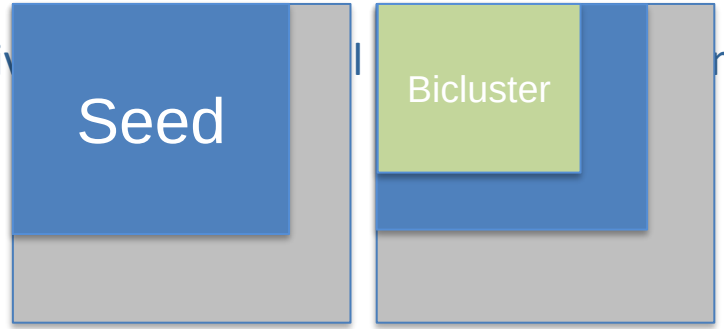


2nd possible bi-clusters computed with errors

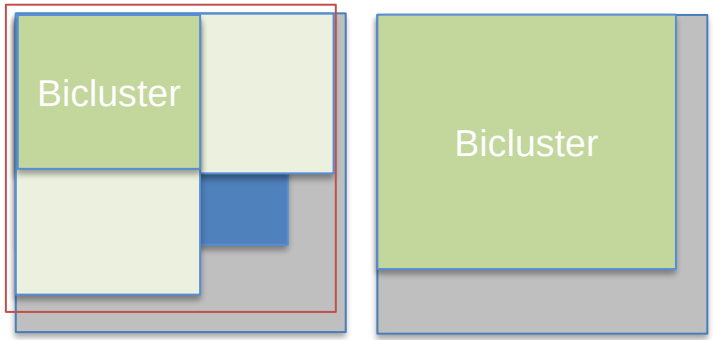
IRISA

4TH STEP: BI-CLUSTERING TO RESOLVE AMBIGUITY

Improve runtime by giv l m start or hint).



Select a dense region and look for first bicluster



Enrichment the initial solution by rows and columns to obtain the final cluster

4TH STEP: BI-CLUSTERING TO RESOLVE AMBIGUITY

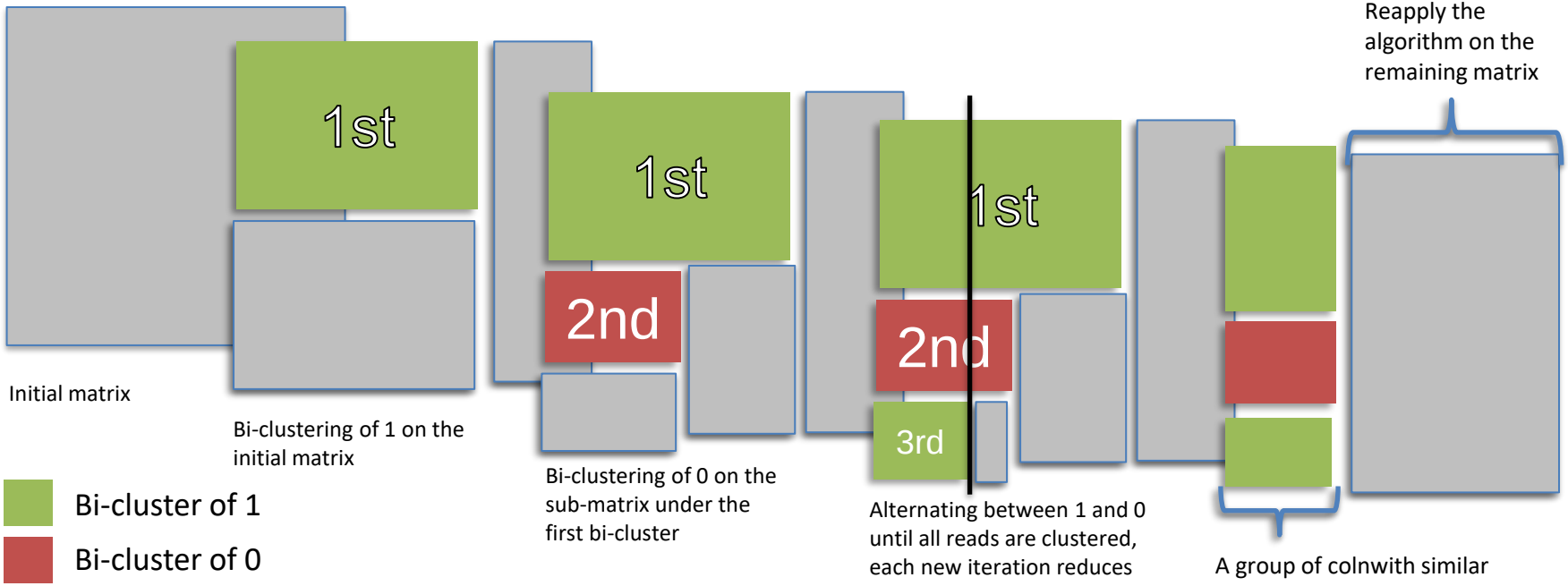
Bipartition matrix

r1	1	1	0	0
r2	1	1	0	0
r3	1	1	0	1
r4	0	0	1	1
r5	0	0	1	1

A matrix that can be separated into 2 regions: region of 1s and region of 0s.

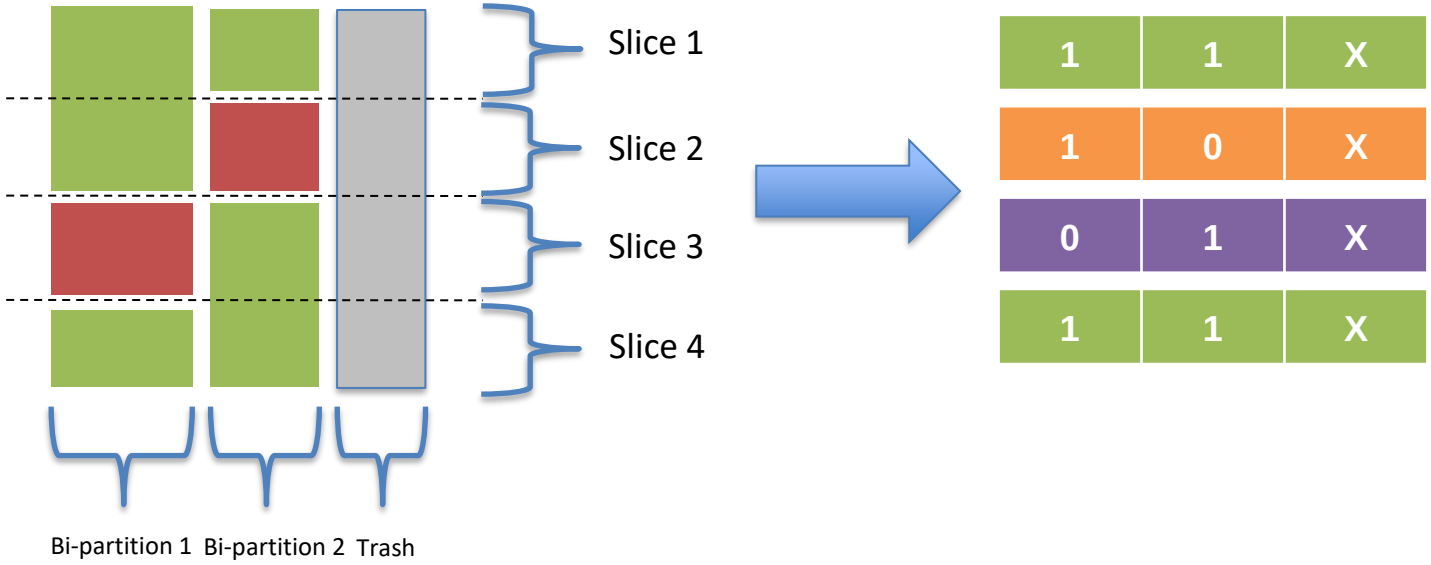
5TH STEP: FINDING A PATTERN/PARTIAL DIVERGENCES/BIPARTITION

A vertical step: Bi-partitioning of the matrix by maximum bi-clusters computation



6TH STEP: FINDING ALL SIGNIFICANT PARTIAL DIVERGENCES/PATTERNS

Reads Splitting



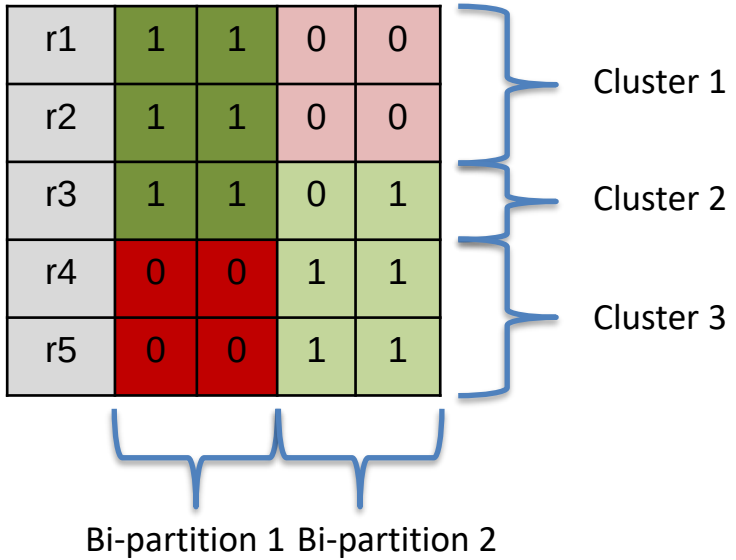
All significant patterns are separated into groups

Identical slices are colored the same

6TH STEP: FINDING ALL SIGNIFICANT PARTIAL DIVERGENCES/PATTERNS

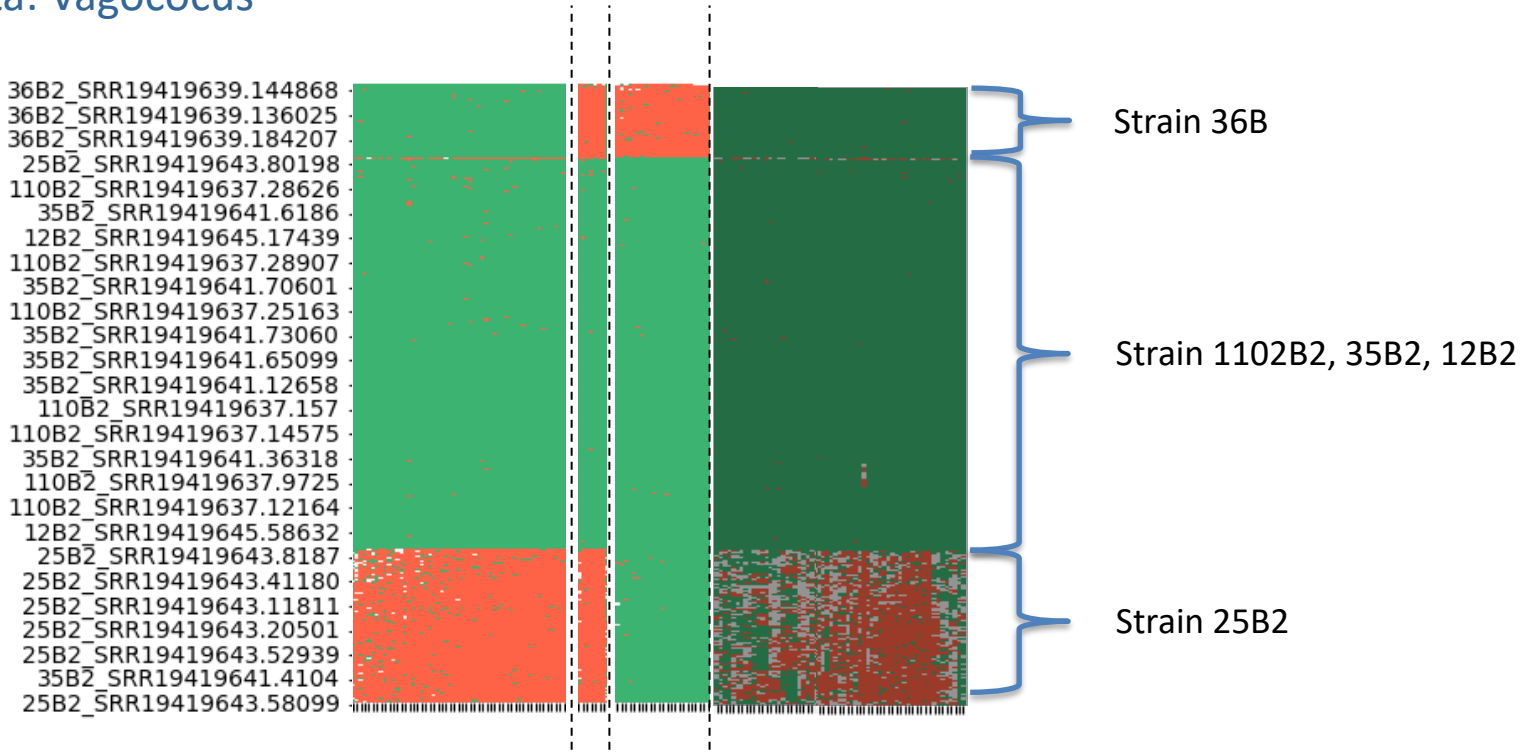
Example

- Start with all the reads as 1 cluster.
  - C = { r1, r2, r3, r4,r5}
- Splitting with the first bi-partition:
  - C_1 = {r1, r2, r3}
  - C_2 = {r4, r5}
- Splitting with the second bi-partition:
  - C_1_1 = {r1, r2}
  - C_1_2 = {r3}
  - C_2 = {r4, r5}



READS SPLITTING

Real data: Vagococcus

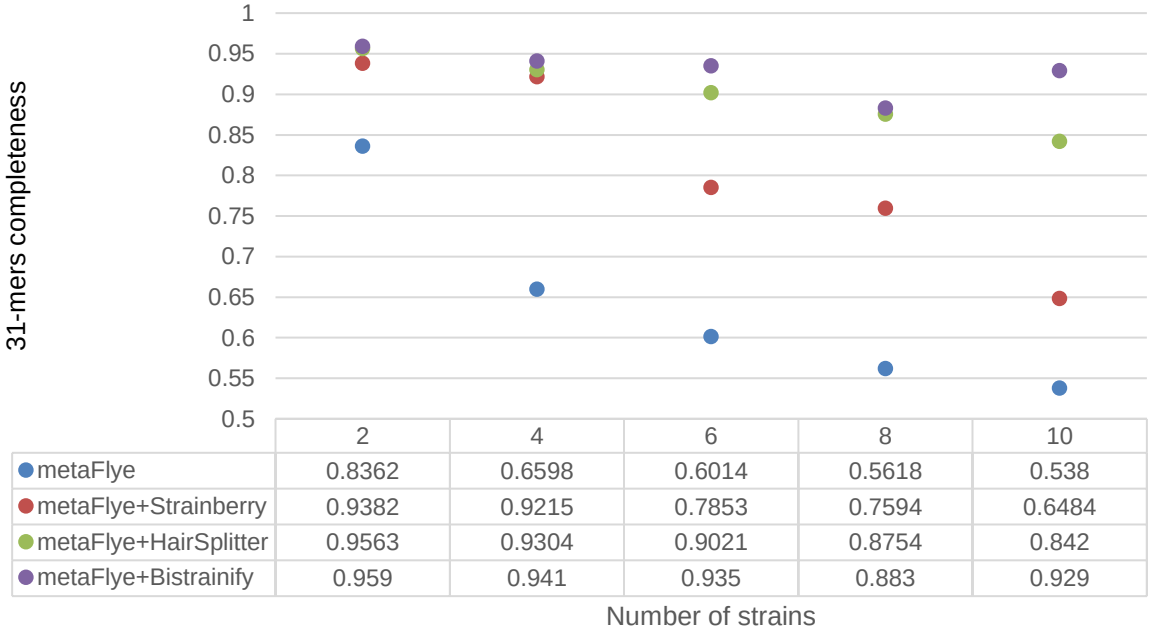


TESTING & RESULTS

Evaluation metric

- The proportion of 31-mers found in the result assembly that are found in the genome.
- Influence factors:
  - Divergence: How different are the strains
  - Coverage: The depth at which each strain is covered
  - Number of strain

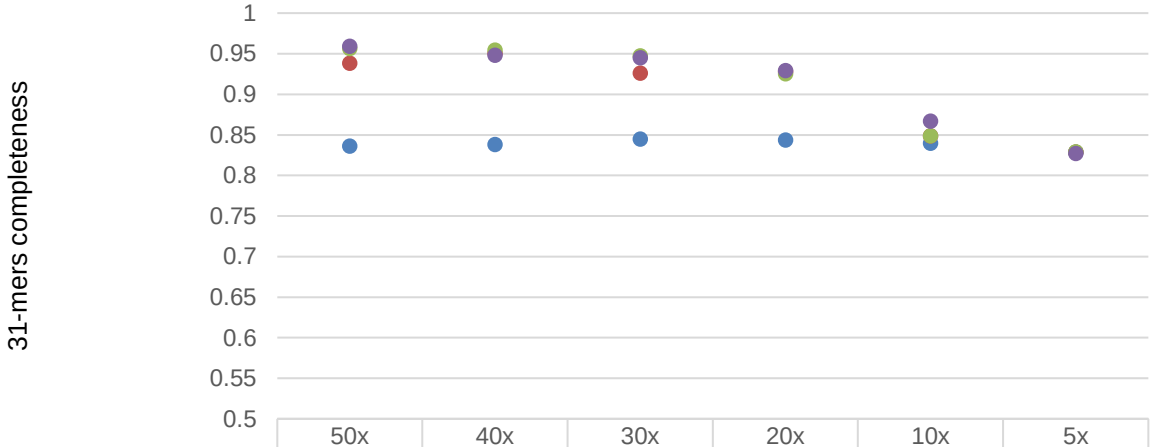
NUMBER OF STRAINS



	2	4	6	8	10
metaFlye	0.8362	0.6598	0.6014	0.5618	0.538
metaFlye+Strainberry	0.9382	0.9215	0.7853	0.7594	0.6484
metaFlye+HairSplitter	0.9563	0.9304	0.9021	0.8754	0.842
metaFlye+Bistrainify	0.959	0.941	0.935	0.883	0.929

Completeness of reconstruction remains close to 1 even with high number of strains

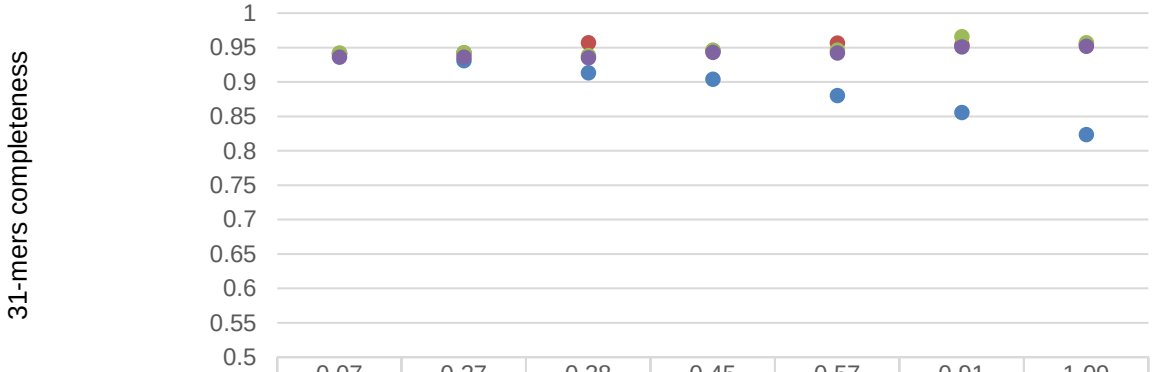
UNEVEN DOWNSAMPLING COVERAGE



	50x	40x	30x	20x	10x	5x
metaFlye	0.8362	0.8379	0.8448	0.8436	0.8394	0.8285
metaFlye+Strainberry	0.9382	0.9504	0.9258	0.9279	0.8486	0.8286
metaFlye+HairSplitter	0.9563	0.9545	0.9473	0.9253	0.8488	0.8289
metaFlye+Bistrainify	0.959	0.948	0.945	0.929	0.867	0.827

Needs at least 20x coverage to be effective

DEGREE OF DIVERGENCE BETWEEN STRAINS



	0,07 ME8067	0,27 LD27-1	0,38 Y5	0,45 EC590	0,57 RM14721	0,91 AMSCJX03	1,09 H5
metaFlye	0.941	0.9308	0.9134	0.9037	0.88	0.8557	0.8235
metaFlye+Strainberry	0.941	0.9418	0.9568	0.9454	0.9566	0.9538	0.9551
metaFlye+HairSplitter	0.9422	0.9424	0.9383	0.9464	0.9462	0.9659	0.9567
metaFlye+Bistrainify	0.936	0.936	0.935	0.943	0.942	0.951	0.952

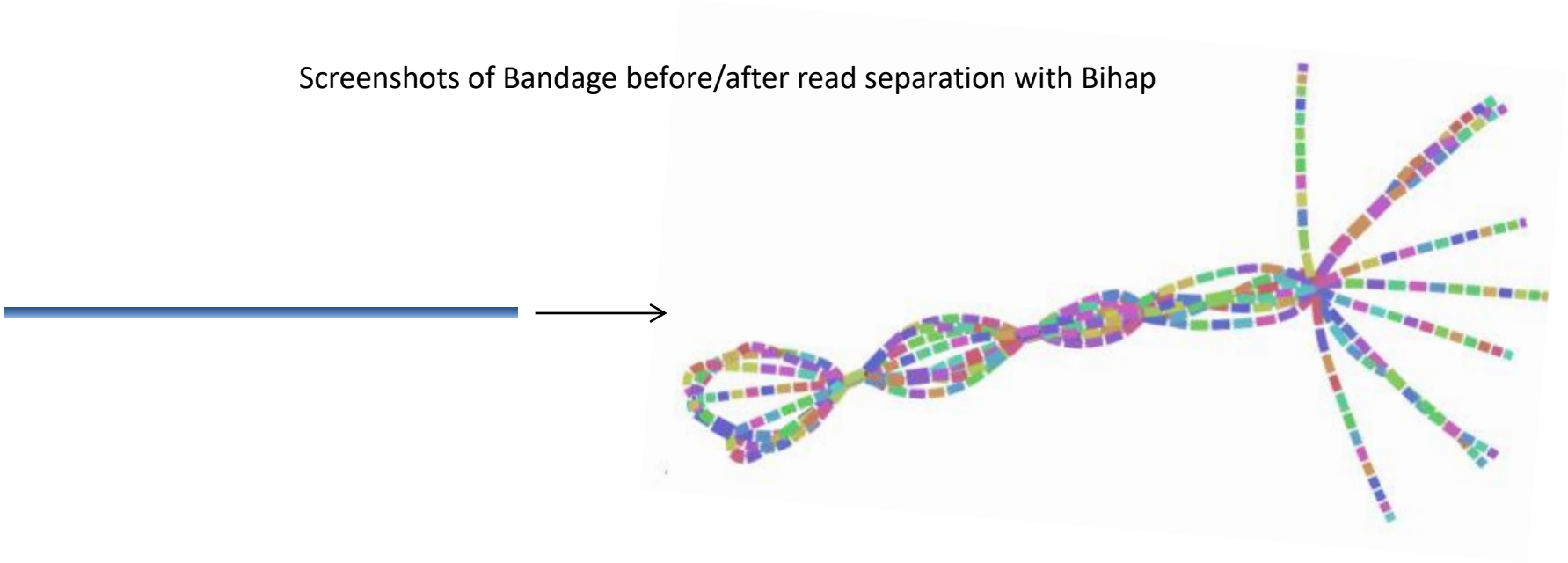
Strain mixed with K12 (ANI divergence)

Comparable sensitivity with Strainberry and HairSplitter

VISUALIZATION

Mix of 10 E. coli strains (simulated Nanopore sequencing)

Screenshots of Bandage before/after read separation with Bihap



RUNTIME (S)

Performance on whole dataset, using GUROBI or CBC Solver

Data Set	No of strains	Runtime (GUROBI)	Runtime (CBC)
E.coli	2	2004.12	2256.9
E.coli	4	4264.22	4893.7
E.coli	6	6894.61	11743.9
E.coli	8	11041.35	22373.5
E.coli	10	17540.87	54615.33
E.coli Hifi	3	1648.38	2238.19
Vagococus	3	5577.22	---

RUNTIME (S)

On individual matrix, with and without divide and conquer

Reads	Positions	Nb of strains	Approach	GUROBI	CBC
450	292	10	D&C	14.2	118.6
			Full ILP	5313.7	---
378	265	8	D&C	4.91	7.07
			Full ILP	2009.21	---
253	143	6	D&C	5.37	10.43
			Full ILP	745.87	1096.67
180	175	4	D&C	0.14	628.92
			Full ILP	0.14	2947.17

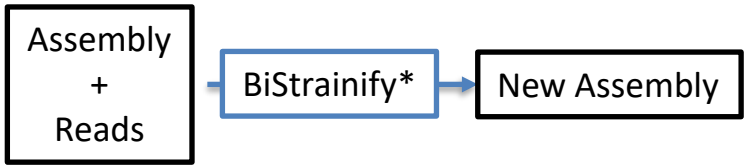
We obtained the same result on the tested instances

PARAMETERS

Param	Value used	Description	User input
Epsilon	0.025	The error rate tolerated for each bicluster	Y
Min_col_num	3	The minimum number of column for a bipartition to be accepted	Y
Window size	5000	The size of a single window	Y
Distance threshold	0.35	Threshold used for divide and conquer step	N
Suspicious position	0.95	The portion of the most popular base, if < 95% then it is suspicious	N
K	10	The number of neighbors for the imputation step	N

CONCLUSION

- High strain separation accuracy even in dataset with high number of strains.
- Our aproach's results confirm its limitations with low coverage.
- It is also not more sensitive than Strainberry or HairSplitter in low heterozygote areas.
- Perspective:
  - Acceleratation through inherent parallelism (each window computaion is independent)
  - Divide the computations in even more narrow matrices (to accelerate ILP without using optimality)
  - Integrate into Hairsplitter to have and end-to-end tool



*we are still working on a name

FOLLOW US!

www.irisa.fr

@irisa_lab

irisa-lab

Institut de Recherche en Informatique et Systèmes Aléatoires