Extracting genomic k-mers in long reads

Pierre Morisse, Thierry Lecroq and Arnaud Lefebvre
pierre.morisse2@univ-rouen.fr

Laboratoire d'Informatique, de Traitement de l'Information et des Systèmes

December 7, 2017

Plan

Outline

- Most recent long reads error correction tools rely on de Bruijn graphs
- For hybrid correction, the DBG is easily built from the short reads' k-mers (LoRDEC [Salmela and Rivals, 2014], Jabba [Miclotte et al., 2016], HG-CoLoR [Morisse et al., 2017])
- For self-correction, the DBG is either built:
  1. From very short k-mers ($k \simeq 8$) of long reads' local alignments (Daccord [Tischler and Myers, 2017])
  2. From short, frequent k-mers ($k \simeq 19$) of the long reads (LoRMA [Salmela et al., 2017])

Outline

- Most recent long reads error correction tools rely on de Bruijn graphs
- For hybrid correction, the DBG is easily built from the short reads' k-mers (LoRDEC [Salmela and Rivals, 2014], Jabba [Miclotte et al., 2016], HG-CoLoR [Morisse et al., 2017])
- For self-correction, the DBG is either built:
  1. From very short k-mers ($k \simeq 8$) of long reads' local alignments (Daccord [Tischler and Myers, 2017])
  2. From short, frequent k-mers ($k \simeq 19$) of the long reads (LoRMA [Salmela et al., 2017])

Outline

- Most recent long reads error correction tools rely on de Bruijn graphs
- For hybrid correction, the DBG is easily built from the short reads' k-mers (LoRDEC [Salmela and Rivals, 2014], Jabba [Miclotte et al., 2016], HG-CoLoR [Morisse et al., 2017])
- For self-correction, the DBG is either built:
  1. From very short k-mers ($k \simeq 8$) of long reads' local alignments (Daccord [Tischler and Myers, 2017])
  2. From short, frequent k-mers ($k \simeq 19$) of the long reads (LoRMA [Salmela et al., 2017])

Outline

- Most recent long reads error correction tools rely on de Bruijn graphs
- For hybrid correction, the DBG is easily built from the short reads' k-mers (LoRDEC [Salmela and Rivals, 2014], Jabba [Miclotte et al., 2016], HG-CoLoR [Morisse et al., 2017])
- For self-correction, the DBG is either built:
  1. From very short k-mers ($k \simeq 8$) of long reads' local alignments (Daccord [Tischler and Myers, 2017])
  2. From short, frequent k-mers ($k \simeq 19$) of the long reads (LoRMA [Salmela et al., 2017])

Outline

- Most recent long reads error correction tools rely on de Bruijn graphs
- For hybrid correction, the DBG is easily built from the short reads' k-mers (LoRDEC [Salmela and Rivals, 2014], Jabba [Miclotte et al., 2016], HG-CoLoR [Morisse et al., 2017])
- For self-correction, the DBG is either built:
  1. From very short k-mers ($k \simeq 8$) of long reads' local alignments (Daccord [Tischler and Myers, 2017])
  2. From short, frequent k-mers ($k \simeq 19$) of the long reads (LoRMA [Salmela et al., 2017])

Outline

Problems

- Computing local alignments is time and memory consuming
- On Oxford Nanopore data, even short, frequent *k*-mers have a high chance to be erroneous

Our aim

Extract genomic *k*-mers from a set long reads, without aligning them

Outline

Problems

- Computing local alignments is time and memory consuming
- On Oxford Nanopore data, even short, frequent k-mers have a high chance to be erroneous

Our aim

Extract genomic k-mers from a set long reads, without aligning them

Datasets

Dataset	Reference genome			Oxford Nanopore data			
	Strain	Reference sequence	Genome size	# Reads	Average length	Coverage	Error rate
A. baylyi	ADP1	CR543861	3.6 Mbp	89,011	4,284	106x	30%
E. coli	K-12 substr. MG1655	NC_000913	4.6 Mbp	22,270	5,999	29x	20%
S. cerevisae	S288C	NC_001133-001148	12.2 Mbp	205,923	5,698	96x	44%

Classical k-mers

Two problems

- Impossible to retrieve all the *k*-mers from the reference genome
- Impossible to discriminate genomic and erroneous *k*-mers

Classical k-mers

Results

With 16-mers from *A. baylyi* (3,525,881 in the genome):

Frequency	16-mers	Genomic 16-mers	Erroneous 16-mers
≥ 1	300,818,977	3,328,299	297,490,678
≥ 3	10,508,757	2,998,219	7,510,538
≥ 5	3,865,005	2,534,737	1,330,268
≥ 10	1,461,662	1,265,323	196,339

Classical k-mers

Results

With 16-mers from *E. coli* (4,513,330 in the genome):

Frequency	16-mers	Genomic 16-mers	Erroneous 16-mers
≥ 1	102,818,754	4,235,462	98,583,292
≥ 3	5,202,351	3,427,611	1,774,740
≥ 5	2,721,907	2,721,907	361,452
≥ 10	531,549	508,657	22,892

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide

A G T A G G A T C T

  2. Insertion of a nucleotide

A G T A G G A T C T

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide

A G T A G G A T C T

2. Insertion of a nucleotide

A G T A G G A T C T

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide

  2. Insertion of a nucleotide

Spaced k-mers

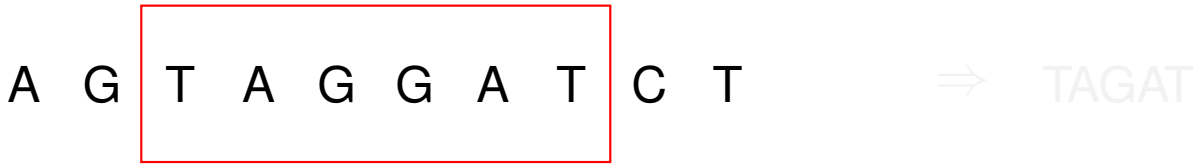
- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide

A G T A G G A T C T ⇒ TAGAT

  2. Insertion of a nucleotide

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide



  2. Insertion of a nucleotide

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide*; pattern: 111011*

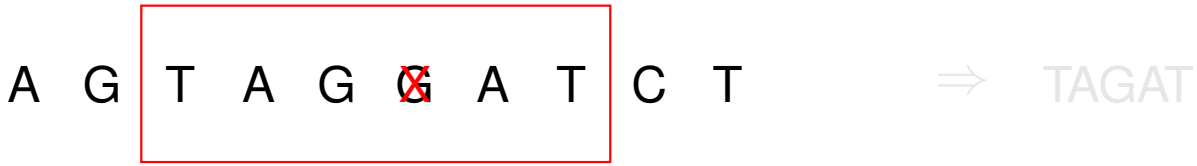
A G T A G G A T C T ⇒ TAGAT

  2. Insertion of a nucleotide *pattern: 111011*

A G T A G G A T C T ⇒ TAGGAT, TAGCAT, TAGTAT, TAGAAT

Spaced k-mers

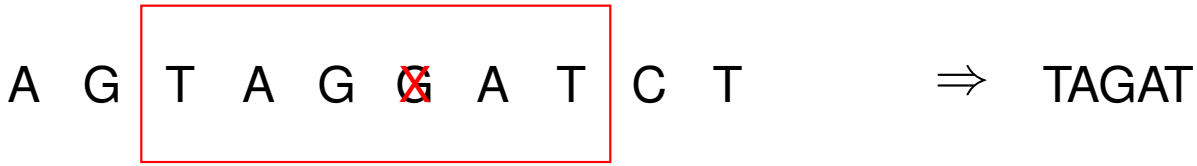
- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide*; pattern: 111011*



  2. Insertion of a nucleotide *pattern: 111011*

Spaced k-mers

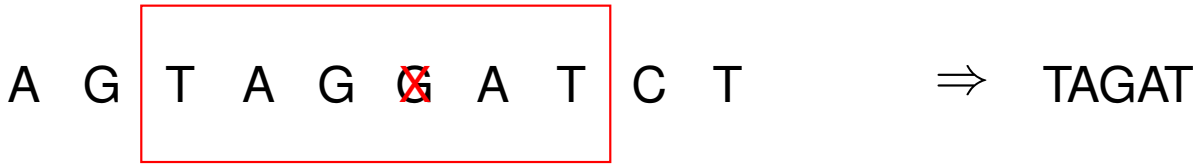
- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide*; pattern: 111011*



  2. Insertion of a nucleotide *pattern: 111011*

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:
  1. Deletion of a nucleotide*; pattern: 111011*



  2. Insertion of a nucleotide*; pattern: 1110111*

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*

A G T A G ~~G~~ A T C T ⇒ TAGAT

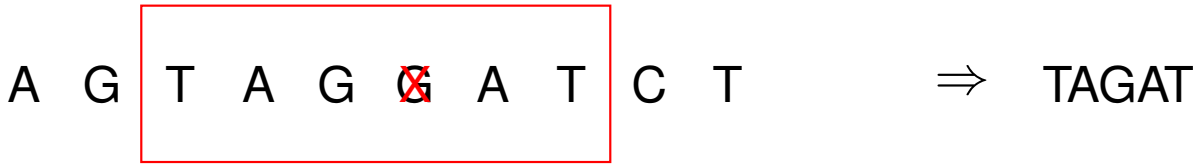
2. Insertion of a nucleotide*; pattern: 1110111*

A G T A G G A T C T ⇒ TAGAGAT, TAGCGAT, TAGGGAT, TAGTGAT

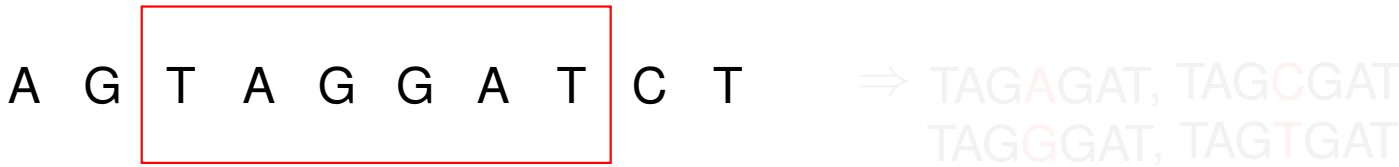
Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*



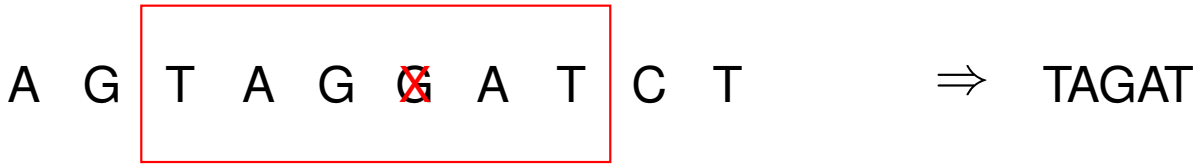
2. Insertion of a nucleotide*; pattern: 1110111*



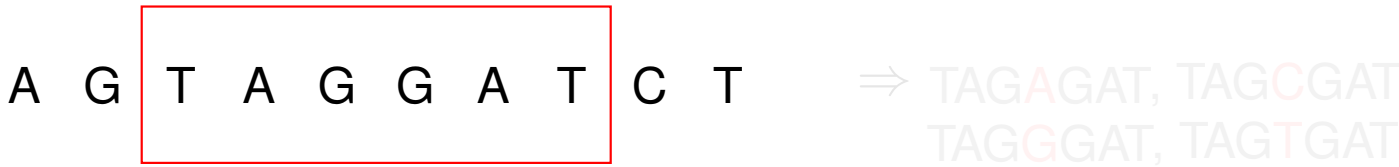
Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*



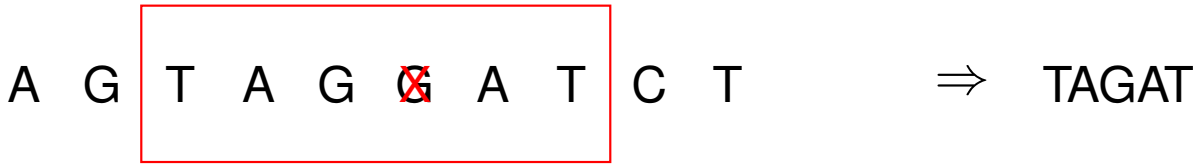
2. Insertion of a nucleotide*; pattern: 1110111*



Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*

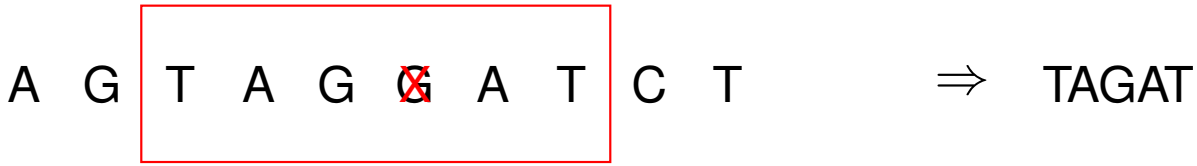


2. Insertion of a nucleotide*; pattern: 1110111*

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*

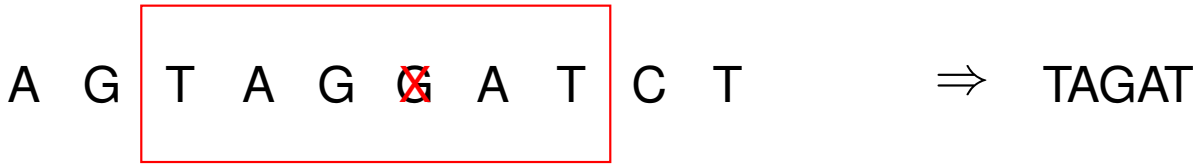


2. Insertion of a nucleotide*; pattern: 1110111*

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*

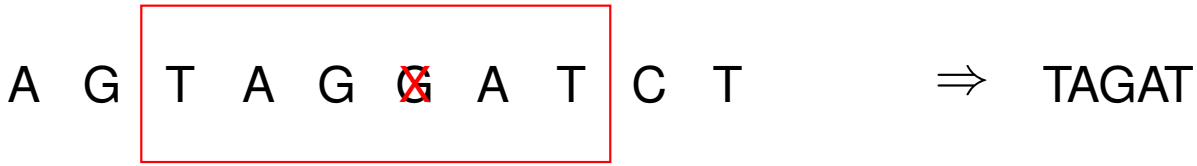


2. Insertion of a nucleotide*; pattern: 1110111*

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*



2. Insertion of a nucleotide*; pattern: 1110111*

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*

A G T A G ~~G~~ A T C T ⇒ TAGAT

2. Insertion of a nucleotide*; pattern: 1110111*

A G T A G V G A T C T ⇒ TAGAGAT, TAGCGAT
TAGGGAT, TAGTGAT

Spaced k-mers

- Idea: simulate corrections to the indel errors in the long reads
- Two cases:

1. Deletion of a nucleotide*; pattern: 111011*

A G T A G ~~G~~ A T C T $\Rightarrow$ TAGAT

2. Insertion of a nucleotide*; pattern: 1110111*

A G T A G V G A T C T $\Rightarrow$ TAGAGAT, TAGCGAT, TAGGGAT, TAGTGAT

Spaced k-mers

Results

Manages to retrieve almost all the k-mers from the reference genome

Problem

Still impossible to discriminate genomic and erroneous k-mers

Spaced k-mers

Results

Manages to retrieve almost all the k-mers from the reference genome

Problem

Still impossible to discriminate genomic and erroneous k-mers

Spaced k-mers

Results

With 16-mers from *A. baylyi* (3,525,881 in the genome), with all patterns containing one zero:

Frequency	16-mers	Genomic 16-mers	Erroneous 16-mers
≥ 1	2,120,061,714	3,514,254	2,116,547,460
≥ 3	1,997,916,471	3,499,165	1,994,417,306
≥ 10	1,116,579,845	3,461,379	1,113,118,466
≥ 30	142,271,582	2,853,079	139,418,503
≥ 50	32,791,370	1,796,485	30,994,885

Spaced k-mers

Results

With 16-mers from *E. coli* (4,513,330 in the genome), with all patterns containing one zero:

Frequency	16-mers	Genomic 16-mers	Erroneous 16-mers
≥ 1	1,825,024,432	4,491,605	1,820,532,827
≥ 3	1,189,940,899	4,472,859	1,185,468,040
≥ 10	240,693,511	4,185,940	236,507,571
≥ 30	15,554,538	1,720,991	13,833,547
≥ 50	2,224,701	457,356	1,767,345

Spaced k-mers

Other tests

- With smaller *k*-mers
- With patterns containing multiple zeros
- Space *k*-mers only on homopolymers
- On PacBio reads

Introduction

- Focusing on words that are absent from a sequence, instead than on those that are present, may bring information
- The number of absent words from a sequence of size n is exponential in n
- Computing all these words would consume too much time and space

Introduction

Definition

A minimal absent word (MAW) of a sequence is a word that does not appear in the sequence, but whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

Example

For the sequence S = AACACACC, the set of minimal absent words is:

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Introduction

Definition

A minimal absent word (MAW) of a sequence is a word that does not appear in the sequence, but whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

Example

For the sequence S = AACACACC, the set of minimal absent words is:

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Introduction

Definition

A minimal absent word (MAW) of a sequence is a word that does not appear in the sequence, but whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

Example

For the sequence S = AACACACC, the set of minimal absent words is:

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Introduction

Definition

A minimal absent word (MAW) of a sequence is a word that does not appear in the sequence, but whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

Example

For the sequence S = AACACACC, the set of minimal absent words is:

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Introduction

Definition

A minimal absent word (MAW) of a sequence is a word that does not appear in the sequence, but whose proper factors (longest prefix, and longest suffix) all occur in the sequence.

Example

For the sequence S = AACACACC, the set of minimal absent words is:

AAA, AACACC, AACC, CAA, CACACA, CCA, CCC

Introduction

Computation

- Various algorithms exist to compute the MAWs of a sequence
- Run in linear time and space
- Based on several data structures: suffix automata [Crochemore et al., 1998], suffix array [Barton et al., 2014], ...

MAWs in long reads

Procedure

- Compute the MAWs of each long read independently
- Store the MAWs of all the long reads into the same set
- Study the MAWs according to their number of occurrences

MAWs in long reads

MAWs display an interesting property:

- Those that are frequent are short, and do appear in the reference genomes

Statistics

When considering MAWs with at least 1000 occurrences as frequent:

Genome	MAWs	MAWs appearing in the genome	MAWs not appearing in the genome
A. baylyi	128,007	127,938	69
E. coli	62,502	62,502	0
S. cerevisae	250,459	250,431	28

MAWs in long reads

MAWs display an interesting property:

- Those that are frequent are short, and do appear in the reference genomes

Statistics

When considering MAWs with at least 1000 occurrences as frequent:

Genome	MAWs	MAWs appearing in the genome	MAWs not appearing in the genome
A. baylyi	128,007	127,938	69
E. coli	62,502	62,502	0
S. cerevisae	250,459	250,431	28

MAWs in long reads

Discriminate k-mers with MAWs

- Use the complementary information brought by the MAWs to determine whether a k-mer is genomic or not
- Consider a k-mer as genomic if it contains frequent MAWs (resp. as erroneous if it does not)

MAWs in long reads

Discriminate k-mers with MAWs

3 approaches:

1. Discriminate a *k*-mer if it contains at least a MAW
2. Discriminate a *k*-mer according to the number of MAWs it contains
3. Discriminate a *k*-mer according to its coverage in MAWs

MAWs in long reads

Discriminate k-mers with MAWs

3 approaches:

1. Discriminate a *k*-mer if it contains at least a MAW
2. Discriminate a *k*-mer according to the number of MAWs it contains
3. Discriminate a *k*-mer according to its coverage in MAWs

MAWs in long reads

Discriminate k-mers with MAWs

3 approaches:

1. Discriminate a *k*-mer if it contains at least a MAW
2. Discriminate a *k*-mer according to the number of MAWs it contains
3. Discriminate a *k*-mer according to its coverage in MAWs

MAWs in long reads

Discriminate k-mers with MAWs

3 approaches:

1. Discriminate a *k*-mer if it contains at least a MAW
2. Discriminate a *k*-mer according to the number of MAWs it contains
3. Discriminate a *k*-mer according to its coverage in MAWs

MAWs in long reads

Unsatisfying results

- Genomic and erroneous *k*-mers display the same statistics
- Still impossible to discriminate genomic and erroneous *k*-mers

MAWs in long reads

Unsatisfying results... Because of the short average length of the MAWs?

Statistics

When filtering out short MAWs, and considering MAWs with at least 20 occurrences as frequent:

Genome	# MAWs	# MAWs appearing in the genome	# MAWs not appearing in the genome
A. baylyi	157,830	146,087	11,743
E. coli	1,875	1,873	3
S. cerevisae	25,916	17,072	8,844

MAWs in long reads

Unsatisfying results... Because of the short average length of the MAWs?

Statistics

When filtering out short MAWs, and considering MAWs with at least 20 occurrences as frequent:

Genome	# MAWs	# MAWs appearing in the genome	# MAWs not appearing in the genome
A. baylyi	157,830	146,087	11,743
E. coli	1,875	1,873	3
S. cerevisae	25,916	17,072	8,844

MAWs in long reads

Results of k-mers discrimination

On *A. baylyi*:

- Only 20% of erroneous *k*-mers contain a MAW...
- ...But only 1/3 of genomic *k*-mers contain one
- No way to further discriminate

MAWs in long reads

Results of k-mers discrimination

On *A. baylyi*:

- Only 20% of erroneous *k*-mers contain a MAW...
- ...But only 1/3 of genomic *k*-mers contain one
- No way to further discriminate

MAWs in long reads

Results of k-mers discrimination

On *A. baylyi*:

- Only 20% of erroneous *k*-mers contain a MAW...
- ...But only 1/3 of genomic *k*-mers contain one
- No way to further discriminate

MAWs in long reads

Results of k-mers discrimination

On *A. baylyi*:

- Only 20% of erroneous *k*-mers contain a MAW...
- ...But only 1/3 of genomic *k*-mers contain one
- No way to further discriminate

Reconstruction

- It is possible to uniquely reconstruct a sequence from its set of MAWs
- Algorithm available in [Mignosi et al., 2002], linear time
- No implementation yet...

Reconstruction

Applications

- Assembly, with corrected long reads
- Correction, with subsets of raw long reads

Reconstruction

Assembly

- Study [Fici et al., 2006]
- Retrieve the MAWs of the original sequence from the MAWs of a set of factors
- Reconstruct the original sequence
- No implementation yet...

Reconstruction

Correction

- Select a long read
- Retrieve similar long reads (Commet [Maillet et al., 2014])
- Compute the MAWs from the subset of long reads
- Reconstruct a consensus sequence from the MAWs

Long reads self-alignment

Daccord

- Align the long reads against each other
- Estimate good / bad quality regions of the reads

Long reads self-alignment

Idea

- Trim the long reads on bad quality regions
- Extract and analyze (spaced) *k*-mers from good quality regions

References I

Barton, C., Héliou, A., Mouchard, L., and Pissis, S. P. (2014).
Linear-time computation of minimal absent words using suffix array.
BMC Bioinformatics, 15(1).

Crochemore, M., Mignosi, F., and Restivo, A. (1998).
Automata and forbidden words.
Information Processing Letters, 67(3):111–117.

Fici, G., Mignosi, F., Restivo, A., and Sciortino, M. (2006).
Word assembly through minimal forbidden words.
Theoretical Computer Science, 359(1):214–230.

References II


Maillet, N., Collet, G., Vannier, T., Lavenier, D., and Peterlongo, P. (2014).
Commet: Comparing and combining multiple metagenomic datasets.
In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*.


Miclotte, G., Heydari, M., Demeester, P., Rombauts, S., Van de Peer, Y., Audenaert, P., and Fostier, J. (2016).
Jabba: hybrid error correction for long sequencing reads.
Algorithms for Molecular Biology, 11:10.

References III

Mignosi, F., Restivo, A., and Sciortino, M. (2002).
Forbidden Words and Fragment Assembly.
In *Developments in Language Theory*, pages 349–358. Springer, Berlin.

Morisse, P., Lecroq, T., and Lefebvre, A. (2017).
HG-CoLoR: Hybrid Graph for the error Correction of Long Reads.
In *Journées Ouvertes en Biologie, Informatique et Mathématiques*, pages 67–74.

Salmela, L. and Rivals, E. (2014).
LoRDEC: Accurate and efficient long read error correction.
Bioinformatics, 30(24):3506–3514.

References IV


Salmela, L., Walve, R., Rivals, E., and Ukkonen, E. (2017).
Accurate selfcorrection of errors in long reads using de Bruijn graphs.
Bioinformatics, 33(6):799–806.


Tischler, G. and Myers, E. W. (2017).
Non Hybrid Long Read Consensus Using Local De Bruijn Graph Assembly.
pages 1–42.