# Hybrid and non hybrid error correction for long reads: LoRDEC and LoRMA

Eric Rivals

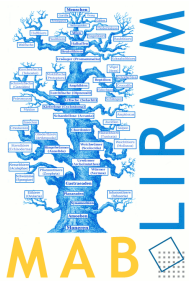
# Hybrid and non hybrid error correction for long reads: LoRDEC and LoRMA

Eric Rivals

Computer Science Lab & Institute Computational Biology, CNRS & Univ. Montpellier

7th Nov. 2016



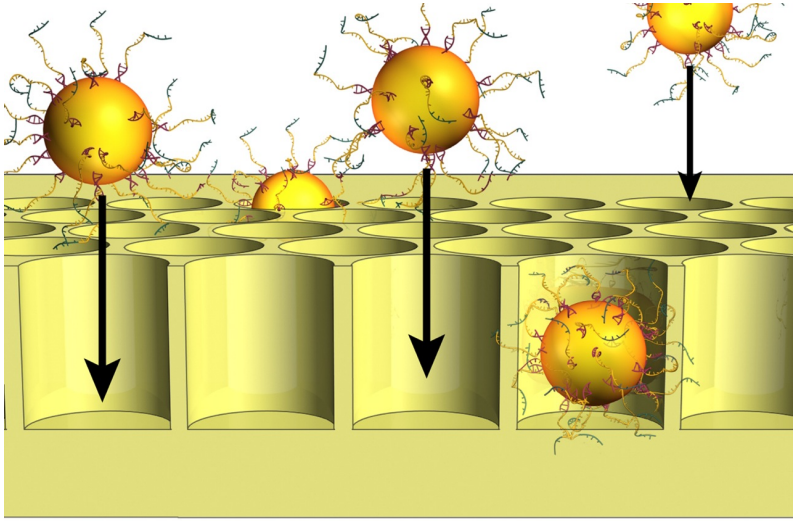


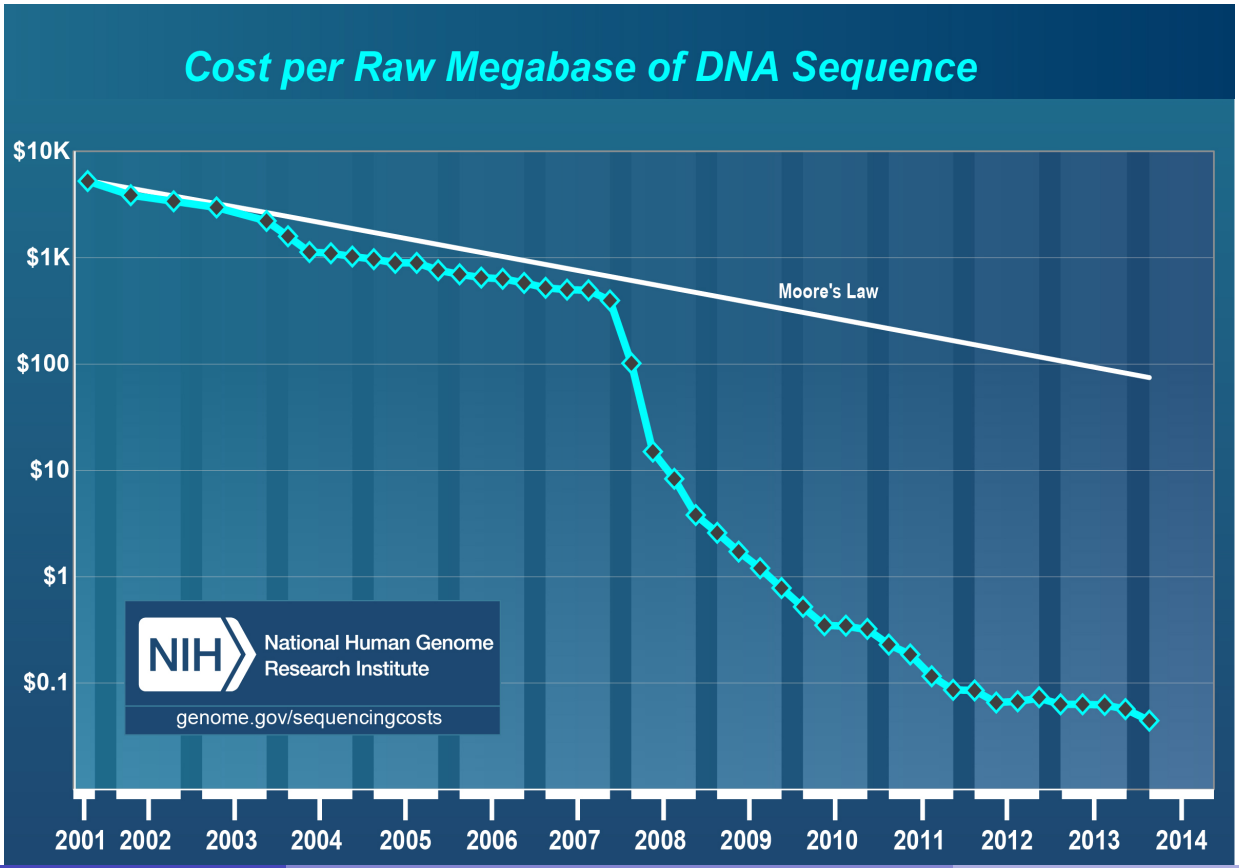


# Outline

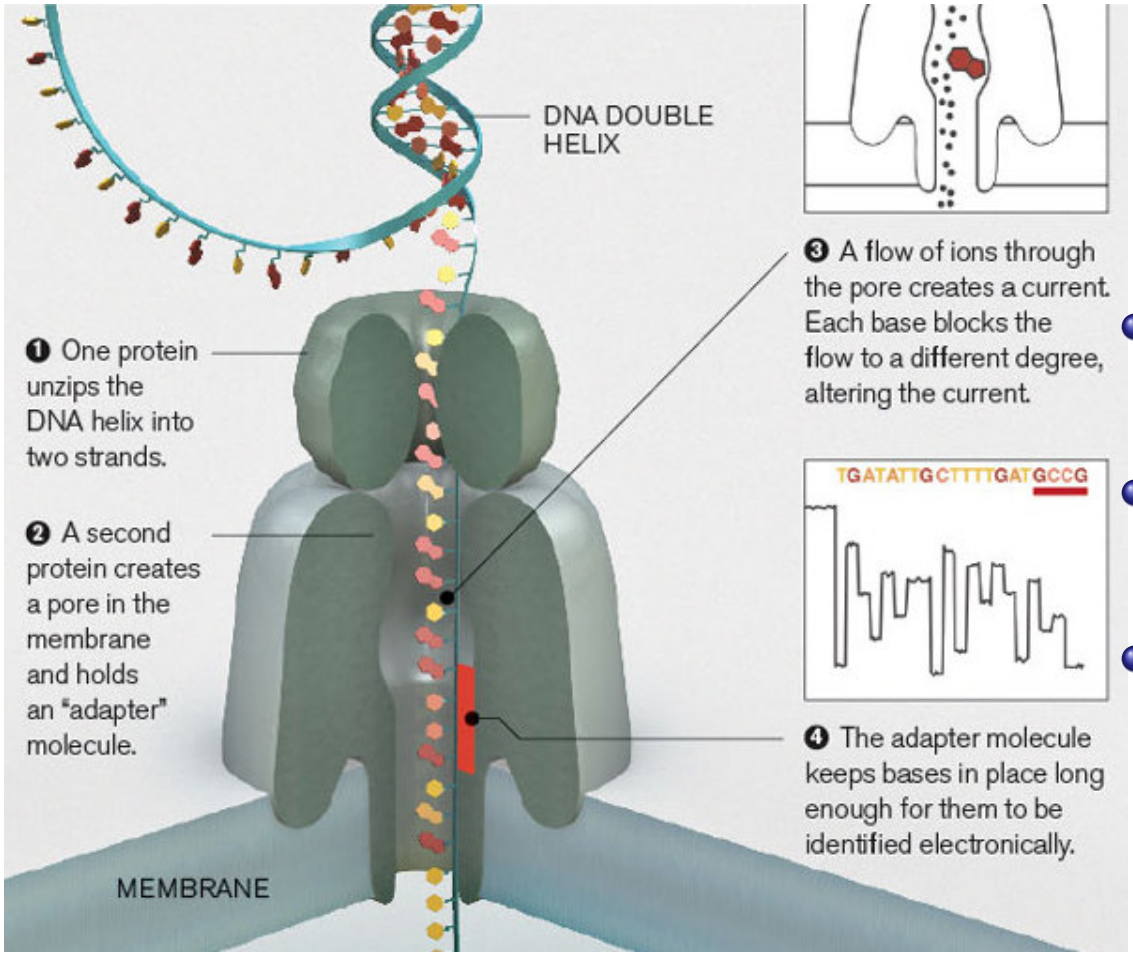
# Outline

# Revolution in DNA sequencing





# Third generation technologies




- PacBio: Pacific Biosciences
  up to 25 Kbp
- Oxford Nanopore MINion
  up to 50 Kbp
- Moleculo synthetic reads
  up to 10 Kbp

©Oxford Nanopore

# Overview of sequencing techniques

| Name | Read Lg | Time | Gb/run | pros / cons |
|---|---|---|---|---|
| 454 GS Flex | 700 | 1 d | 0.7 | long / indels |
| Illumina HiSeq X | 2*300 | 3 d | 200 | short/cost |
| Illumina NextSeq 500 | 2*300 | 3 d | 150 | PE, single/idem |
| SOLID (LifeSc) | 85 | 8 d | 150 | long time |
| Ion Proton | 200 | 2 h | 100 | new |
| Illumina TrueSeq | 10-8500 | – | 4 | synthetic reads |
| PacBio Sciences | 10-40000 | 0.3 d | 3 | high error rate |
| Oxford MINion | 10-50000 | 1 d | 0.8 | high error rate |

The vast majority of errors for PacBio and Oxford are insertions & deletions.

# Context

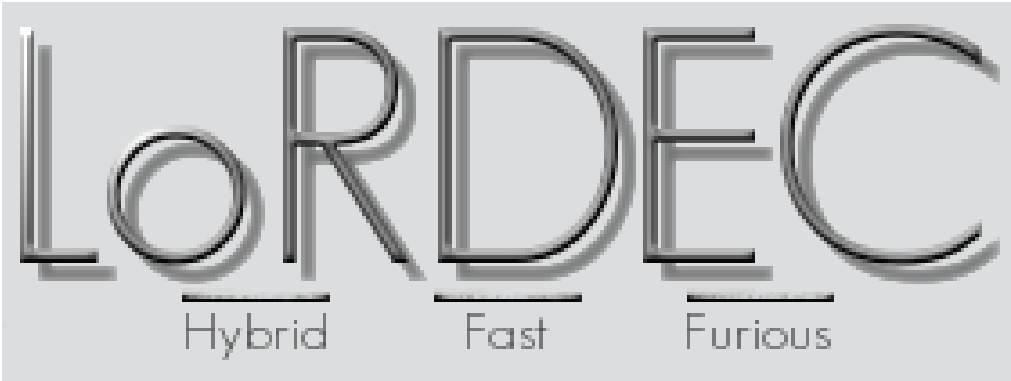
- 3rd generation sequencing technologies yield longer reads
- PacBio Single Molecule Real Time sequencing:
  much longer reads (up to 25 Kb) but much higher error rates
- Error correction is required
  1. self correction: using long reads only
  2. hybrid correction: using short reads to correct long reads

# Context

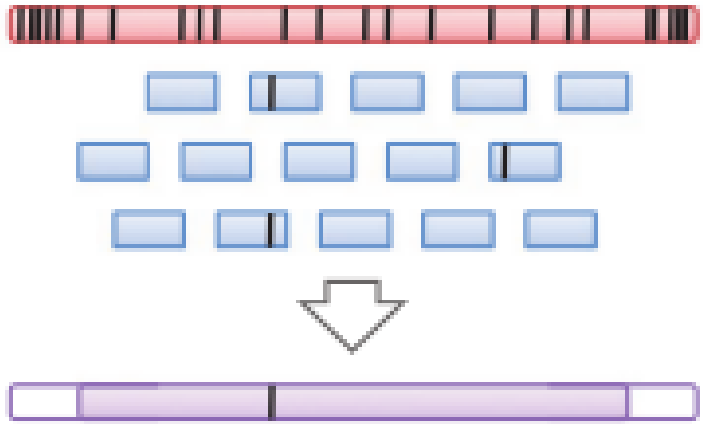
- 3rd generation sequencing technologies yield longer reads
- PacBio Single Molecule Real Time sequencing:
  much longer reads (up to 25 Kb) but much higher error rates
- Error correction is required
  1. self correction: using long reads only
  2. hybrid correction: using short reads to correct long reads

# Context

- 3rd generation sequencing technologies yield longer reads
- PacBio Single Molecule Real Time sequencing:
  much longer reads (up to 25 Kb) but much higher error rates
- Error correction is required
  1. self correction: using long reads only
  2. hybrid correction: using short reads to correct long reads



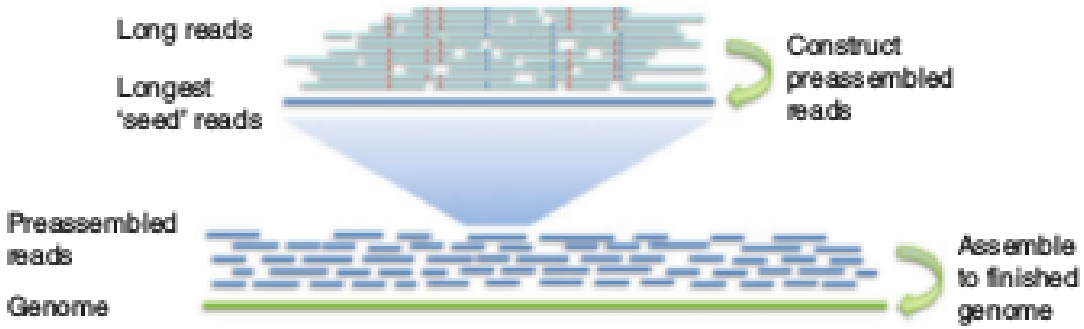
# Hybrid correction methods



[Koren et al, Nat. Bio. 2012]

- Short reads are aligned to long reads
- a consensus is applied to correct part of the long read

# Self correction methods




[Chin et al, Nat. Met. 2013]

Long reads are corrected with shorter reads from same technology

# Other hybrid PacBio error correction programs

- PacBioToCA [Koren et al. 2012]
- AHA [Bashir et al. 2012]
  inside the assembler
- LSC [Au et al. 2012]
  compress homopolymers before alignment

All follow an alignment based strategy (e.g. BLAST like)

# Other hybrid PacBio error correction programs

- PacBioToCA [Koren et al. 2012]
- AHA [Bashir et al. 2012]
  inside the assembler
- LSC [Au et al. 2012]
  compress homopolymers before alignment

All follow an alignment based strategy (e.g. BLAST like)

- `proovread` [Hackl et al. 2014]: alignment & chimera detection
- Jabba [Miclotte et al. 2015]: LoRDEC's approach + MEM based alignment
  variable length seeds for anchoring the LR on graph
- `CoLoRMap` [Haghshenas et al. 2016]: alignment & local assembly

# Hybrid correction and assembly

- ECtools [Lee et al. bioRxiv 2014]
  assemble SR into unitigs, assemble unitigs and LR with Celera

- Nanocorr [Goodwin et al. bioRxiv 2014]
  recruit SR for a LR using BLAST,
  select SR with Longest Increasing Subsequence (LIS)
  compute consensus
  assembly with Celera

- NaS (Nanopore) [Madoui et al BMC Genomics 2015]
  recruit SR for each LR and reassemble the LR sequence
  complex pipeline

# Hybrid correction and assembly

- ECtools [Lee et al. bioRxiv 2014]
  assemble SR into unitigs, assemble unitigs and LR with Celera

- Nanocorr [Goodwin et al. bioRxiv 2014]
  recruit SR for a LR using BLAST,
  select SR with Longest Increasing Subsequence (LIS)
  compute consensus
  assembly with Celera

- NaS (Nanopore) [Madoui et al BMC Genomics 2015]
  recruit SR for each LR and reassemble the LR sequence
  complex pipeline

All need to assemble SR

# Motivation

*LR correction programs "require high computational resources and long running times on a supercomputer even for bacterial genome datasets".*

[Deshpande et al. 2013]

# Motivation

*LR correction programs "require high computational resources and long running times on a supercomputer even for bacterial genome datasets".*

[Deshpande et al. 2013]

*For a 1 Gb plant genome, correction of 18x PacBio with 160x Illumina required* 600000 *CPU hours with EC-tools !*

# Contributions

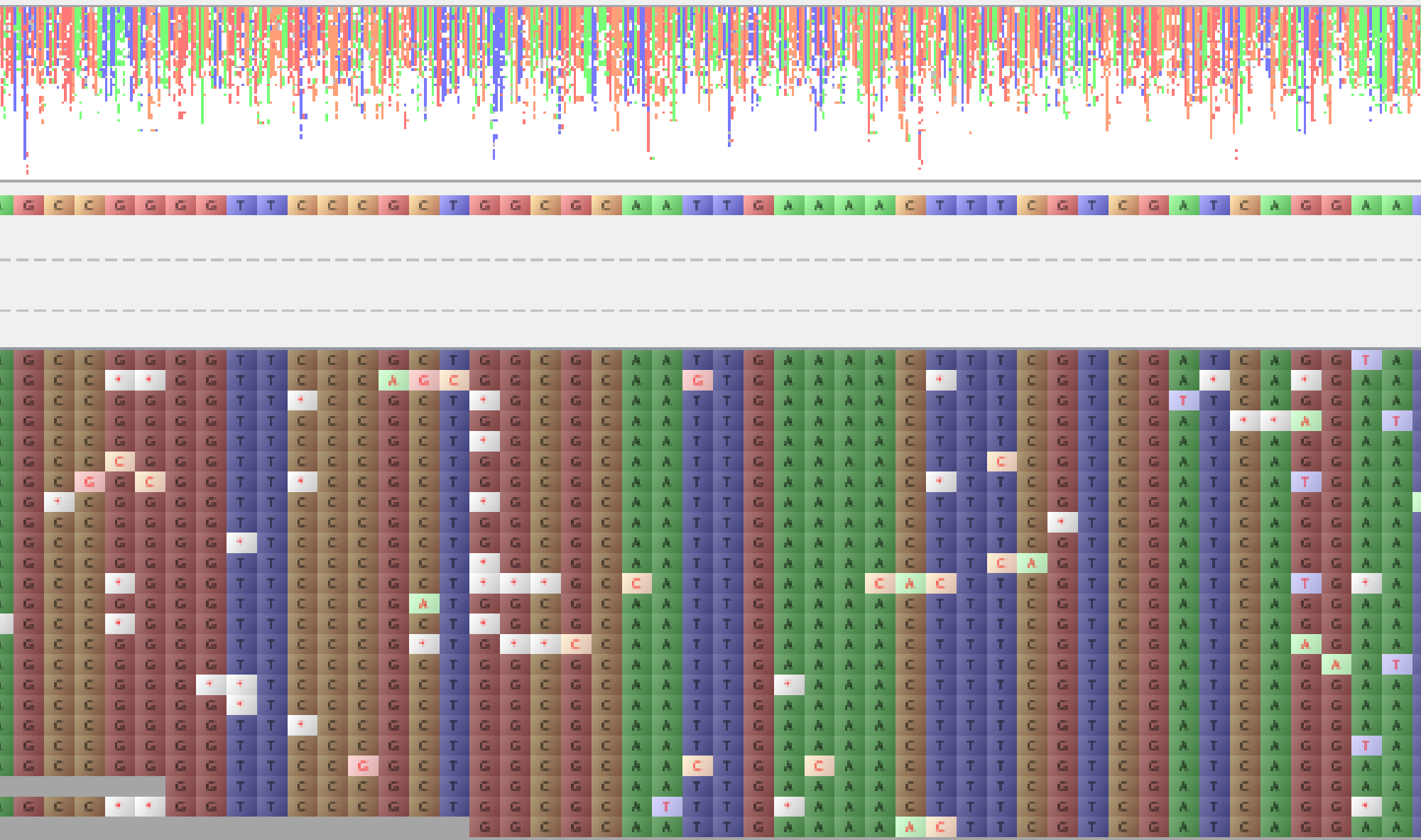
LoRDEC

- a new and efficient hybrid correction algorithm
- based on De Bruijn Graphs (DBG) of short reads
- avoids the time consuming alignments (of SR on LR)

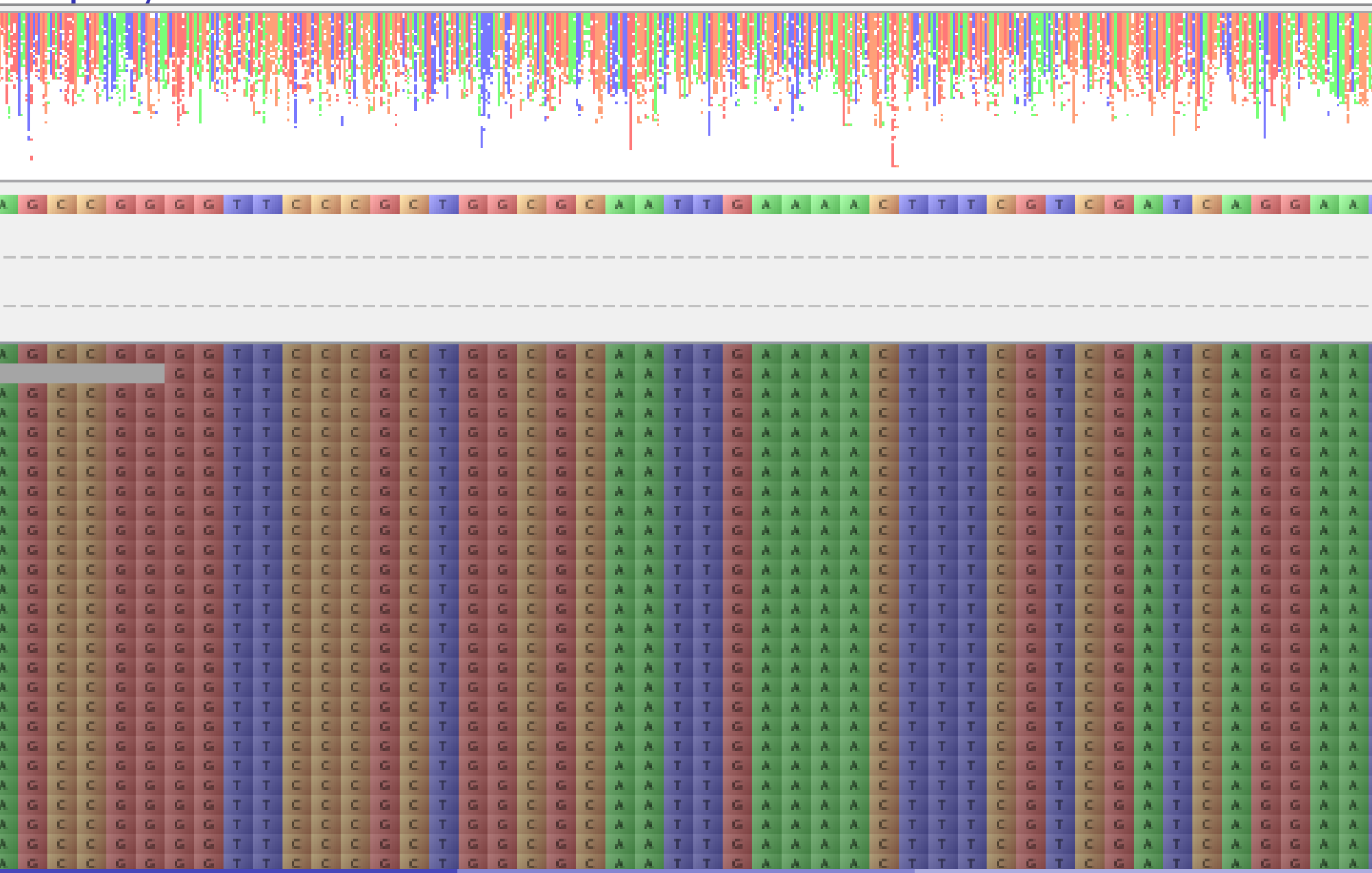
LoRMA

- a complementary tool to LoRDEC for self correction of long reads
- a pipeline that iterates LoRDEC and apply LoRMA

# Aperçu of raw and corrected PacBio reads



# Aperçu of raw and corrected PacBio reads



# Outline

# Algorithm overview

1. build a de Bruijn graph of the short reads

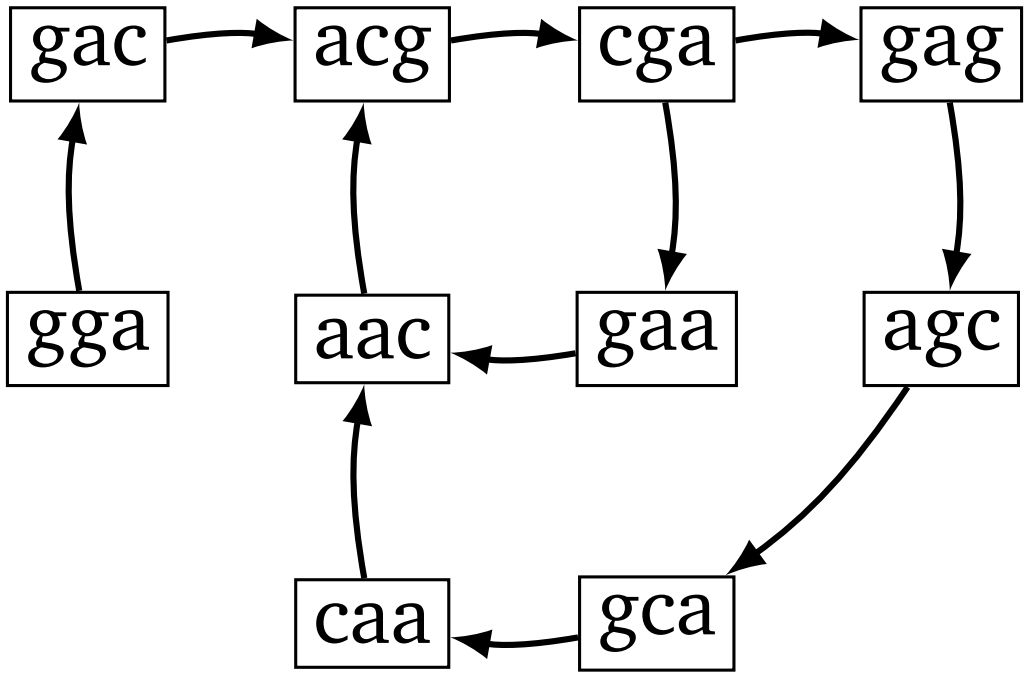
# Algorithm overview

1. build a de Bruijn graph of the short reads

   the graph represents the short reads in compact form


2. take each long read in turn and attempt to correct it
   1. correct internal regions,
   2. correct end regions of the long read

# Example of short read DBG of order 3




$S = \{ggacgaa, cgaac, gacgag, cgagcaa, gcaacg\}$

The DBG is built from the set of short reads (Illumina)

using the GATB library.

# Filtering $k$-mers of short reads

## Filtering $k$-mer rationale

Because errors are randomly positioned

Erroneous $k$-mers have low expected occurrence numbers

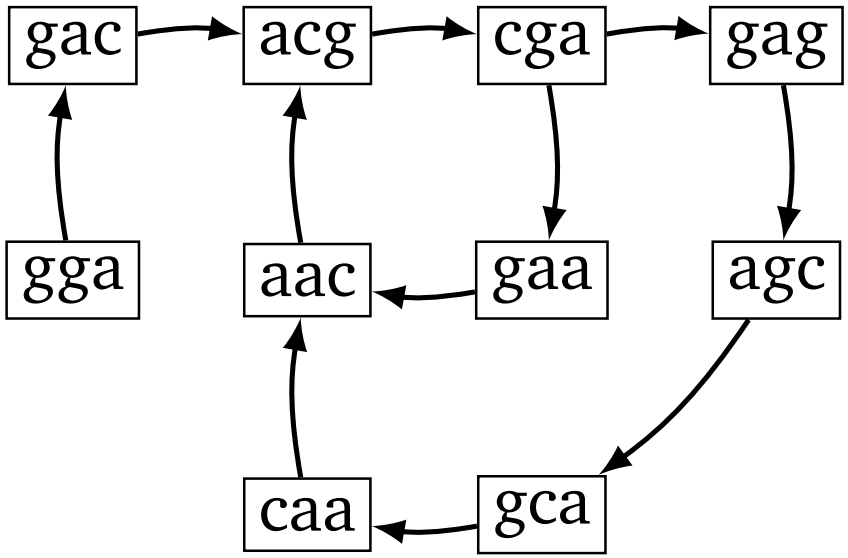
Threshold based filter $s$: minimum number of occurrences in short reads

All $k$-mers present more than $s$ times are called solid $k$-mers

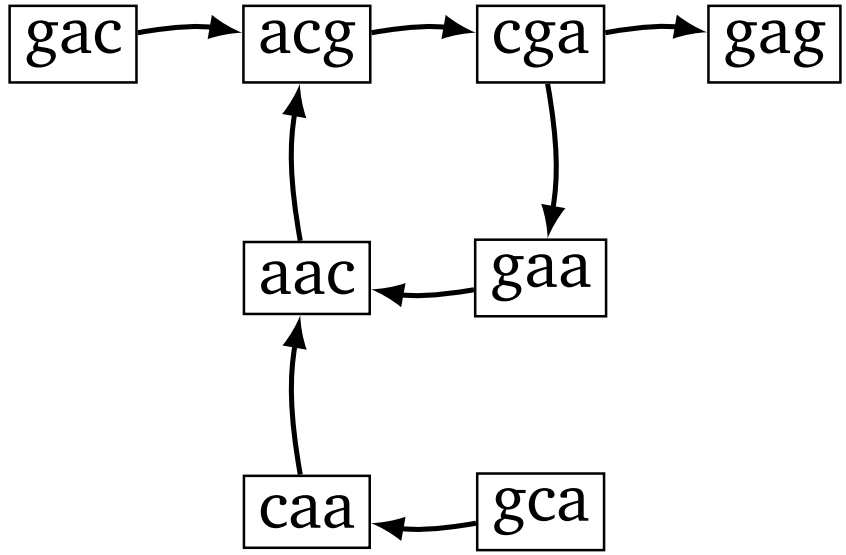
and kept in the de Bruijn Graph

# Example of filtered short read DBG of order 3

raw graph

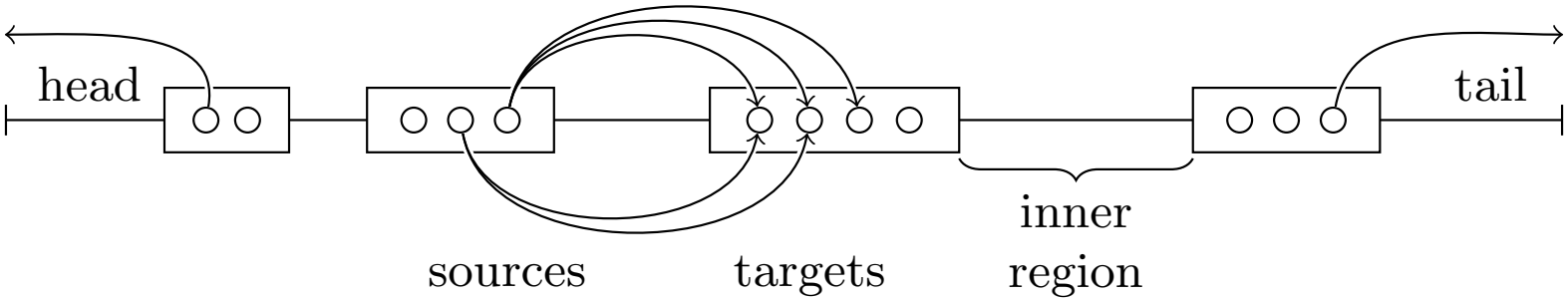



filtered with $s = 1$




$S = \{ggacgaa, cgaac, gacgag, cgagcaa, gcaacg\}$

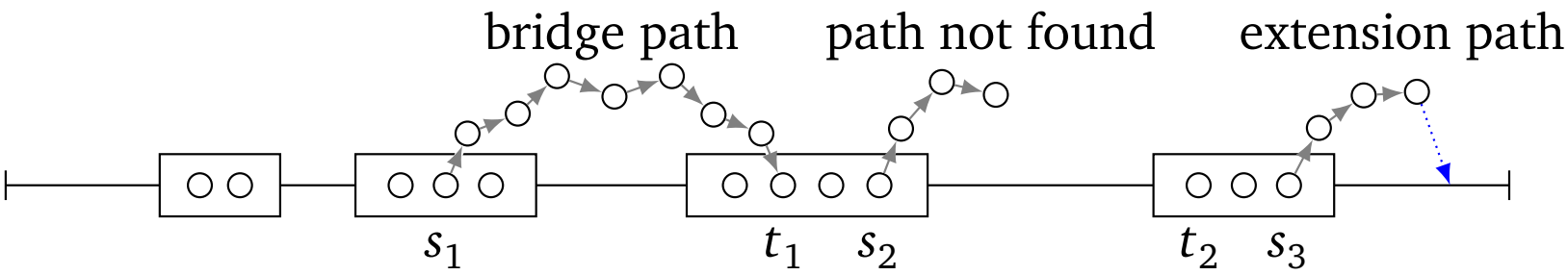
# Long read sequence is partitioned




○: solid $k$-mers of the long read

- Solid $k$-mers are a priori correct piece of the sequences
- we correct the region between two solid $k$-mers

# Long read is corrected with DBG

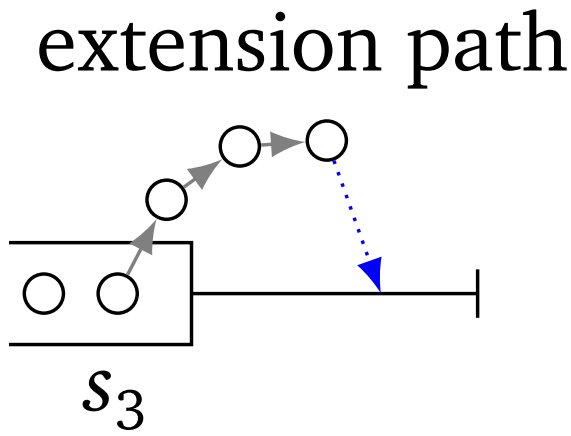



For each putative region of a long read:

- align the region to paths of the de Bruijn graph
- find best path according to edit distance
- limited path search

# LoRDEC: Correcting read ends

- Find a path in DBG starting from the extreme solid $k$-mer
- Maximize length of the prefix of the end to correct
- Minimize edit distance between the path and the prefix of the end
- Find best extension maximizing an alignment score



# Correction algorithm

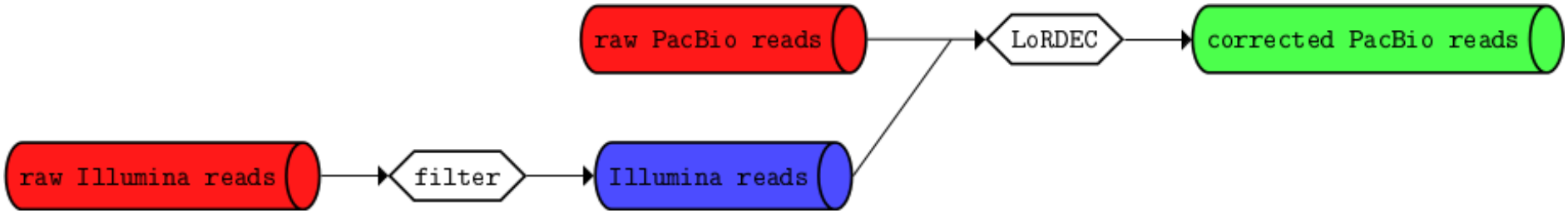
1. Correct inner region:
    1. depth first search traversal of paths between source and target $k$-mers
    2. node wise: minimal edit distance computation with seq region
2. Correct end region:
3. Paths optimisation:
    1. build a graph of all correction paths for current read
    2. finding a shortest path between the first and last solid k-mers Dijkstra algorithm

# Trimming and splitting (optional)

- Classify each base as solid if it belongs to at least one solid $k$-mer and weak otherwise
- LoRDEC outputs solid bases in upper case characters and weak ones in lower case characters
- Corrected reads can be trimmed and/or split:
  1. Trim weak bases from both ends of the read
  2. Extract all runs of solid bases from the corrected reads

- Output of LoRDEC:
  >read1
  acgtgaGTAGTCGAGTagcgtagG
  TGGATCGAGCTAGggggt
- Trimmed read:
  >read1
  GTAGTCGAGTagcgtagGTGGATCG
  AGCTAG
- Trimmed and split reads:
  >read1_1
  GTAGTCGAGT
  >read1_2
  GTGGATCGAGCTAG

# LoRDEC correction pipline



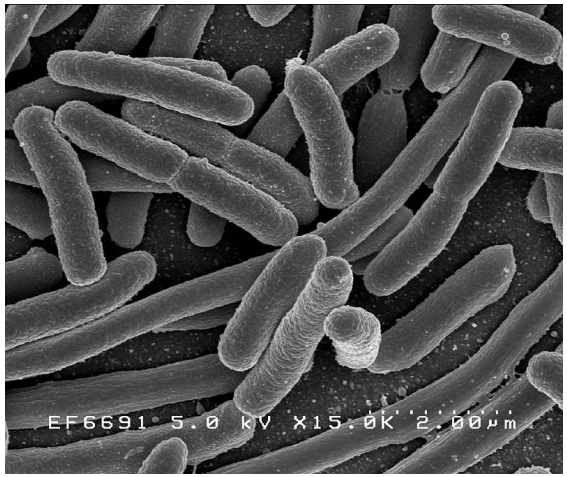

- Filtering short-reads data for quality value and adapter presence `cutadapt` [Martin, 2012]

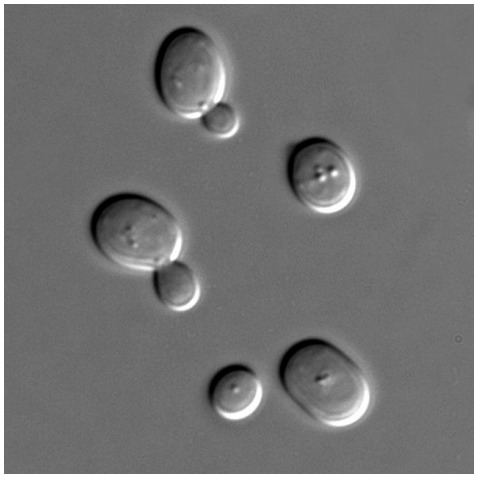
- Long reads correction with LoRDEC.
  Two parameters must be set :
  - $k$-mer length – default $k = 19$
  - threshold : minimum abundance for a $k$-mer to be solid
    that is, to be included in the de Bruijn graph

# Outline

# Data sets

| | E. coli | Yeast | Parrot |
|---|---|---|---|
| Genome size | 4.6 Mbp | 12 Mbp | 1.23 Gbp |
| PacBio coverage | 21x | 129x | 5.5x |
| Illumina coverage | 50x | 38x | 28x |

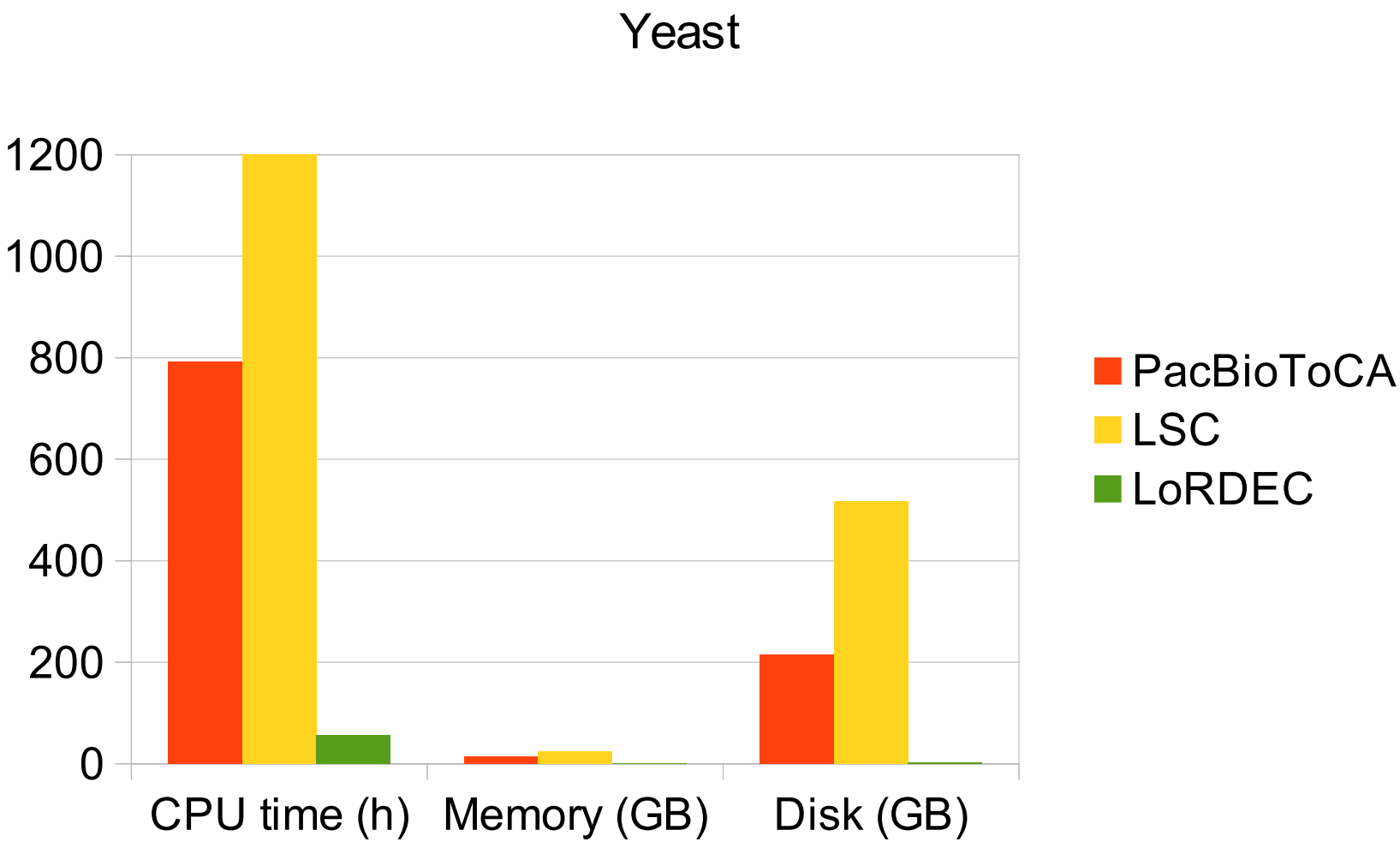


# Results: time and memory

| Data | Method | CPU time | Elapsed time | Memory | Disk |
|---|---|---|---|---|---|
| *E. coli* | PacBioToCA | 45 h 18 min | 3 h 12 min | 9.91 | 13.59 |
| | LSC | 39 h 48 min | 2h 56 min | 8.21 | 8.51 |
| | LoRDEC | 2 h 16 min | 10 min | 0.96 | 0.41 |
| Yeast | PacBioToCA | 792 h 41 min | 21 h 57 min | 13.88 | 214 |
| | LSC | 1200 h 46 min | 130 h 16 min | 24.04 | 517 |
| | LoRDEC | 56 h 08 min | 3 h 37 min | 0.97 | 1.63 |
| Parrot | LoRDEC | 568 h 48 min | 29 h 7 min | 4.61 | 74.85 |

# Runtime, memory and disk usage



# Evaluation methods

Two ways:

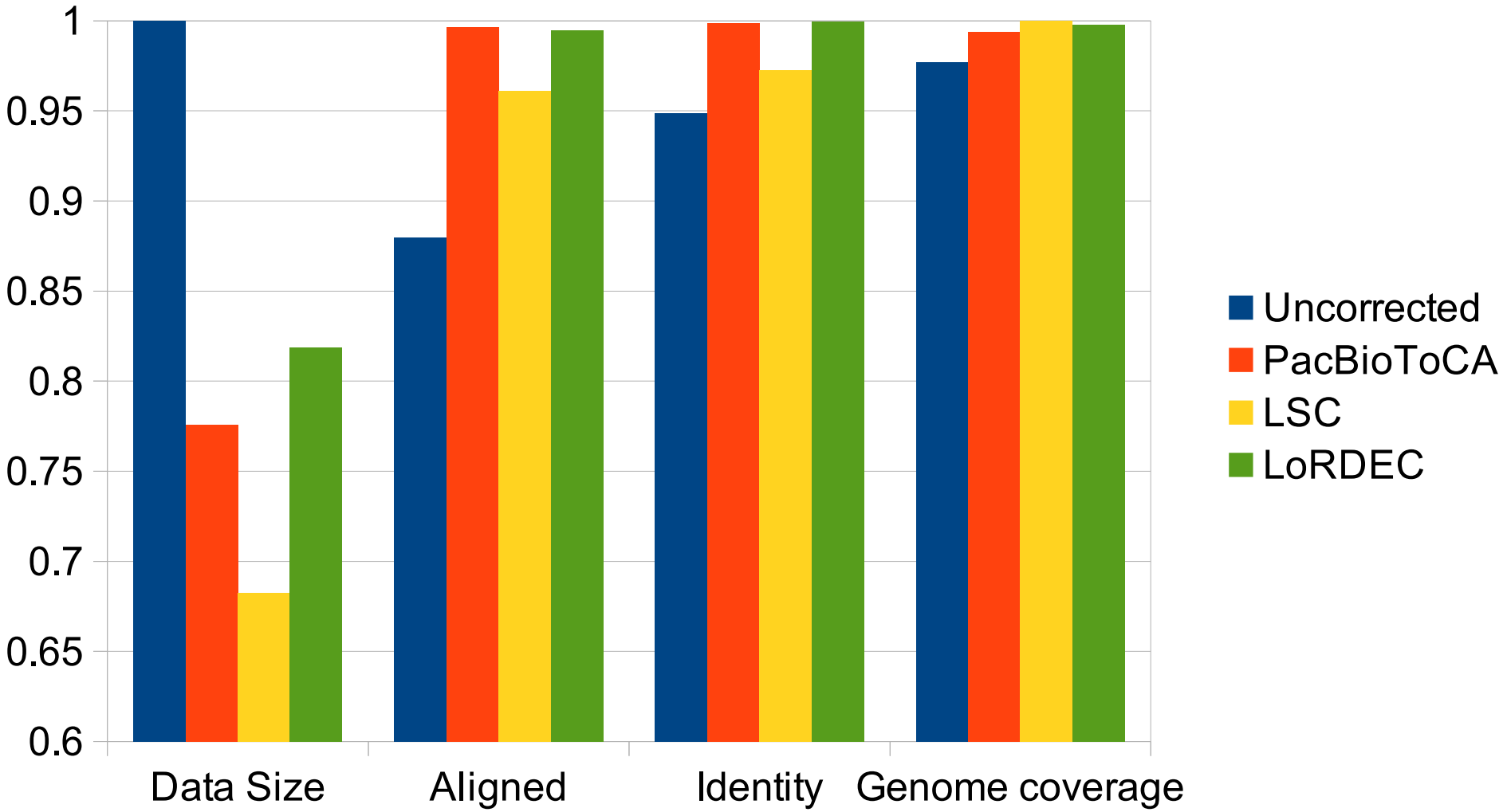
1. how do the reads align to the genome?
2. how do raw and corrected reads differ in their alignments?

Using the Error Correction Toolkit [Yang et al. 2013] we compute

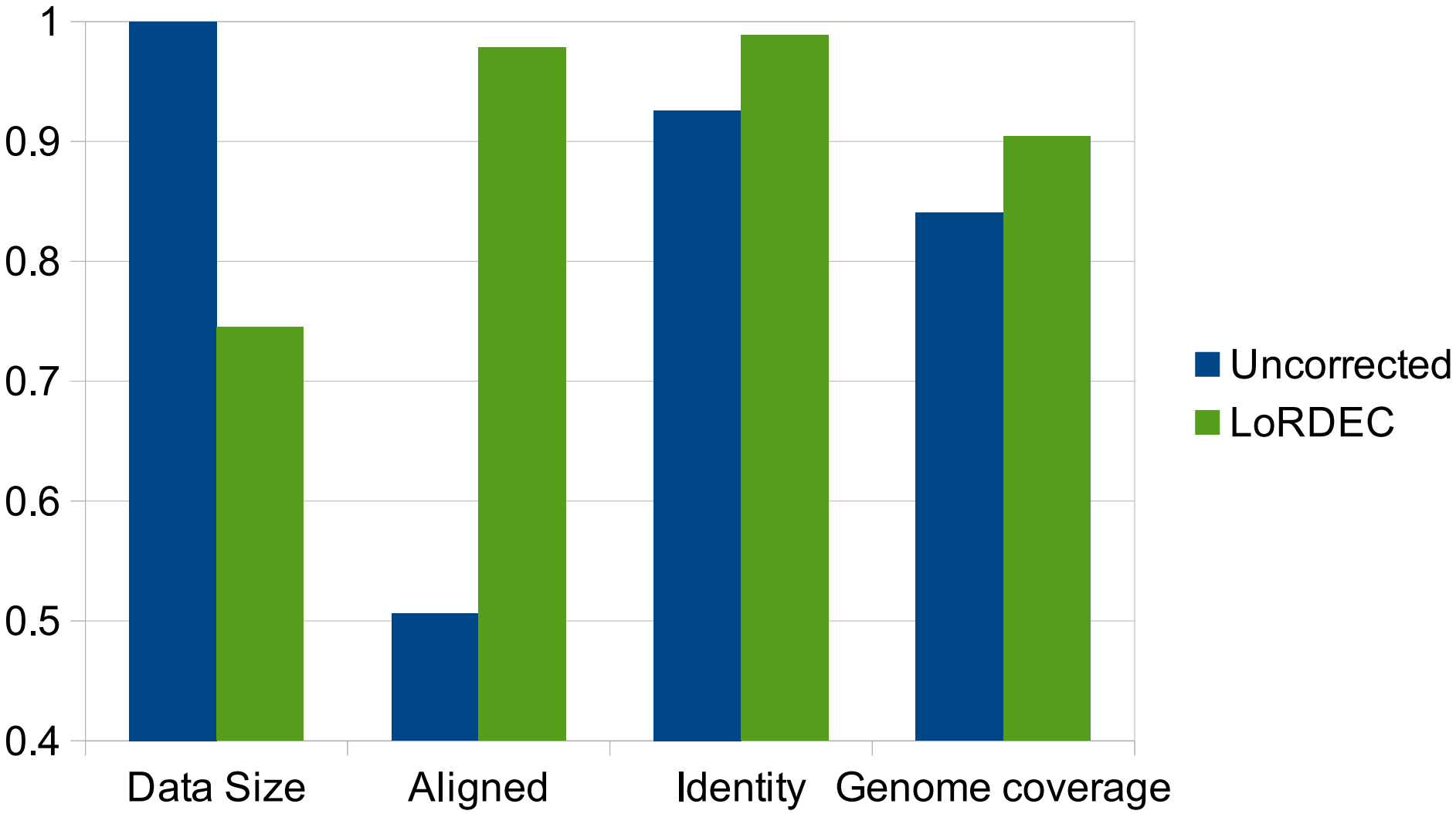
- Sensitivity = TP/(TP+FN)
  how well does the tool recognise erroneous positions?

- Gain = (TP-FP)/(TP+FN)
  how well does the tool remove errors without introducing new ones?

# Error correction performance: *E. coli*



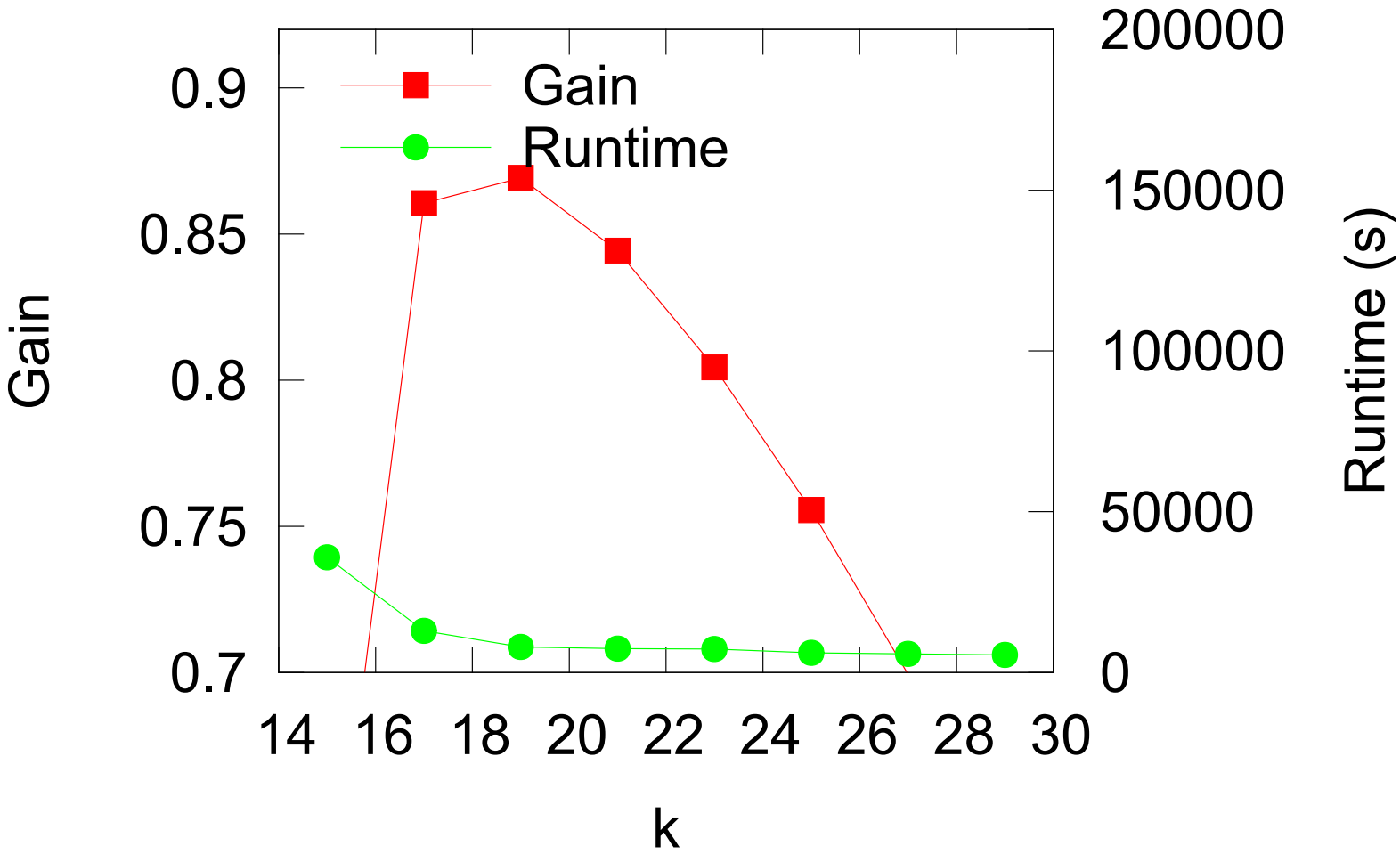
# Error correction performance: Parrot



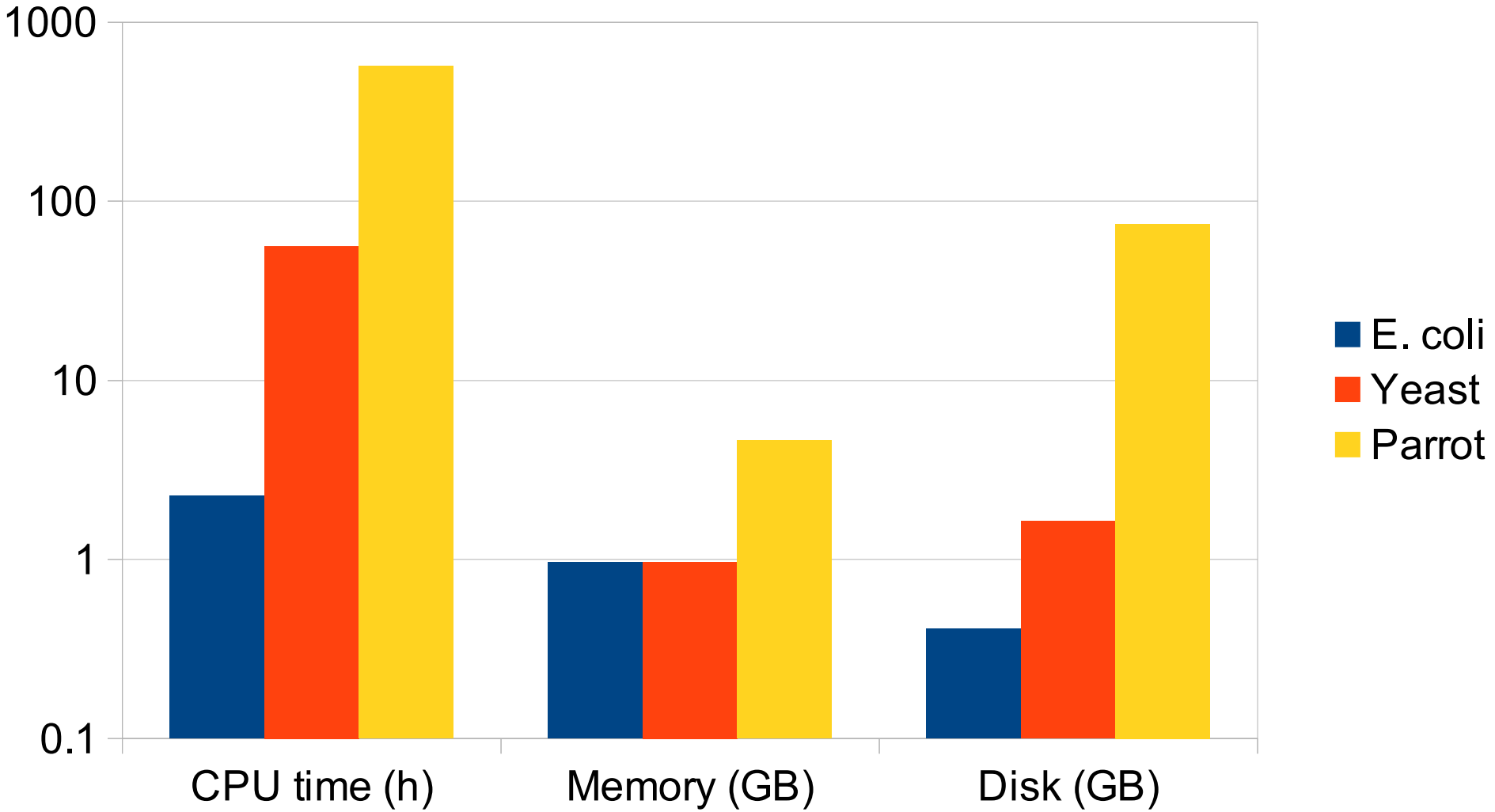
# Sensitivity and gain results

| Data | Method | Sensitivity | Gain |
|---|---|---|---|
| *E. coli* | PacBioToCA | NA | NA |
| | LSC | 0.2865 | 0.2232 |
| | LoRDEC | 0.9090 | 0.8997 |
| Yeast | PacBioToCA[1] | NA | NA |
| | LSC | 0.3246 | 0.2596 |
| | LoRDEC | 0.8427 | 0.8194 |
| Parrot | LoRDEC | 0.8962 | 0.8544 |

# Parameters: E. coli



# Scalability of LoRDEC



# Scalability of LoRDEC

Mais transcriptome data

- Illumina HiSeq : 194 million of reads, 29 Tbp
- PacBio : 276000 reads, 168 Gbp
- LoRDEC time: 12 hours
- LoRDEC memory: 5 Gbytes

# Chicken transcriptome with PacBio

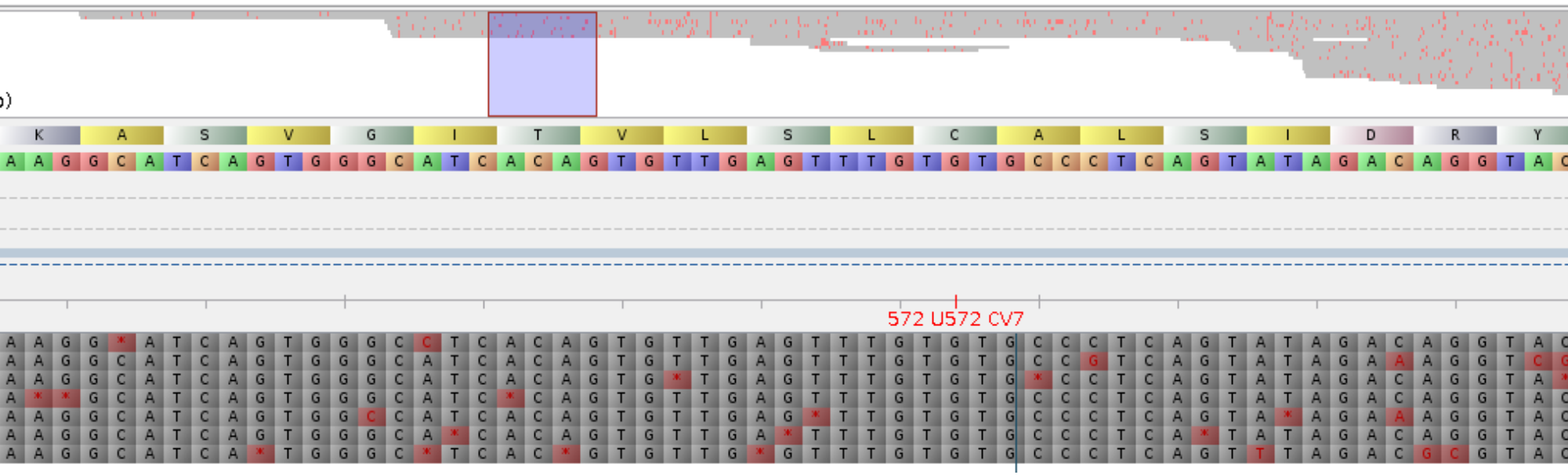
| PacBio data | Raws | Corrected and trimmed |
|---|---|---|
| # reads (x1000) | 1 849 | 1 848 |
| # reads > 1Kbp (x1000) | 687 | 569 |
| Max length of reads (kbp) | 12.2 | 11.9 |
| Total length (Gbp) | 1.98 | 1.77 |
| %GC | 48.08 | 47.28 |
| Avg length (bp) | 1 075 | 960 |

# Chicken transcriptome with PacBio

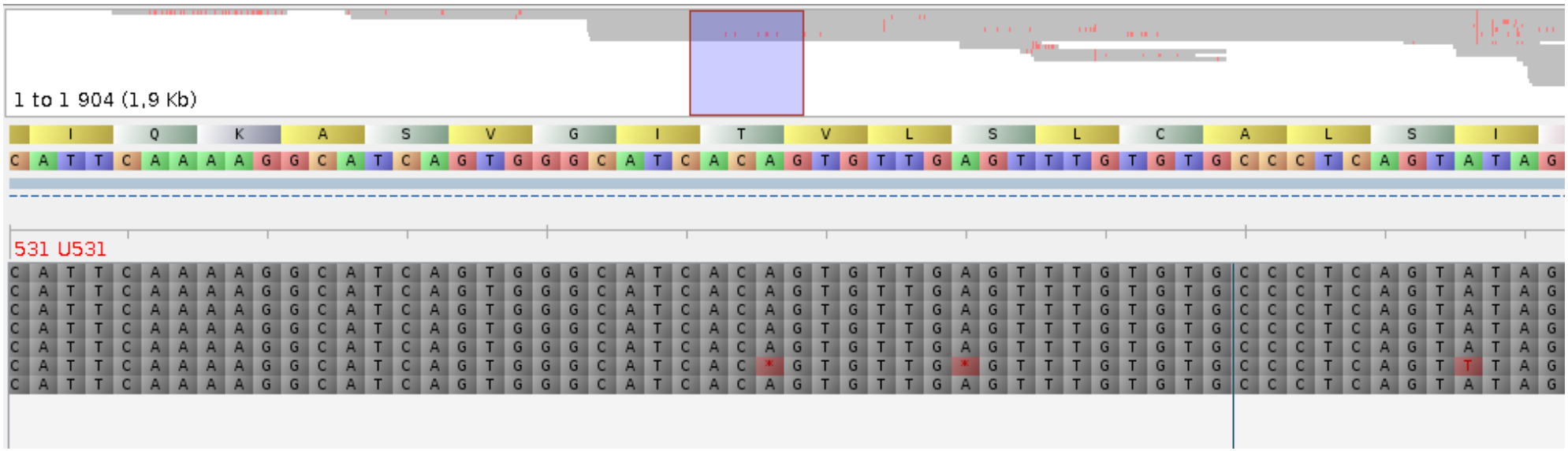
After correction and mapping with BWA-MEM [Li H., 2013]
on ref. transcriptome (1 RNA per gene)

- 5% more transcripts covered with uniquely mapping reads
- 80% id in alignments vs 66% before correction

# Aperçu of raw and corrected PacBio RNA reads



# Aperçu of raw and corrected PacBio RNA reads



# Correcting E. coli Nanopore MINIon data

- Raw reads + quast
- Corrected reads + quast

| Nanopore data | Raw | Corrected |
|---|---|---|
| Nb reads | 3463 | 2749 |
| Nb reads $\geq$ 1kbp | 3420 | 2685 |
| Total length (Mbp) | 22 | 17 |
| Unaligned bases (%) | 99.99 | 7.60 |
| Genome fraction (%) | 0.02 | 96.59 |

Quast [Gurevich et al. 2013]

# MINion S. aureus data

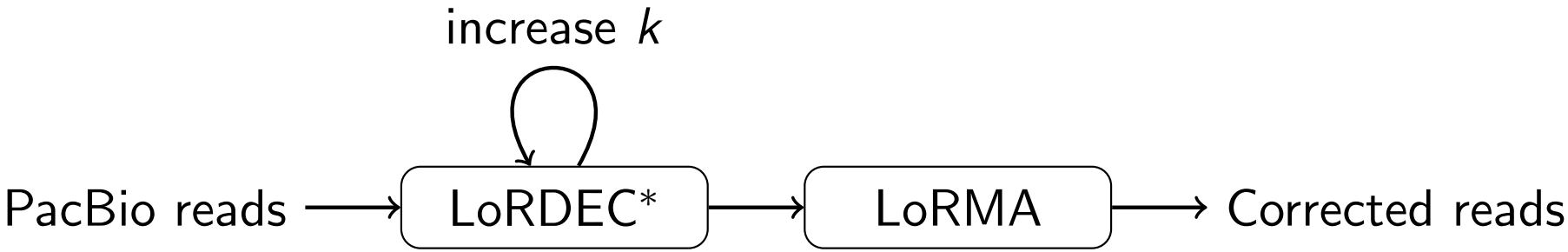
Mapping of reads with BWA-MEM onto the reference genome with appropriate options

- ref génome: 2.8 Mbp
- MINIon sequencing coverage $14x$
- gain for $k = 17$ and $s = 2$ reaches 69%
- $99,9$ % genome covered by corrected reads
- 65 % genome at median coverage 8x
- 79% identity instead of 66 % without correction

# Outline

# Overview of LoRDEC*+LoRMA

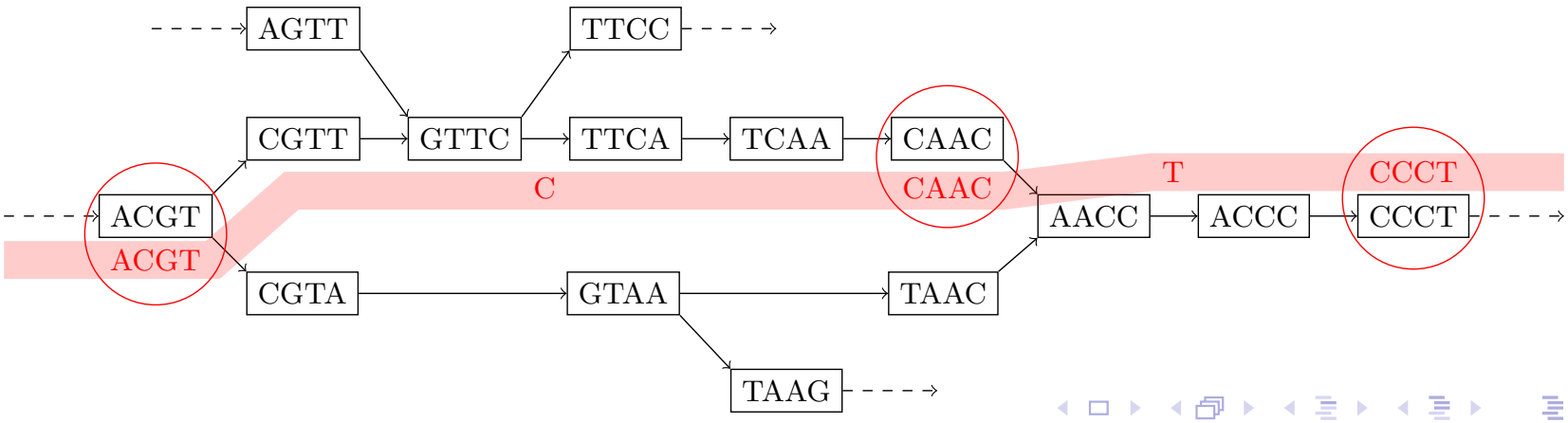
- Modify LoRDEC to run on long reads only $\Longrightarrow$ LoRDEC*
- Run LoRDEC* iteratively with increasing $k$
- Polish the result with multiple alignments $\Longrightarrow$ LoRMA



# LoRDEC

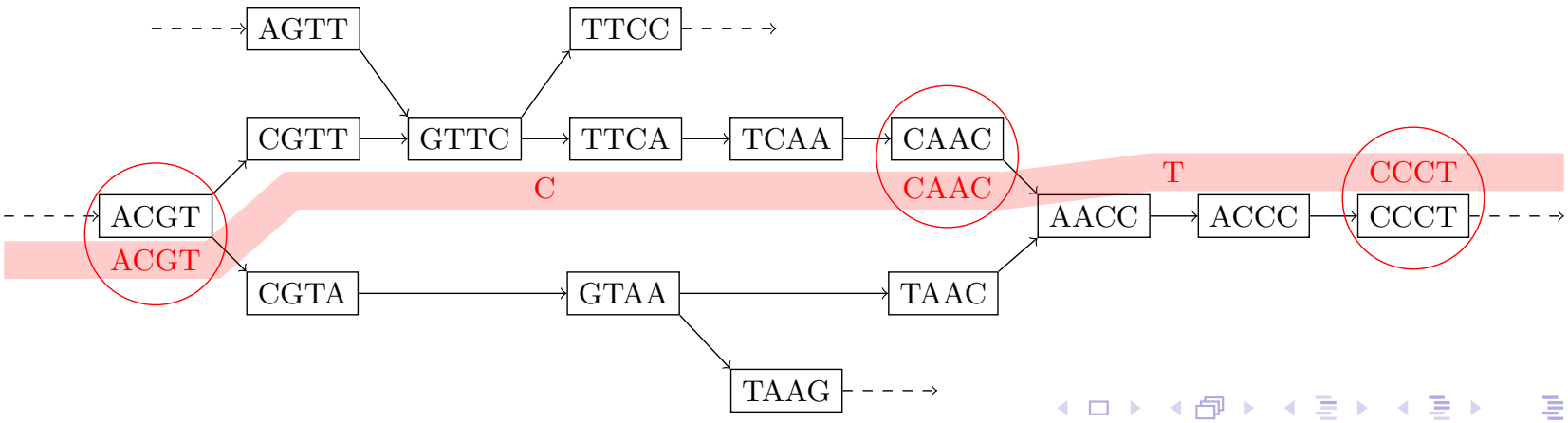
- Build a de Bruijn graph of the short reads

- For each long read:
  - Classify $k$-mers: solid (= in the DBG) and weak
  - Find paths in the DBG between the solid $k$-mers
  - Minimize edit distance between the long read and the path's string



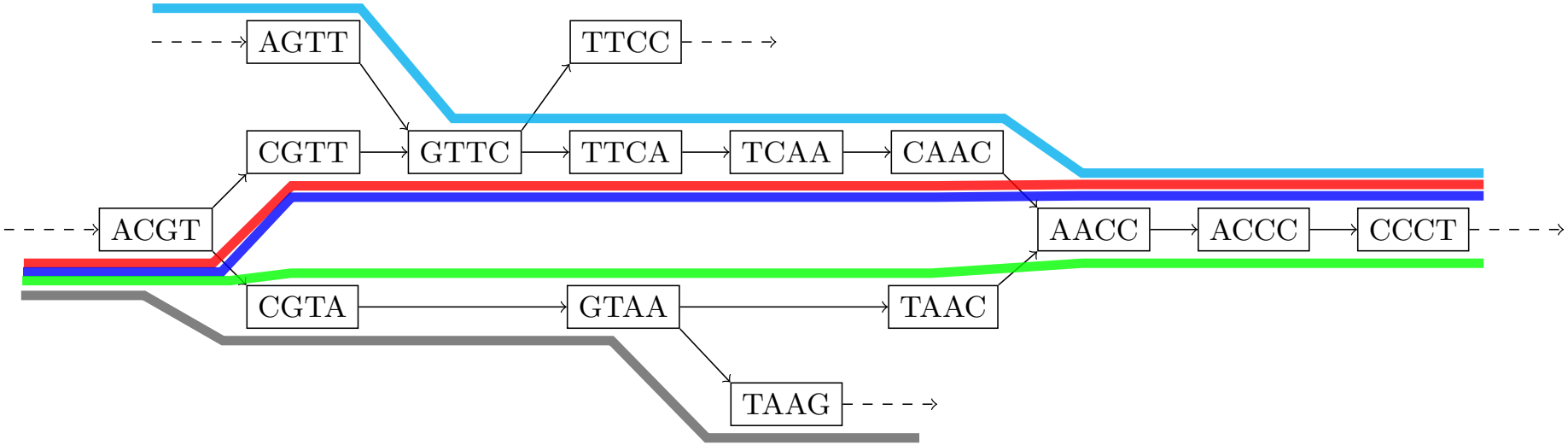
# LoRDEC*

- Build a de Bruijn graph of the LONG reads
  - Use a small $k$ such that
    the genomic $k$-mers are expected to be found in the reads
  - Use an abundancy threshold to differentiate
    between correct and erroneous $k$-mers
- For each long read:
  - Classify $k$-mers: solid (= in the DBG) and weak
  - Find paths in the DBG between the solid $k$-mers
  - Minimize edit distance between the long read and the path's string
  - Select a correcting path only if all possibilities have been explored.



# LoRMA

- Build a de Bruijn graph of the reads
- Annotate the graph by threading each read through the graph
- For each read find its friends, i.e. the most similar reads
- Use a multiple alignment of a read and its friends to correct the read



# Outline

# Evaluation method

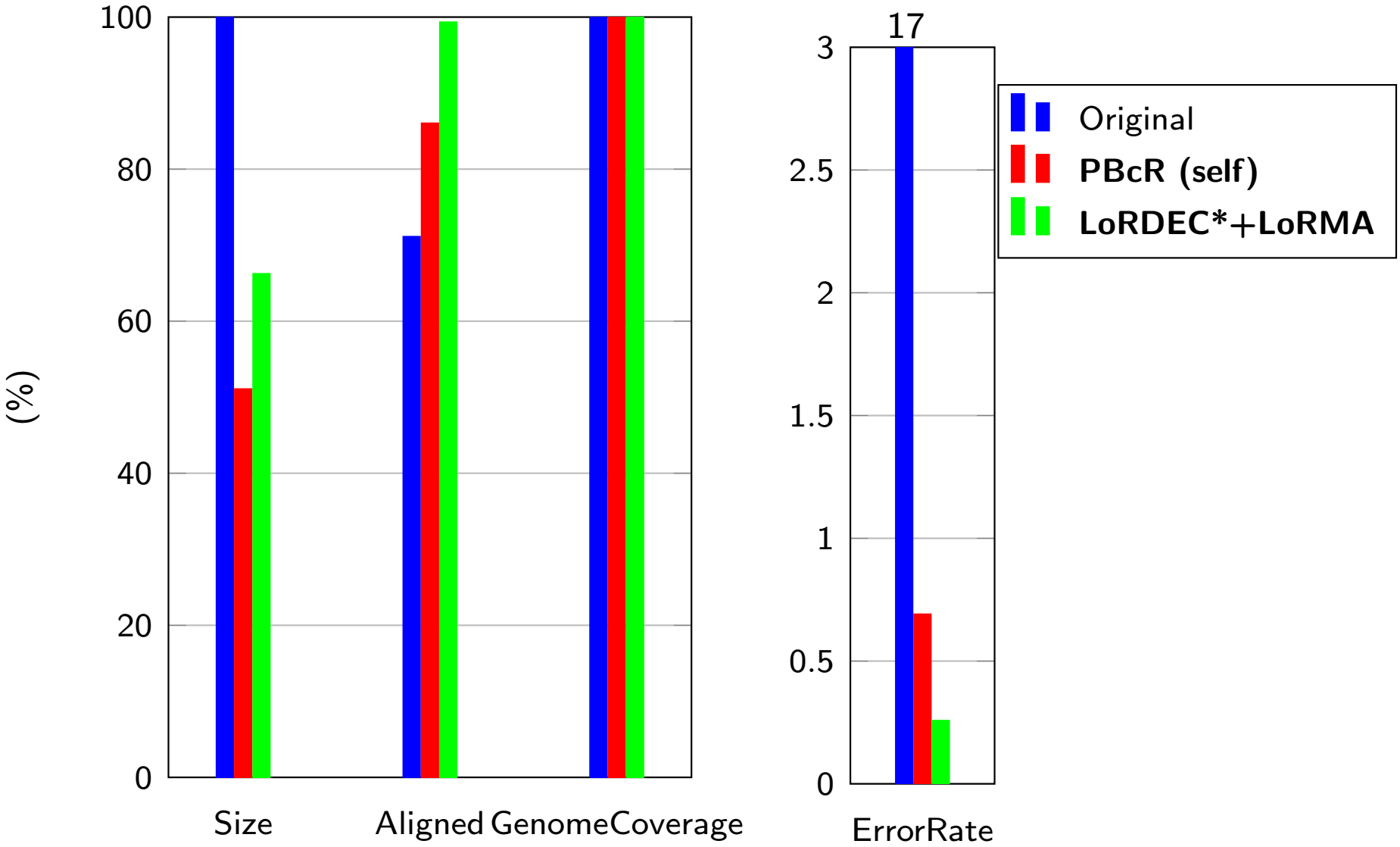
Process

1. Align the raw and corrected reads to the genome with BLASR [Chaisson et Tesler, 2012]
2. Consider a single best alignment.

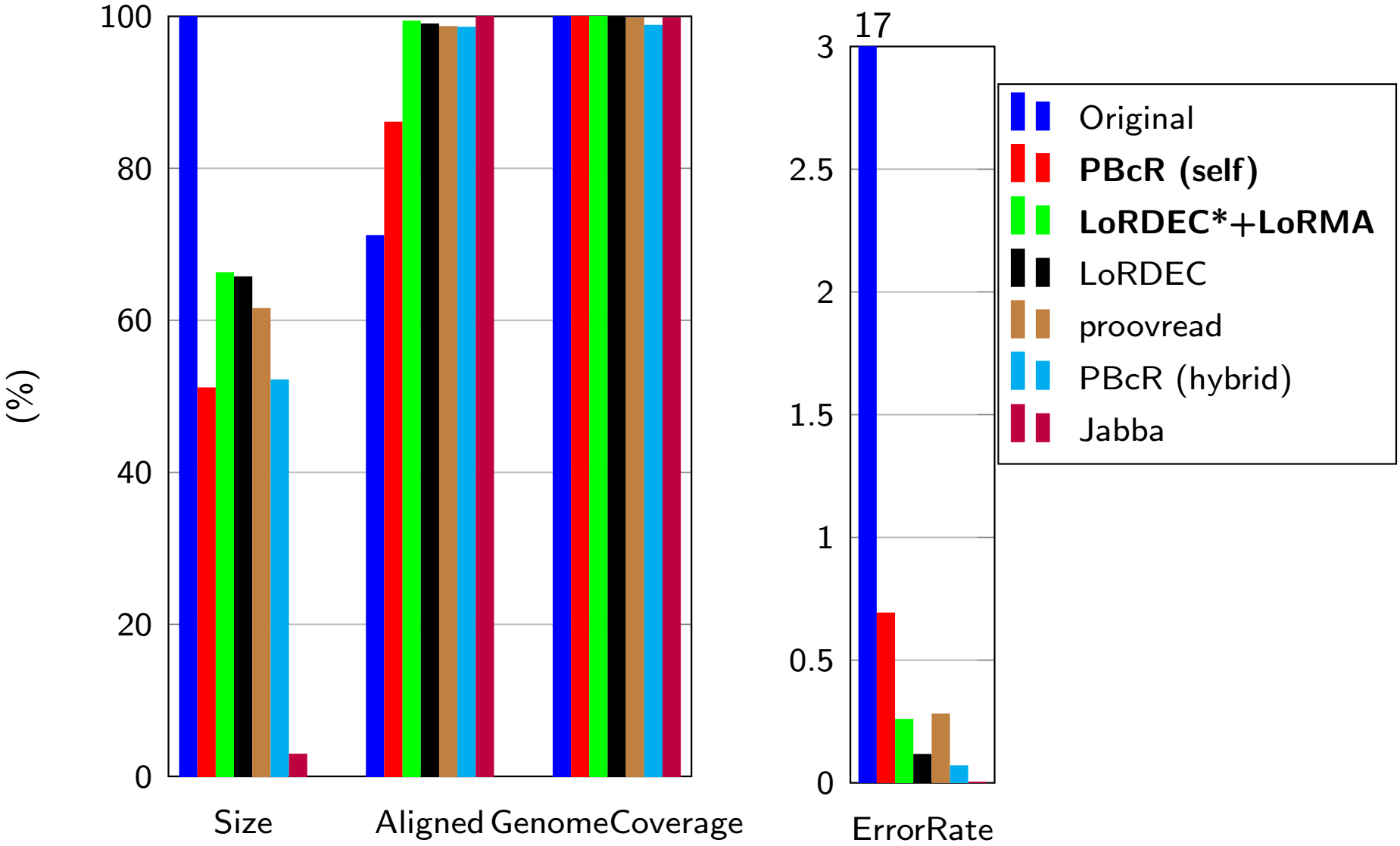
Compute following metrics

- total size of corrected reads
- total aligned size of corrected
- error rate of aligned regions (nb erroneous positions / aligned length)
- genome coverage

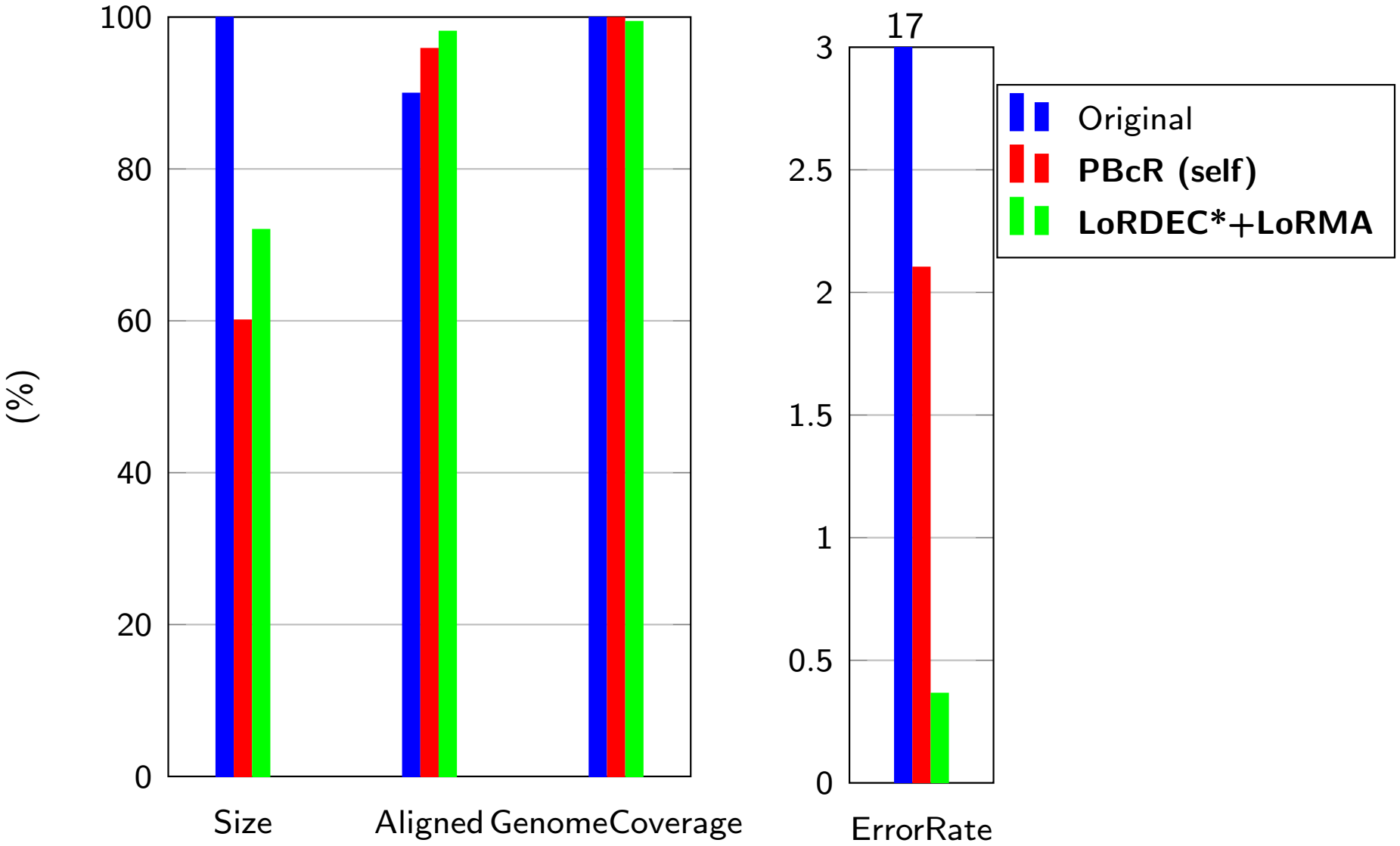
# Selfcorrection: *E. coli* with $k = 19, 40, 61$



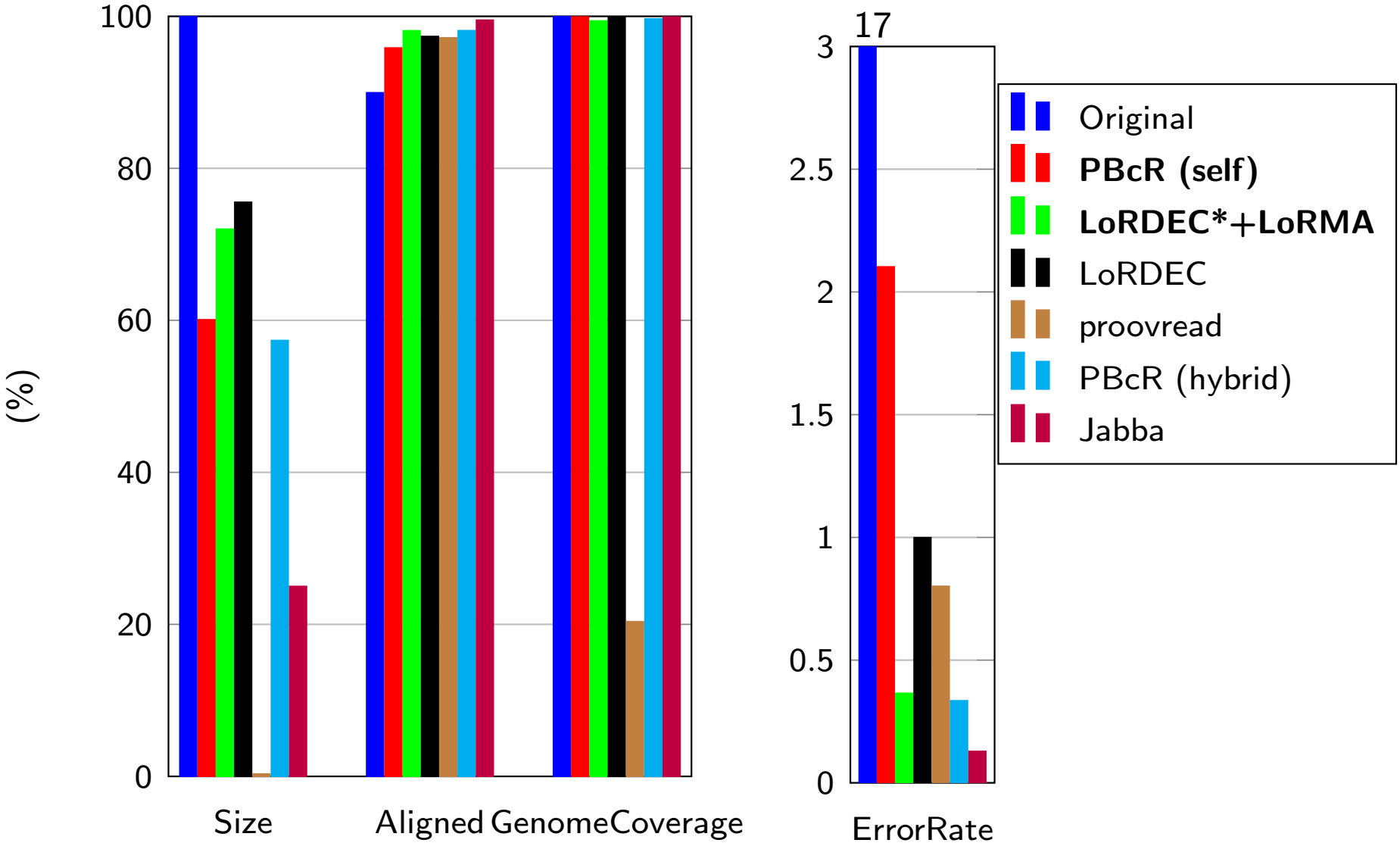
# Selfcorrection and hybrid correction: *E. coli*



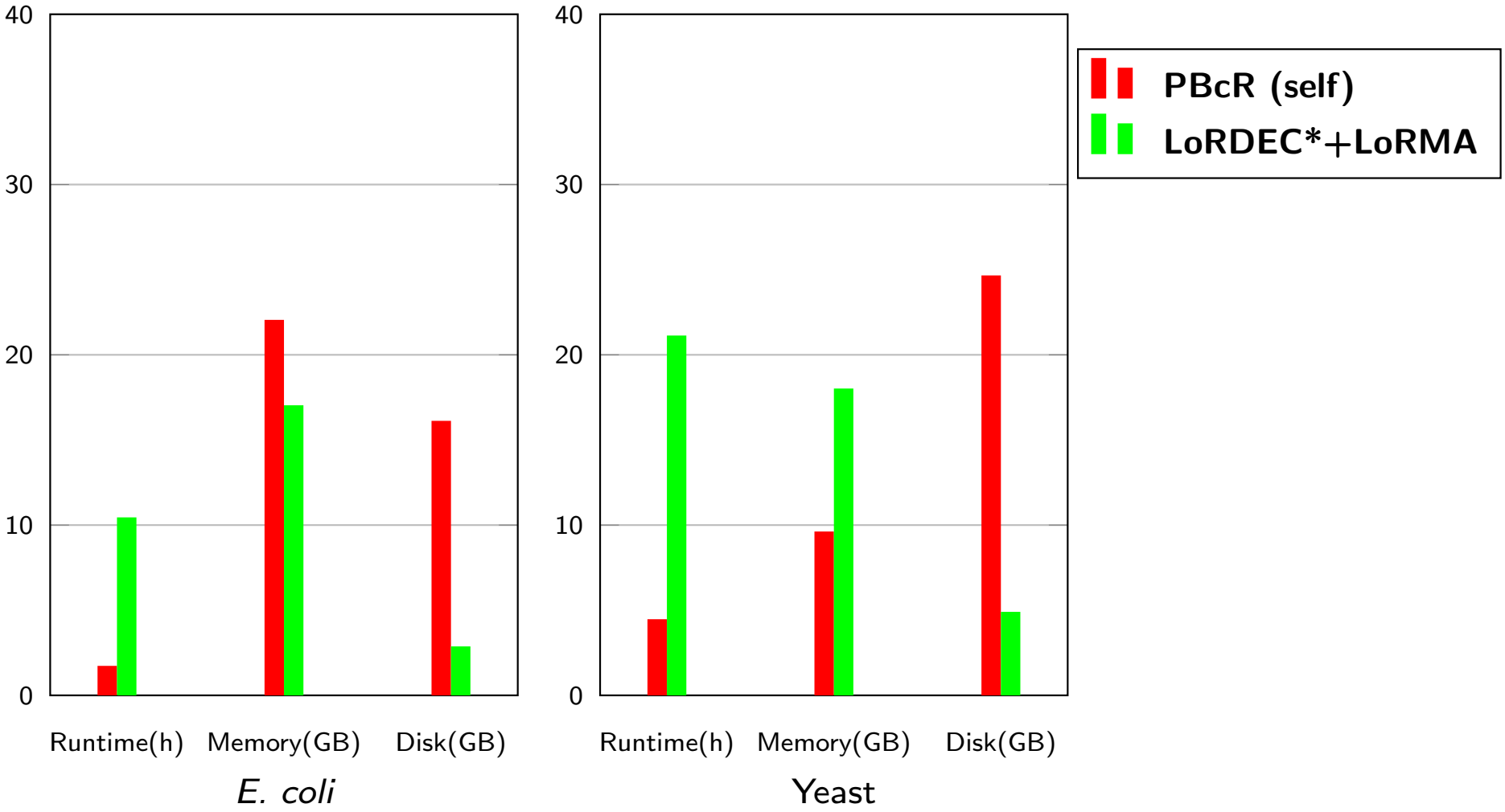
# Selfcorrection: Yeast



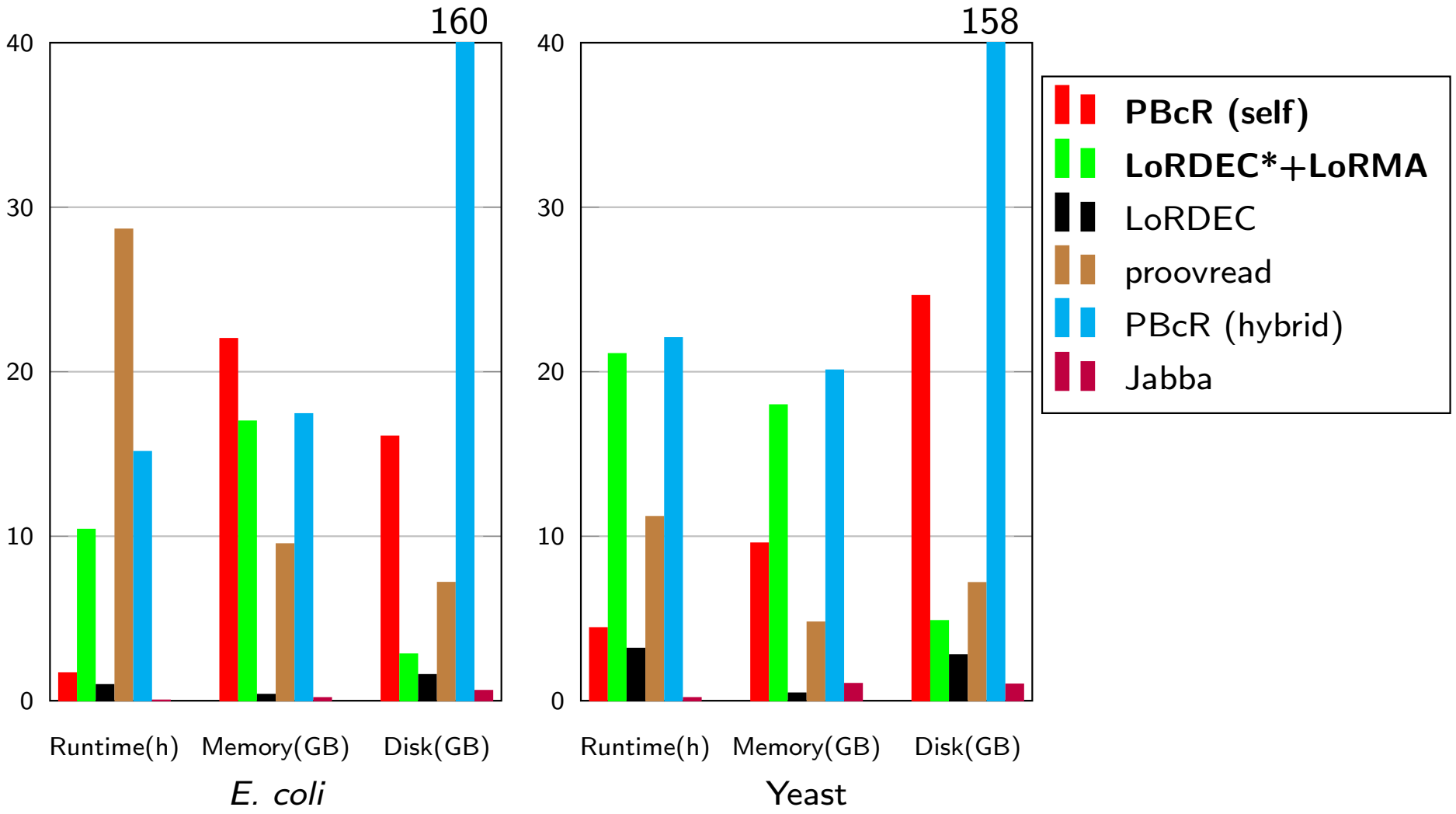
# Selfcorrection and hybrid correction: Yeast



# Selfcorrection: Resources



# Selfcorrection and hybrid correction: Resources



# Outline

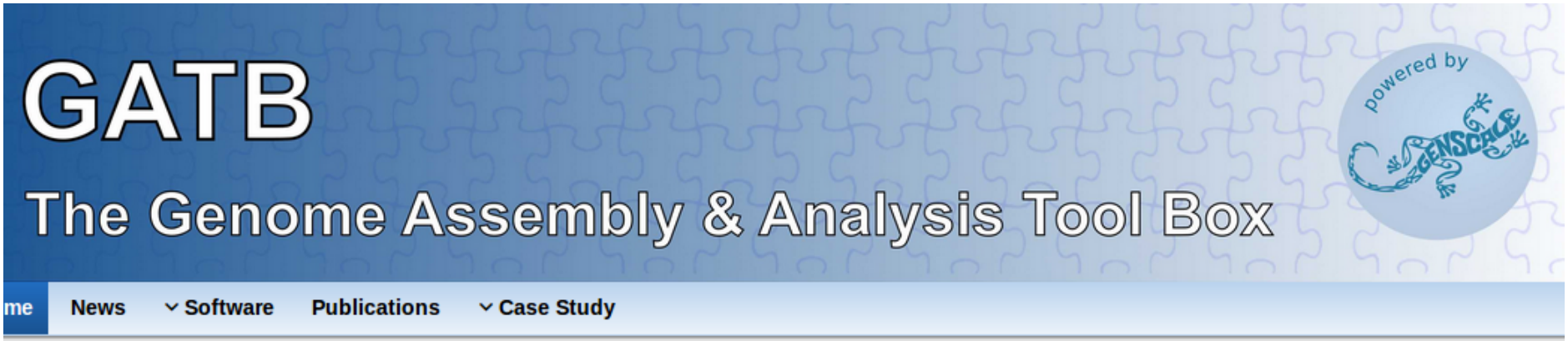
# Take home message

LoRDEC is

- at least 6 times faster than previous methods
- uses at least 93% less memory than previous methods
- corrects both PacBio & Nanopore reads
- scales up to vertebrate cases
- achieves similar accuracy as state-of-the-art methods.

LoRDEC is freely available at http://atgc.lirmm.fr/lordec/

# LoRDEC and LoRMA use GATB




http://gatb.inria.fr

# Conclusions

LoRDEC*+LoRMA [Bioinformatics 2016]:

- DBG based initial correction of sequencing errors in long read data
- Further polishing with multiple alignments
- Accurate selfcorrection method, needs high coverage (75×)
- Future: improve memory footprint and running time
- Freely available at http://www.cs.helsinki.fi/u/lmsalmel/LoRMA/

# LoRDEC and LoRMA publications

*LoRDEC: accurate and efficient long read error correction*

*L. Salmela, E. Rivals*

*Bioinformatics, doi:10.1093/bioinformatics/btu538, 30 (24): 3506-3514, 2014.*

*Accurate selfcorrection of errors in long reads using de Bruijn graphs*

*L. Salmela, R. Walve, E. Rivals, E. Ukkonen*

*Bioinformatics, doi: 10.1093/bioinformatics/btw321, 2016.*

# Funding and acknowledgements

UNIVERSITY OF HELSINKI
FACULTY OF SCIENCE

HELSINKI INSTITUTE FOR INFORMATION TECHNOLOGY

ACADEMY OF FINLAND

SYSCOL
SYSTEMS BIOLOGY OF COLORECTAL CANCER

UNIVERSITÉ MONTPELLIER

LIRMM

cnrs
dépasser les frontières

AGENCE NATIONALE DE LA RECHERCHE
ANR

IBC
Institut de biologie computationnelle

Colib'read

Thank you for your attention!

Questions?

Thanks to L. Salmela, R. Wake, E. Ukkonen, A. Makrini