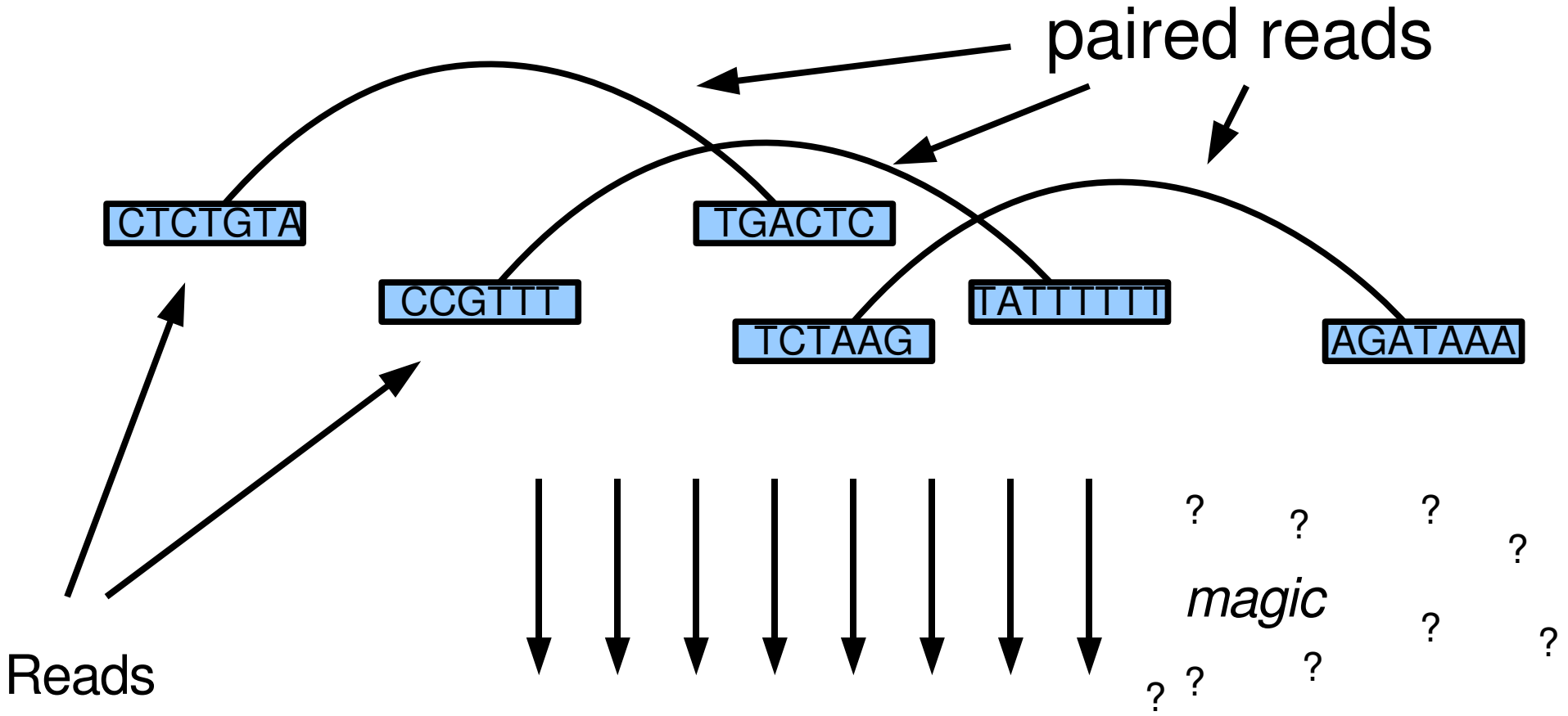
ARACHNE: A Whole-Genome Shotgun Assembler

Serafim Batzoglou,David B. Jaffe, Ken Stanley, Jonathan Butler, Sante Gnerre, Evan Mauceli, Bonnie Berger, Jill P. Mesirov, and Eric S. Lander

Presented by Ilya Sutskever

Problem: **ab-initio** genome assembly

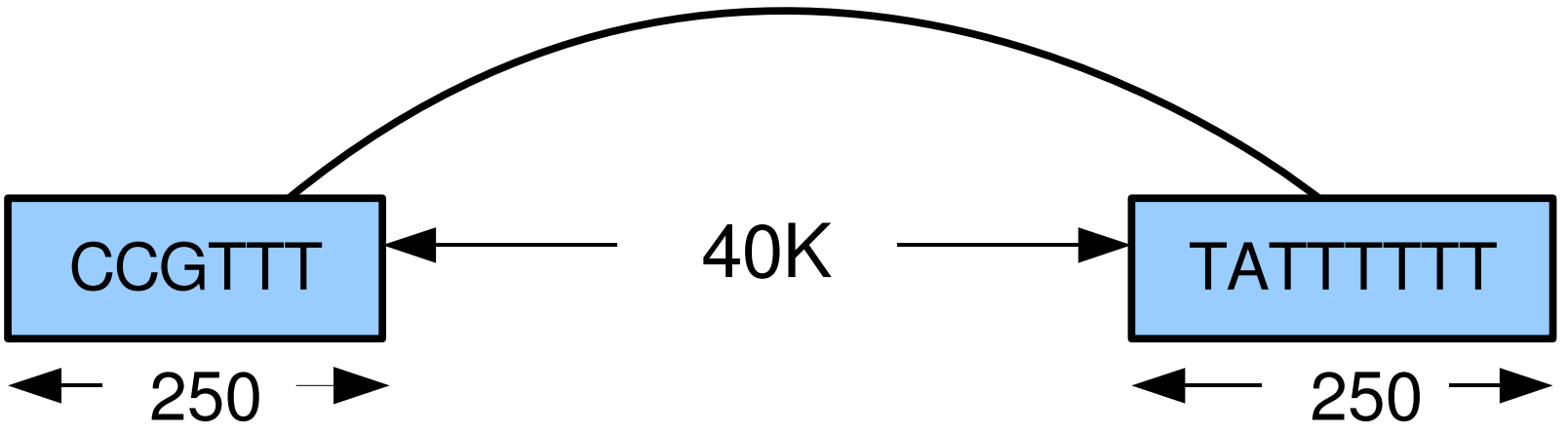


Assembled genome

ACGTACCGTTTGACTCTAGTATCTTCTAGTAGATATTTTTTTTTTAGATAAAA

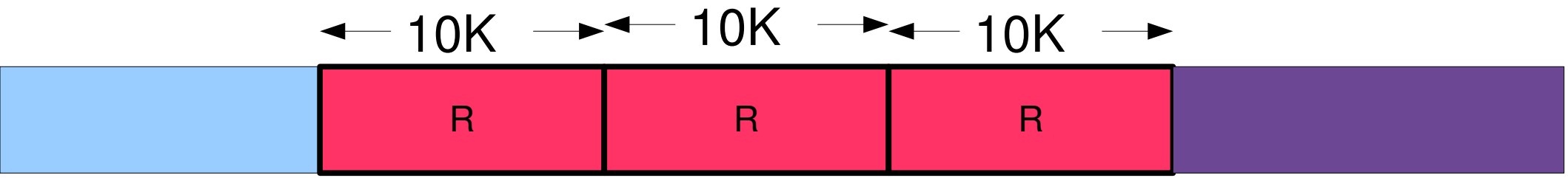
Sanger sequencing

- Recover genome from the paired reads
- Paired reads have very long known distance (40K+noise)
- Each read is moderately long (250-500)



Why whole-genome assembly hard?

- Easy If No Repeats.
  - Every method works: just grow overlapping reads.
  - May not even need paired reads.
- Almost unsolvable with repeats.
  - “Which repeat did the read come from?”
  - (Question to the audience: is it *always* true that the more repeats an organism has, the more “evolved” it is?)

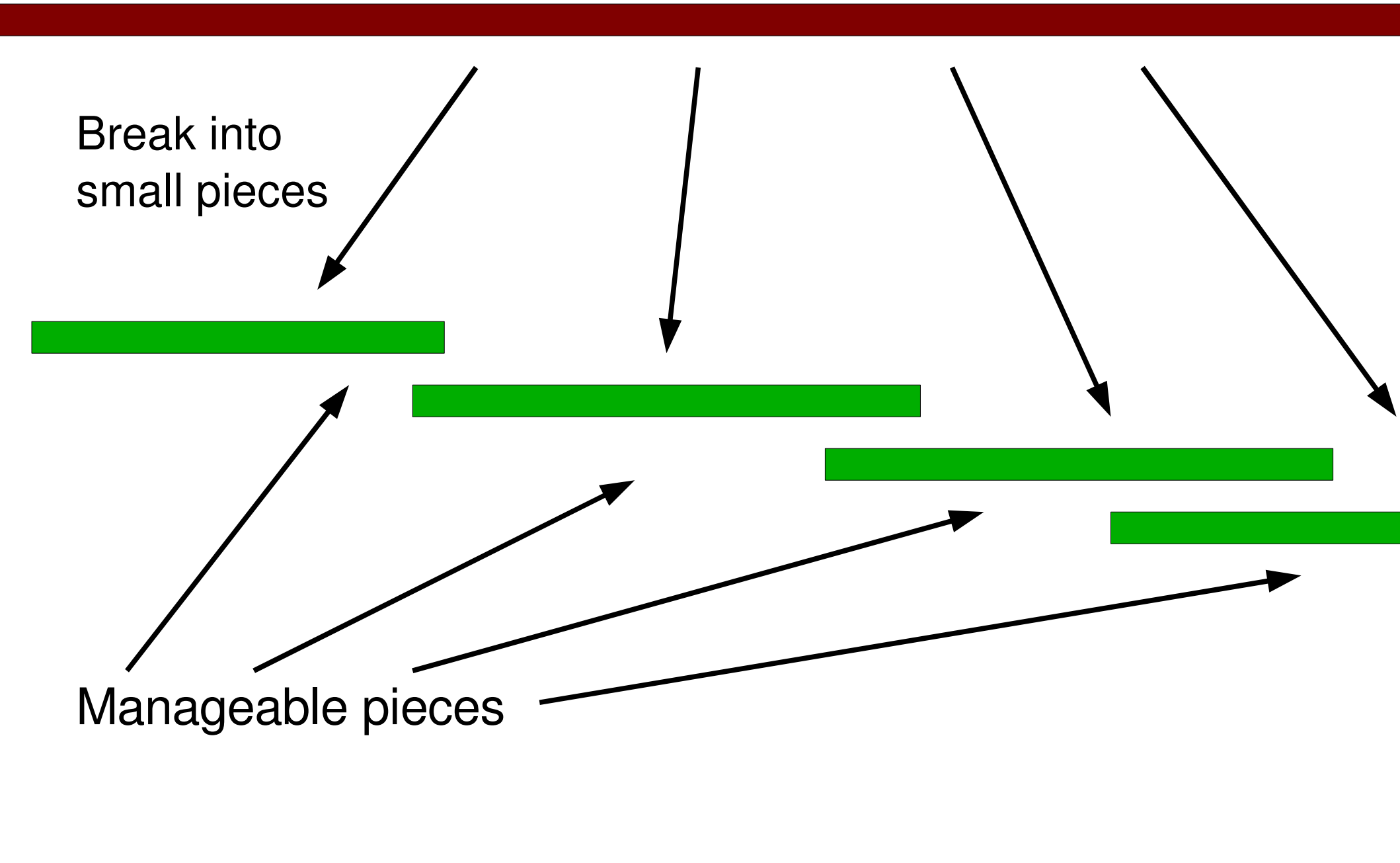


Why is it an important problem?

- Because it is **cheaper** than Hierarchical Shotgun (used to sequence human genome).
  - Divide and Conquer: break genome to small bits.
  - Sequence each bit.
  - But much more expensive than NGS.
- Has more potential for personalized genome assembly.

Hierarchical Shotgun

Original, unmanageable Genome



This paper's contribution

- An assembly algorithm that copes with repeats using Sanger reads as inputs.

Talk Outline

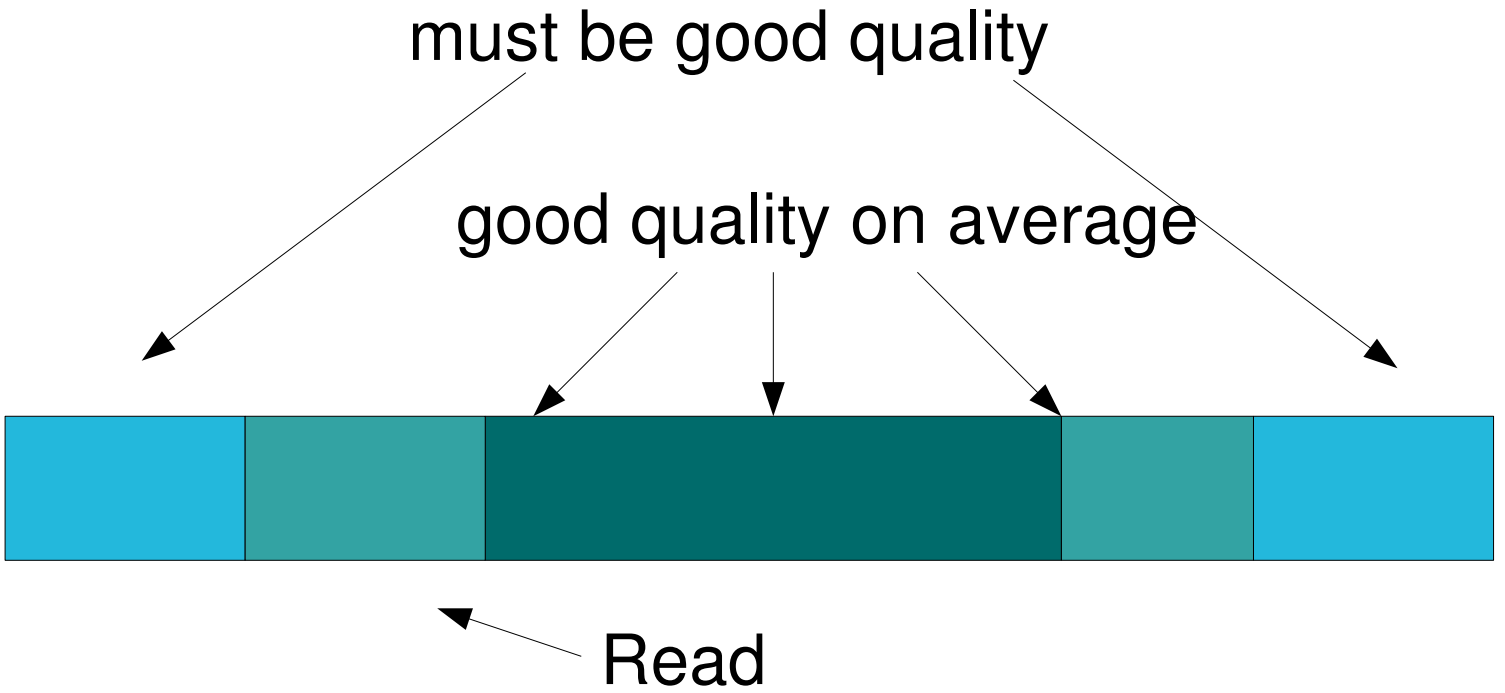
- Description of Algorithm.
- Discussion of Results.
- irrelevance to NGS.

ARACHNE: high level steps

1) Throw away low-quality paired-reads.
2) Align overlapping reads
    1) Compute neighbors.
3) Correct errors and evaluate alignmetns.
4) Grow paired reads into “good” contigs, and up to repeat boundaries.
5) Determine who is a repeat and who is not.
6) Use the repeats to fill in the gaps between the non-repeats.
7) **Output**: a few very long contigs.

Step 1: clean up data

- Make sure that all reads have a sufficiently high quality score.
- Especially Near the boundaries.
- Make sure its not similar to E. Coli genome.



Step 2: Align Overlapping Reads (to fix errors and find neighbors)

1) Use a sorted table of all 24-mers appearing in data, and their locations.

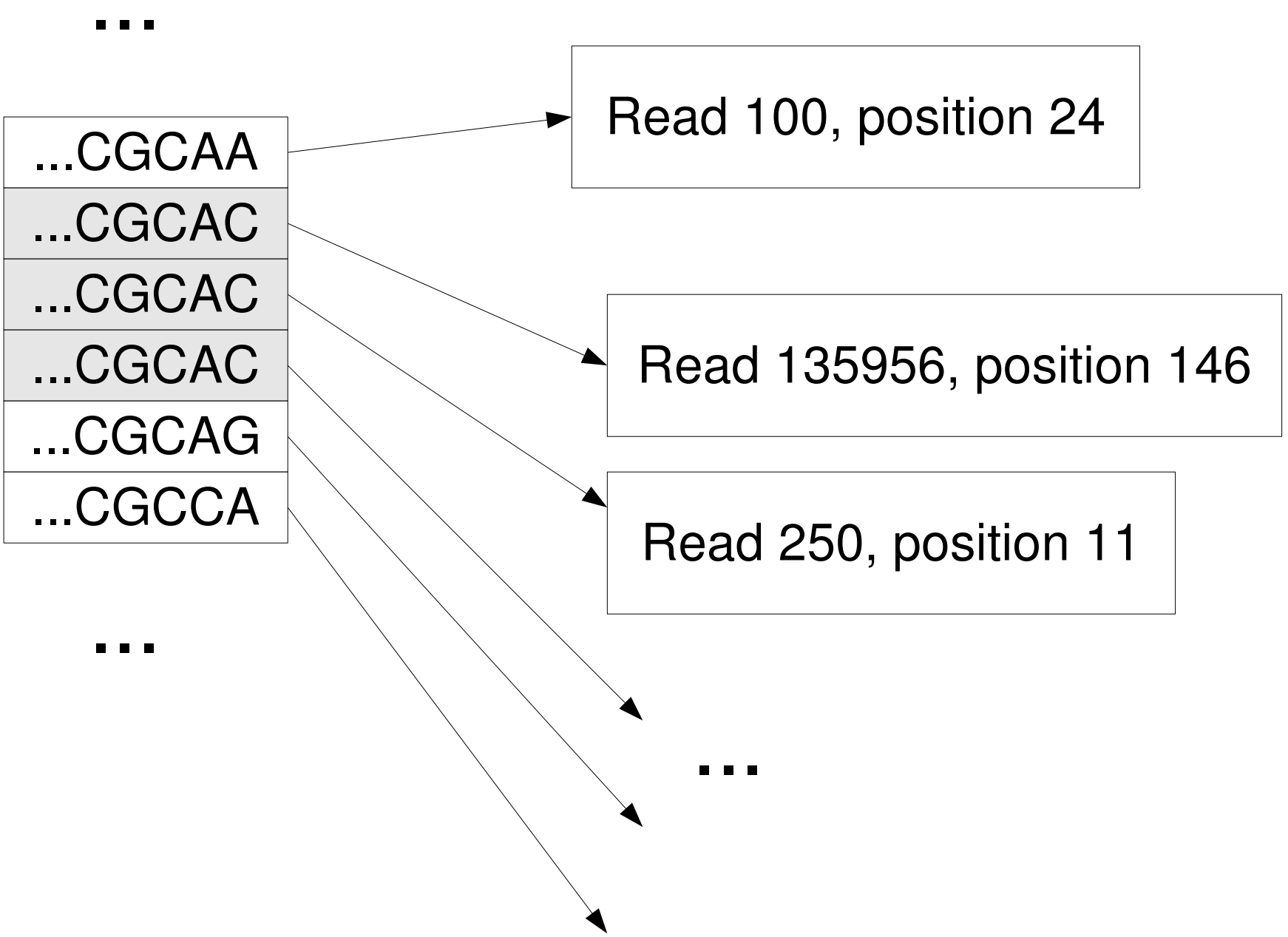
2) Produce a list of all overlapping reads.

3) Approximately align all reads sharing a 24-mer.

4) Use DP to exactly align all close-enough reads.

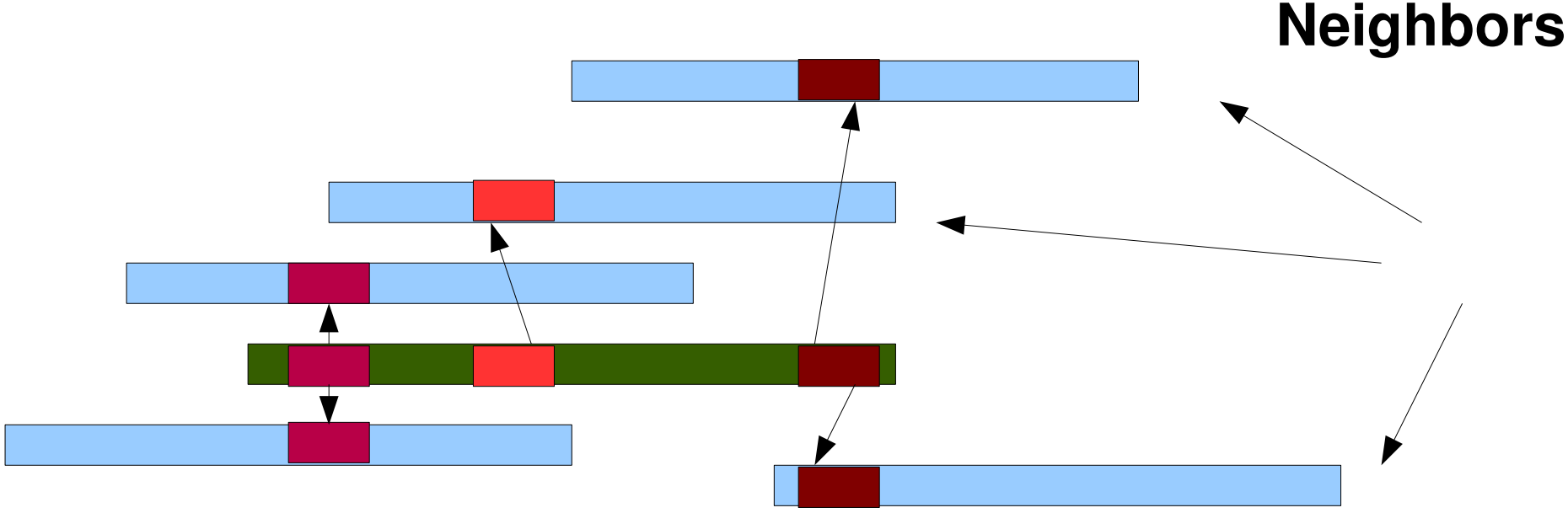
5) This is inapplicable to NGS, since the reads have length 24 at most.

Q-mer table (Q=24)



Computing Neighbors

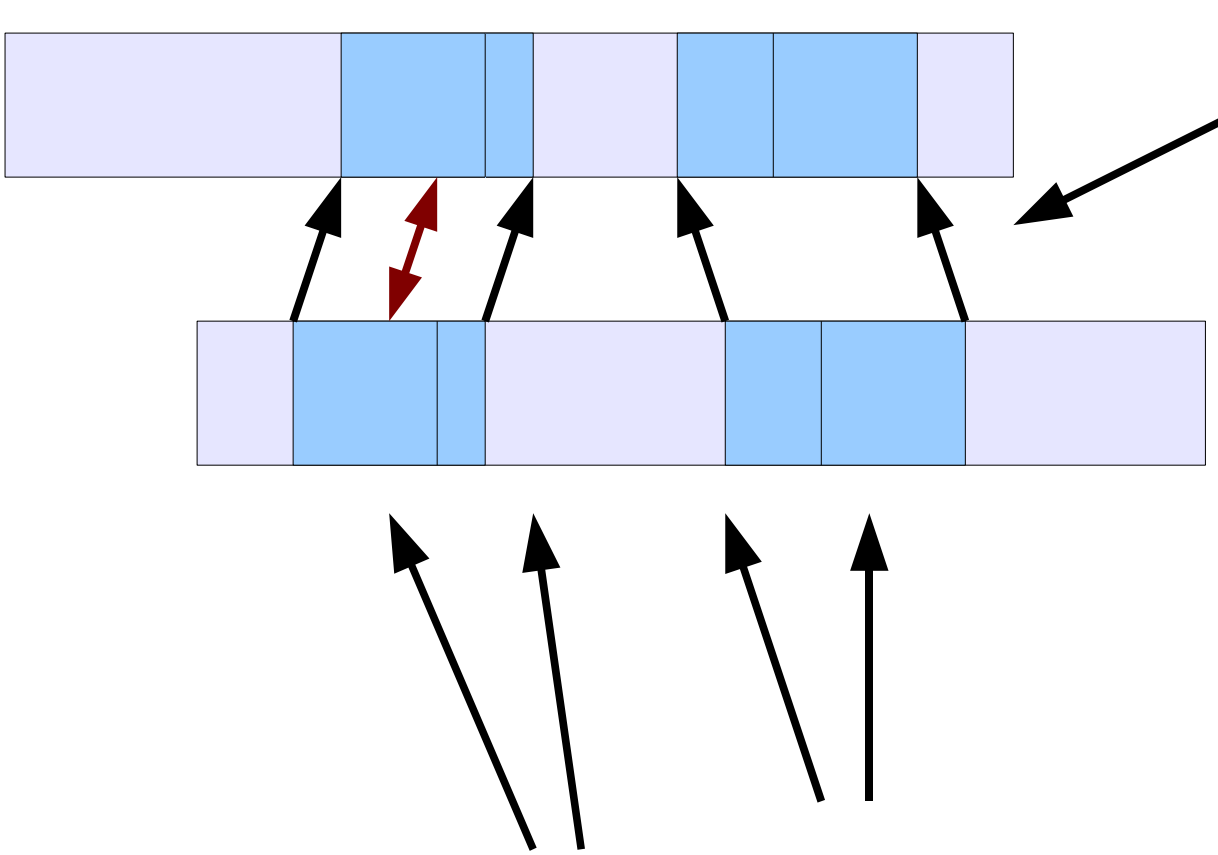
- Given a read, we can efficiently find all other reads that share a q-mer.
- Can find all “neighboring” reads efficiently.
- Essential subroutine in what follows



Align Overlapping Reads: details

- For each pair of reads sharing a Q-mer:
  - Merge overlapping Q-mers contained in both reads.
  - Extend the shared Q-mers to some alignment.
  - Refine Alignment with DP
- Note: we do not make use of the “paired” aspect of the reads here.

Aligning reads that share a Q-mer

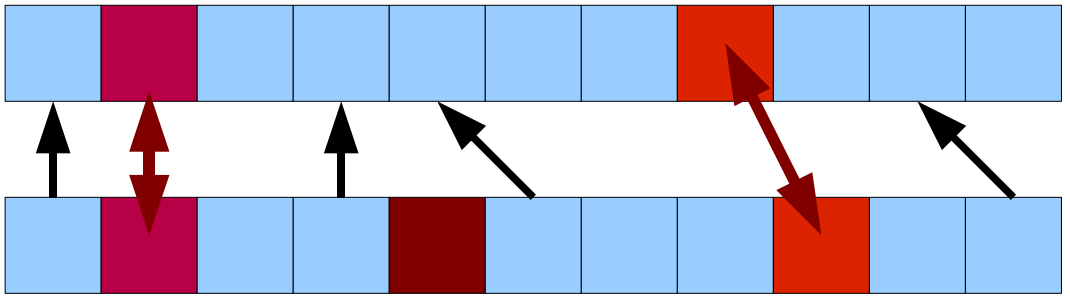


The initial alignment, to be refined by DP.

Shared Overlapping Q-mers are merged. Some mistakes are allowed. This initializes an alignment.

Details regarding alignments

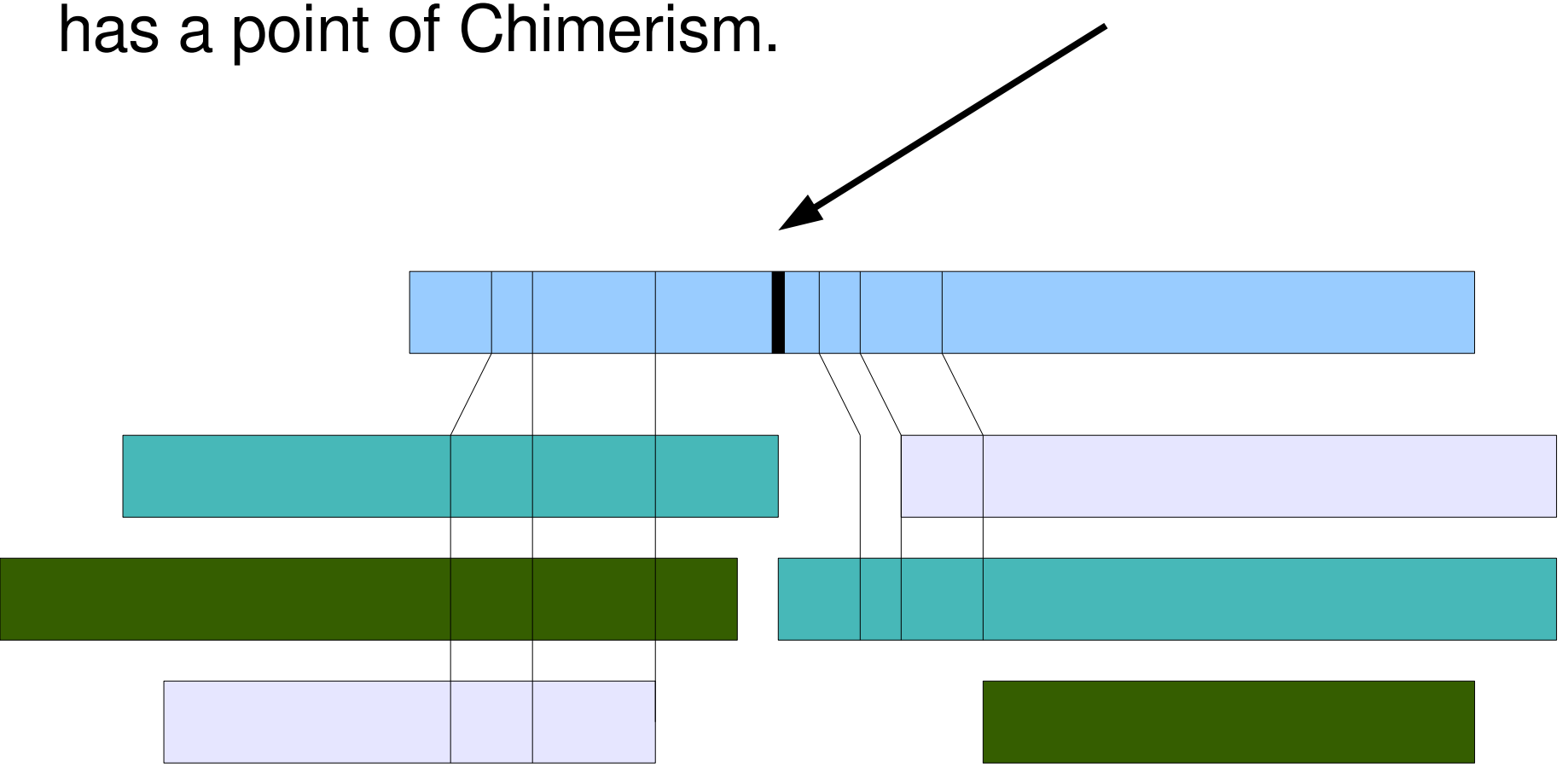
- Each alignment has a penalty score: the amount of change it makes, depending on the quality of the bases.



- Very bad alignment disqualify both reads.
- Chimeric reads are also removed.
- Reads are error-corrected to match the majority vote.

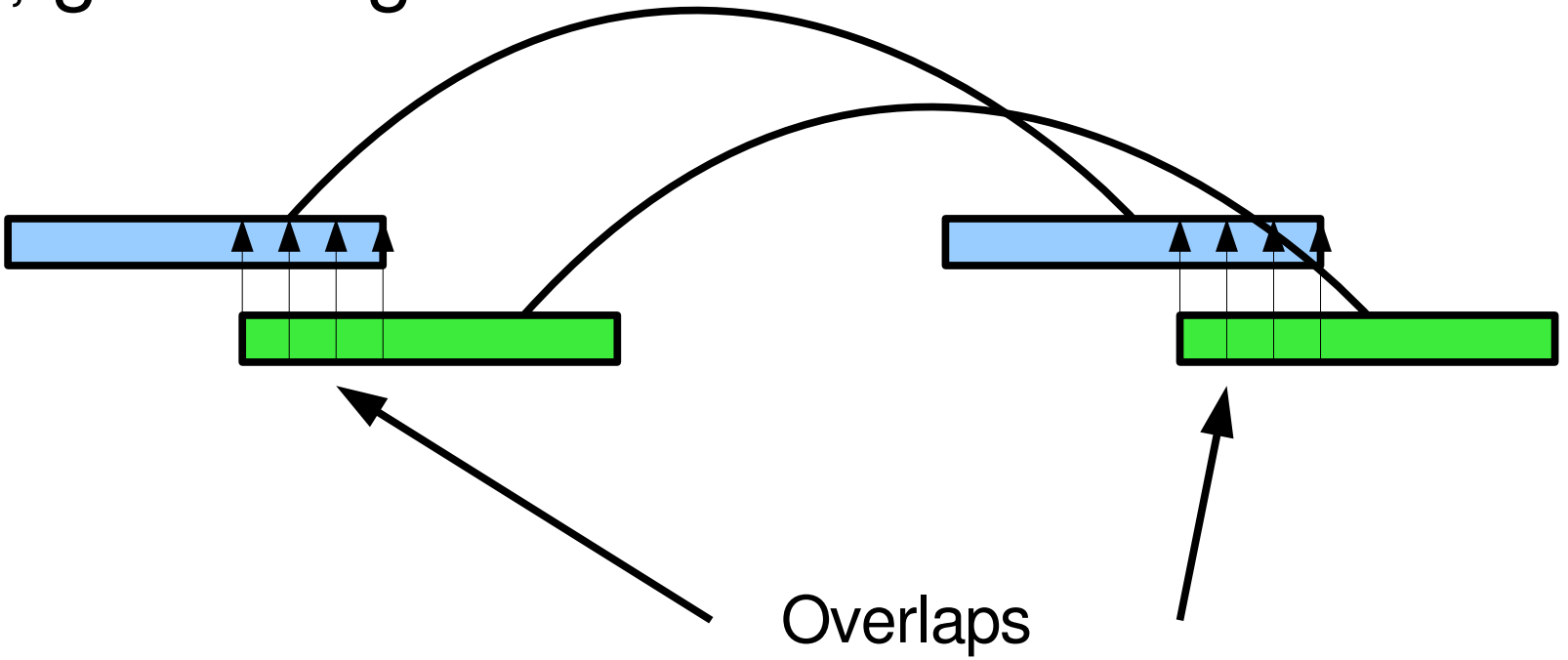
Chimeric Reads detection

Chimeric Read. To find it, the algorithm verifies that it has a point of Chimerism.



Assembling Contigs

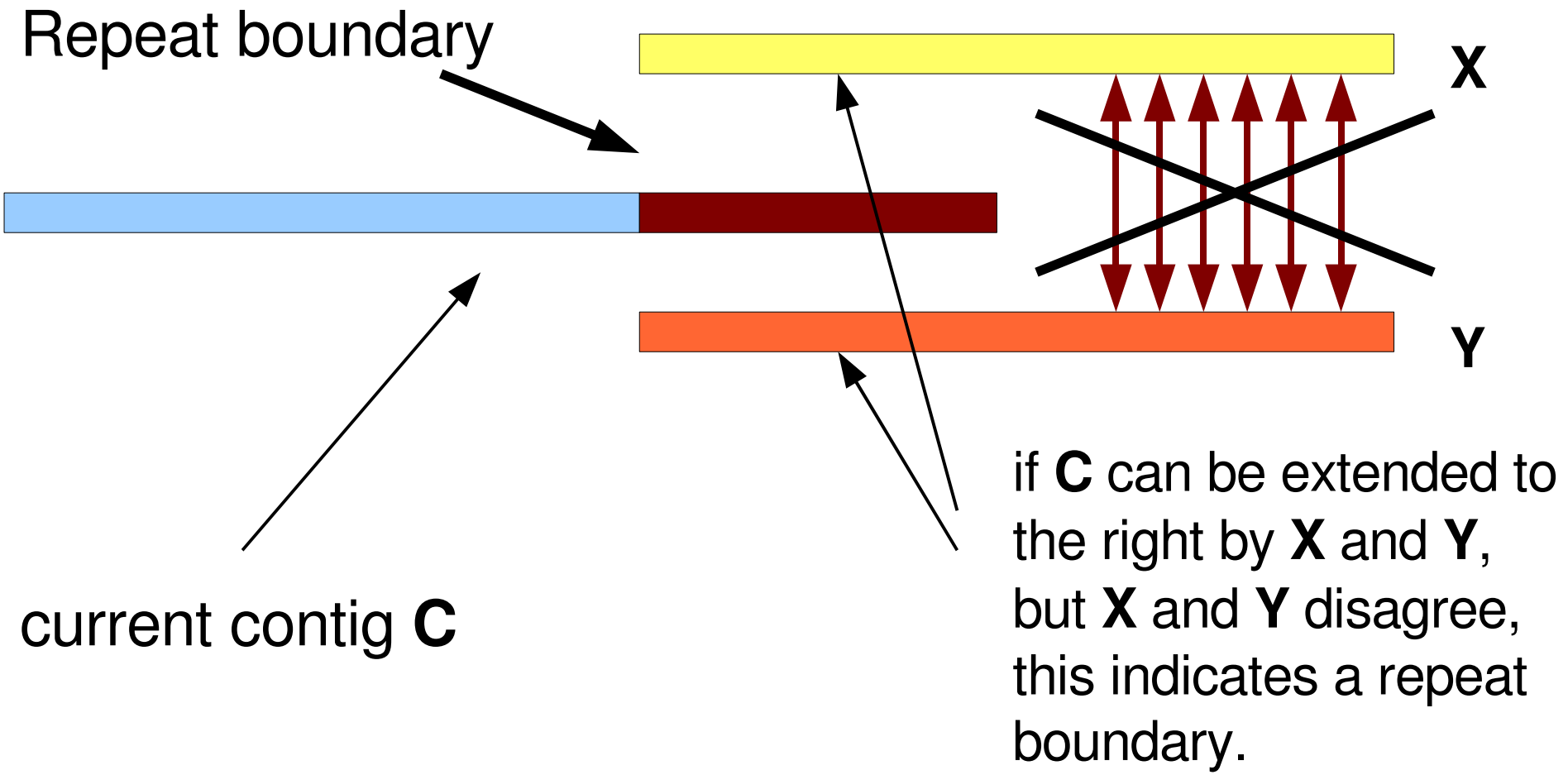
- Merge pairs of reads if they overlap on both ends, get contig:



- Treat the contig as a large paired read;
- Iterate.

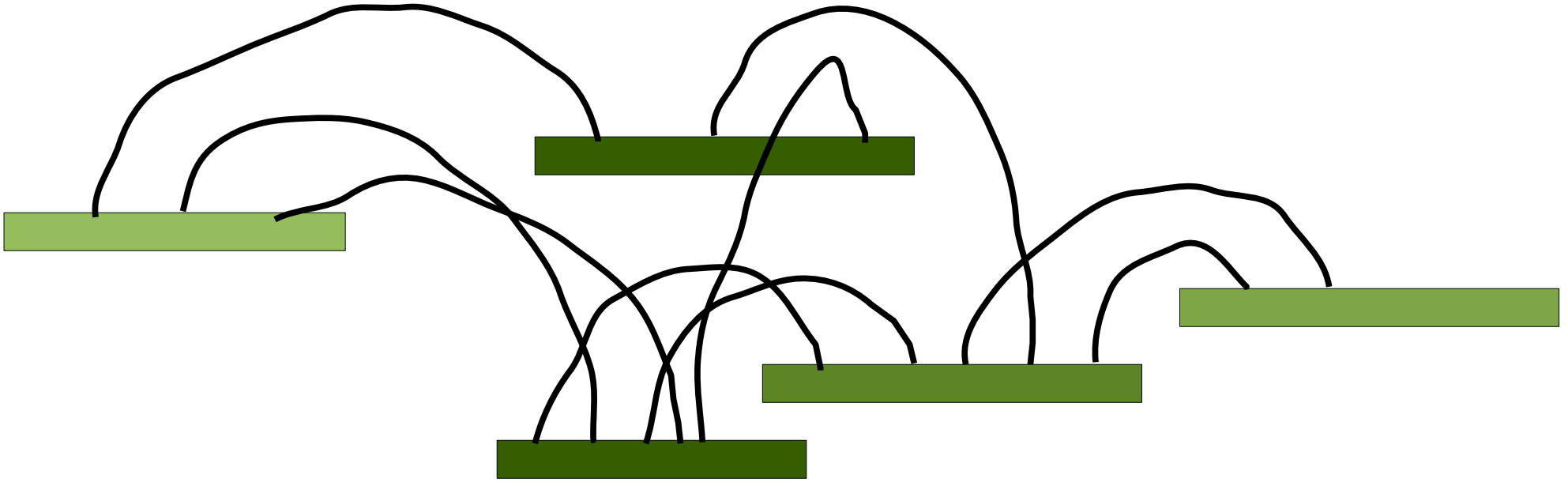
But avoid repeat boundaries.

- Check if a position is a repeat boundary:



What do we have?

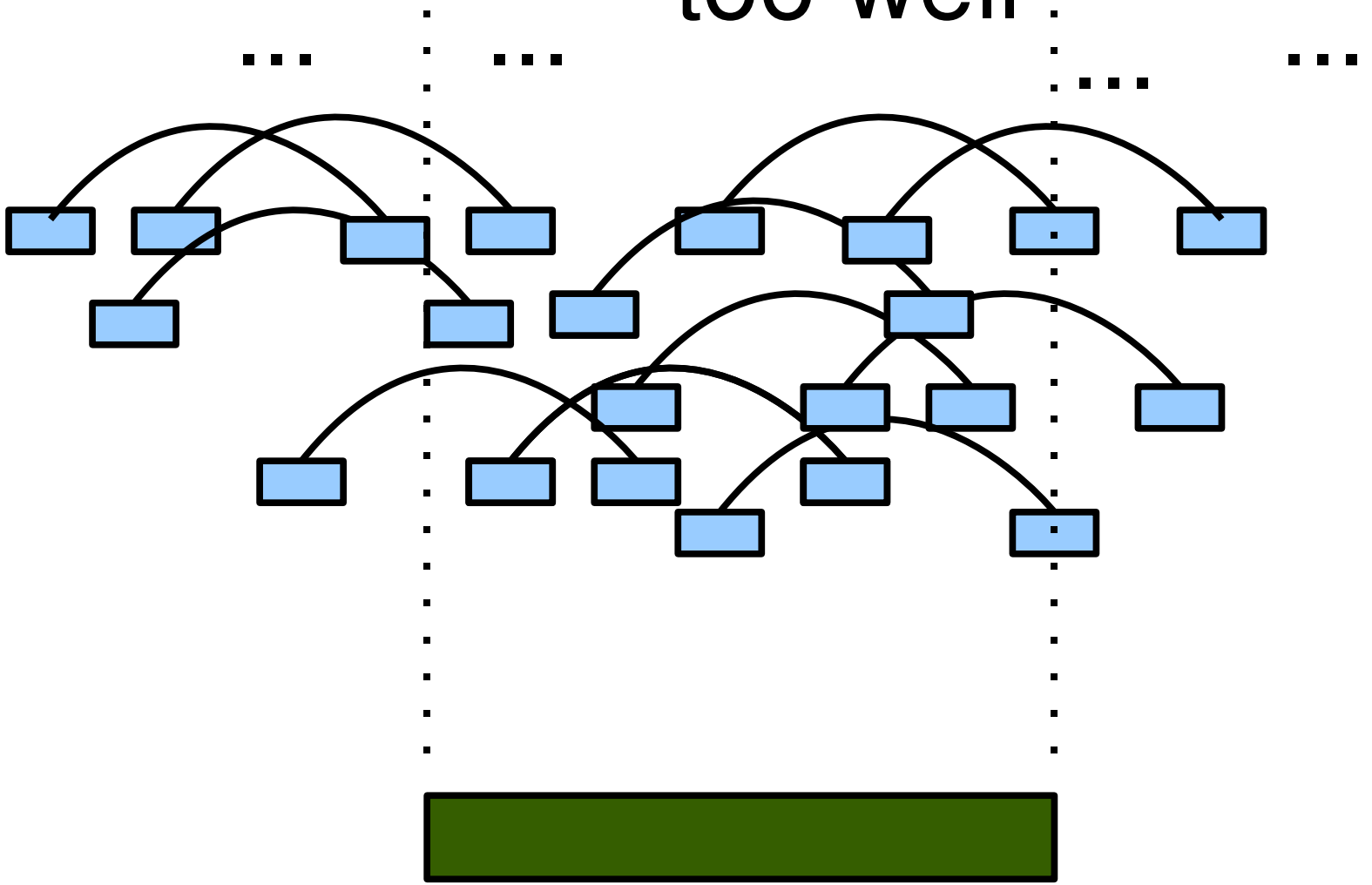
- We have long contigs with long distance “links”, most of which do not cross repeats boundaries.



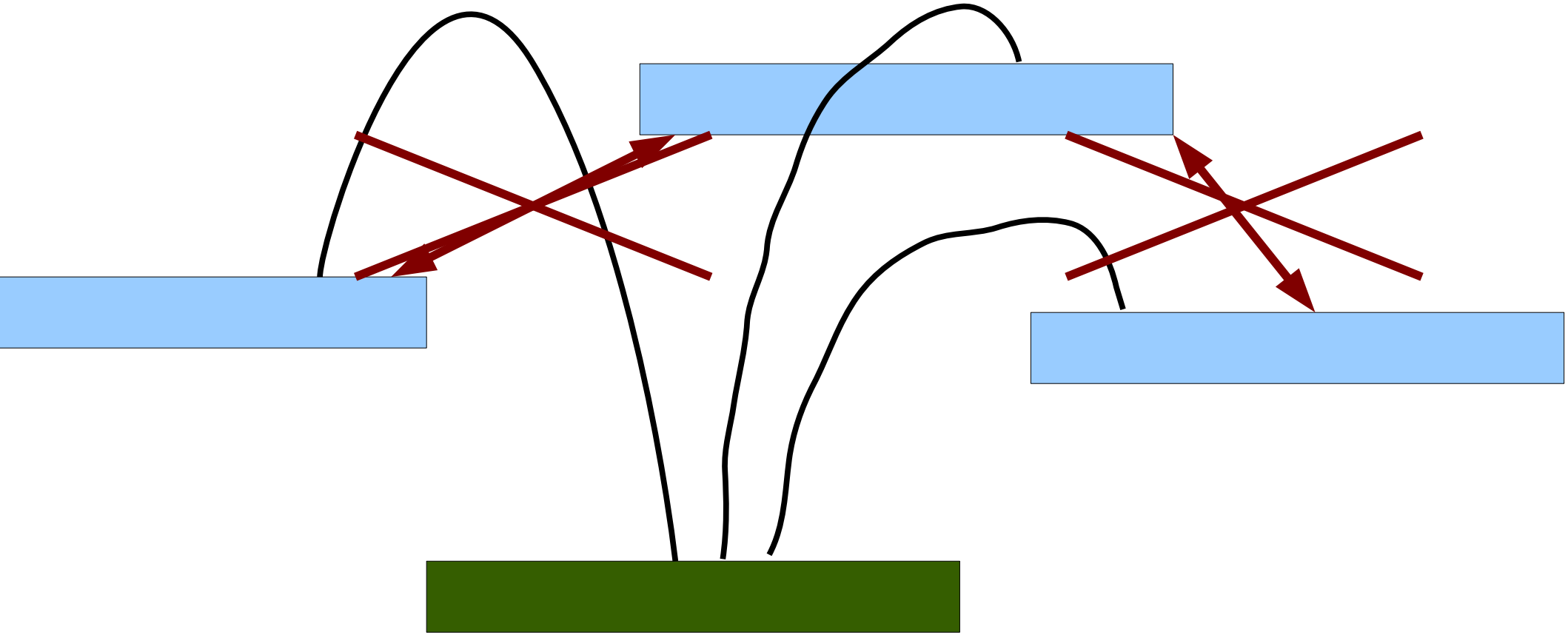
Which contig is a repeat?

- We can grow contigs that mostly avoid repeat boundaries.
- So each contig is either a repeat or a non-repeat.
- A contig is a repeat if
  - they have high depth of coverage
  - links to conflicting contigs

Repeat contig detection: covered too well

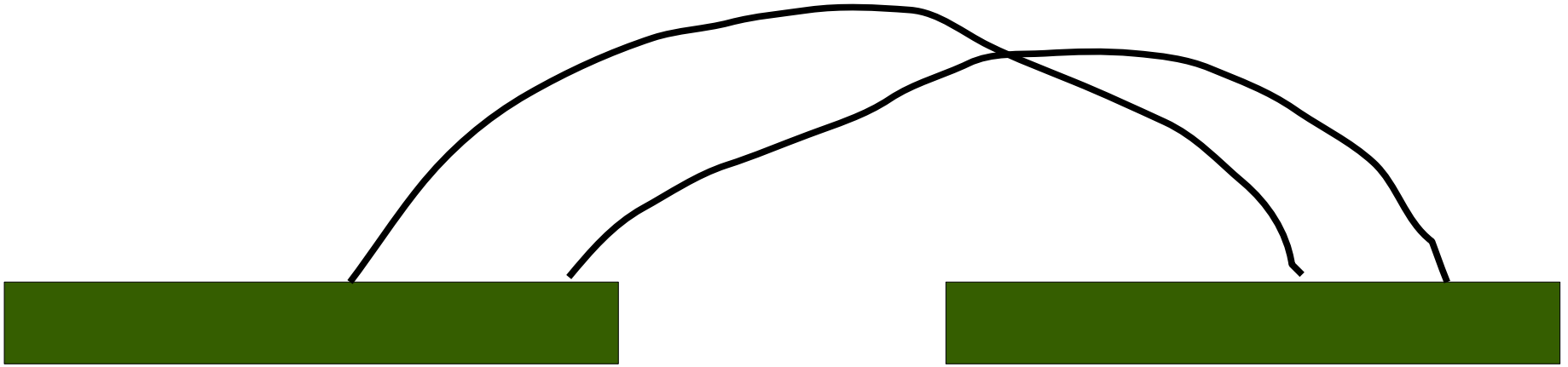


Repeat contig detection: links to highly nonoverlapping contigs



Assembling Supercontigs

- Take all non-repeating contigs.
- Using the links, join super contigs. But there can be gaps now.



non-repeat contig

non-repeat contig

Fill the gaps with repeats

- Use the links from the repeat configs to fill the gaps.
- If a repeat config has enough links, it can be used to fill the empty space.
- Obtain a small number of very long contigs.

Results

- Synthetic experimental data:
  - Take a good genome
  - Produce reads at random
  - Assign realistic quality scores (by matching to existing reads)
- But: the reads are not taken uniformly from the genome.
- 10-fold and 5-fold coverage.
- Links: 40K and 4K, ratio 20:1 or 10:1

Table of results (10-fold coverage)

	H. Influenzae	*S. cerevisiae*	*D. melanogaster*	*Human 21*	*Human 22*
Length (MB)	1.8	12	120	33.8	33.5
% Gen. in contigs	98.80%	96.10%	97.90%	96.70%	95.30%
Supercontig:					
N50 Length (KB)	1192	1177	5143	3986	3011
BP accuracy	**45.3**	**43.6**	**43.4**	**42.8**	**41.3**
Missasemblies:	2	6	115	14	32
Mean insert length		350	990		400
Mean delete lengt	440	470	1660	360	430

Table of results (5-fold coverage)

	H. Influenzae	*S. cerevisiae*	*D. melanogaster*	*Human 21*	*Human 22*
Length (MB)	1.8	12	120	33.8	33.5
% Gen. in contigs	97.10%	92.40%	95.40%	95.00%	92.00%
Supercontig:					
N50 Length (KB)	629	1732	4258	3278	3197
BP accuracy	**32.3**	**32.6**	**33**	**32.3**	**32.1**
Missasemblies:	6	6	175	43	63
Mean insert lengtl	380		670	90	390
Mean delete lengt	290	3790	1600	220	340