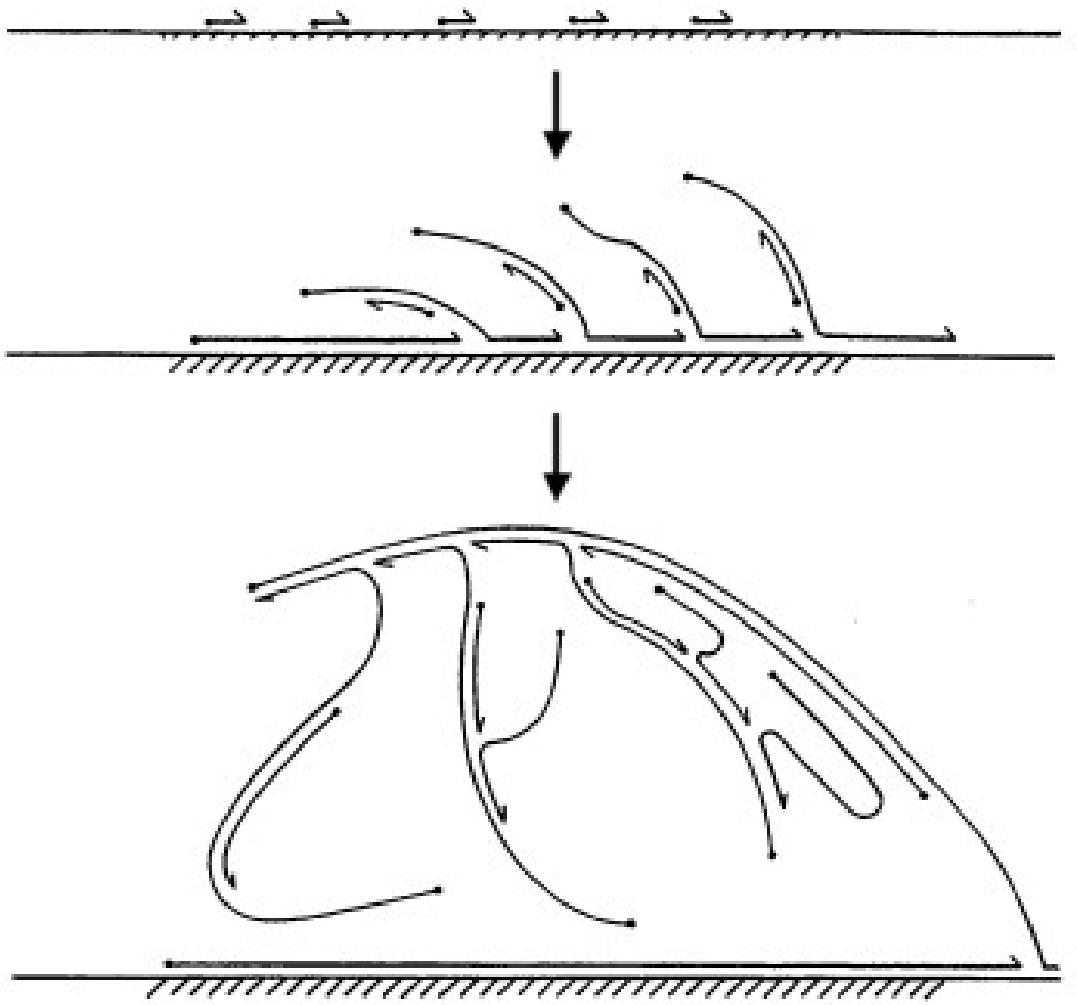
Don't throw away *chimeras!*
(comics edition)

Petar Ivanov

Mentors:
Sergey Nurk
Anton Bankevich

Jul–Aug 2013
Algorithmic Biology Lab
St. Petersburg Academic University

Background: MDA



Lizardi – Multiple displacement amplification, US 6280949 B1, 2001

Background: MDA

- *Multiple Displacement Amplification* (MDA) is the currently used method for *single cell* DNA amplification
- MDA generates *chimeric* (i.e. not really existing in the genome) DNA rearrangements in the amplified DNA
- All the genome assemblers try to reduce and eliminate the chimeric reads

...but we can do better

Our dream

Make use of the proximity (span) expectation between both ends of the chimera:

- in path extending **(to be presented)**
- in scaffolding **(in the future)**

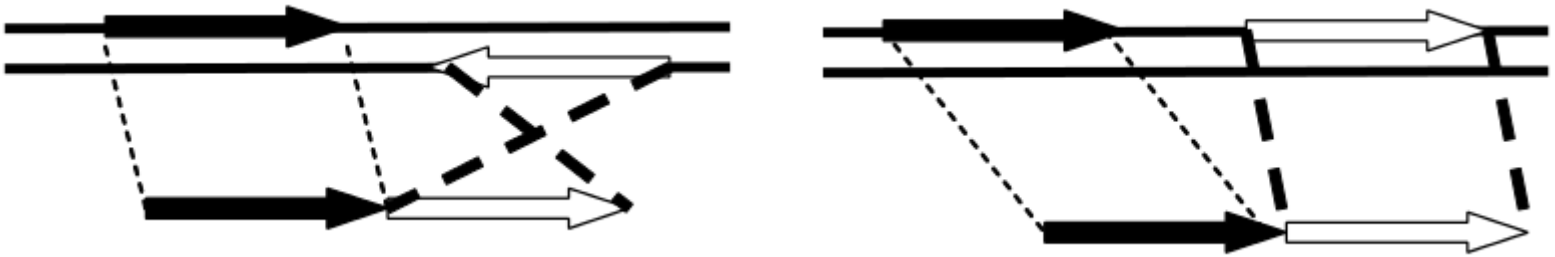
Pipeline

- (done before) MDA lane + SPAdes → simplified de Bruijn Graph → identify chimeras
- (current work) chimeras + reference → statistics
- (current work) de Bruijn Graph + statistics → chimeric path-chooser
- (future plan) integrate the chimera-flavored path-chooser in production SPAdes
- (future plan) chimera-based scaffolding

Chimeras: inverted *vs* direct

Reads 85% vs 15% (Lasken on *E. coli*)

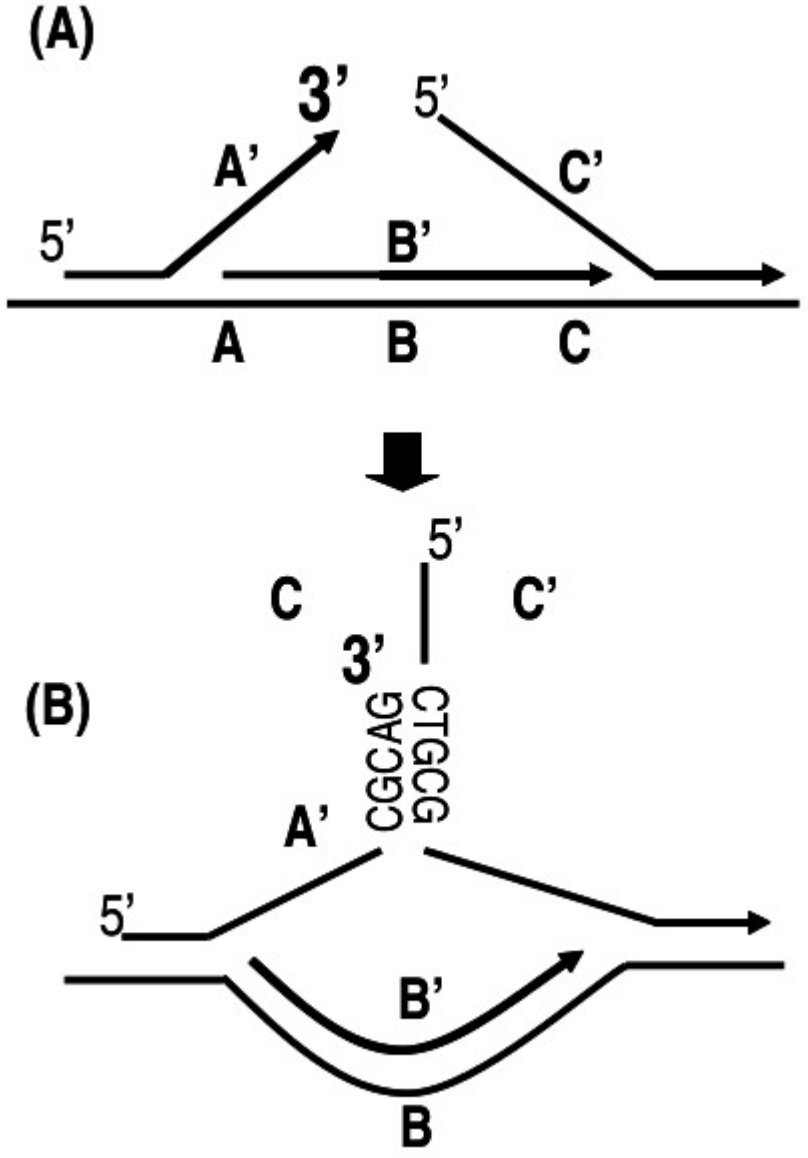
Chimeras 71% vs 29% (Ours on *E. coli*)

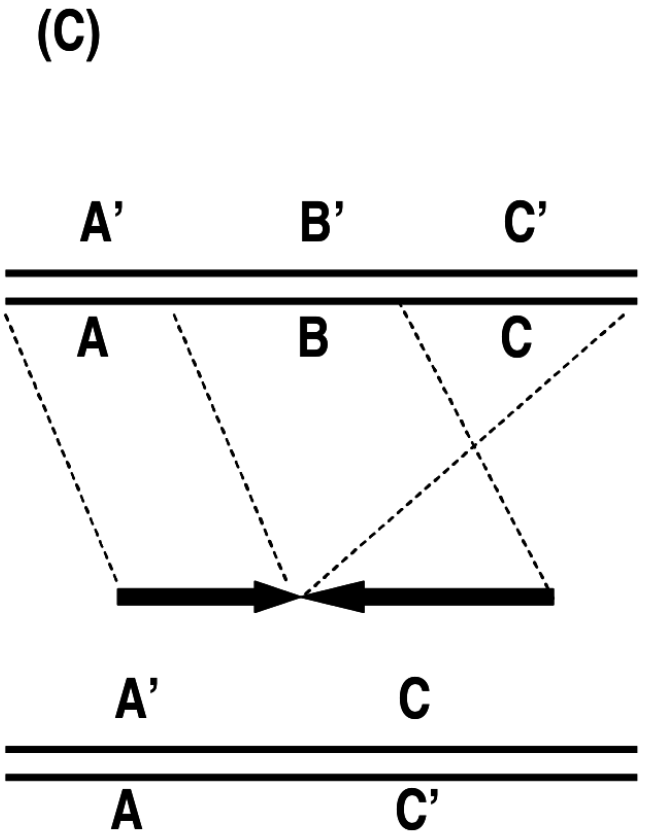


inverted direct

Lasken, Stockwell – “Mechanism of chimera formation during the Multiple Displacement Amplification reaction”, 2007

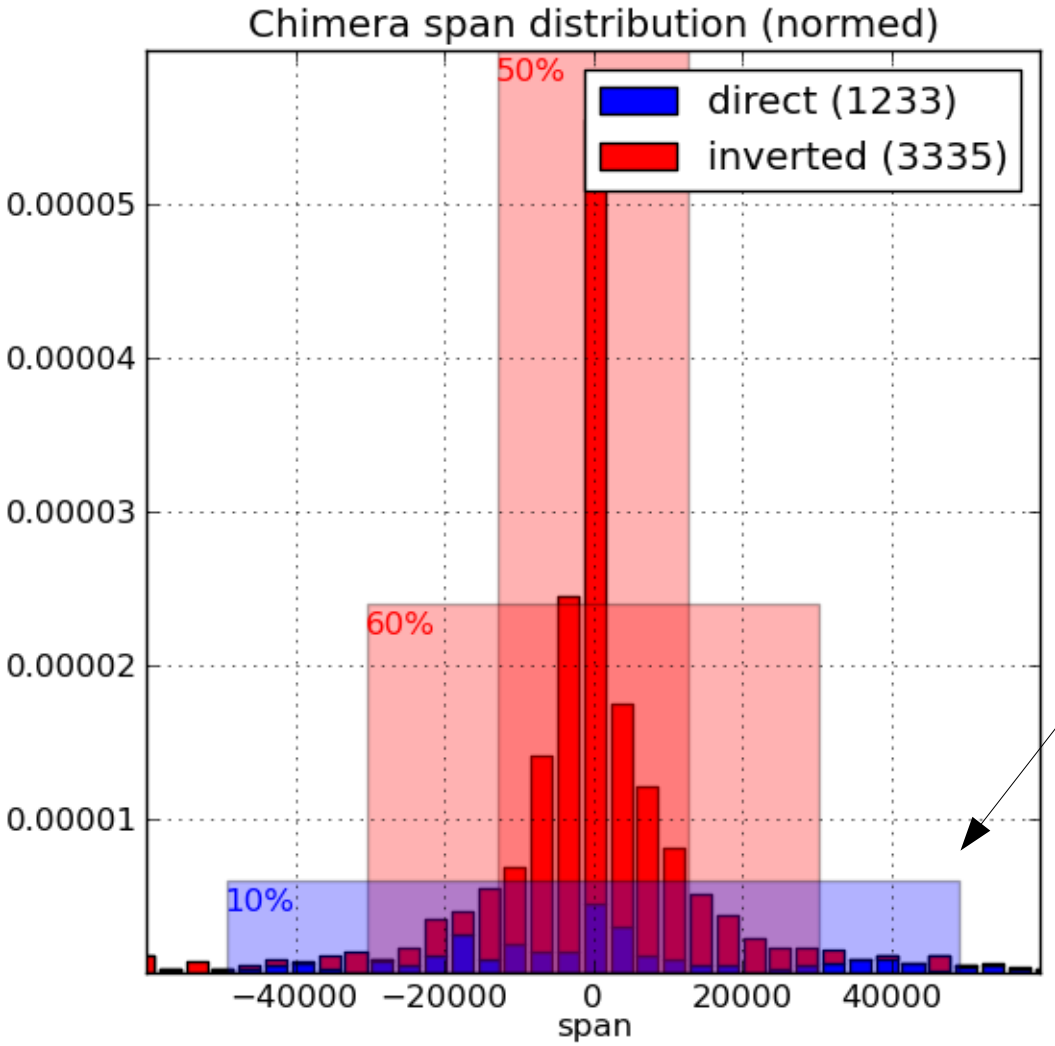
Inverted chimera formation





Lasken, Stockwell – "Mechanism of chimera formation during the Multiple Displacement Amplification reaction", 2007

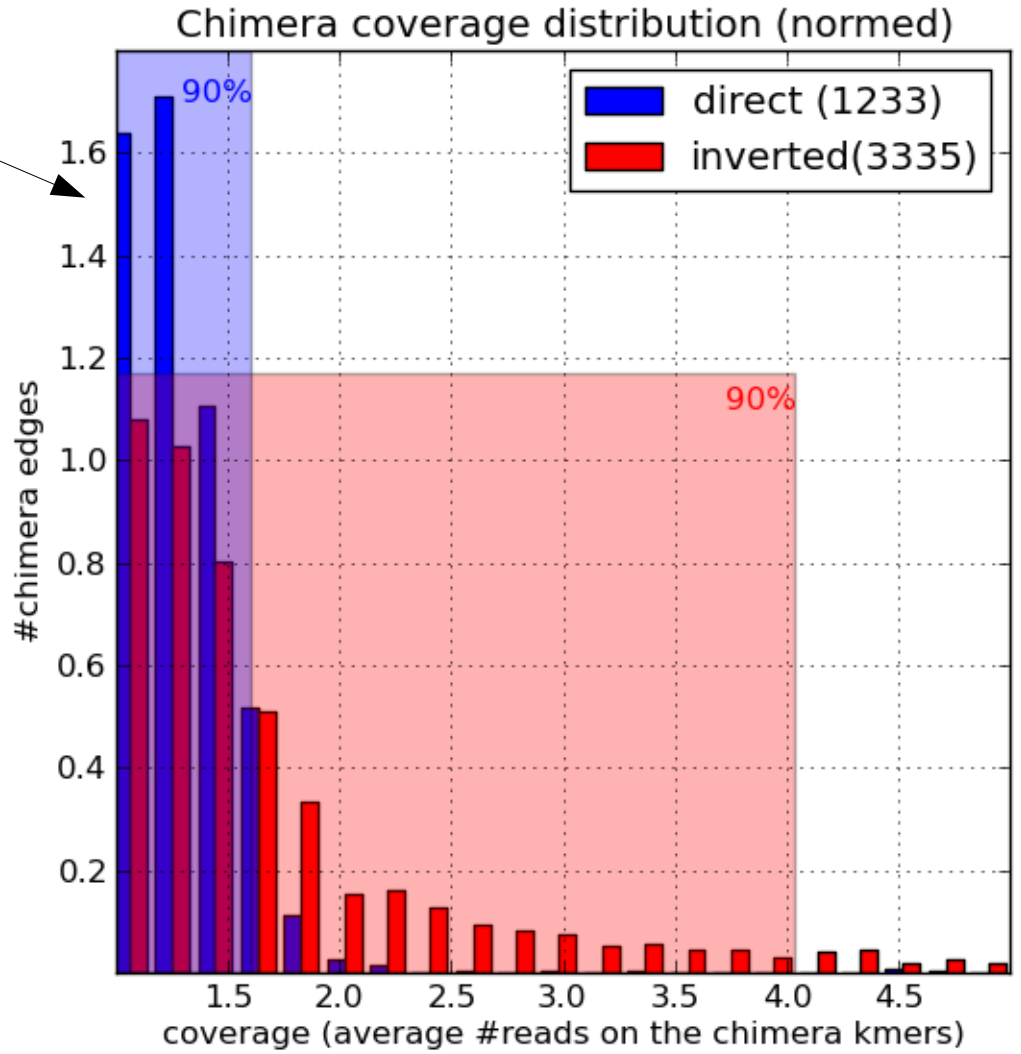
Symmetric & unimodal
Nice!



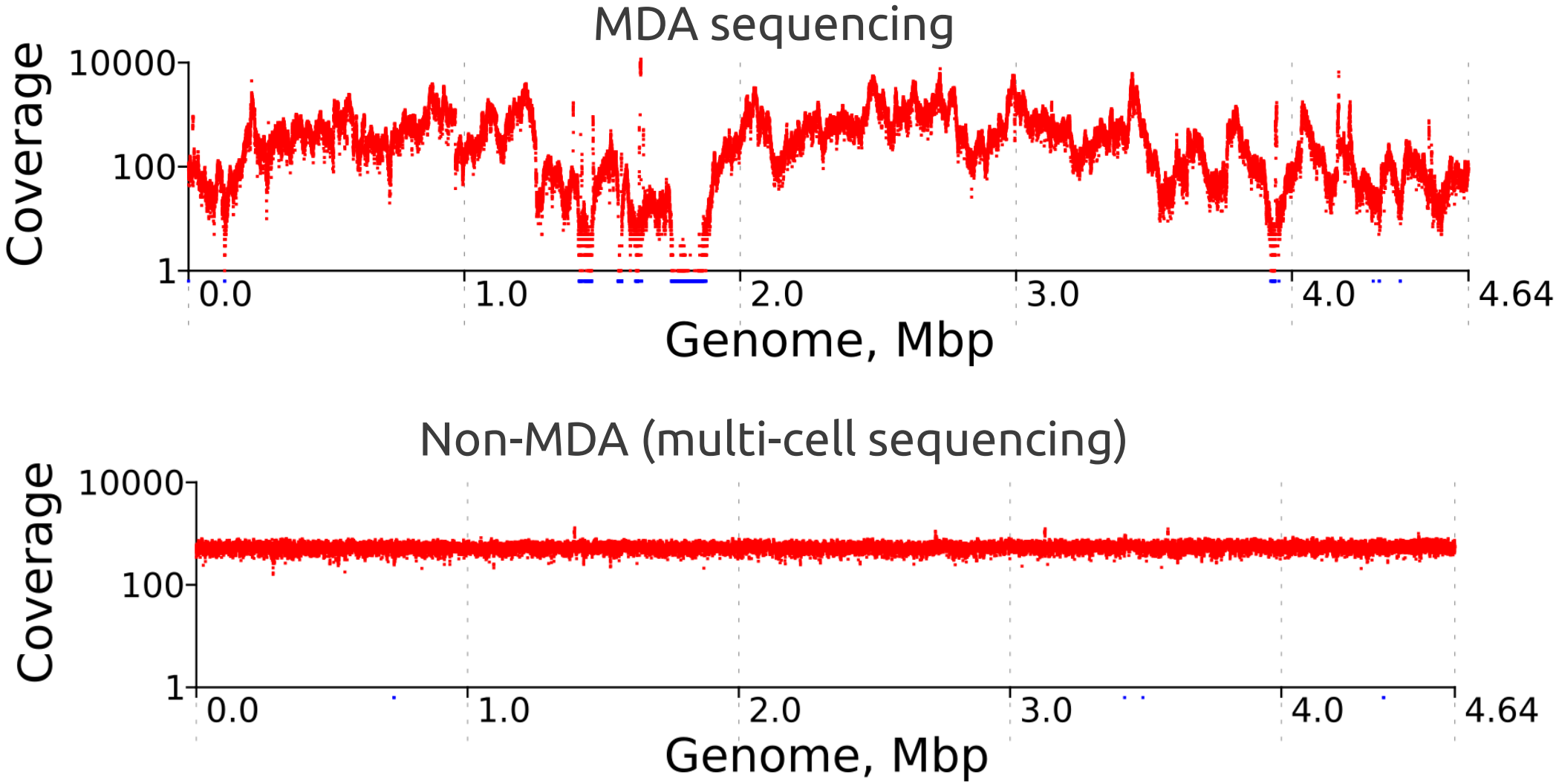
Big variance

Inverted chimeras have greater coverage (?!)

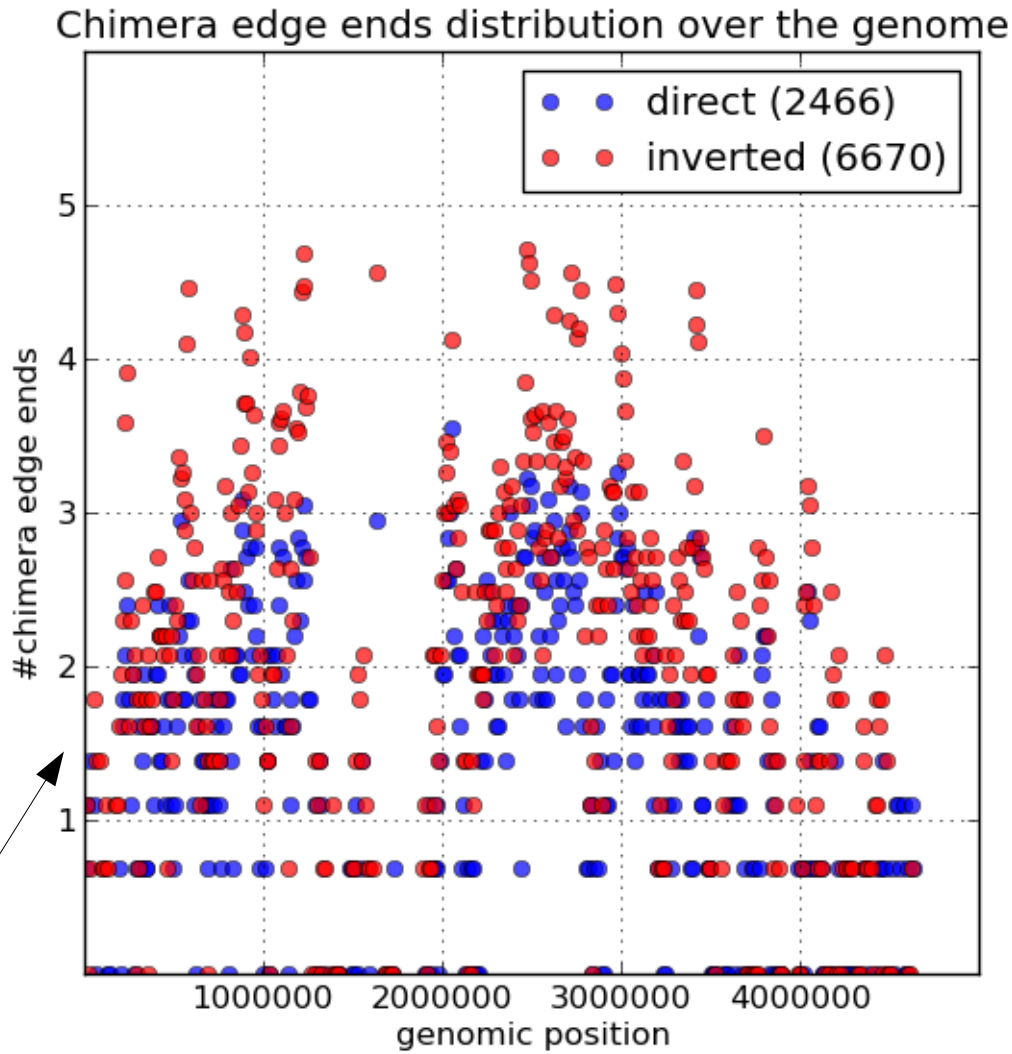
Noise?



Reads distribution

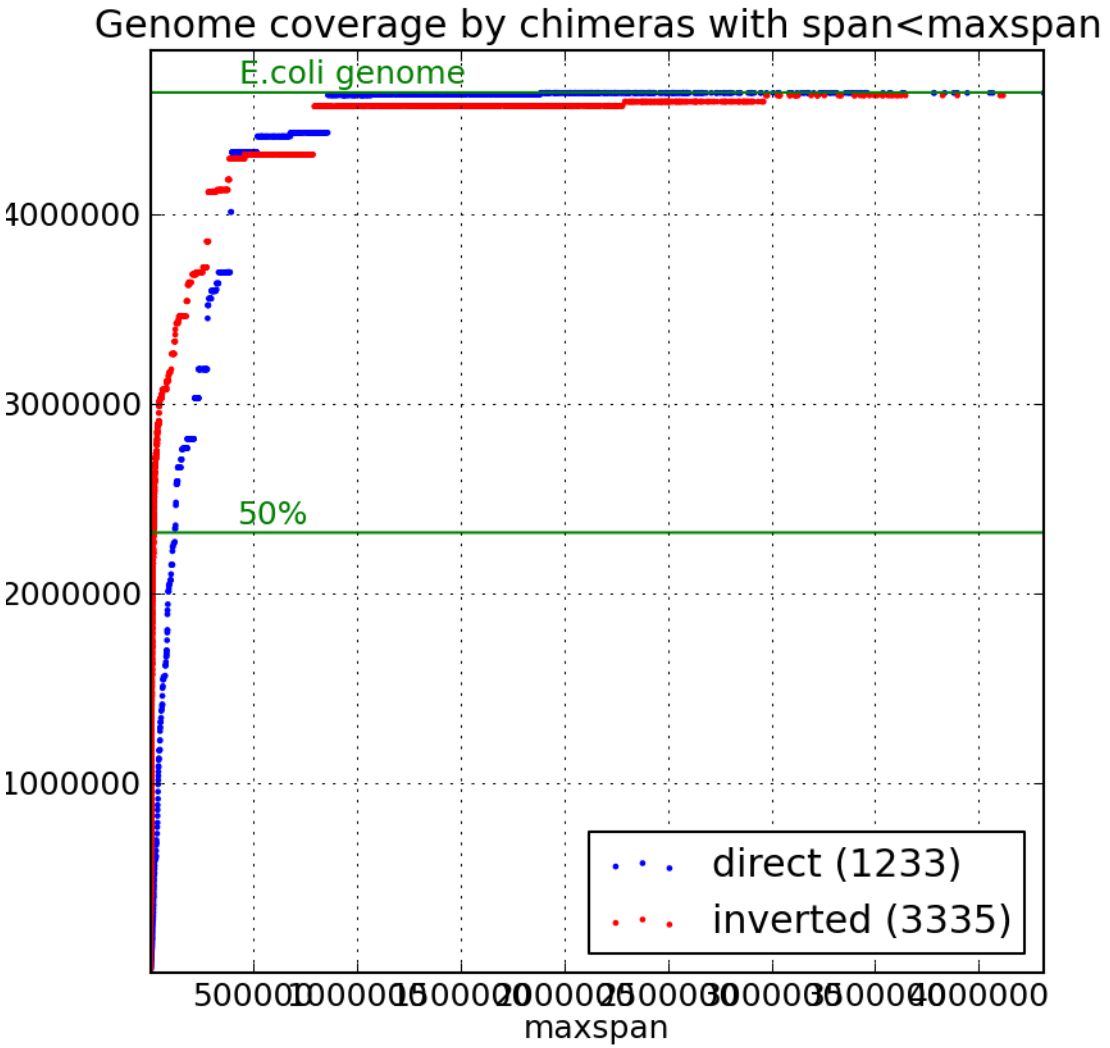


Chimeras on the genome are as fuzzy as reads

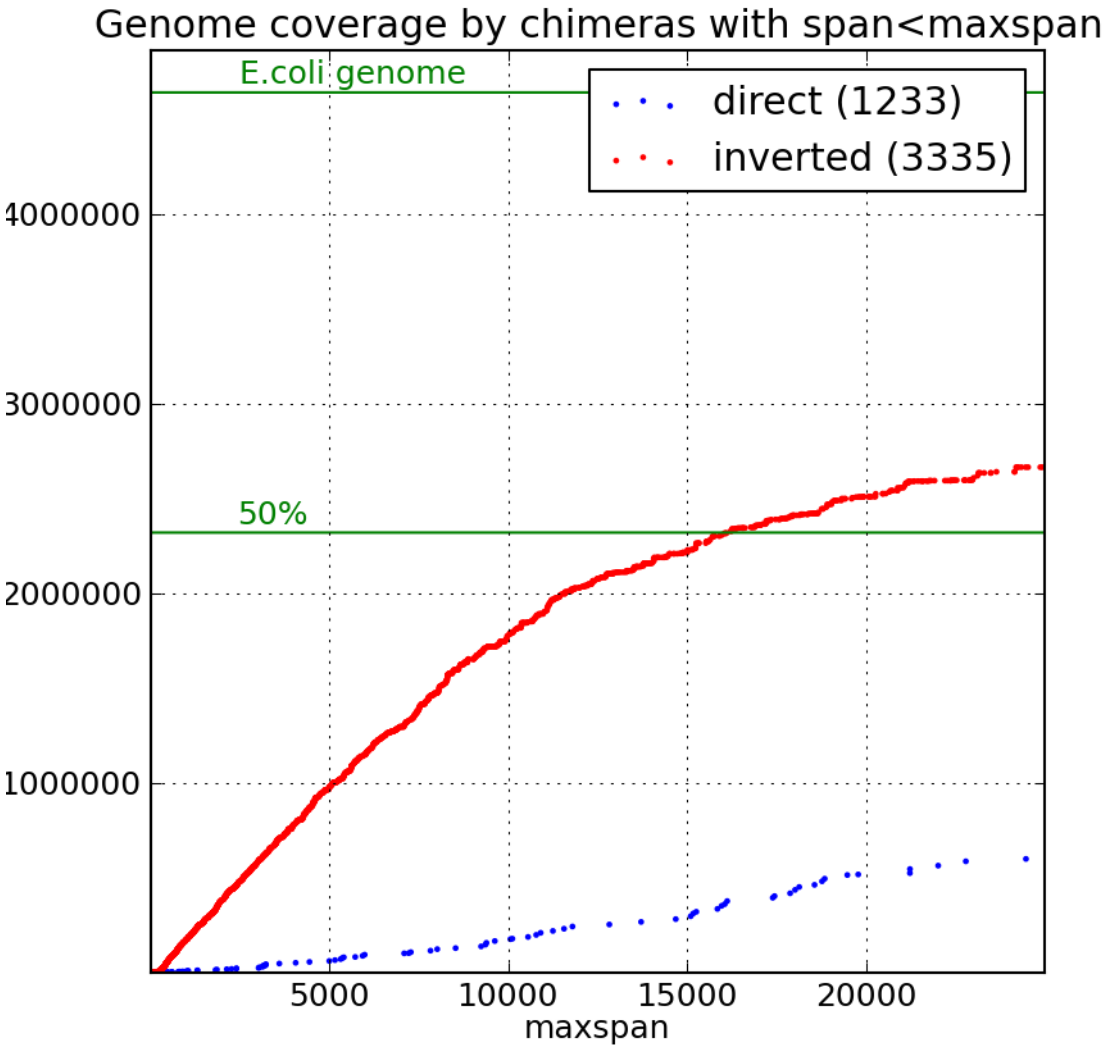


Log-scale

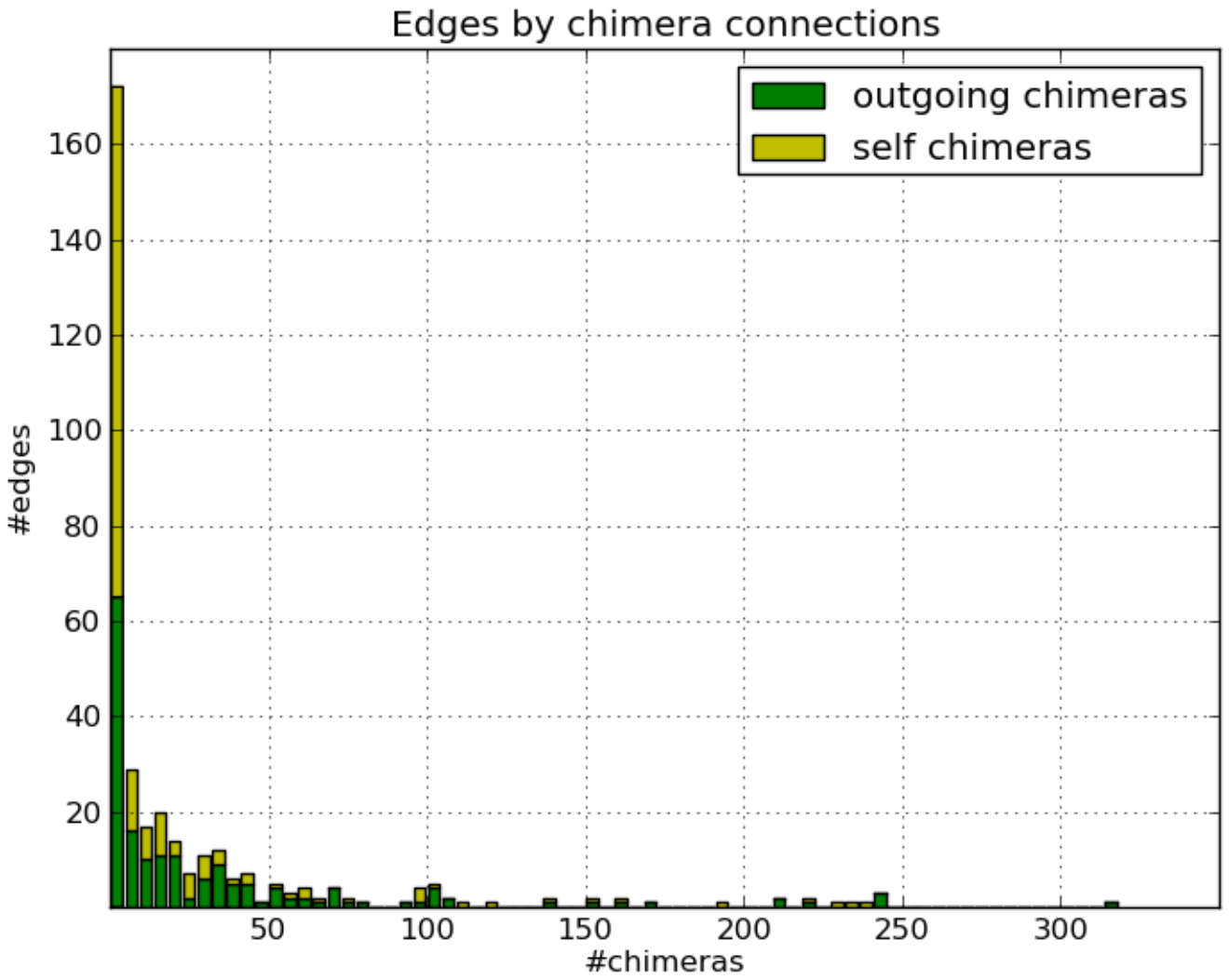
Upper bound of informativeness



<15k span chimeras cover 50% of *E. coli*



How many edges can we join together

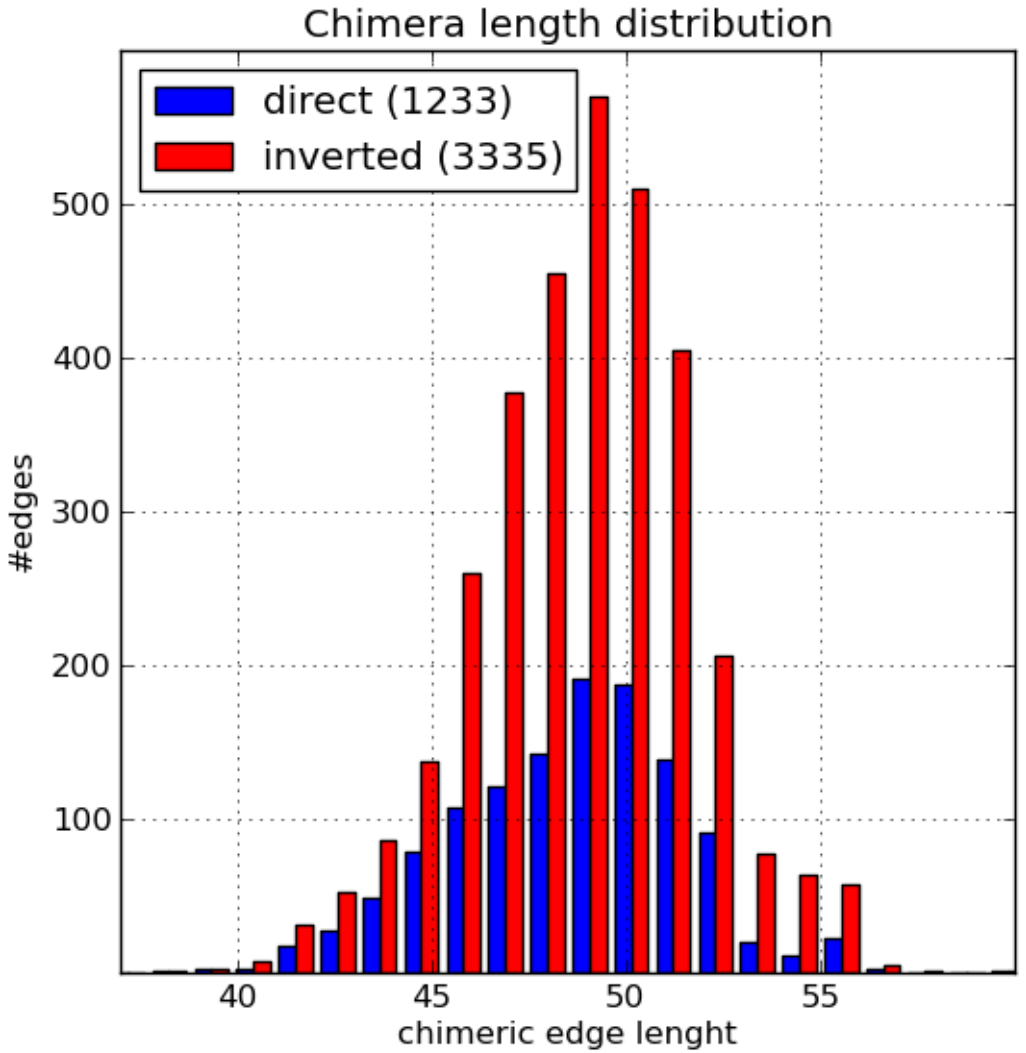


Chimera length

```
5'-CCAGTGAATTTCACTTCGCCAACG-3'
                     3'-GGTTGCCCGACCGGGTGTTCAAC-5'
5'-TGTACTAAAAGGGTAGTCAGAAAAA-3'
                     3'-TCTTTTTGGTCCAGAGCTAAAAT-5'
5'-GAGGCAACATTTGATCGTCAGTG-3'
                   3'-CAGTCATACTTTTCAGGCACCGTCG-5'
5'-CGCCAGGAAACATTGCACACCACGC-3'
                    3'-GTGGGGCGCTAGCGCTCCGTTTGG-5'
5'-CATTCCCGGAATTACATATCTTT-3'
                  3'-TATAGAAAAAGTAATCCGTCACCGGA-5'
5'-GCATATCTCCATCCTGAGTGACGC-3'
                  3'-CTCATTGCGAAAACCAACCCGCTCTT-5'
5'-TTTGAAATATCCACTATTAAGCTAGTGTTTAACG-3'
                           3'-CACAAATTGCGTCGGAA-5'
```

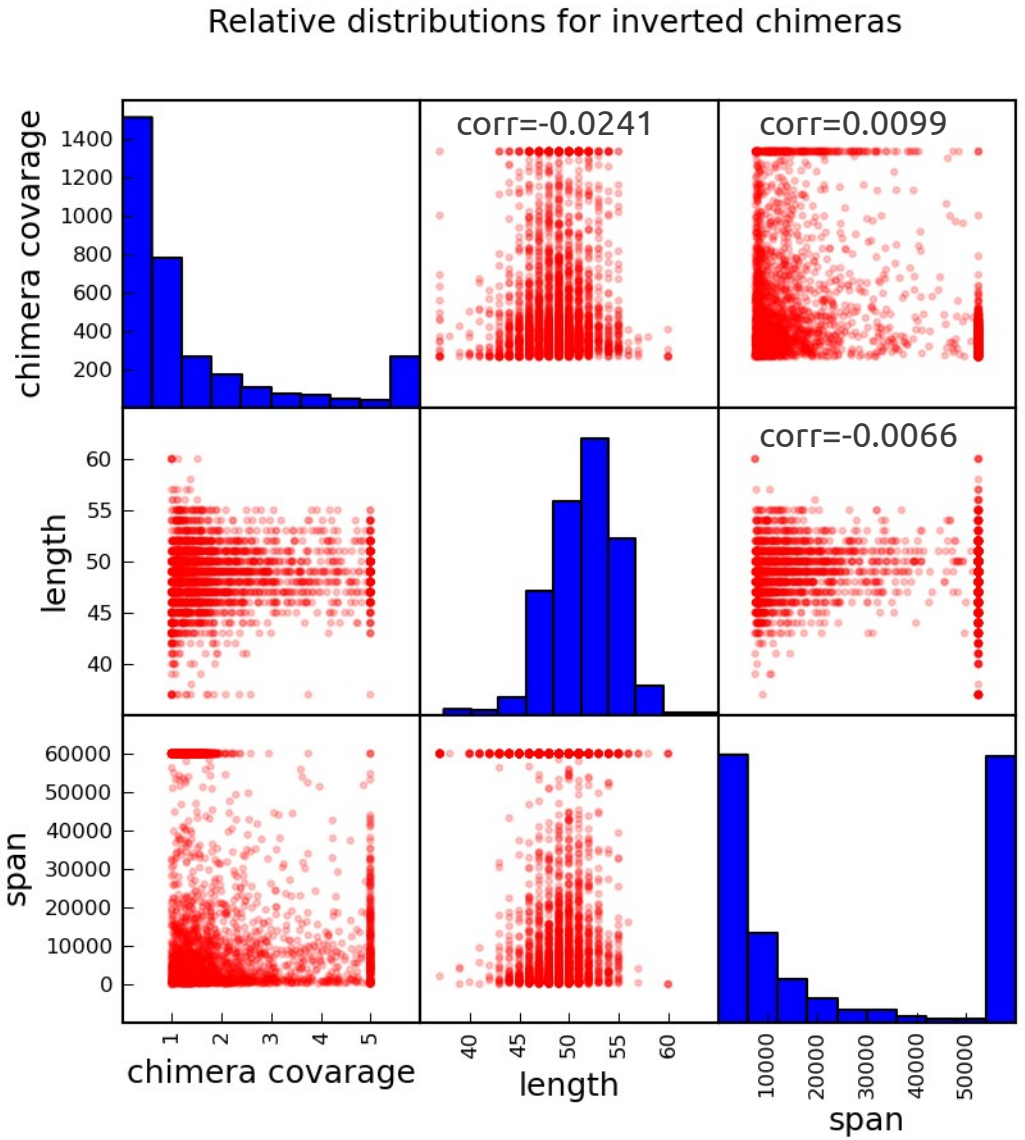
Lasken, Stockwell – “Mechanism of chimera formation during the Multiple Displacement Amplification reaction”, 2007

Length can also vote for chimeras

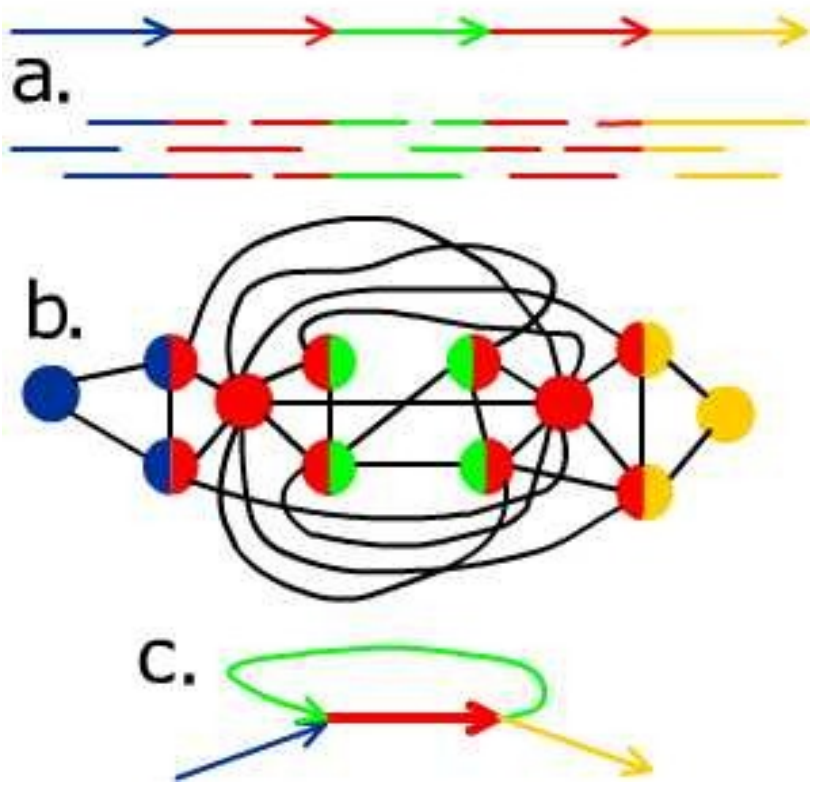


Mean of 49 for both chimera types (agreement with Lasken & Stockwell)

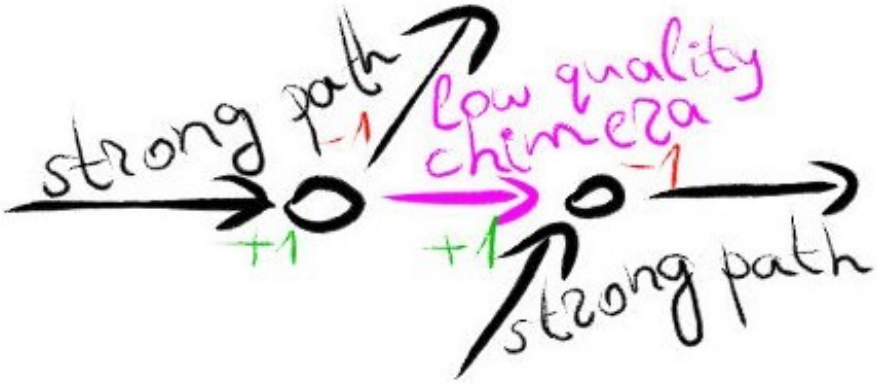
No correlation... :"(



de Bruijn Graph



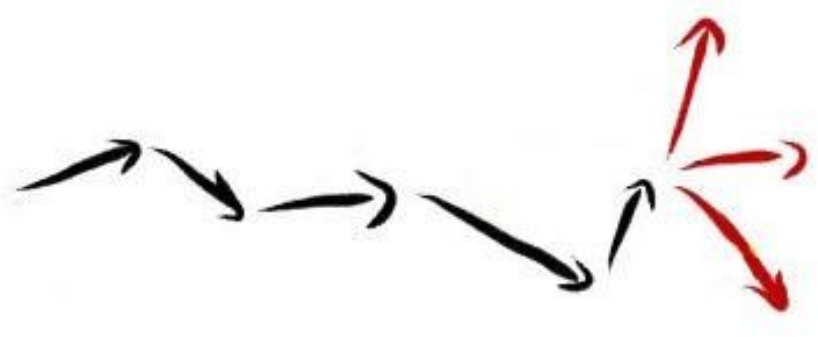
Our hero...
the De Bruijn Graph



How chimera looks like

image by P. Pevzner, H. Tang, M. Waterman

Path extender queries choosers

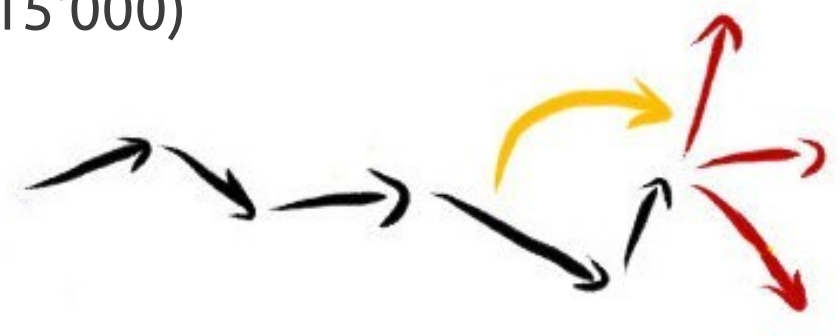


`Given`: **path** + set of **outgoing edges**
`Return`: the correct **extension edge** (if any)
or an empty set if not sure
`Idea`: lets make a path chooser on chimeras!

SPAdes has several different choosers
(by paired-end reads, mate-pair reads, long reads, etc.)

Naïve path chooser

Considers only **chimeras** with
span ≤ *maxspan* (~15'000)

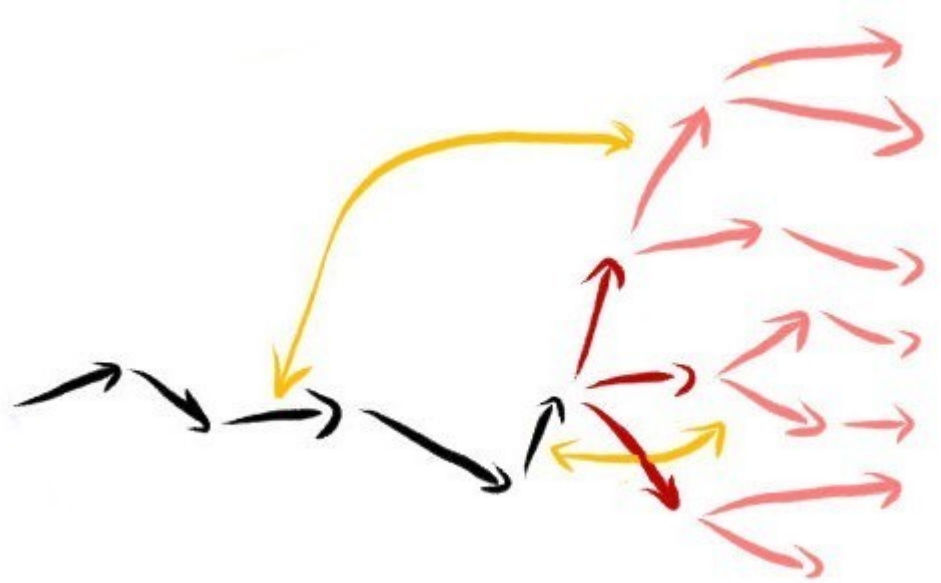


Choose the **edge** connected to the **path**
by a maximum number of **chimeras**

```
chooser invocations: 416
           corrects: 3
         incorrects: 0
```

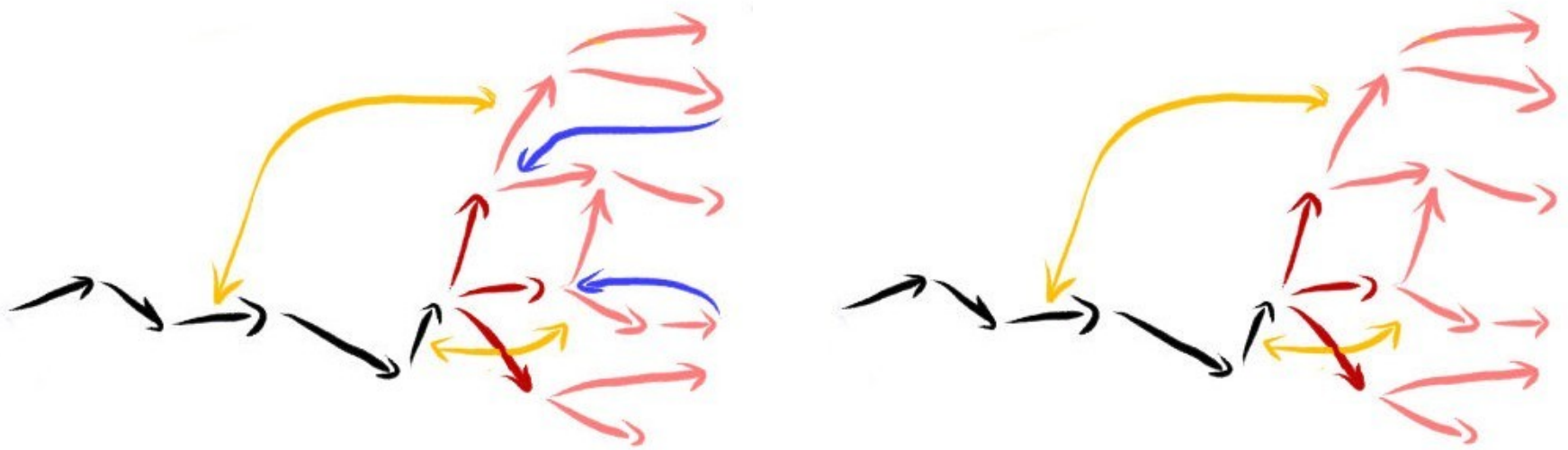
Concl: nice but the #chimeras is not enough

Not **edges** but **paths**



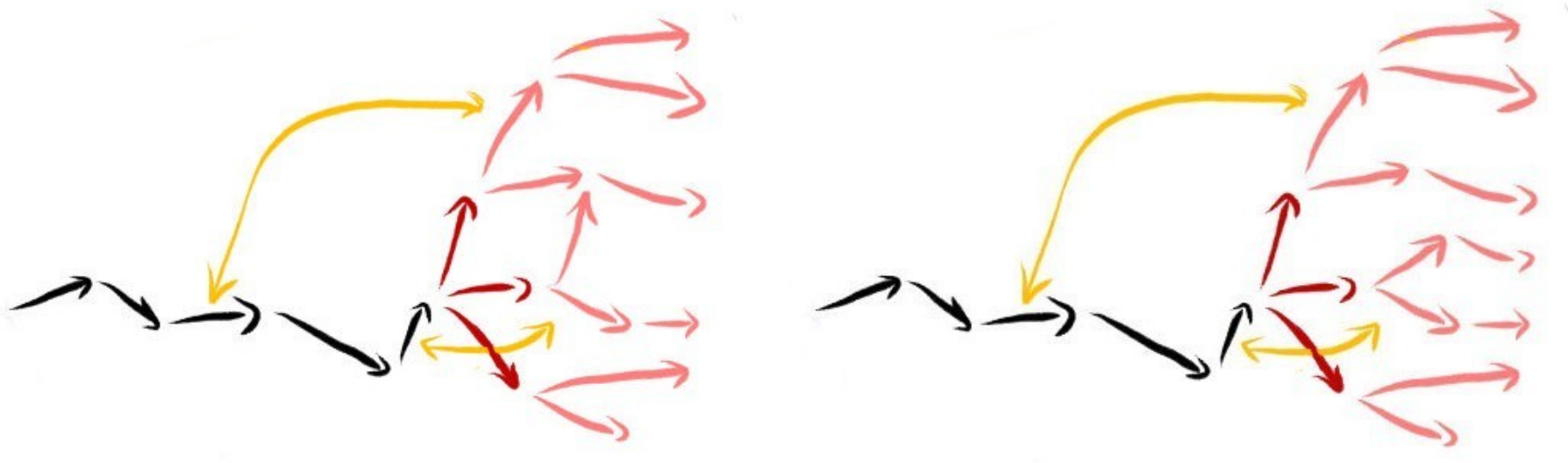
Consider not only the **chimeras** to **edges** but to whole **path extensions** — choose the one connected to the **path** by a maximum number of **chimeras**

Cycles?



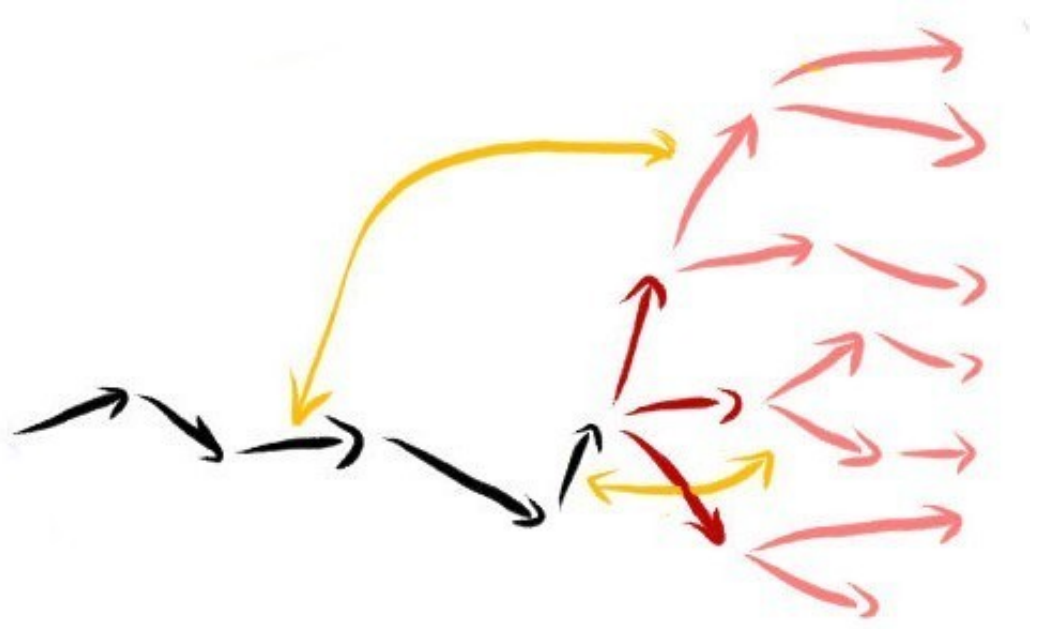
Kill them somehow
(e.g. by constructing a **DFS** tree)

Not a tree?



In case of a **D**irected **A**cyclic **G**raph
duplicate the joint **path extensions**

A tree again



So the notion of “path” gets clear

Result scheme

Let {**result**^{i}} be a subset of **edges**, s.t.

- starting at **result**^{i} there is an **extension path** connected with at least *min#chimeras* (~2÷3)

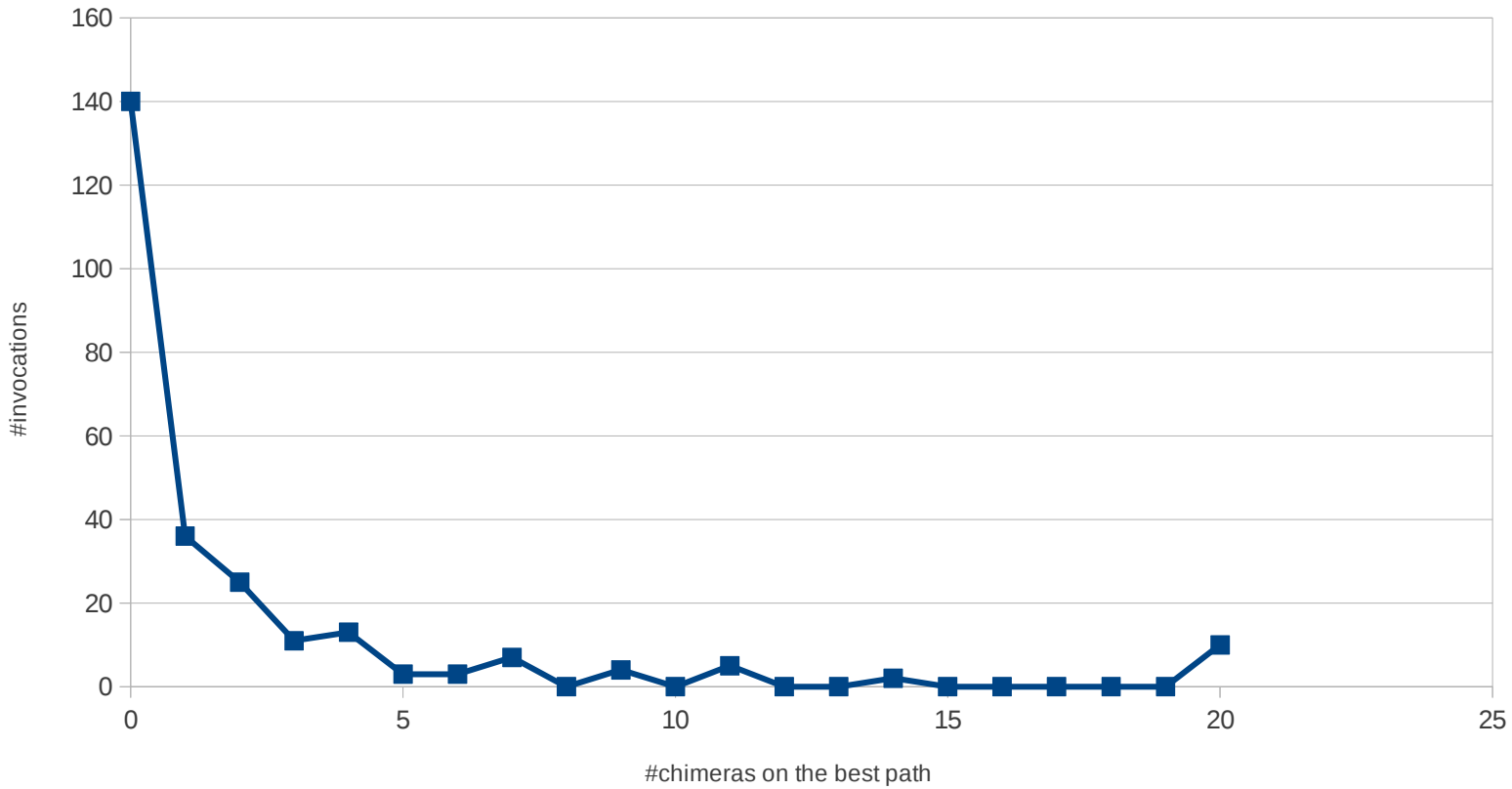
Return nothing in case of:

- multiple results; or
- another extension edge leading to a path with ≥ *leader_coef* (~1÷1.5) times the #chimeras in the best path

Else:

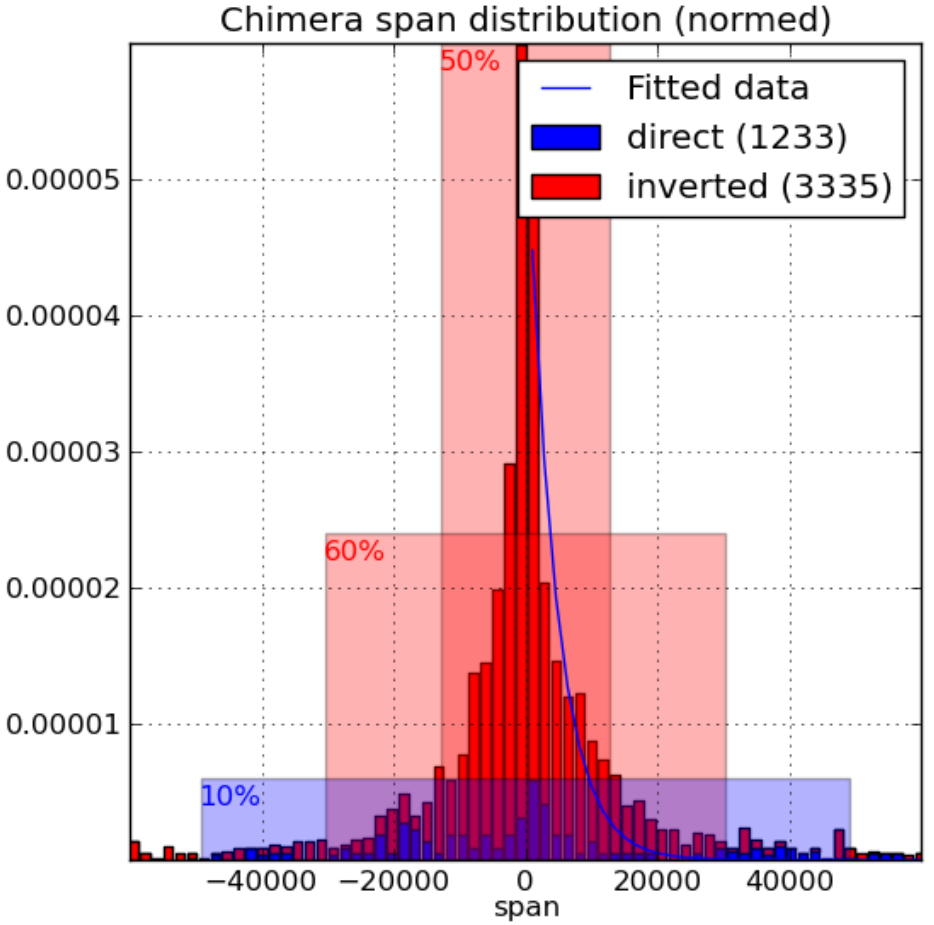
- Return {**result**^{i}} which has a single or no edges

#invocations by #chimeras in the best path



We need more chimeras, folks...

Artificial chimeras



Blue curve:
λ*exp(–λ*x)
λ=1/16'145

The span distribution
of the **inverted chimeras** is almost exponential

E.coli results

Artificial inverted chimeras	Edges with chimeras	Corrects	Incorrects	#tmp
50000	676	51	8	213
20000	626	44	2	208
10000	570	35	3	210
5000	496	27	1	211
0	332	19	1	212

Edges: 1935
Natural inverted chimeras: 3336
Natural direct chimeras: 1331
MaxSpan: 15000
min#chimeras: 3
Invocations: ~430÷460

Not yet ready for production :")

Assembly	Scaffolds without any choosers	Current best results	+ Chimera chooser
N50	67332	109825	121369
#misassemblies	3	2	3

Some useless parameters we tried

- Give up if no long edges in the **extension path**
- Don't allow **extension path** to continue to the **path**: only few such situations
- Limit the number of forks on the **extension path**: **no** visible correlation with correctness
- Limit the number of edges the **extension path**

Pain in the ass: Correctness

It is tricky to certify the chooser correctness:

- Sometimes the given **path** doesn't match anywhere in the genome

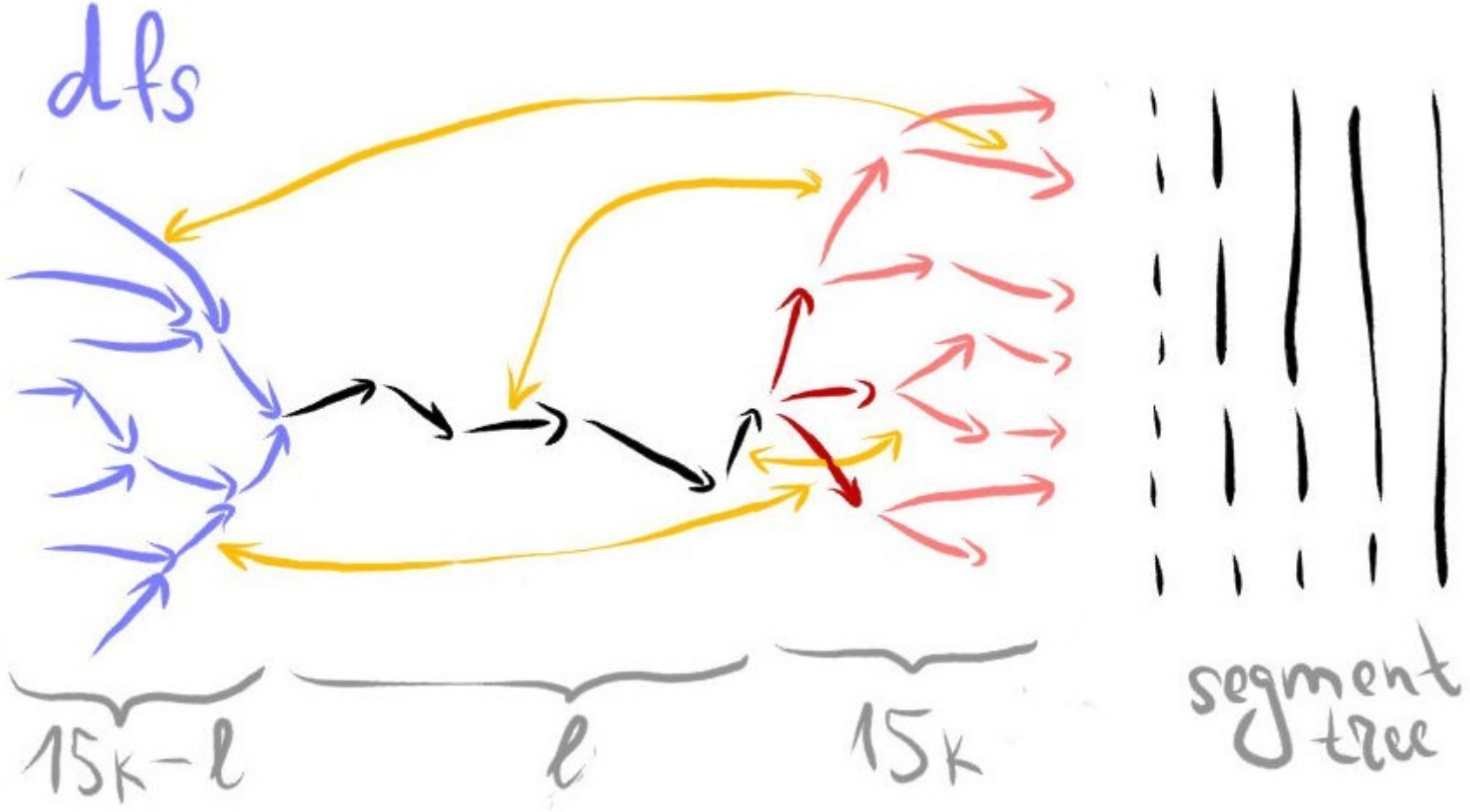
Different heuristics are tried with no right way to do it

Fundamental problems

Too few chimeras

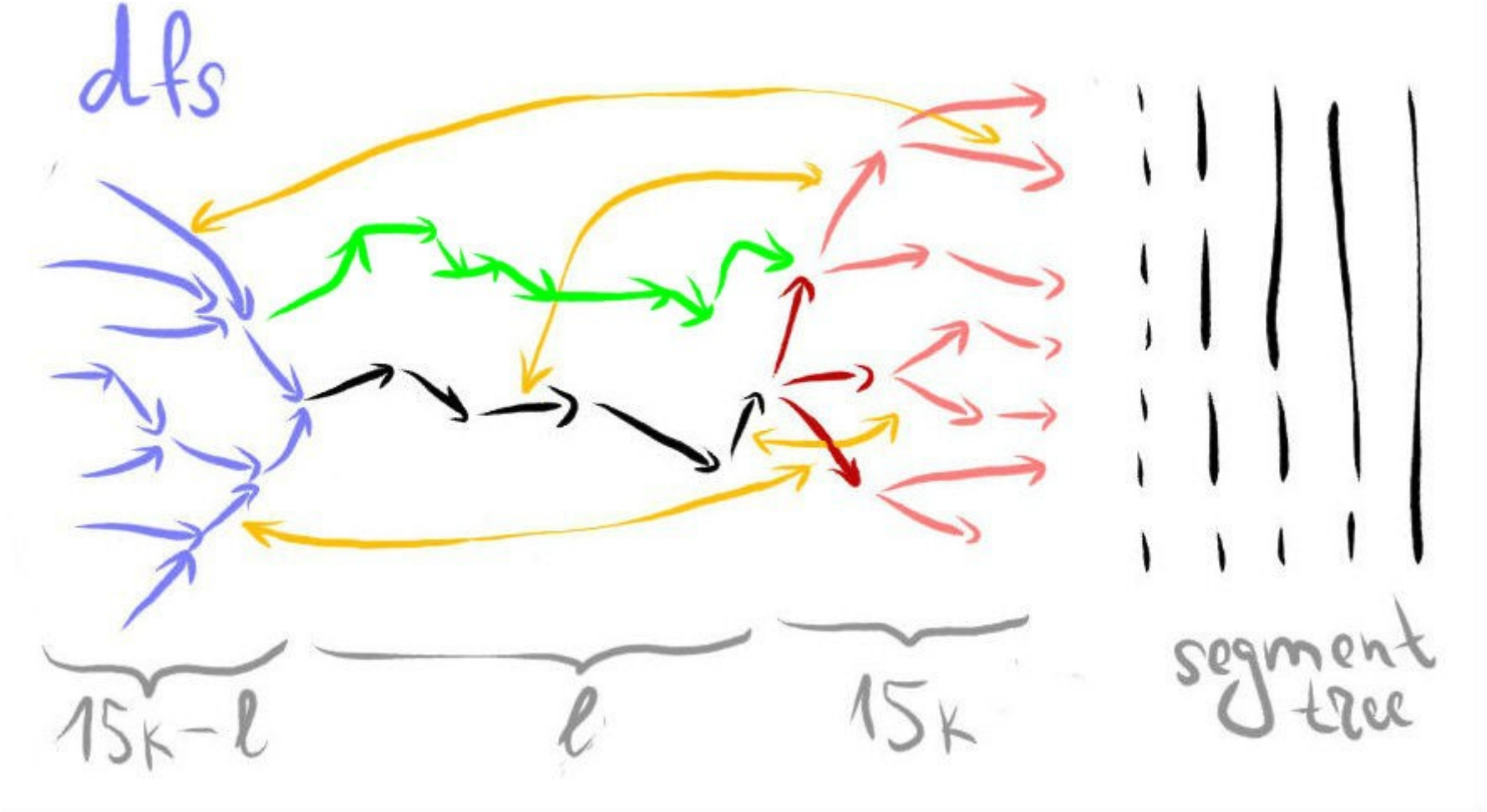
- Can we ask the biologists to produce more chimera?
- Let's try to use even more chimeras...

Joint optimization of the incoming and outgoing paths



Recurse the **incoming tree** while updating the #chimeras going to all the **extension paths**

Joint optimization of the incoming and outgoing paths



Recurse the **incoming tree** while updating the #chimeras going to all the **extension paths**

Joint optimization of the incoming and outgoing paths

O(?): finds BypassingPaths from v ∈ incoming tree o v' ∈ extension paths
θ(N + C*logC): map the chimeras going to the extension paths
θ(C*logC + N*logC): recurse the incoming tree maintaining a segment tree for the #chimeras in all extension paths

N – #vertices, **C** – #chimeras

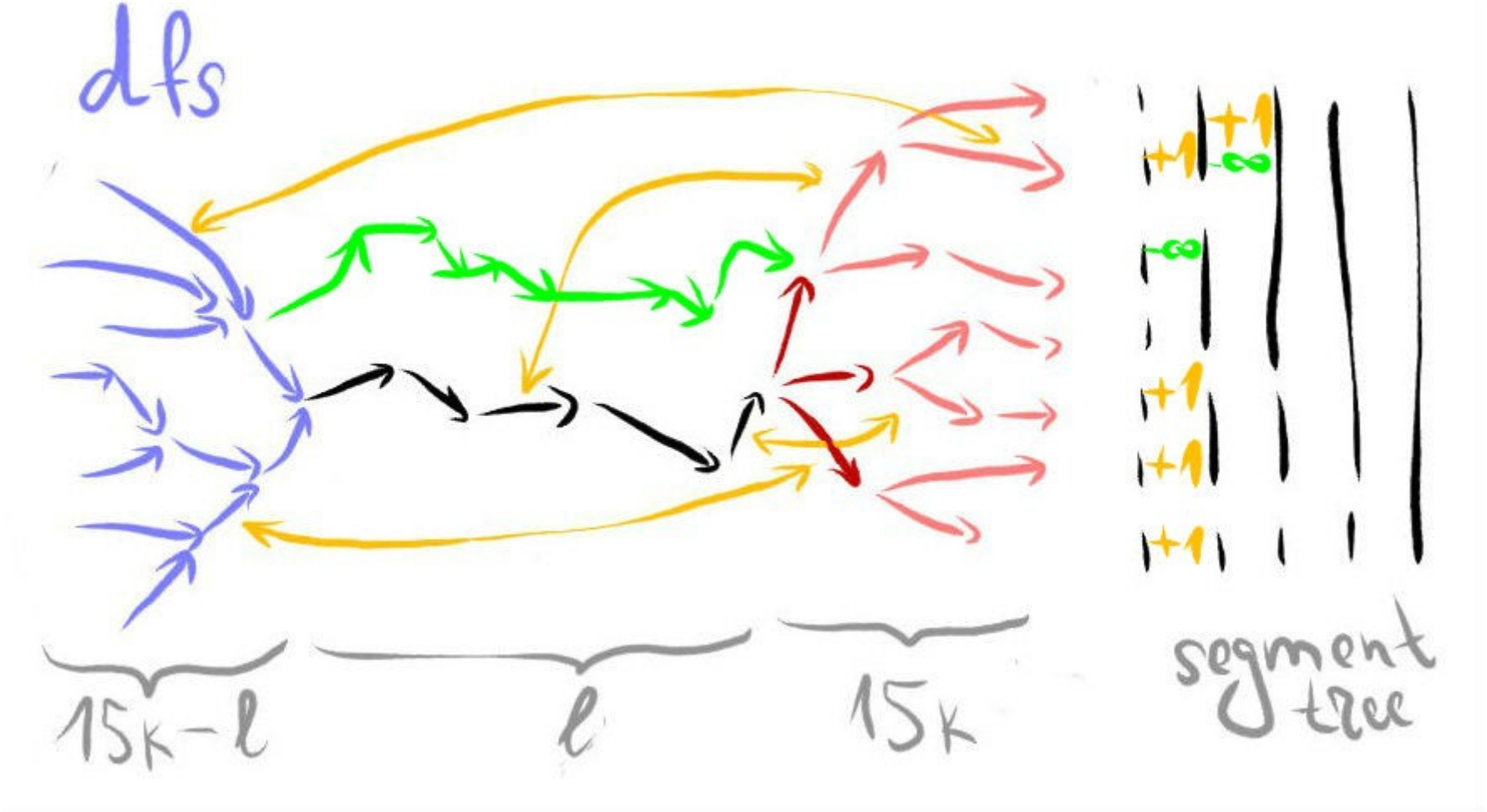
```
dfs(v, w):                          // v, w ∈ incoming tree:
  UpdateBypassings(w, -inf)
  UpdateChimeras(v, w, +1)
  for edge(u, v) ∈ Edges, u ∈ incoming tree:
    dfs(u, v)
  UpdateChimeras(v, w, -1)
  UpdateBypassings(w, +inf)

UpdateBypassings(v, val):           // v ∈ incoming tree:
  for edge(v, v') ∈ BypassingPath, v' ∈ extension:
    updateSegTree(v', val)

UpdateChimeras(v, w, val):          // v, w ∈ incoming tree:
  for chimera from edge(v, w) to edge(v', w') ∈ extension:
    updateSegTree(w', val)

updateSegTree(v', val):             // v' ∈ extension:
  chimeras := updateVal(v', val)
  updatePathRes(v', chimeras)
```

Joint optimization of the incoming and outgoing paths



Recurse the **incoming tree** while updating the #chimeras going to all the **extension paths**

Future plans

- Get better graph visualizations for debugging
- Test on other datasets (S.aureus)
- Ask the technicians to increase #chimeras
- Chimeras → Scaffolder
- *Inverted vs Direct* chimera classify by graph topology
- Define a probabilistic interpretation

Thank you for discussing :")

Let we use the MDA “bugs”(chimeras) to assemble single cells better than multi-cells!

kitties by Denis Sazhin
http://habrahabr.ru/users/centaurus/