LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

PAGANTEC: OPENMP PARALLEL ERROR CORRECTION FOR NEXT-GENERATION SEQUENCING DATA

Markus Joppich, Tony Bolger, Dirk Schmidl
Björn Usadel and Torsten Kuhlen

Markus Joppich
Lehr- und Forschungseinheit Bioinformatik
Institut für Informatik
LMU München

PAGANtec

Probabilistic **A**lgorithm for **G**enome **A**ssembly of **N**ext-Generation Sequencing Data

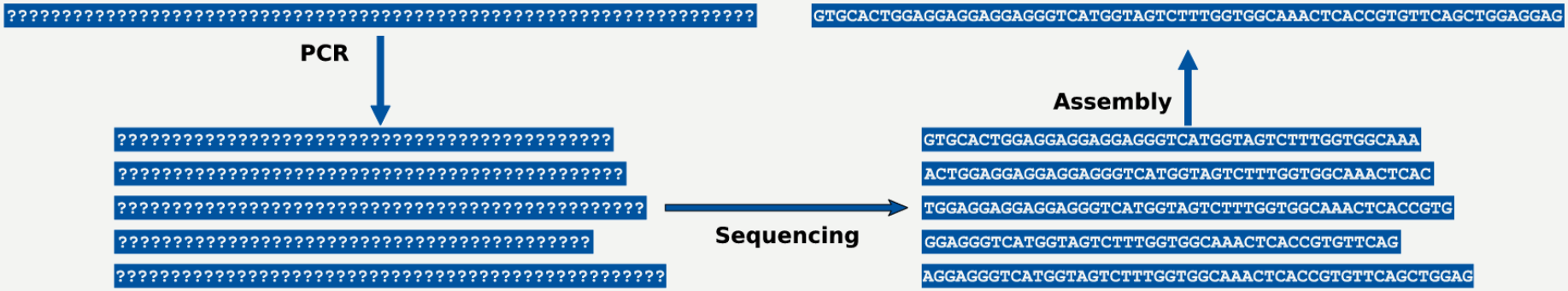
transcriptome **e**rror **c**orrection

USER EXPERIENCE ON OPENMP

1. Biological Background
2. PAGANtec
3. OpenMP Parallelism in PAGANtec
4. Conclusion

Biology / Sequencing

- DNA consists of words $w = \Sigma^* = \{A, T, G, C\}^*$ from a 4 letter alphabet
  - **bases** or base-pairs (bp)
- Sequencing tries to translate from **molecule** to **textual** representation
- Many technologies
  - Sanger (<1000bp)
  - **NGS** (~100 – 300 bp)



Raw Read Data

```
>SequenceID1
ATGACAACAGAGATATCAGA
@
1231241212312AABASDA
```

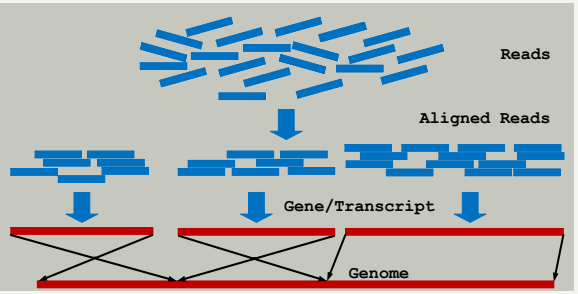
```
>SequenceID2
ATGACAACATAGATATCAGA
@
1231241212312AABASDA
```

- Fragments/Sequences are called **reads**
- Sequencing is **error-prone**
  - Up to 10% for early Illumina, 30% Nanopore
- Huge amount of **data**: several 100GB
  - Not easy to spot mistakes **by eye**

Assembly

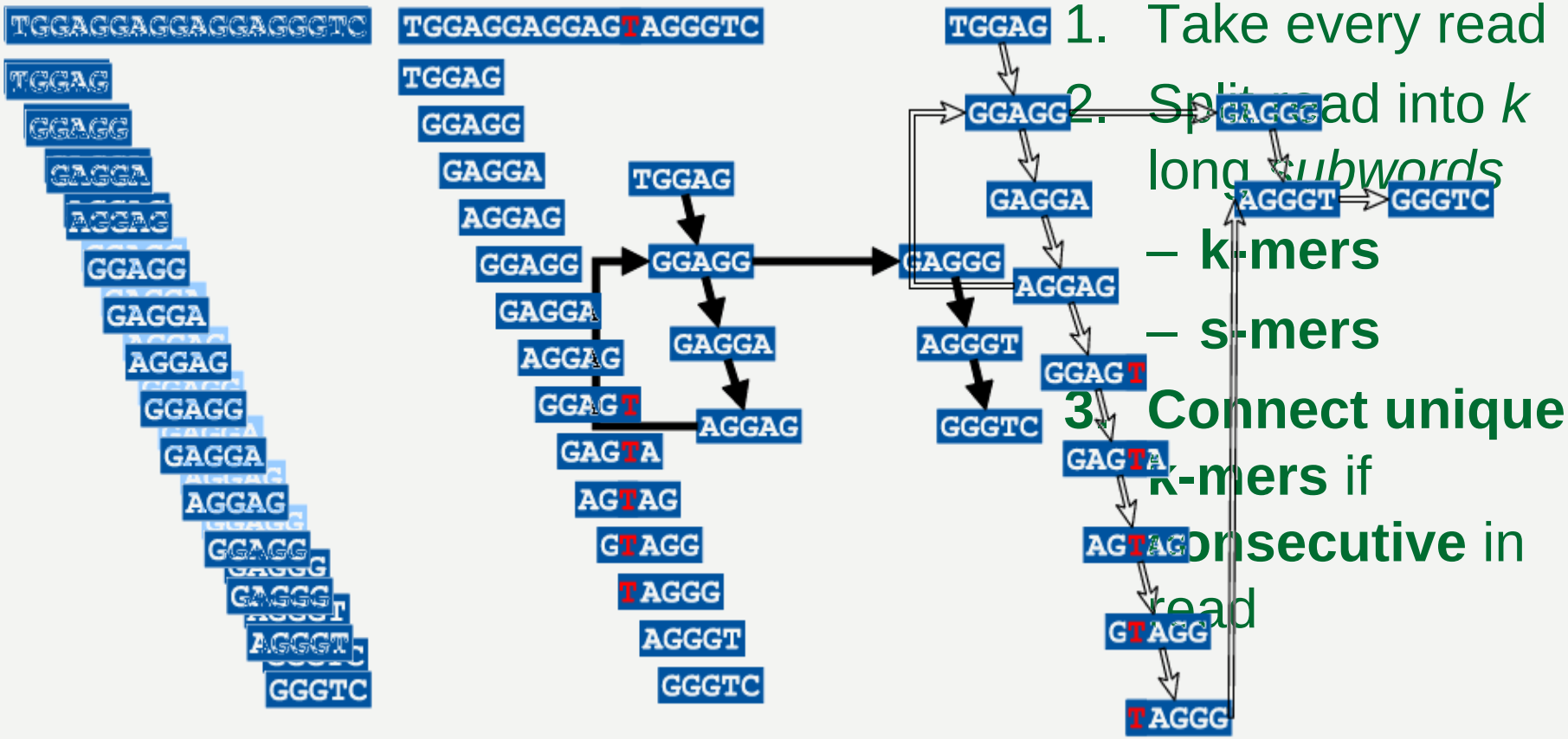
- **Assembly**: put reads together to get original sequence
- **Assembly Problem:**

> "Maximize the likelihood that the assembly is the source of the set of reads"



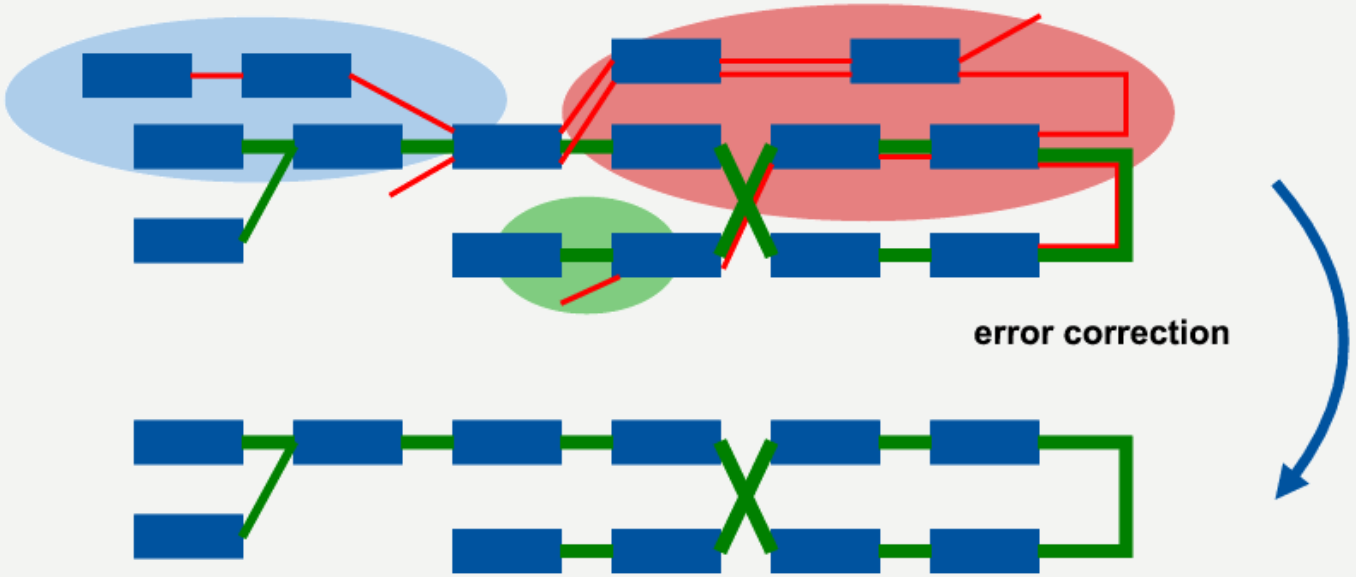
- Several Methods:
  - Overlap / Layout / Consenus or **De Bruijn Graph**
  - Problem complexity ~ Graph complexity

DBG: Graph Structure



1. Take every read
2. Split read into *k* long *subwords*
   - **k-mers**
   - **s-mers**
3. **Connect unique k-mers** if **consecutive** in read

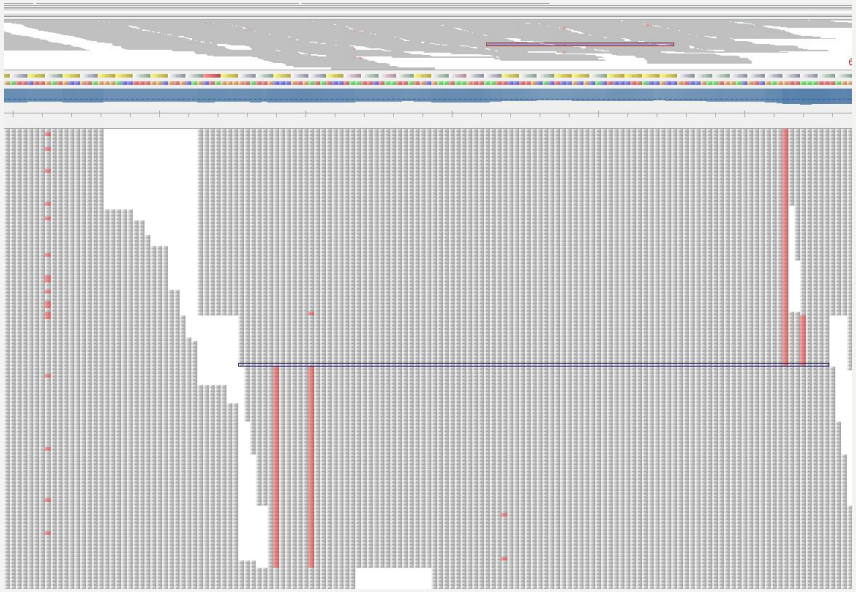
Error Correction



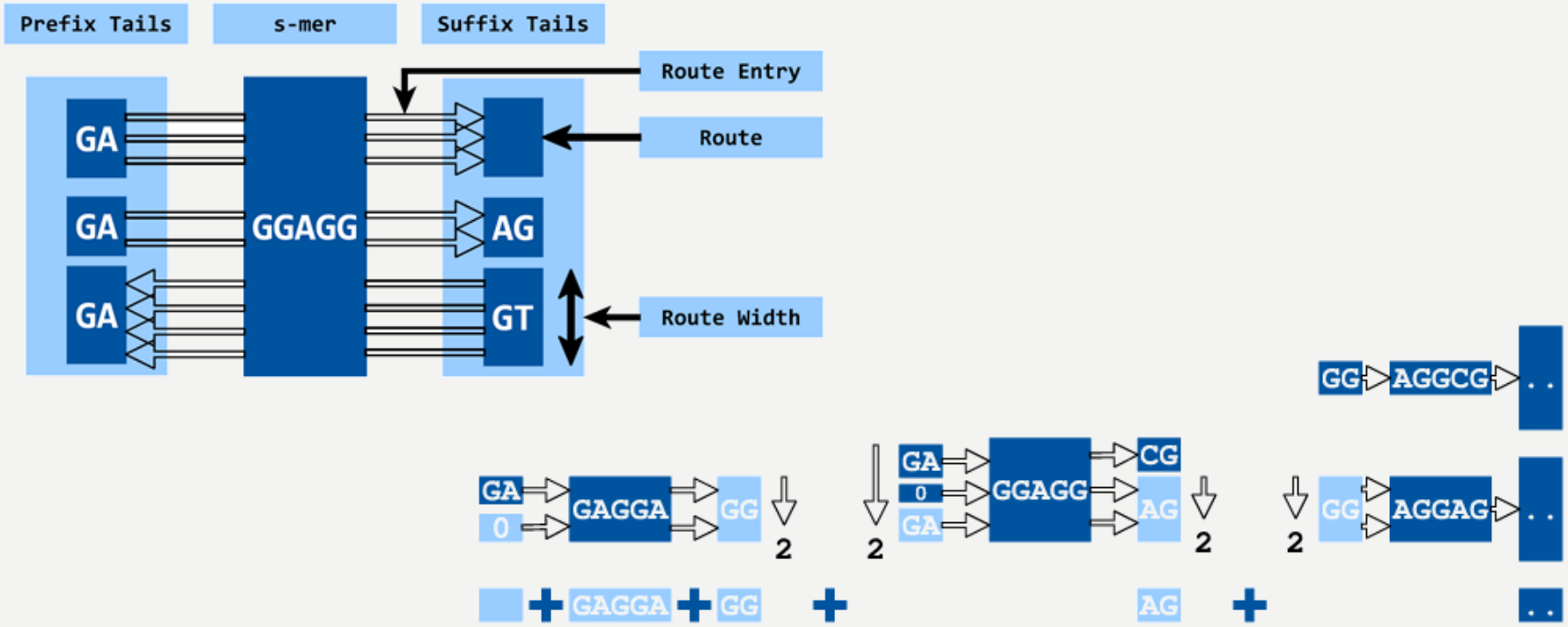
- Several Structures can be observed in **k-mer** graph:
  - **Tangles, Spurs and Bubbles**
  - due to **errors** in reads, make assembly harder
- Essentially **error correction** removes such artifacts

Error Correction

- There are several error correctors available
  - Mostly for **genomes**
  - Different **characteristics**
- ***SEECER*** error corrector for transcriptomes [Le 2013]
- Results not satisfying enough



PAGAN Graph Structure

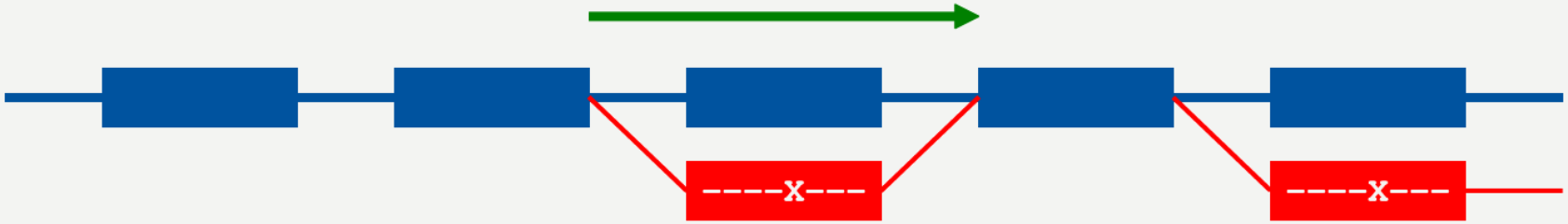


- Allows **lossless** storage of **reads** in a k-mer-graph
- Single reads can be **extracted**

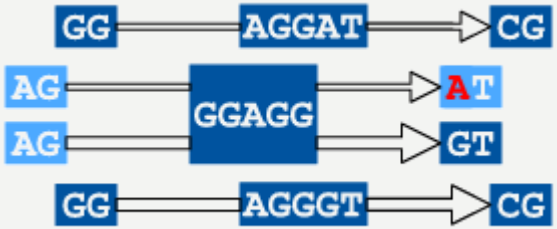
PAGANtec: algorithmic

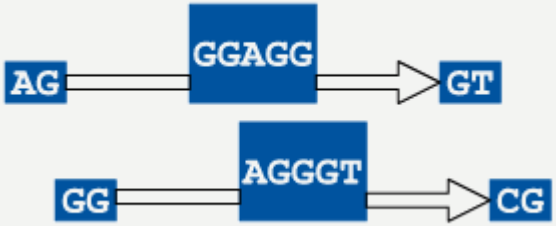
- Correction strategey depends on the location of error
  - Error at an **end**

  - Or forming a **bubble** or **long spur**



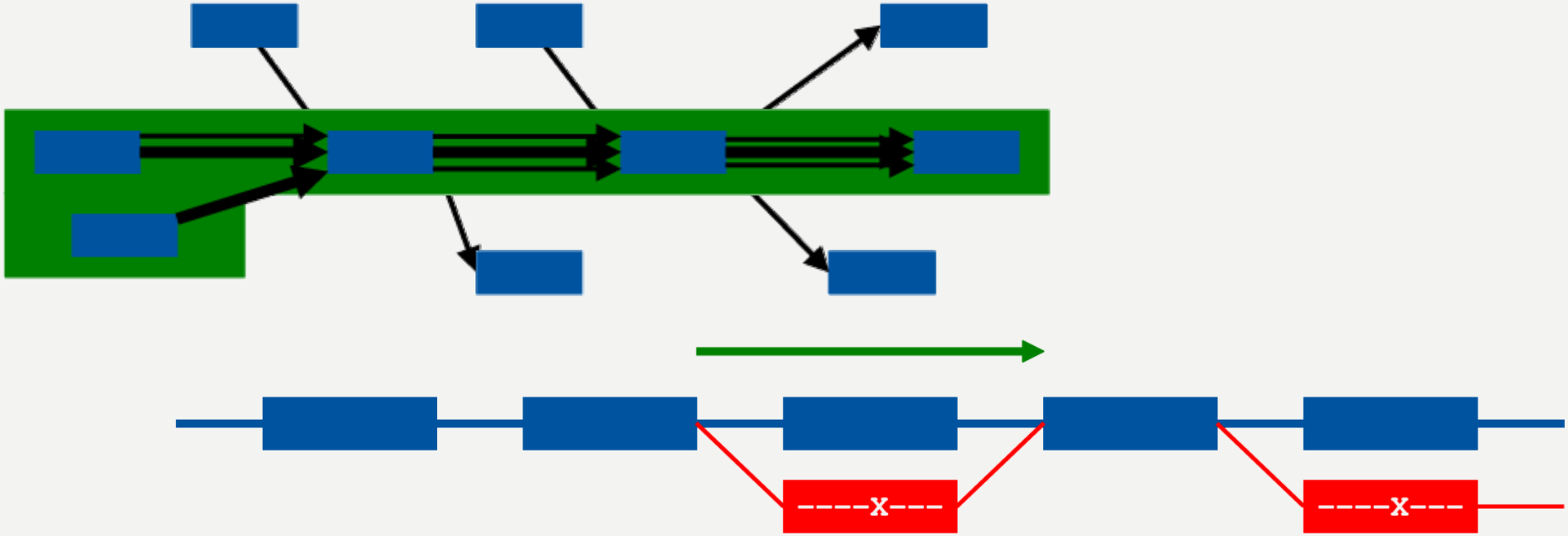
PAGANtec: algorithmic





- **Short tip** correction
  - Detect erroneous sequence
  - Replace by correct sequence
    - Attention: tails can be used by other reads!

PAGANtec: algorithmic



- Long **tip/Bubble** correction
  - **Detect** erroneous sequence
  - **Store** correction and corrected sequence
    - Not depending on other reads!

PAGANtec: results

- **Good correction** in general and in detail!
  - No structures forming
- However slower.
  - Easy parallelization

PAGANtec: filter

- Each algorithm for
  - Short **tips**
  - Long **tips**
  - **Bubbles**
- Essentially same steps
  - **Detect** Errors
  - Find **correction**
  - **Store/Apply** correction

Algorithm

```
1: function APPLYFILTER(graph, smers)
2:
3:     for smer ∈ smers do
4:         for route ∈ smer do
5:             for route entry ∈ route do
6:                 CORRECTROUTEENTRY(graph, route entry)
7:             end for
8:         end for
9:     end for
10: end function
```

PAGANtec: parallelism

- Apply
  `#pragma omp parallel for`

Algorithm

```
1: function APPLYFILTER(graph, smers)
2:
3:     for smer ∈ smers do
4:         for route ∈ smer do
5:             for route entry ∈ route do
6:                 CORRECTROUTEENTRY(graph, route entry)
7:             end for
8:         end for
9:     end for
10: end function
```

$> 10^6$

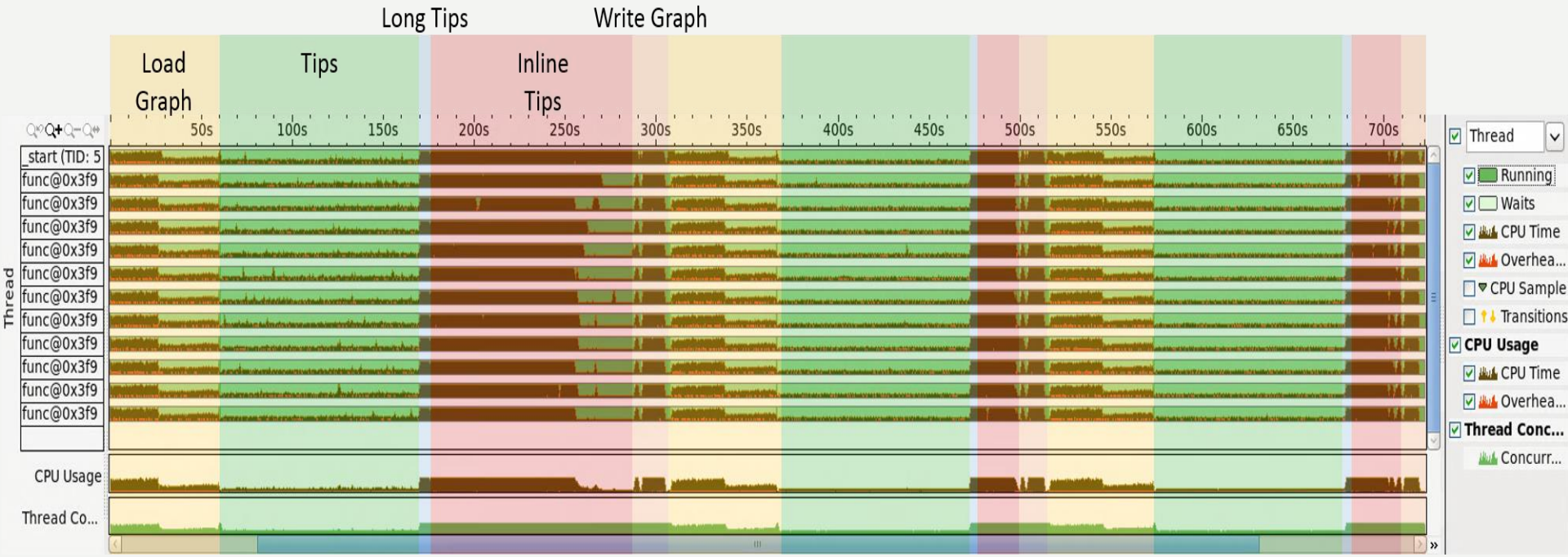
$\sim < 10^1$

$\sim 10^0 - 10^1$

Width of smer

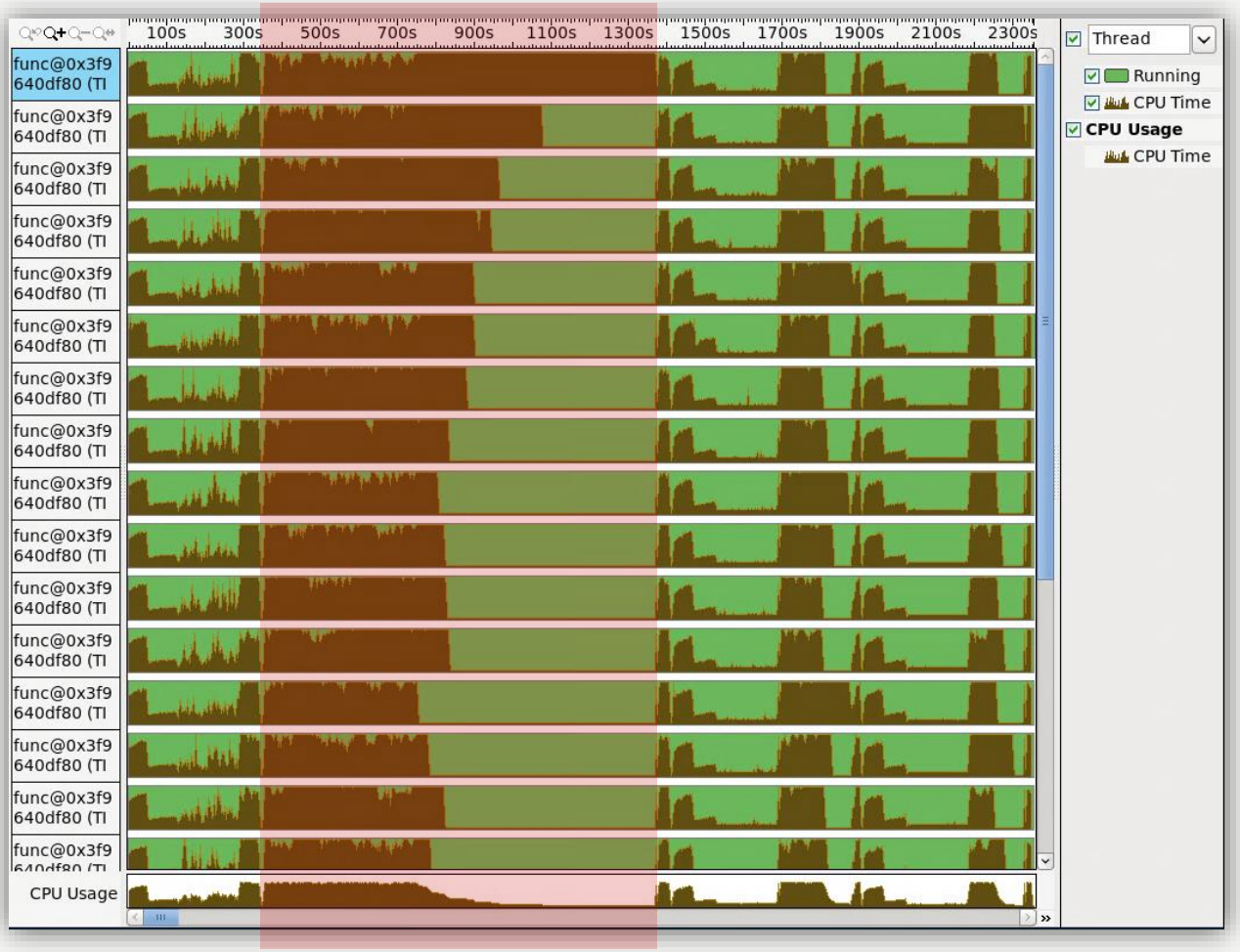
- FilterTips: **race conditions / data race**!
- Correction Storage: **Threadsafe** Hash-Map

PAGANtec: load imbalance

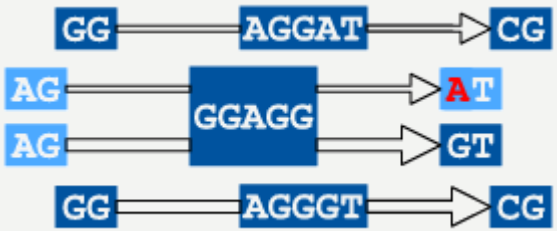


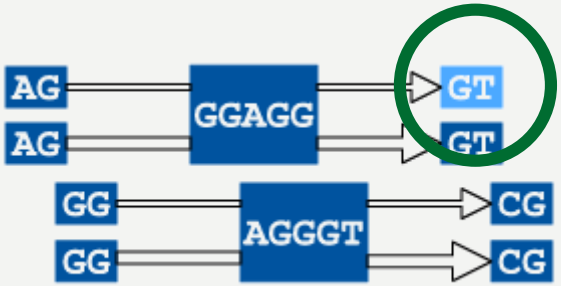
- Several stages identifiable
- Problems detected:
  - Tips have a low **CPU usage**!
  - One thread takes essentially **longer** than others!

PAGANtec: load imbalance



PAGANtec: characteristics





Low CPU usage?

Locking!

Algorithm

```
1: function APPLYFILTER(graph, smers)
2:     # pragma omp parallel for schedule(dynamic)
3:     for smer ∈ smers do
4:         for route ∈ smer do
5:             for route entry ∈ route do
6:                 CORRECTROUTEENTRY(graph, route entry)
7:             end for
8:         end for
9:     end for
10: end function
```

Very coarse grained!

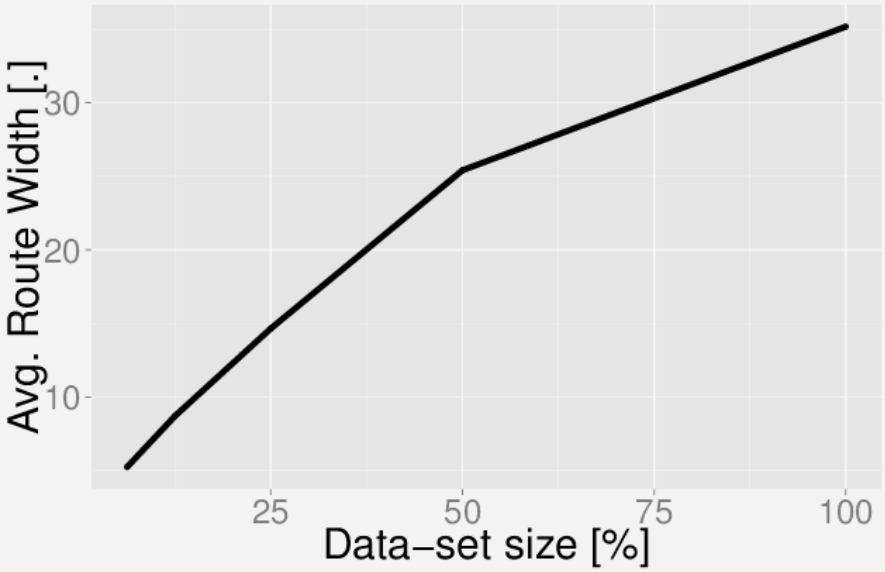
Load Imbalance?

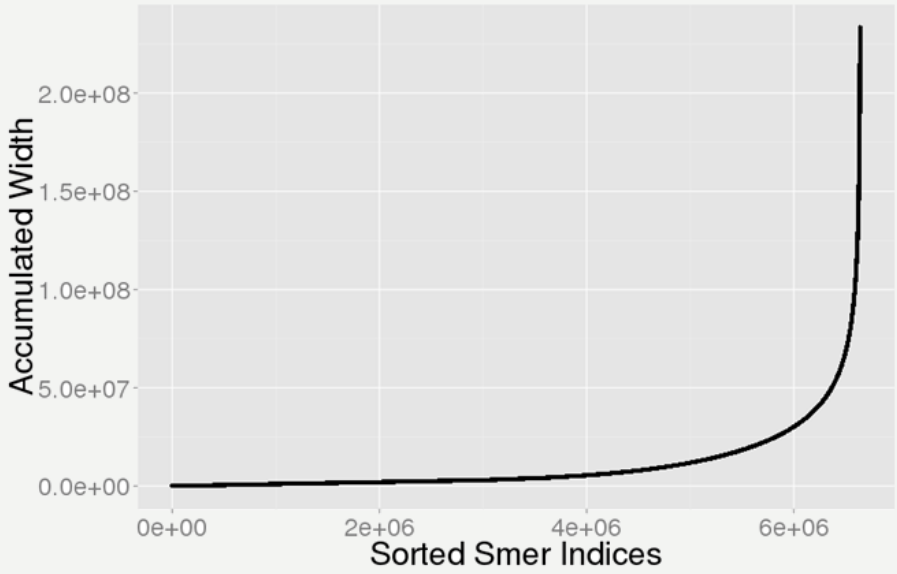
First reason:
`omp parallel for`

PAGANtec: characteristics

Second reason: **busy k-mers**

- Low average route width for whole dataset $O(10^1)$
- Accumulated Width explains behaviour
  - Very **few s-mers** account major part of width!





Lessons Learned

- OpenMP `parallel` **`task`** construct
  - Create batches of s-mers with 2000 route entries
  - Create extra tasks for routes with > 500 entries

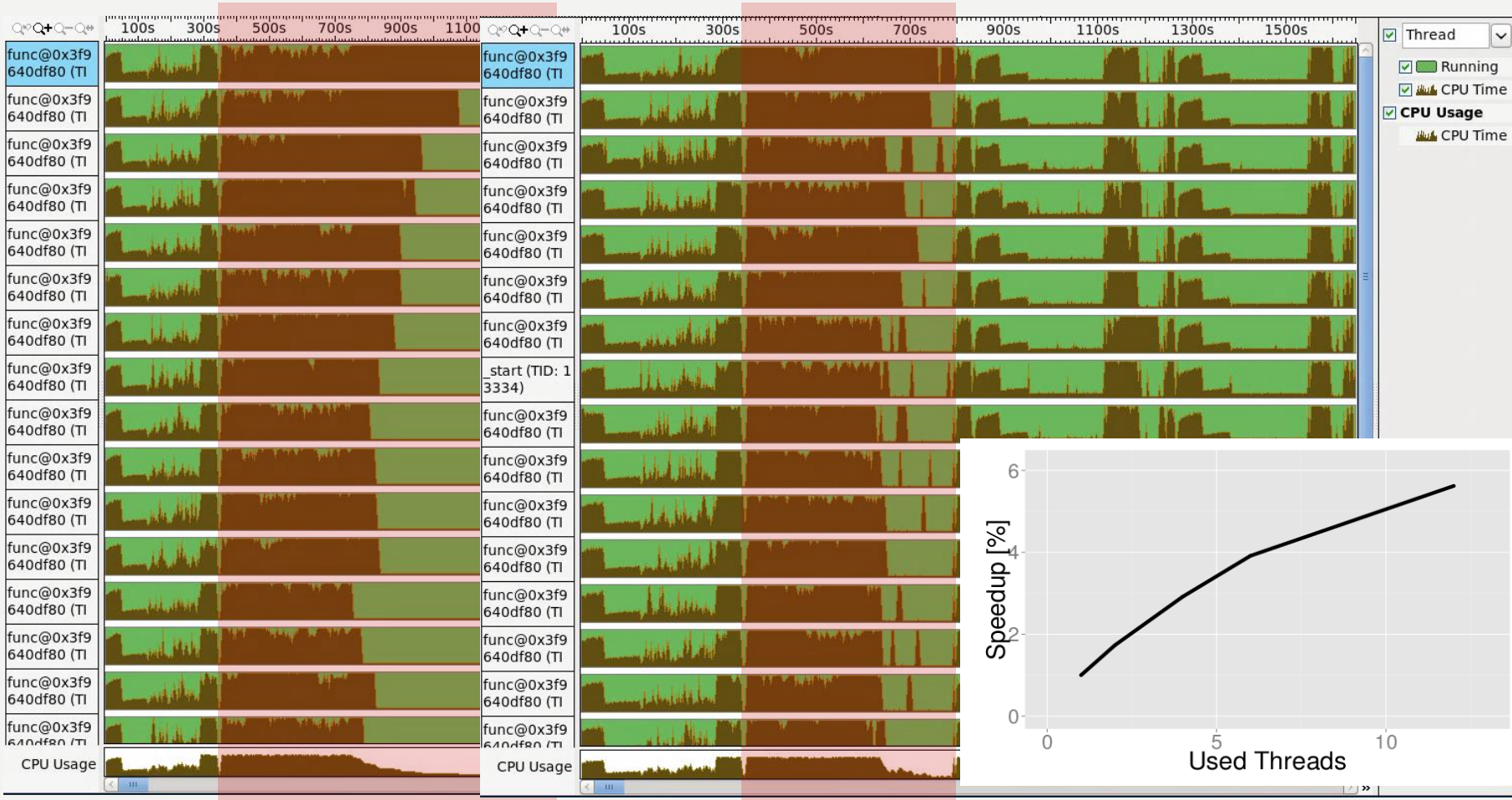
Algorithm 1.4 `TASK` work-sharing construct implementation

```
1: function APPLYFILTER(graph, smers)
2:     #pragma omp parallel
3:     #pragma omp single
4:             ▷ For each chunk of |smervec| s-mers with less than 2000 route entries
5:     for each smervec ⊆ smers ∧ |routeentries(smervec)| ≤ 2000 do
6:         #pragma omp task untied firstprivate(smervec)          ▷ Outer task
7:         for route ∈ smer do
8:             for each entries ⊆ route ∧ |routeentries(entries)| ≤ 500 do
9:                 #pragma omp task untied firstprivate(entries)   ▷ Inner task
10:                {
11:                CORRECTROUTEENTRIES(graph, entries)
12:                }
13:            end for
14:        end for
15:    end for
16: end function
```

PAGANtec: parallelism

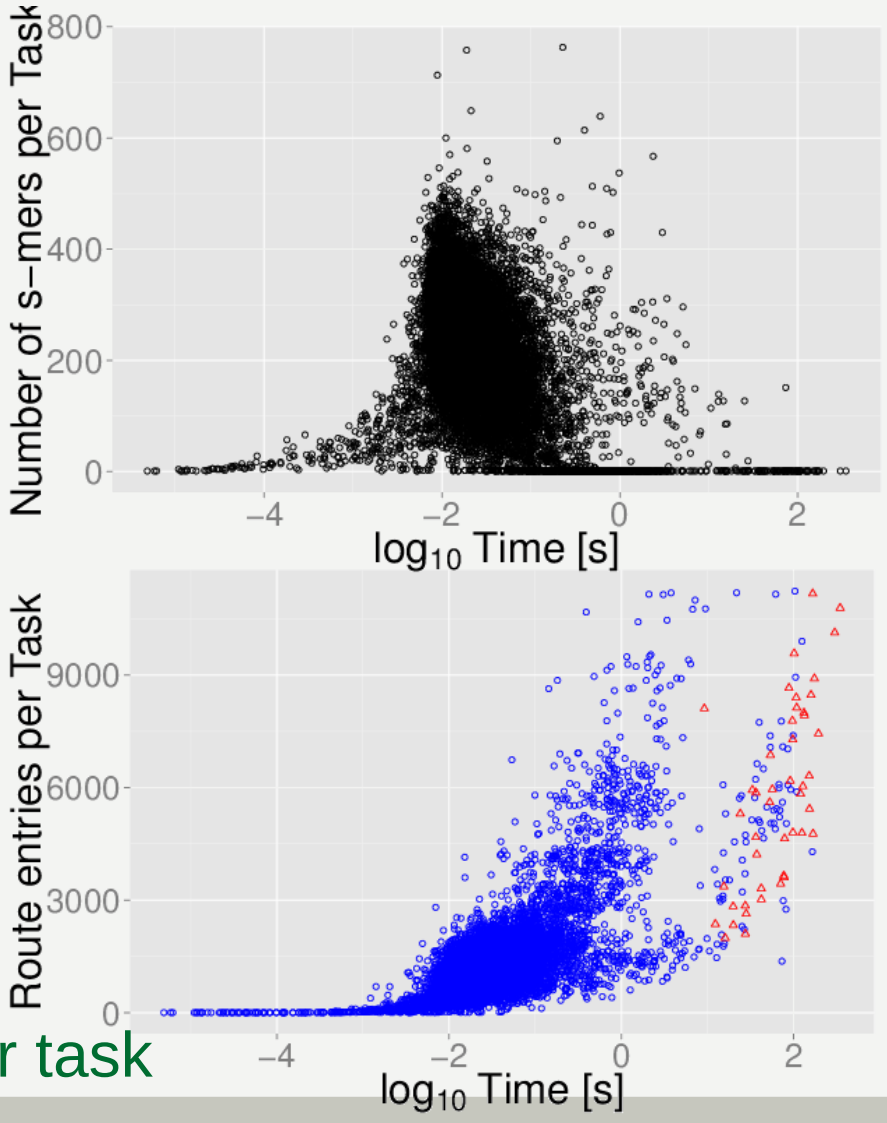
IMPROVED

- Make use of `parallel tasks`



PAGANtec: revised

- Why is there still a load imbalance?
  - How many s-mers are there per task?
  - How many route entries (width) are per task?



Red Ticks: > 1 inner task

Conclusion

- Convoluted graph structure
  - Not easily splittable into same chunks
  - Locking poses a problem

- `#pragma omp task` greatly **improves** performance
  - Not possible with `#pragma omp parallel for`
  - Manual *creation of chunks*

- Still difficulties from coarse grained **structure**
  - **Prioritize** chunk to avoid imbalance
  - Instead of leaving it at end of queue

Thank you for your attention!

Dirk Schmidl & Torsten Kuhlen

Tony Bolger & Björn Usadel

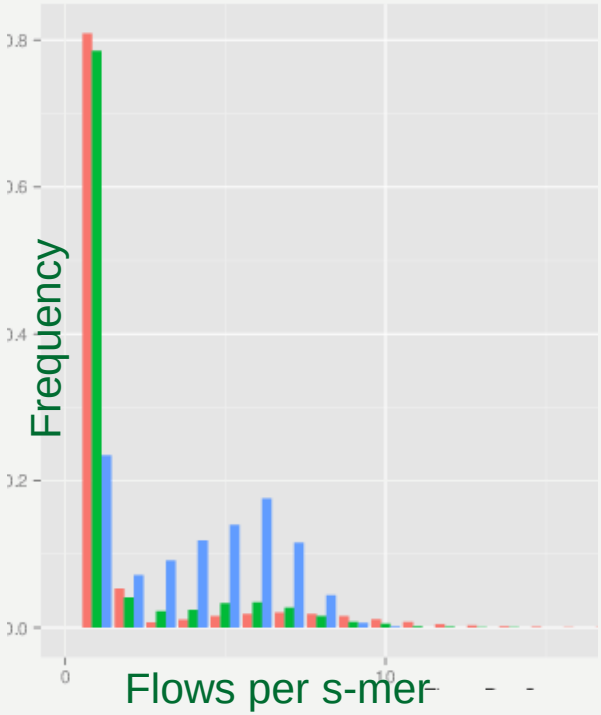
Literature

- [Le 2013] Le, H.S., Schulz, M.H., McCauley, B.M., Hinman, V.F., Bar-Joseph, Z.: Probabilistic error correction for RNA sequencing. Nucleic acids research 41(10), e109 (May 2013)

Chunk Creation

```
void* blubb

#pragma omp parallel
{
    #pragma omp single
    {

        while (newline)
        {

            blubb->push_back(f(newline));

            #pragma omp task vector(blubb) if (blubb->size() > 200) end( blubb->size() > 0 )
            {
            }
                #pragma omp task firstprivate(blubb)
            }   {
                }
    }
    #pragma omp barrier //we create additional task when barrier (or implicit barrier, e.g. end of scope)
    //and end-clause evaluates true (blubb->size() > 0)
}
        }

        #pragma omp task firstprivate(blubb)
        {
        }

    }
}
```

- Manual effort to create chunks
- Pattern observed frequently with tasks!
- Often leads to some (minor) code duplication
- Integrate into OpenMP-clause?



$$G = (V, E)$$
$$V = \{n | n \in k - mers\}$$
$$E = \{(i \quad j) | suff(j, k-1) == pref(i, k-1)\}$$