Nanopore Sequencing Technology and Tools for Genome Assembly:

Computational Analysis of the Current State, Bottlenecks and Future Directions

Damla Senol Cali, Jeremie S. Kim, Saugata Ghose,
Can Alkan and Onur Mutlu

Contact: dsenol@andrew.cmu.edu

February 16, 2019

Carnegie Mellon

SAFARI

ETH zürich

Bilkent University

Nanopore Sequencing & Tools

Nanopore Sequencing Technology and Tools for Genome Assembly: Computational Analysis of the Current State, Bottlenecks and Future Directions

Damla Senol Cali [1,*], Jeremie S. Kim [1,3], Saugata Ghose [1], Can Alkan [2*] and Onur Mutlu [3,1*]

[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA
[2]Department of Computer Engineering, Bilkent University, Bilkent, Ankara,Turkey
[3]Department of Computer Science, Systems Group, ETH Zürich, Zürich, Switzerland

Damla Senol Cali, Jeremie S. Kim, Saugata Ghose, Can Alkan, and Onur Mutlu. **"Nanopore Sequencing Technology and Tools for Genome Assembly: Computational Analysis of the Current State, Bottlenecks and Future Directions."** *Briefings in Bioinformatics* (2018).

BiB Version

arXiv Version

Executive Summary

- **Motivation: Nanopore sequencing** is an emerging and a promising technology with its ability to generate **long reads** and provide **portability**.
- **Problem:**
  - **High error rates** of the technology
  - **Critical importance of the tools** to 1) overcome the high error rates of the technology, and 2) enable fast, real-time data analysis.
- **Goal:** Analyze the multiple steps and the associated tools in the genome assembly pipeline using nanopore sequence data.
- **Key Contributions:**
  - Analysis of the tools in multiple dimensions: **accuracy**, **performance**, **memory usage** and **scalability**.
  - New **bottlenecks** and **tradeoffs** that different combinations of tools lead to
  - **Guidelines** for both **practitioners** and **tool developers**

Outline

- Background and Motivation
  - Nanopore Sequencing Technology
  - Comparison with Prior Technologies
  - Nanopore Genome Assembly Pipeline
  - Our Goal
- Experimental Methodology
- Results and Analysis
- Conclusion

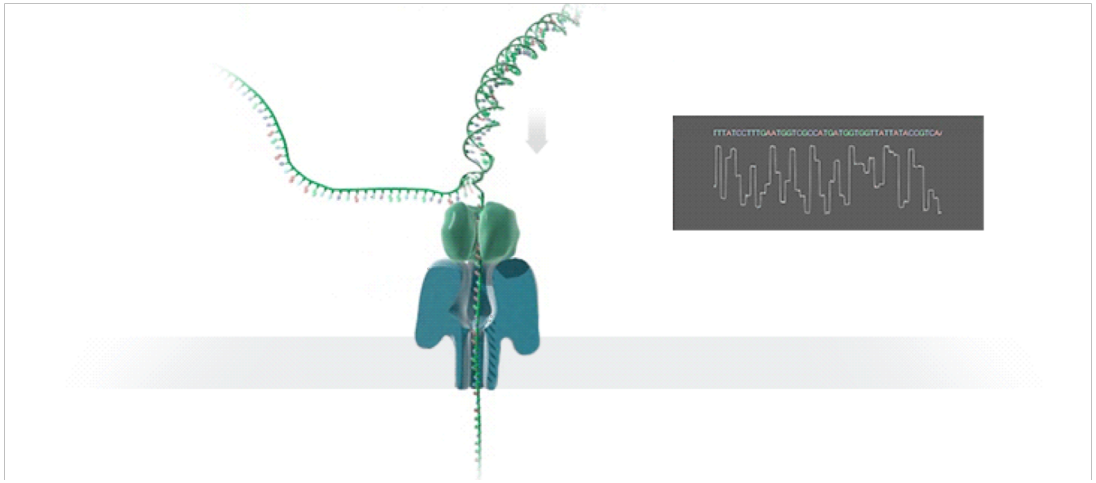
Nanopore Sequencing Technology

- **Nanopore sequencing** is an emerging and a promising single-molecule DNA sequencing technology.

- First nanopore sequencing device, **MinION**, made commercially available by **Oxford Nanopore Technologies** (ONT) in **May 2014.**
  - Inexpensive
  - Long read length (>882Kbp)
  - Produces data in real time
  - Pocket-sized and portable

Nanopore Sequencing

- **Nanopore** is a nano-scale hole.
- In nanopore sequencers, an **ionic current** passes through the nanopores.
- When the DNA strand passes through the nanopore, the sequencer measures the **change in current**.
- This change is used to identify the bases in the strand with the help of **different electrochemical structures** of the different bases.



Damla Senol Cali

Why Nanopore Sequencing?

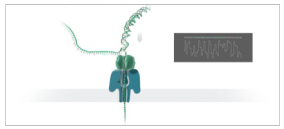
Nanopore Sequencing Technology

- Do *not* require an amplification step before the sequencing process,
- Do *not* require any labeling of the DNA or nucleotide for detection during sequencing,
- Allow sequencing of **very long reads**, and
- Provide portability, low cost and high throughput.
- One major drawback: high error rates (~10-15%)

(Prior) High-Throughput Sequencing Technologies

- Require an amplification step before the sequencing process,
- Require labeling of the DNA or nucleotide for detection during sequencing,
- Generate billions of short but accurate reads,
- Provide high throughput, high speed and low cost,
- Suffers from massive amount of data and short reads, which poses challenges due to the repetitive sequences in the genome.

Nanopore Genome Assembly Pipeline



Raw signal data

Basecalling

DNA reads

Read-to-Read Overlap Finding

Overlaps

Assembly

Assembly

Draft assembly

Read Mapping (Optional)

Mappings of reads against draft assembly

Improved assembly

Polishing (Optional)

Our Goal

- Comprehensively analyze the multiple steps and the associated state-of-the-art tools in genome assembly pipelines using nanopore sequence data in terms of **accuracy**, **performance**, **memory usage**, and **scalability**.

- Reveal **bottlenecks** and **trade-offs** that different combinations of tools lead to.

- Provide **guidelines** for both **practitioners**, such that they can determine the appropriate tools and tool combinations that can satisfy their goals, and **tool developers**, such that they can make design choices to improve current and future tools.

Outline

- Background and Motivation
- **Experimental Methodology**
- Results and Analysis
- Conclusion

Experimental Methodology

Name	Model	CPU specifications	Main memory specifications
System 1	40-core Intel® Xeon® E5-2630 v4 CPU @ 2.20GHz	20 physical cores 2 threads per core 40 logical cores with hyper-threading**	128GB DDR4 2 channels, 2 ranks/channel Speed: 2400MHz
System 2 (desktop)	8-core Intel® Core i7-2600 CPU @ 3.40GHz	4 physical cores 2 threads per core 8 logical cores with hyper-threading**	16GB DDR3 2 channels, 2 ranks/channel Speed: 1333MHz
System 3 (big-mem)	80-core Intel® Xeon® E7-4850 CPU @ 2.00GHz	40 physical cores 2 threads per core 80 logical cores with hyper-threading**	1TB DDR3 8 channels, 4 ranks/channel Speed: 1066MHz

Experimental Methodology (cont.)

Accuracy Metrics

- Average Identity
  - Percentage **similarity** between the assembly and the reference genome
  - Higher ($\simeq 100\%$) is preferred
- Coverage
  - **Ratio of the #aligned bases** in the reference genome to the length of reference genome
  - Higher ($\simeq 100\%$) is preferred
- Number of mismatches
  - Total number of **single-base differences** between the assembly and the reference genome
  - Lower ($\simeq 0$) is preferred
- Number of indels
  - Total number of **insertions and deletions** between the assembly and the reference genome
  - Lower ($\simeq 0$) is preferred

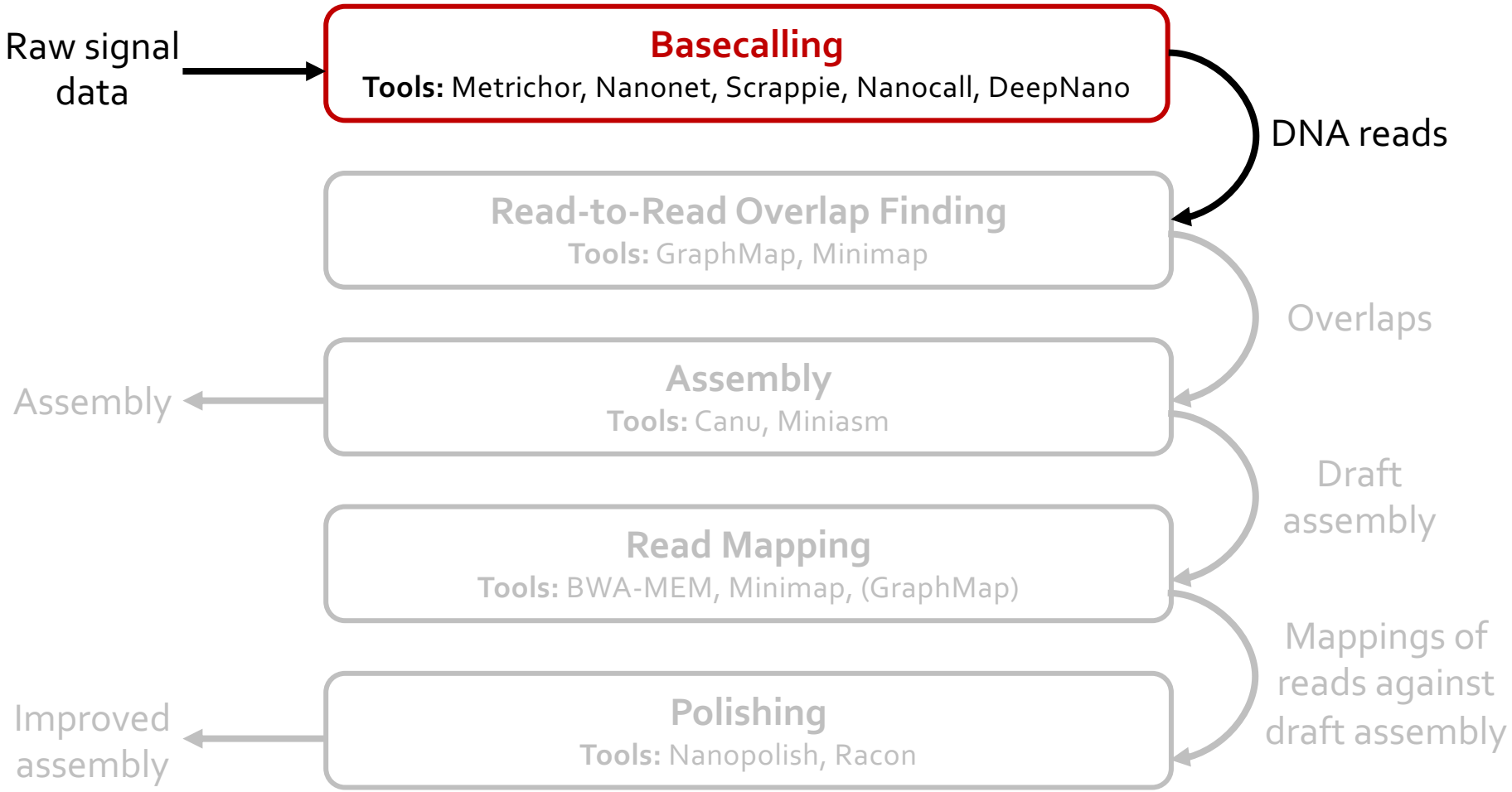
Performance Metrics

- Wall clock time
- Peak memory usage
- Parallel speedup

Outline

- Background and Motivation
- Experimental Methodology
- **Results and Analysis**
  - **Basecalling Tools**
    - **Accuracy**
    - **Performance**
  - Read-to-Read Overlap Finding Tools
  - Assembly Tools
  - Read Mapping and Polishing Tools (optional)
- Conclusion

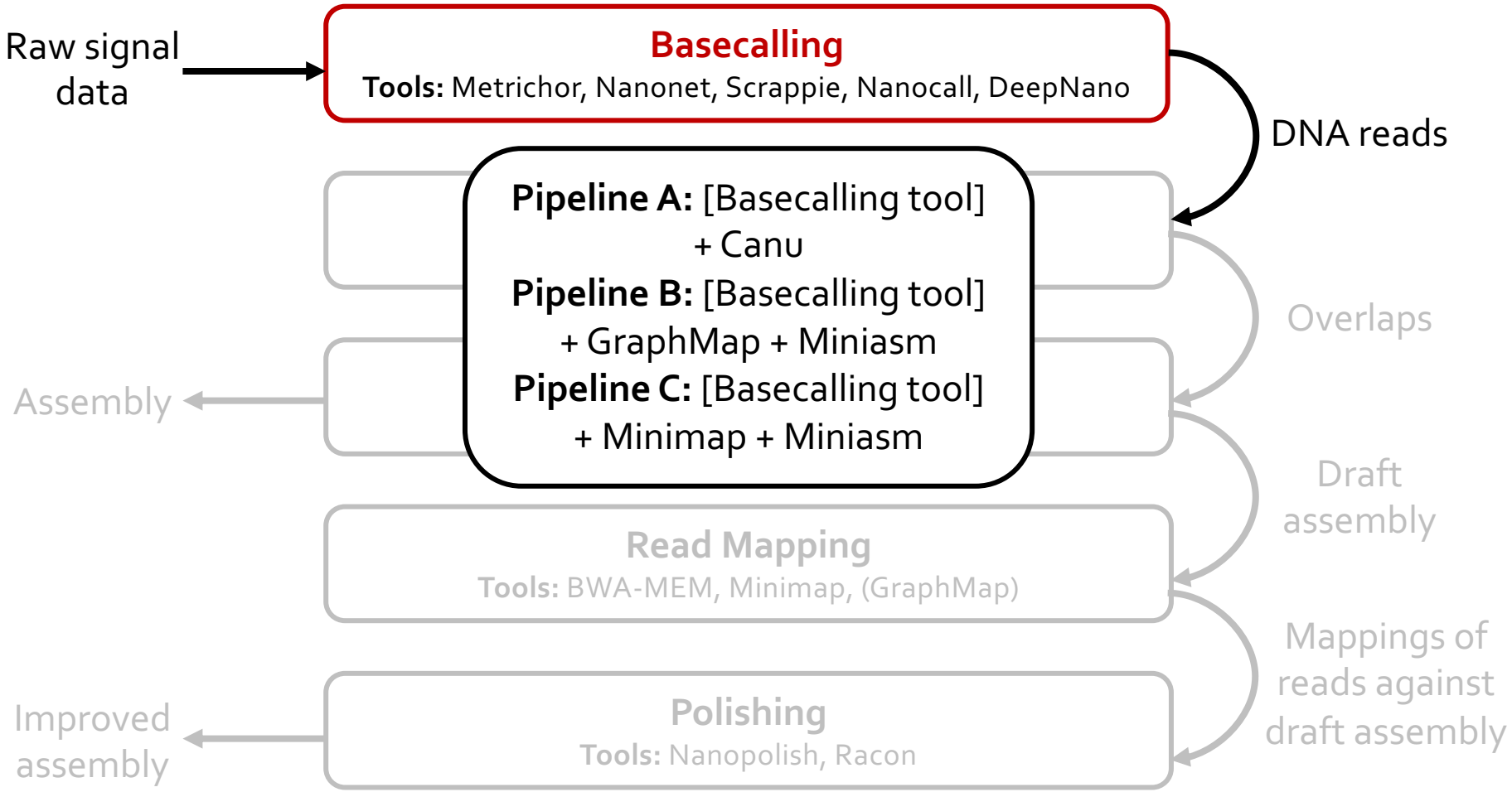
Nanopore Genome Assembly Pipeline



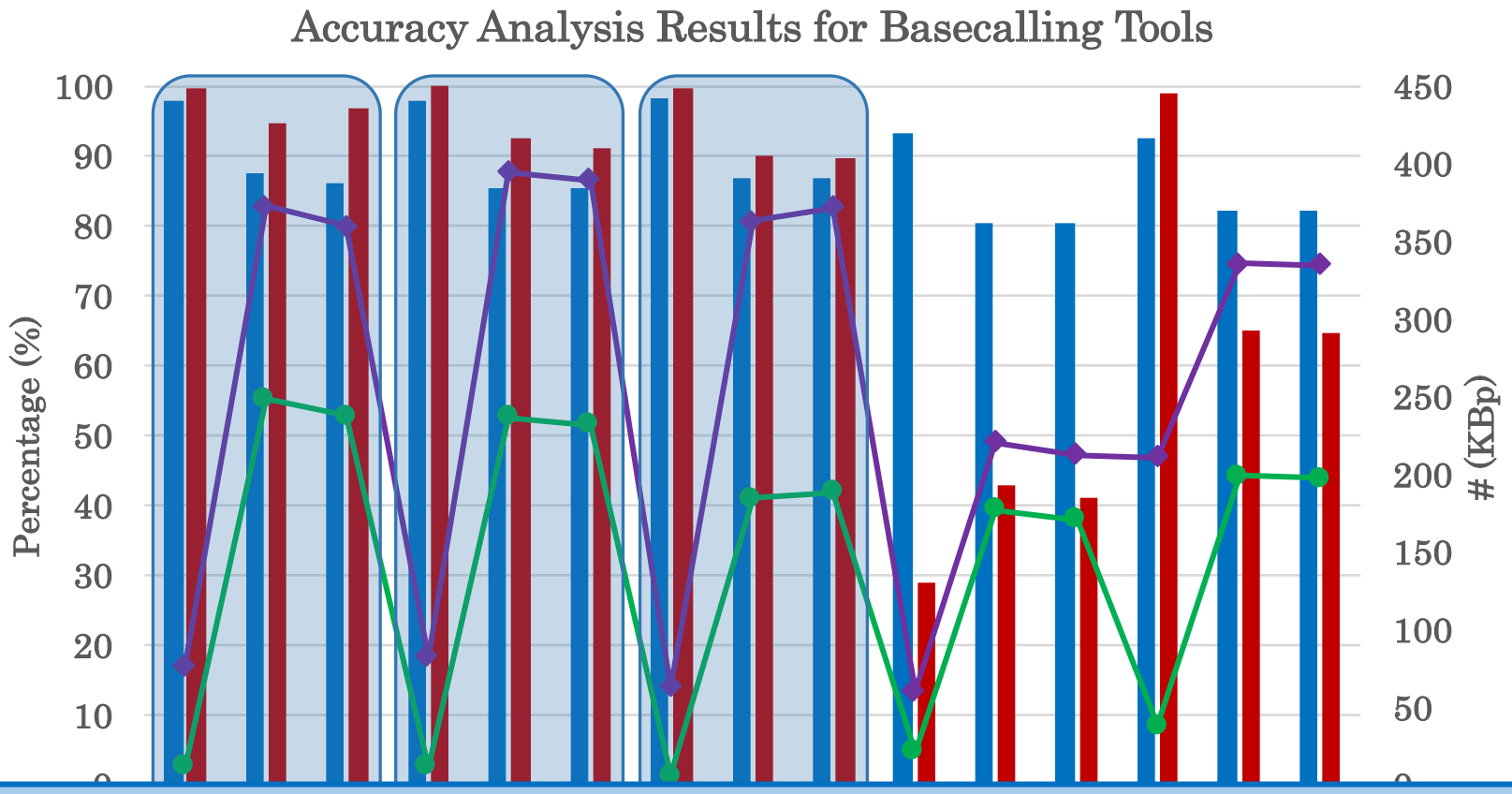
Basecalling Tools

- Metrichor
  - ONT's cloud-based basecaller
  - Uses recurrent neural networks (**RNN**) for basecalling
- Nanonet
  - ONT's offline and open-source alternative for Metrichor
  - Uses **RNN** for basecalling
- Scrappie
  - ONT's newest basecaller that explicitly addresses basecalling errors in homopolymer regions
- Nanocall [David+, Bioinformatics 2016]
  - Uses Hidden Markov Models (**HMM**) for basecalling
- DeepNano [Boža+, PloS One 2017]
  - Uses **RNN** for basecalling

Nanopore Genome Assembly Pipeline

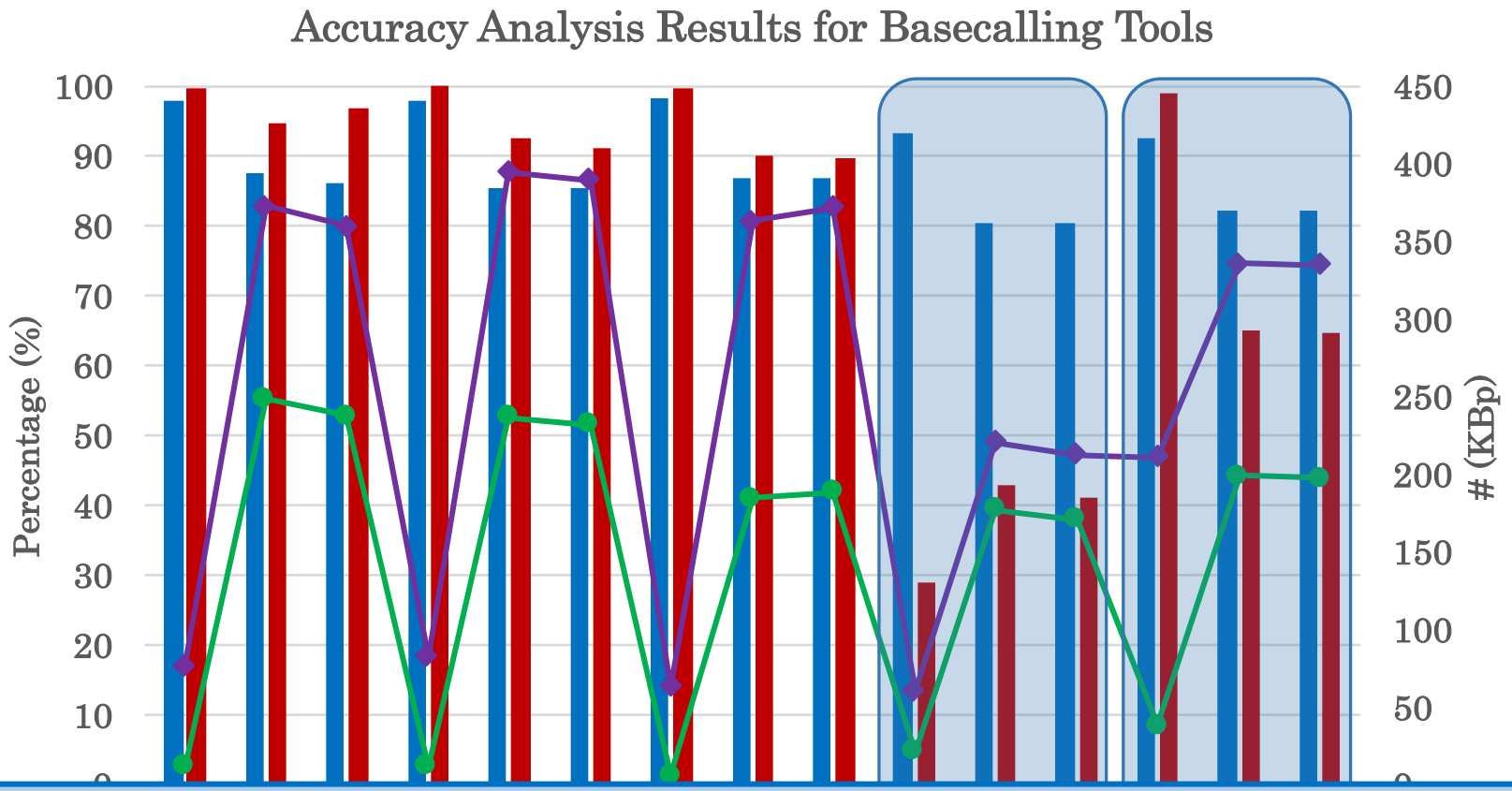


Basecalling – Accuracy



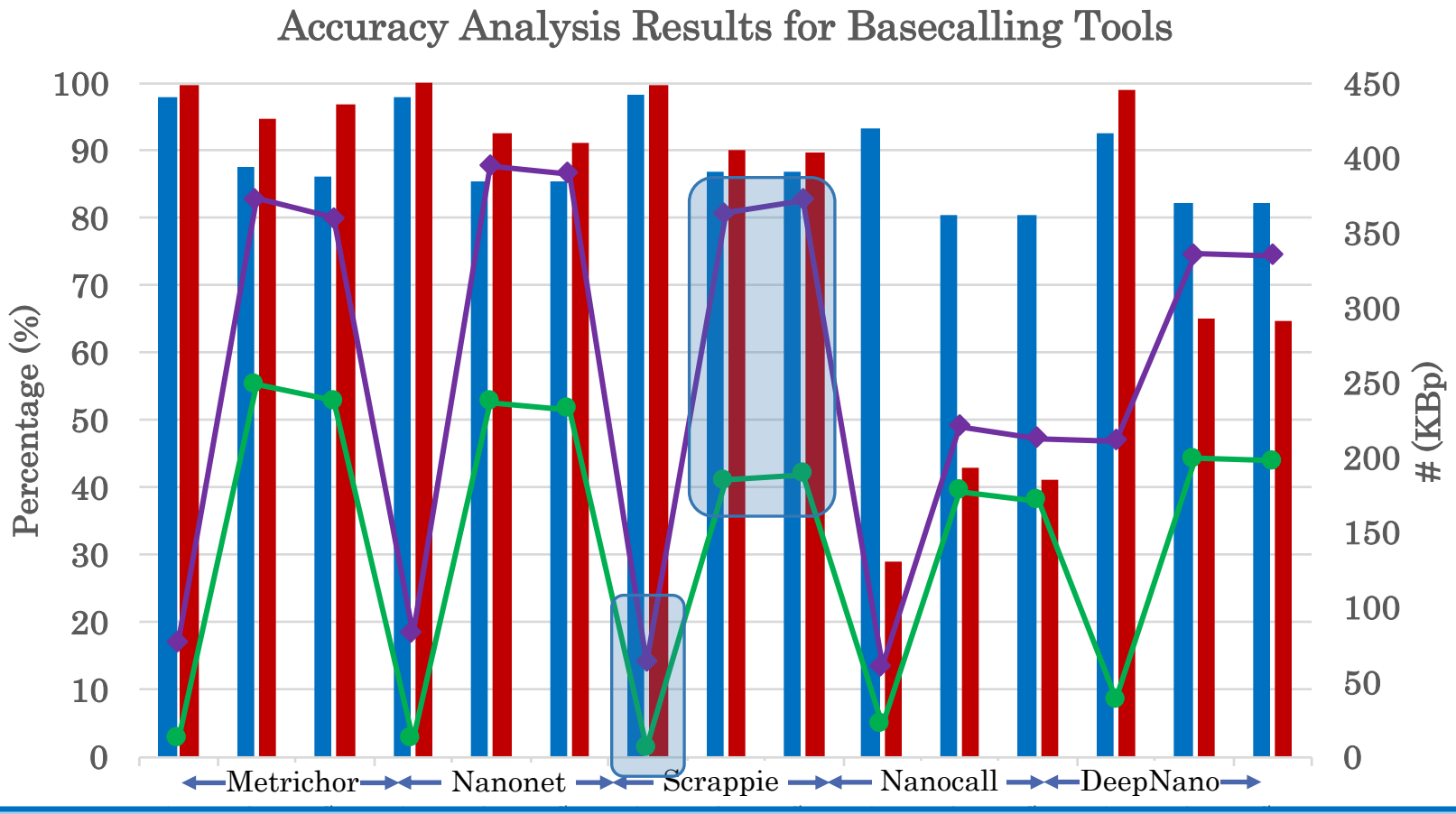
Observation 1-a: *Metrichor, Nanonet and Scrappie have similar identity and coverage trends among all of the evaluated scenarios.*

Basecalling – Accuracy



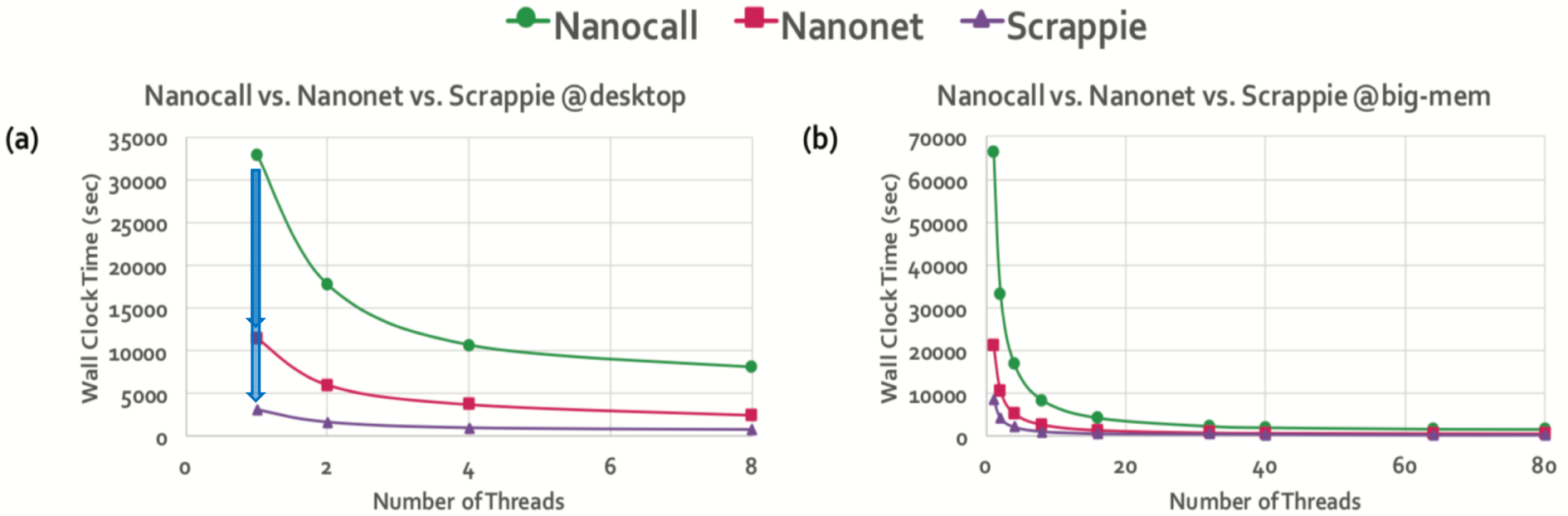
Observation 1-b: *However, Nanocall and DeepNano cannot reach these three basecallers' accuracies: they have lower identity and lower coverage.*

Basecalling – Accuracy



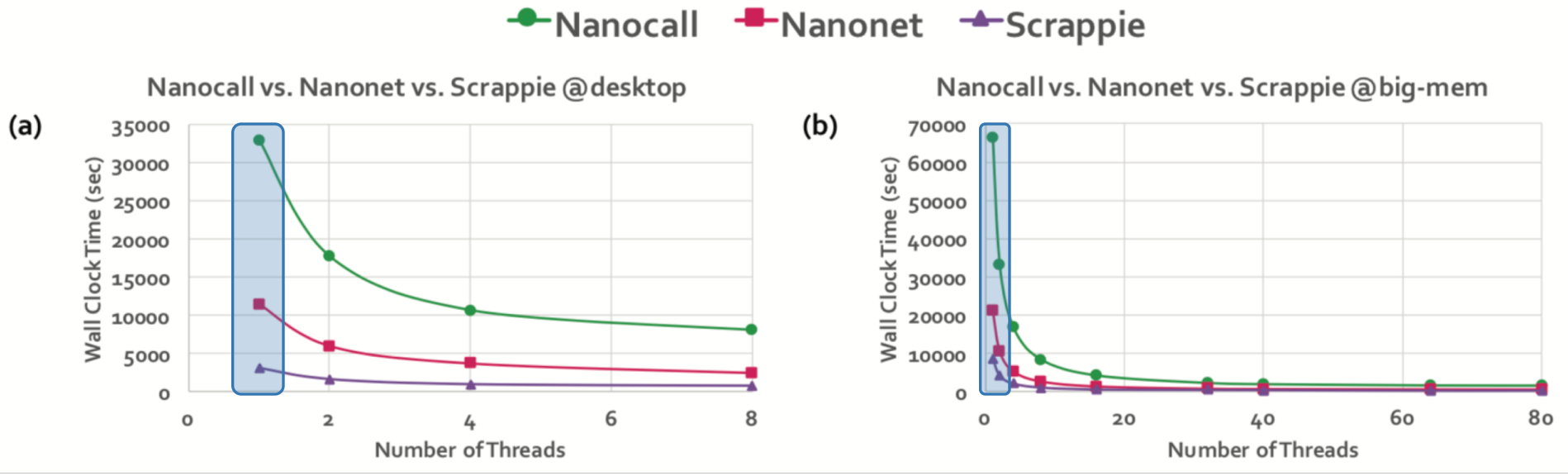
Observation 1-c: *Scrappie has the highest accuracy with the lowest number of mismatches and indels.*

Basecalling – Speed



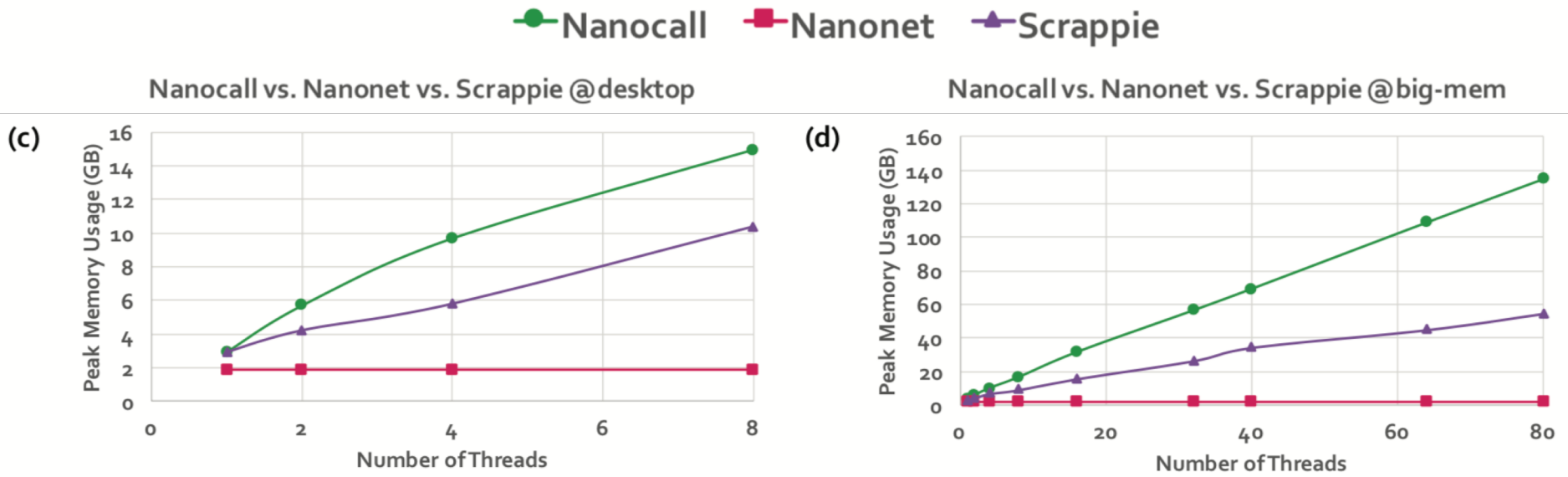
Observation 2: *RNN-based basecallers, Nanonet and Scrappie are faster than HMM-based basecaller, Nanocall.*

Basecalling – Speed



Observation 3: *When #threads=1, desktop is approximately 2x faster than big-mem because of desktop's higher CPU frequency. It is an indication that all of these three tools are computationally expensive.*

Basecalling – Memory



Observation 4: *Scrappie and Nanocall have a linear increase in memory usage when number of threads increases. In contrast, Nanonet has a constant memory usage for all evaluated thread units.*

Basecalling – Summary

- The choice of the tool for the basecalling step plays an important role to overcome the high error rates of nanopore sequencing technology.

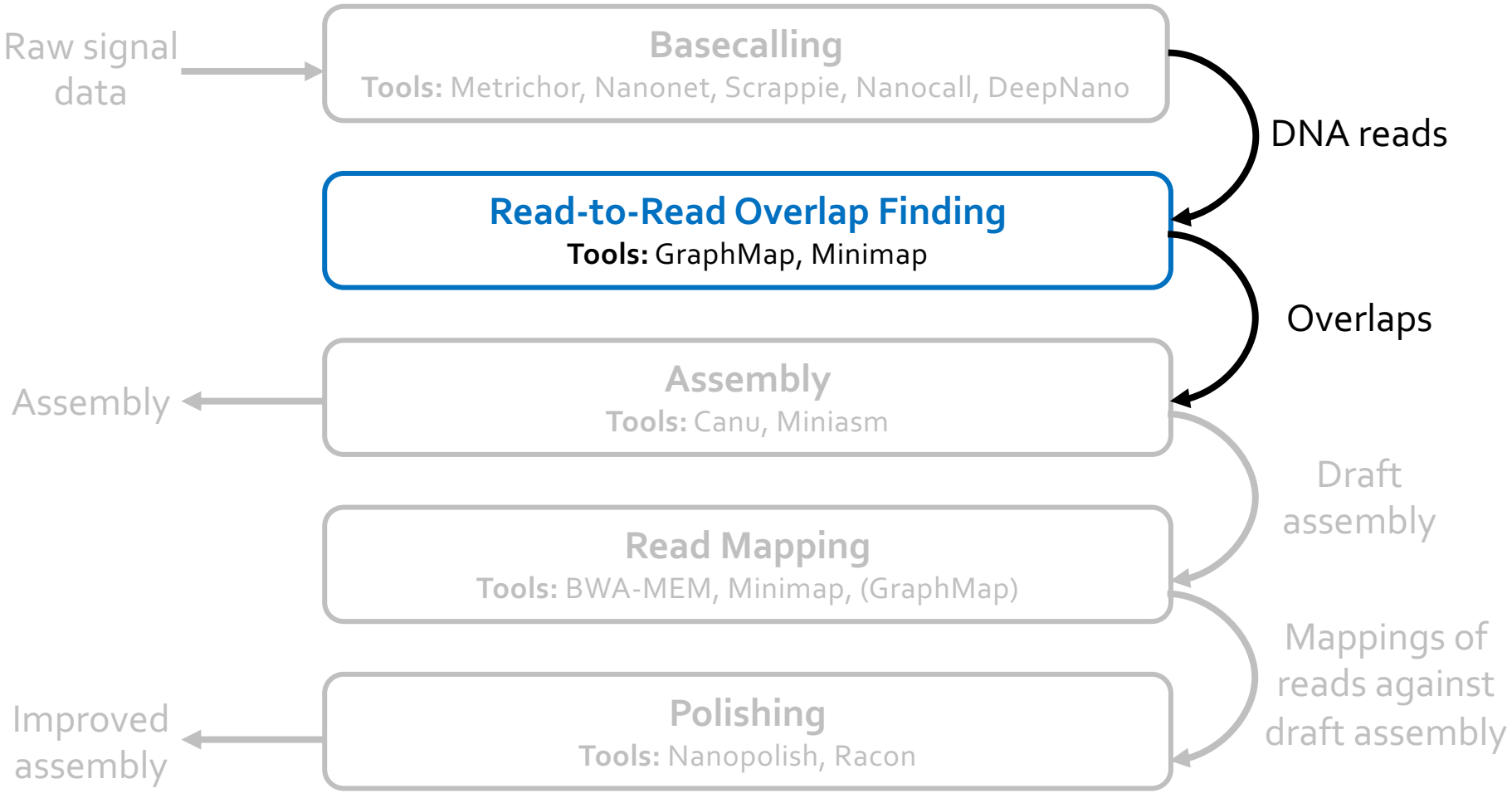
- Basecalling with RNNs (e.g. Metrichor, Nanonet, Scrappie) provides higher accuracy and higher speed than basecalling with HMMs.

- The newest basecaller of ONT, Scrappie, also has the potential to overcome the homopolymer basecalling problem.

Outline

- Background and Motivation
- Experimental Methodology
- **Results and Analysis**
  - Basecalling Tools
  - **Read-to-Read Overlap Finding Tools**
    - **Accuracy**
    - **Performance**
  - Assembly Tools
  - Read Mapping and Polishing Tools (optional)
- Conclusion

Nanopore Genome Assembly Pipeline

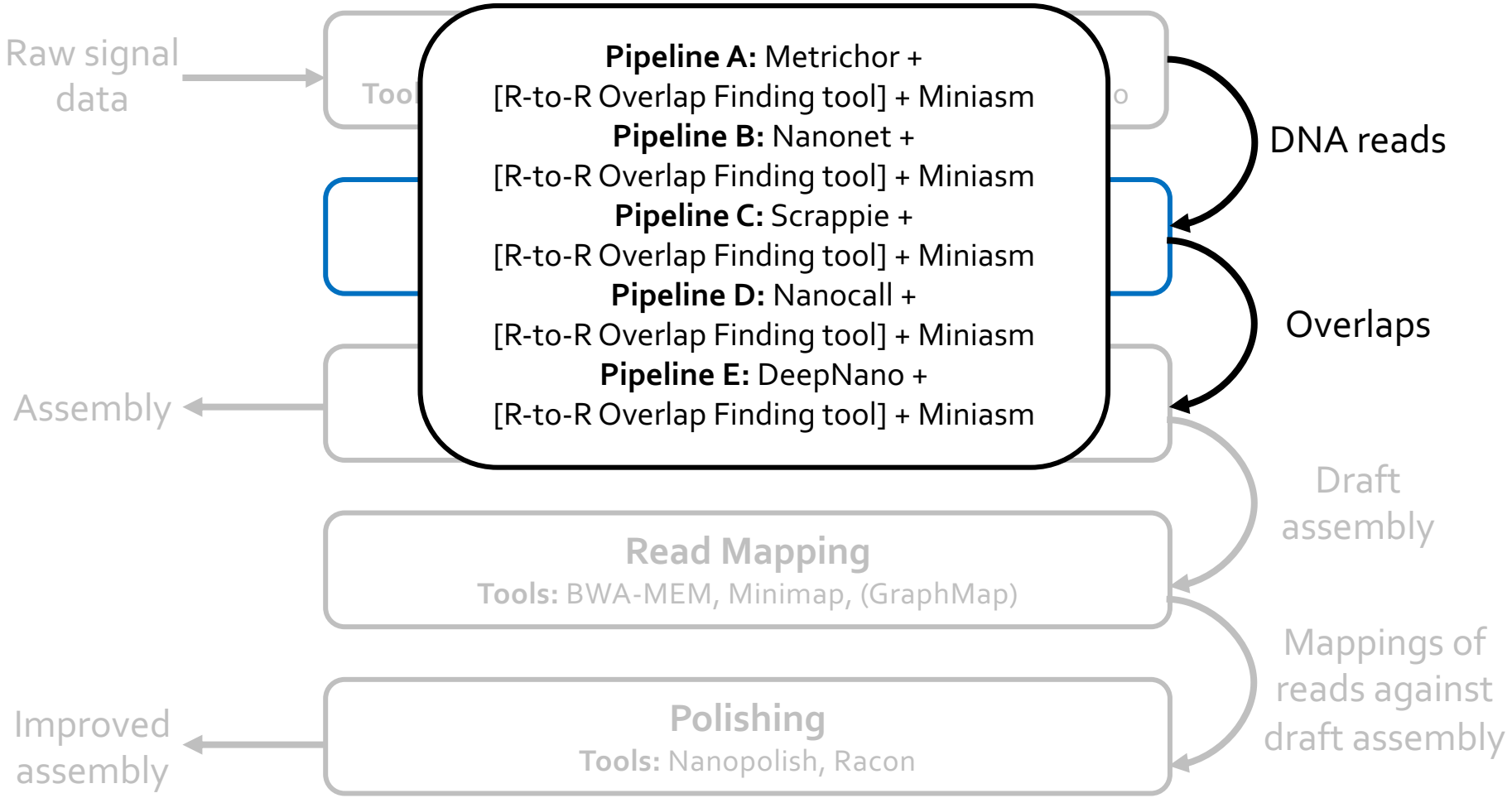


Read-to-Read Overlap Finding Tools

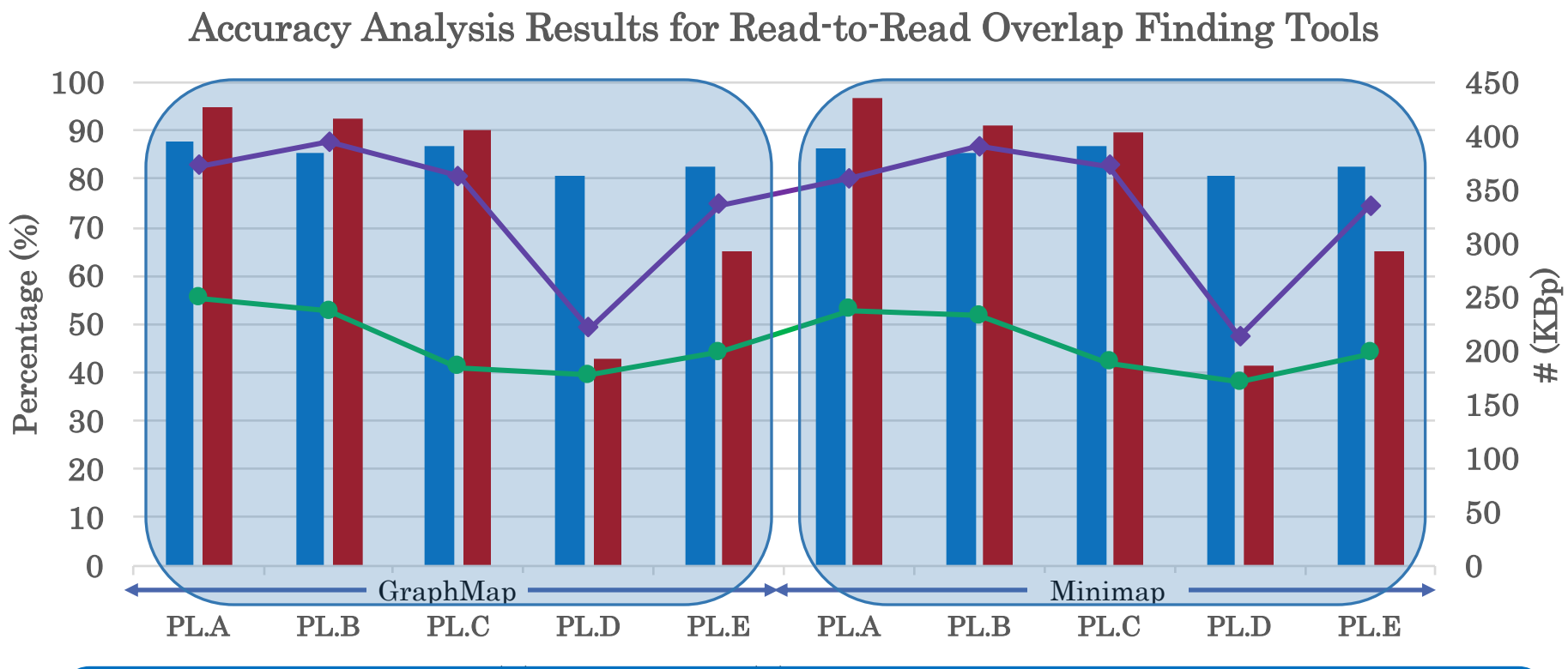
- GraphMap [Sović+, Nature Communications 2016]
  - First partitions the entire read data set into *k*-length substrings (i.e., *k*-mers), and then stores them in a hash table with the positions.
  - Detects the overlaps by finding the *k*-mer similarity between any two given reads, using the generated hash table.

- Minimap [Li+, Bioinformatics 2016]
  - Partitions the entire read data set into *k*-mers, but instead of creating a hash table for the full set of *k*-mers, finds the minimum representative set of *k*-mers, called *minimizers*, and creates a hash table with only these minimizers.
  - Finds the overlaps between two reads by finding minimizer similarity.

Nanopore Genome Assembly Pipeline

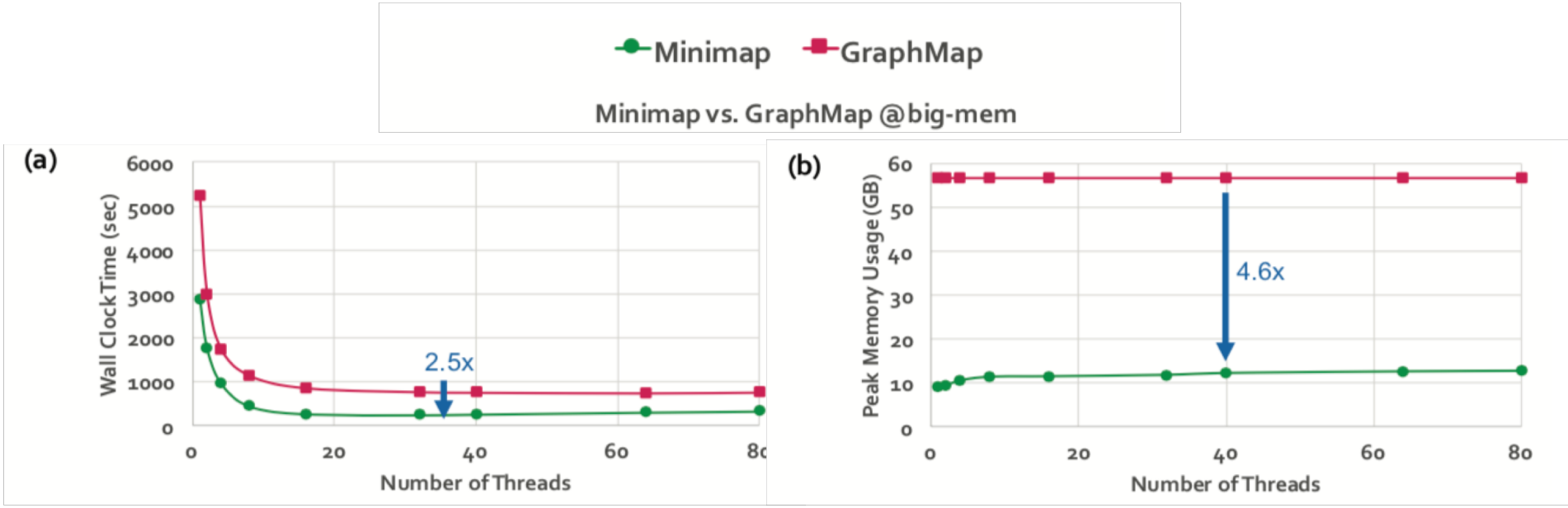


R-to-R Overlap Finding – Accuracy



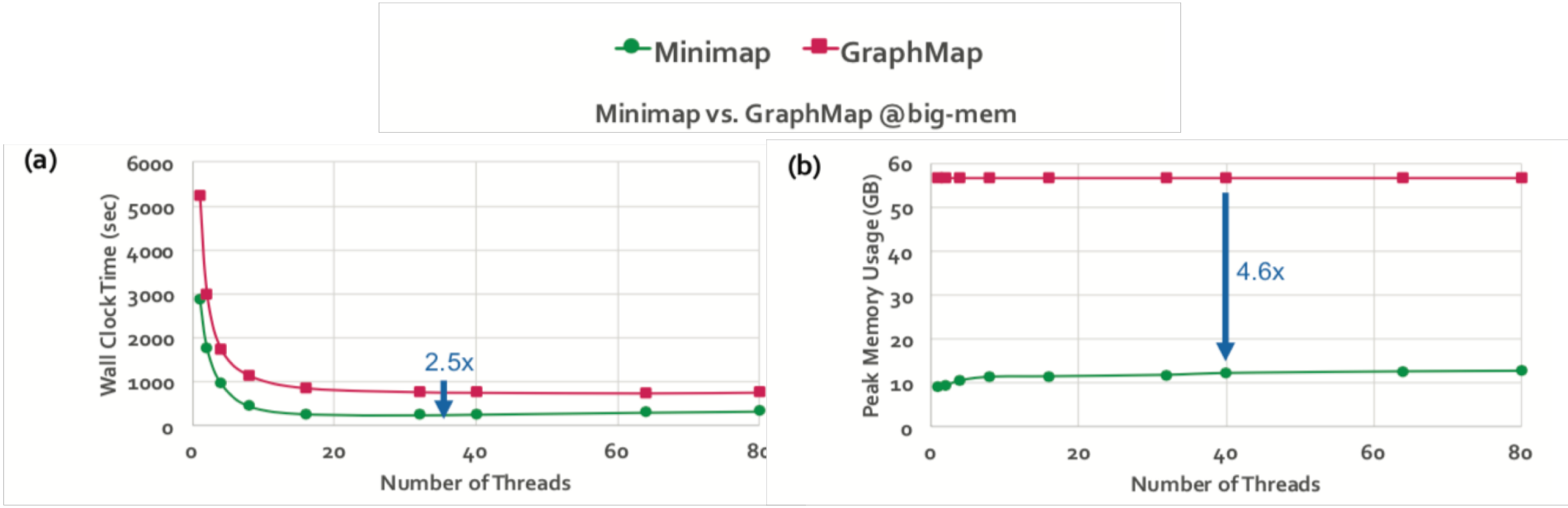
Observation 5: *Pipelines with GraphMap or Minimap end up with similar accuracy results.*

R-to-R Overlap Finding – Performance



Observation 6: *The memory usage of both GraphMap and Minimap is dependent on the hash table size but independent of number of threads. Minimap requires 4.6x less memory than GraphMap, on average.*

R-to-R Overlap Finding – Performance



Observation 7: *Minimap is 2.5x faster than GraphMap, on average. Since in Minimap, the size of dataset that needs to be scanned is greatly shrunk by storing minimizers instead of k-mers, it performs much less computation than GraphMap.*

R-to-R Overlap Finding – Summary

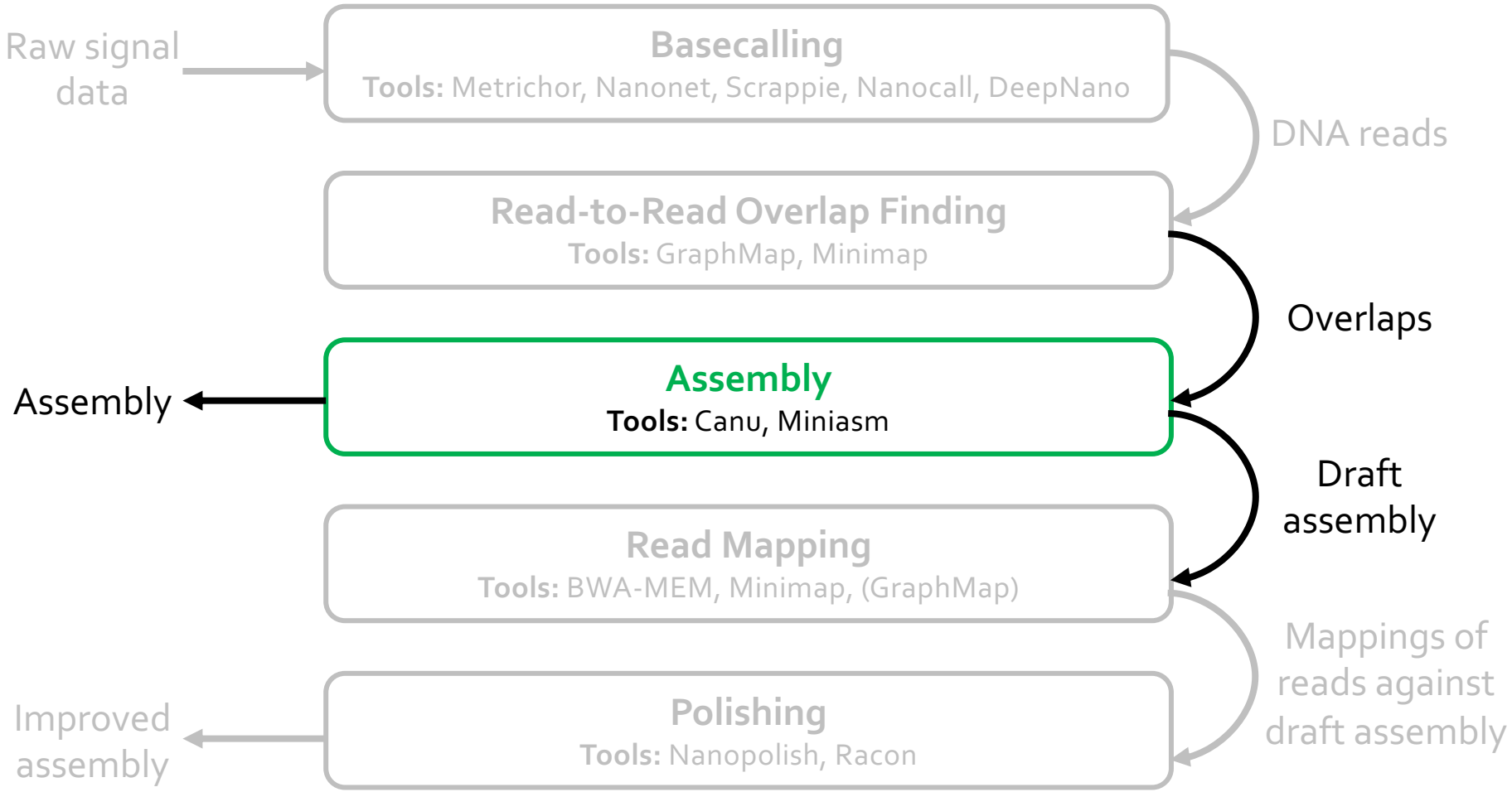
- Storing minimizers instead of all *k*-mers, as done by Minimap, does *not* affect the overall accuracy of the first three steps of the pipeline.

- By storing minimizers, Minimap has a much lower memory usage and thus much higher performance than GraphMap.

Outline

- Background and Motivation
- Experimental Methodology
- **Results and Analysis**
  - Basecalling Tools
  - Read-to-Read Overlap Finding Tools
  - **Assembly Tools**
    - **Accuracy**
    - **Performance**
  - Read Mapping and Polishing Tools (optional)
- Conclusion

Nanopore Genome Assembly Pipeline



Assembly Tools

- Canu [Koren+, Genome Research 2017]
  - Performs error-correction as the initial step of its own pipeline
    - Improves the accuracy of the bases in the reads
    - Computationally-expensive
  - After the error-correction step, finds overlaps between corrected reads and constructs a draft assembly

- Miniasm [Li+, Bioinformatics 2016]
  - Skips the error-correction step, and constructs the draft assembly from the uncorrected read overlaps computed in the previous step.
  - Lowers computational cost but the accuracy of the draft assembly depends directly on the accuracy of the uncorrected basecalled reads.

Assembly – Accuracy & Performance

Observation 8: *Canu provides higher accuracy than Miniasm, with the help of the error-correction step that is present in its own pipeline. On average, Canu provides 96.1% identity whereas Miniasm provides 84.4% identity.*

Observation 9: *Canu is much more computationally intensive and greatly (i.e., by 1096.3x) slower than Miniasm, because of its very expensive error-correction step.*

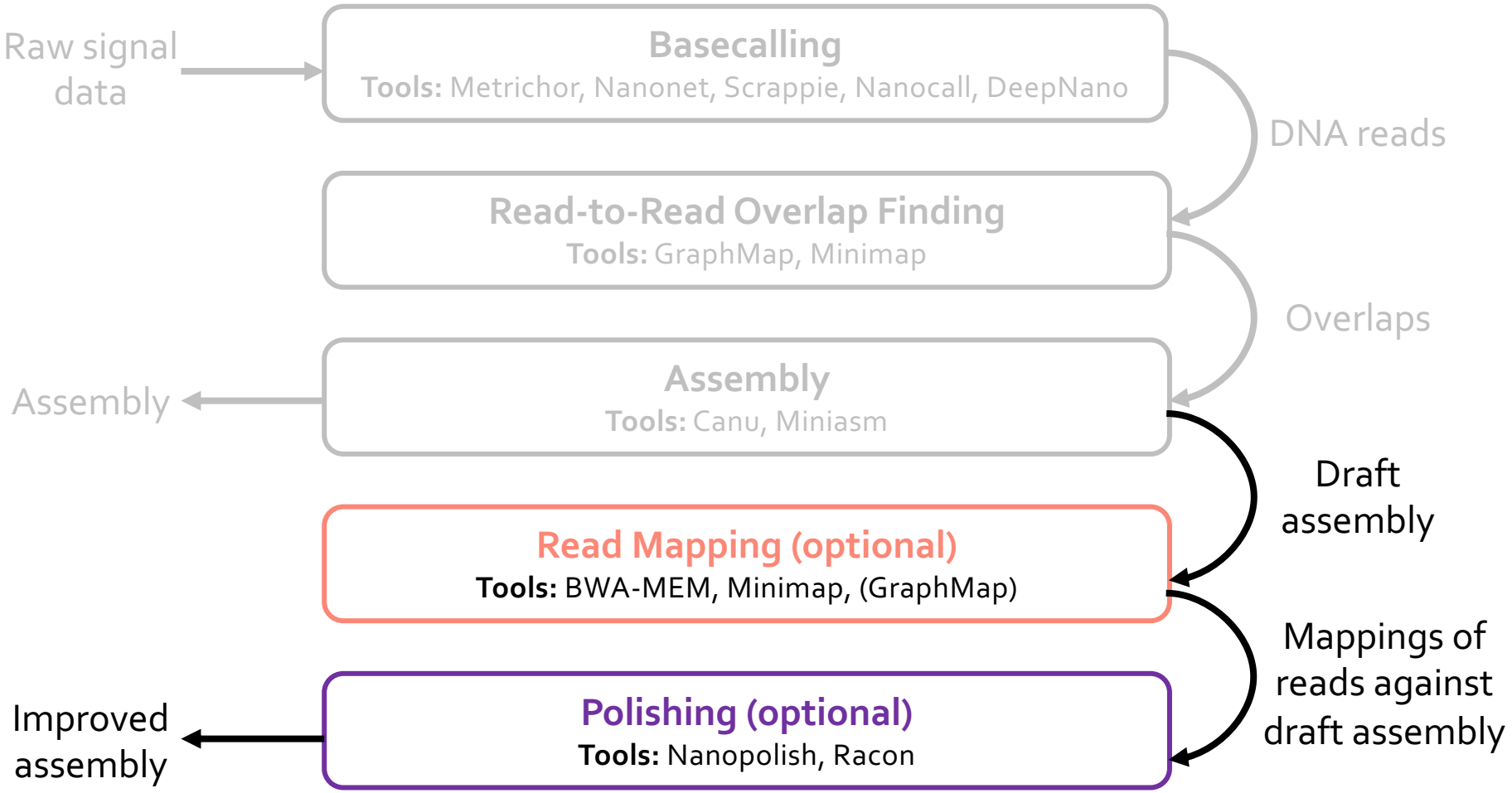
Assembly – Summary

- There is a trade-off between accuracy and performance when deciding on the appropriate tool for the assembly step.
- Canu produces highly accurate assemblies, but it is resource intensive and slow. In contrast, Miniasm is a fast assembler, but it cannot produce as accurate draft assemblies as Canu.
- Miniasm can potentially be used for fast initial analysis and then further polishing can be applied in the next step to produce higher-quality assemblies.

Outline

- Background and Motivation
- Experimental Methodology
- **Results and Analysis**
  - Basecalling Tools
  - Read-to-Read Overlap Finding Tools
  - Assembly Tools
  - **Read Mapping and Polishing Tools (optional)**
- Conclusion

Nanopore Genome Assembly Pipeline



Read Mapping & Polishing – Summary

- Further polishing can significantly increase the accuracy of the assemblies.
- Pipelines with Minimap and Racon can provide a significant speedup compared with the pipelines with BWA-MEM and Nanopolish, while resulting with high-quality consensus sequences.
- More details in the paper..

Outline

- ❑ Background and Motivation
- ❑ Experimental Methodology
- ❑ Results and Analysis
- ❑ **Conclusion**

Future Implications

- The choice of the tool for **basecalling** plays a critical role in overcoming the high error rates of nanopore sequencing technology.
  - RNNs perform better than HMMs in terms of both accuracy and performance.

- Since **parallelizing** the tool can increase the memory usage, dividing the input data into **batches**, or limiting the memory usage of each thread, or **dividing the computation** instead of dividing the dataset between simultaneous threads can prevent large increases in memory usage, while providing **performance benefits from parallelization**.

- In the future, **laptops** may become a popular platform for running genome assembly tools, as the **portability** of a laptop makes it a good fit for **in-field analysis**.
  - Greater memory constraints
  - Lower computational power, and
  - Limited battery life.

Conclusion

- **Motivation: Nanopore sequencing** is an emerging and a promising technology with its ability to generate **long reads** and provide **portability**.
- **Problem:**
  - **High error rates** of the technology
  - **Critical importance of the tools** to 1) overcome the high error rates of the technology, and 2) enable fast, real-time data analysis.
- **Goal:** Analyze the multiple steps and the associated tools in the genome assembly pipeline using nanopore sequence data.
- **Key Contributions:**
  - Analysis of the tools in multiple dimensions: **accuracy**, **performance**, **memory usage** and **scalability**.
  - New **bottlenecks** and **tradeoffs** that different combinations of tools lead to
  - **Guidelines** for both **practitioners** and **tool developers**

Nanopore Sequencing & Tools

Nanopore Sequencing Technology and Tools for Genome Assembly: Computational Analysis of the Current State, Bottlenecks and Future Directions

Damla Senol Cali [1,*], Jeremie S. Kim [1,3], Saugata Ghose [1], Can Alkan [2*] and Onur Mutlu [3,1*]

[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA
[2]Department of Computer Engineering, Bilkent University, Bilkent, Ankara,Turkey
[3]Department of Computer Science, Systems Group, ETH Zürich, Zürich, Switzerland

Damla Senol Cali, Jeremie S. Kim, Saugata Ghose, Can Alkan, and Onur Mutlu. "Nanopore Sequencing Technology and Tools for Genome Assembly: Computational Analysis of the Current State, Bottlenecks and Future Directions." *Briefings in Bioinformatics* (2018).

BiB Version

arXiv Version

Nanopore Sequencing Technology and Tools for Genome Assembly:

Computational Analysis of the Current State, Bottlenecks and Future Directions

Damla Senol Cali, Jeremie S. Kim, Saugata Ghose,
Can Alkan and Onur Mutlu

Contact: dsenol@andrew.cmu.edu

February 16, 2019

Carnegie Mellon

SAFARI

ETH zürich

Bilkent University

Backup Slides

Genome Sequencing

Genome

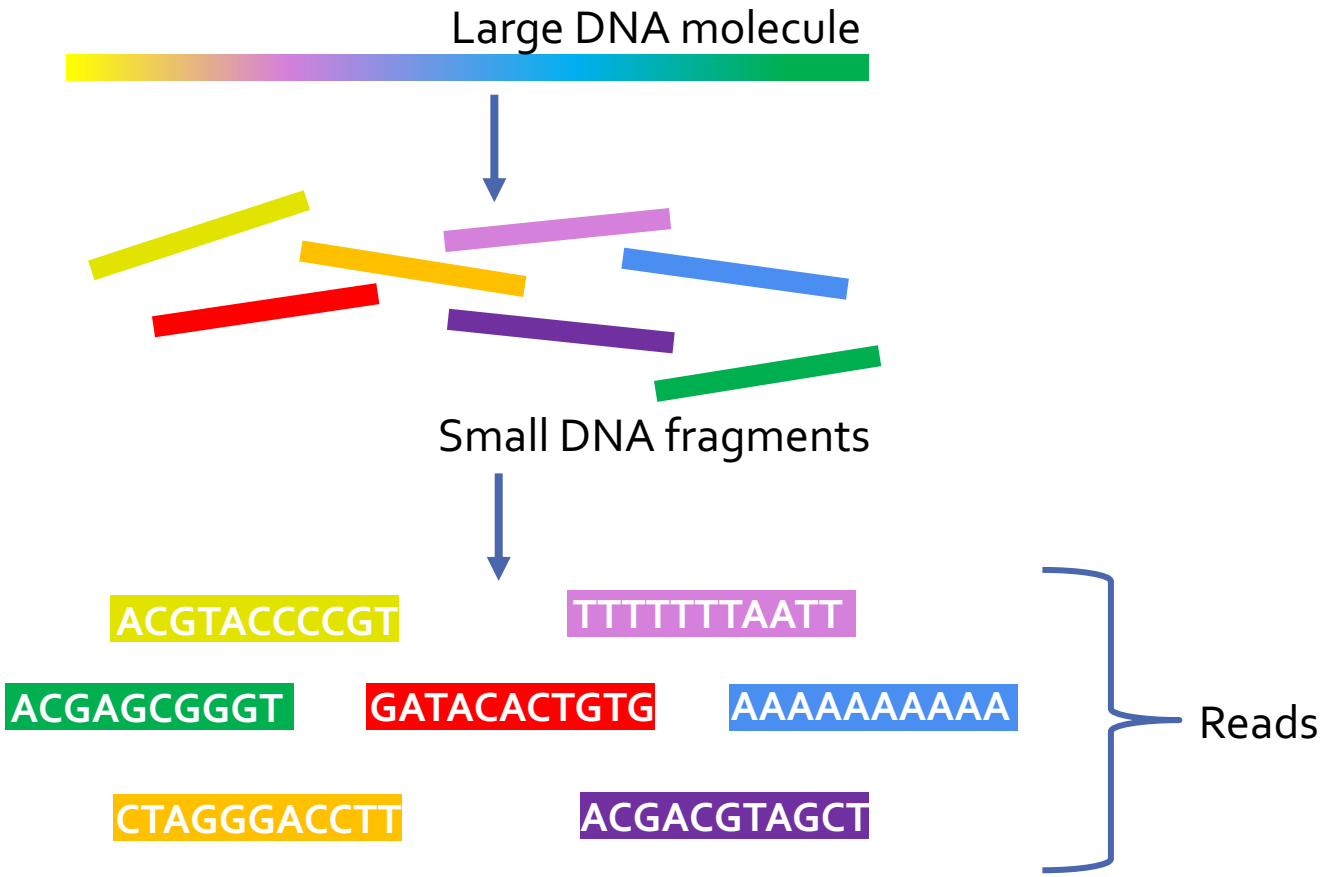
DNA

Genome sequencing is the process of determining the order of the DNA sequence in an organism's genome.

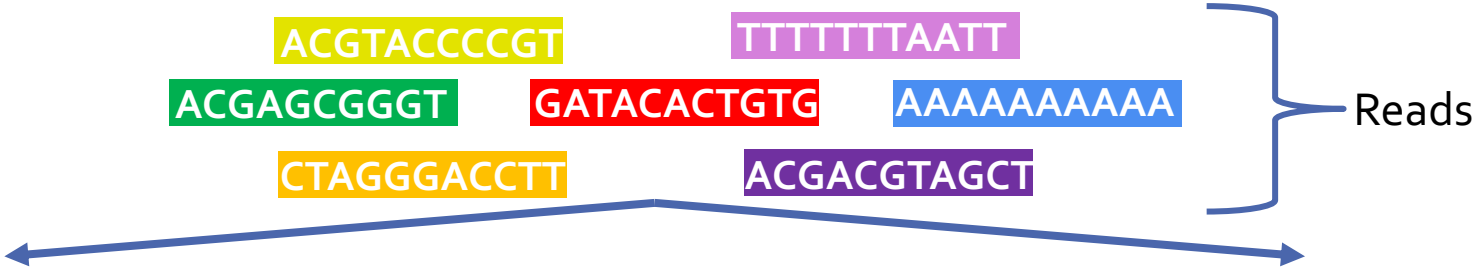
Genome Sequencing plays a pivotal role in:

- Disease discovery
- Personalized medicine
- Evolution
- Forensics

Genome Sequencing

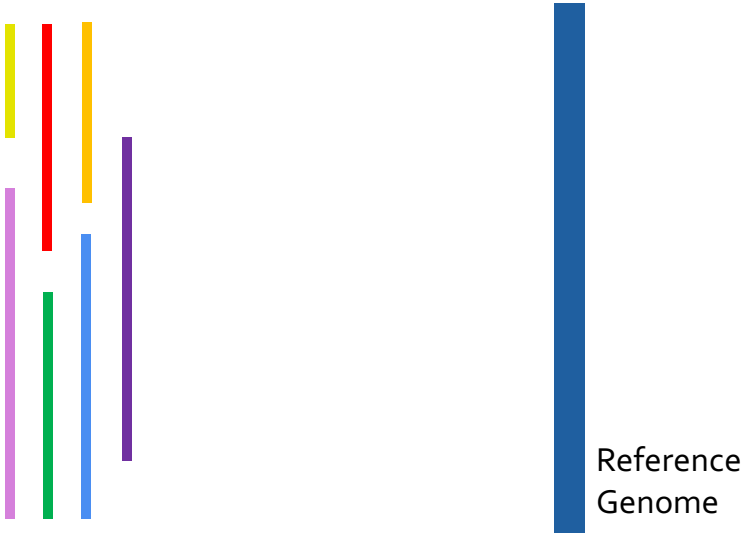


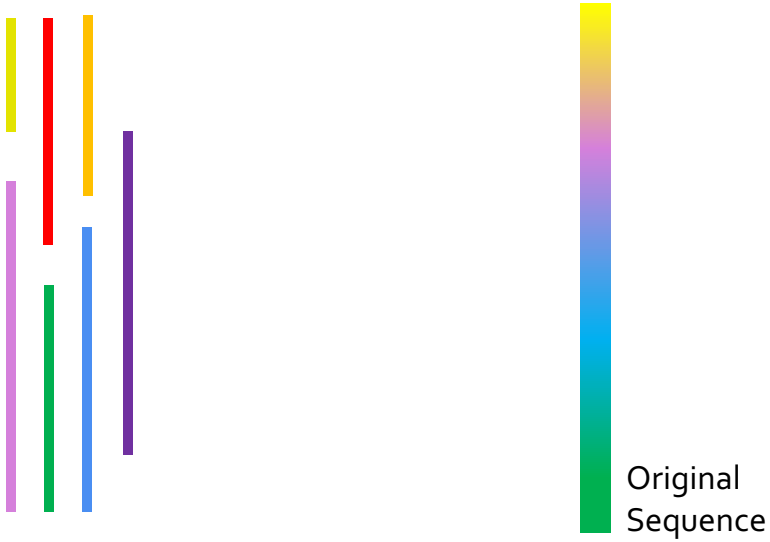
Genome Sequence Analysis



Read Mapping, method of aligning the reads against the reference genome in order to **detect matches and variations.**

De novo Assembly, method of merging the reads in order to **construct** the original sequence.





High-Throughput Sequencing

High-throughput sequencing (HTS) technology:

- Has dominated the sequencing market since 2000, and
- Generates billions of short reads in a cheap and fast way, but
- Suffers from massive amount of data and short reads, which poses challenges to read mapping and to de novo assembly due to the repetitive sequences in the genome.

Solution(s):

- Successful computational tools that can process and analyze this amount of data quickly and accurately, or
- New alternative sequencing technologies that can produce longer reads.

Advantages of Nanopore Sequencing

Nanopores are suitable for sequencing because they:

- Do *not* require any labeling of the DNA or nucleotide for detection during sequencing,
- Rely on the electronic or chemical structure of the different nucleotides for identification,
- Allow sequencing of very long reads, and
- Provide portability, low cost and high throughput.

Challenges of Nanopore Sequencing

- One major drawback: high error rates
- Nanopore sequence analysis tools have a critical role to:
  - Overcome high error rates, and
  - Take better advantage of the technology
- Faster tools are critically needed to:
  - Take better advantage of the real-time data production capability of MinION, and
  - Enable fast, real-time data analysis

Step 1: Basecalling

Raw signal data

Basecalling

DNA reads

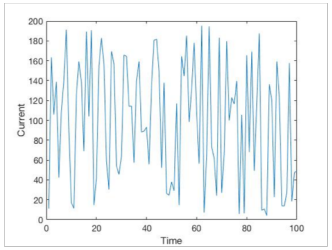
Read-to-Read Overlap Finding

Overlaps

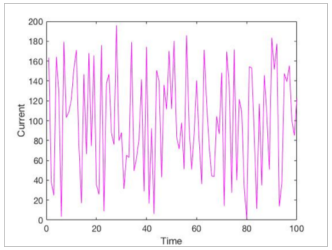
Assembly

Assembled genome

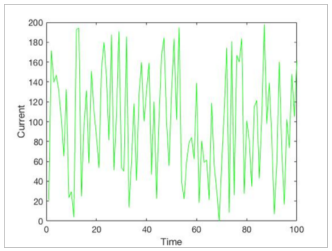
Translates the **raw signal** output into bases to generate **DNA reads**.



ACTGTCGAGTCGTAGAGA…TTT

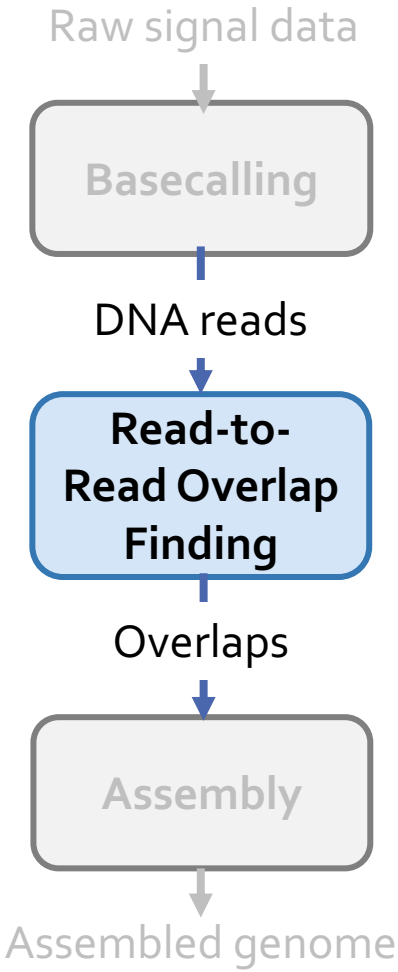


TTTGTCGAGTCGTAGAGA…TAG



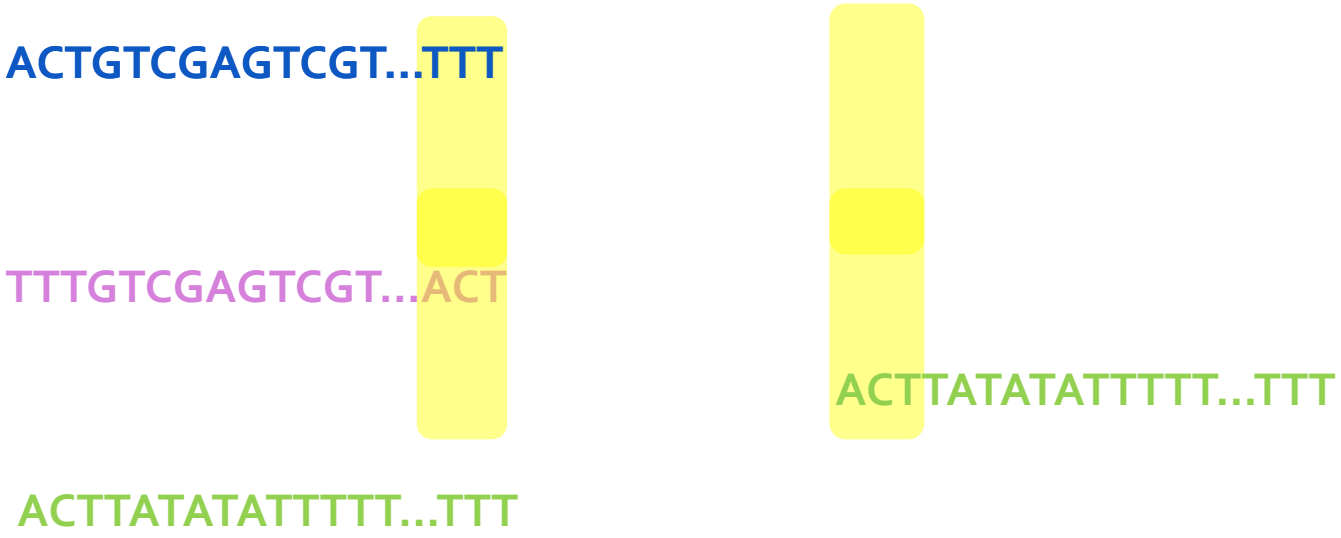
TAGTATATATTTTGGGGT…TAA

Step 2: Read-to-Read Overlap Finding

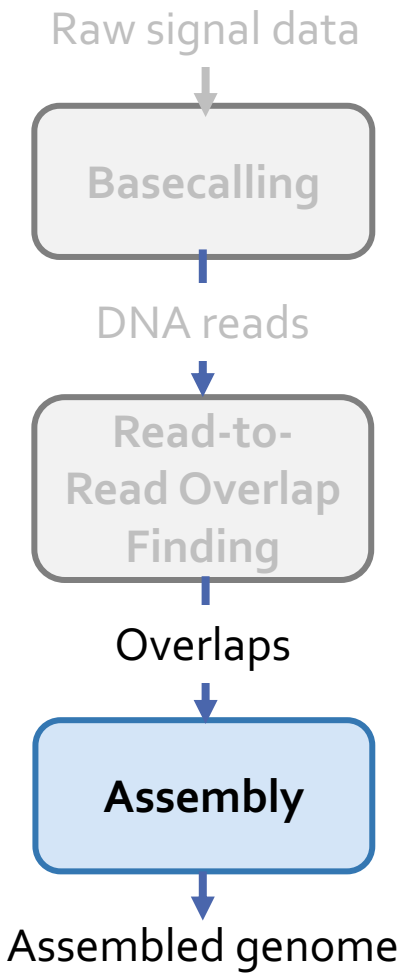


Read-to-read overlap

- is a **common sequence between two reads,** and
- occurs when the matched regions of these reads **originate from the same part of the complete genome.**



Step 3: Assembly



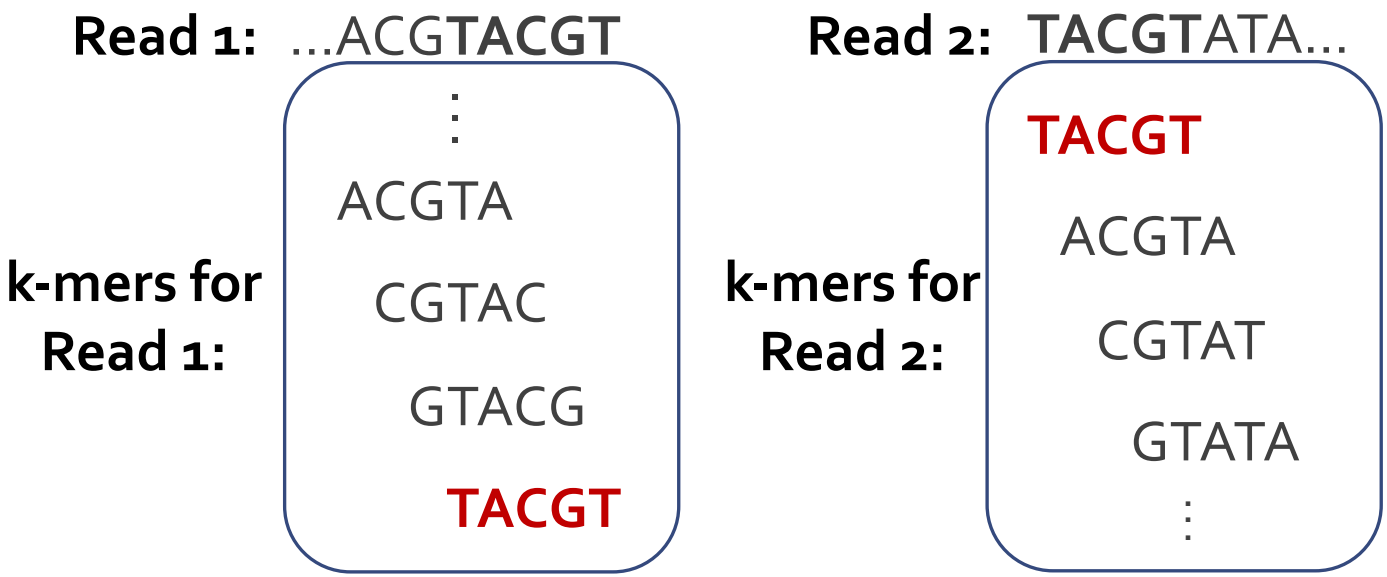
Assembly algorithms,

- generate an **overlap graph** with the overlaps from the previous step,
- **traverse** this graph, then
- **construct** the assembled genome.

Which one is correct?

GraphMap vs. Minimap

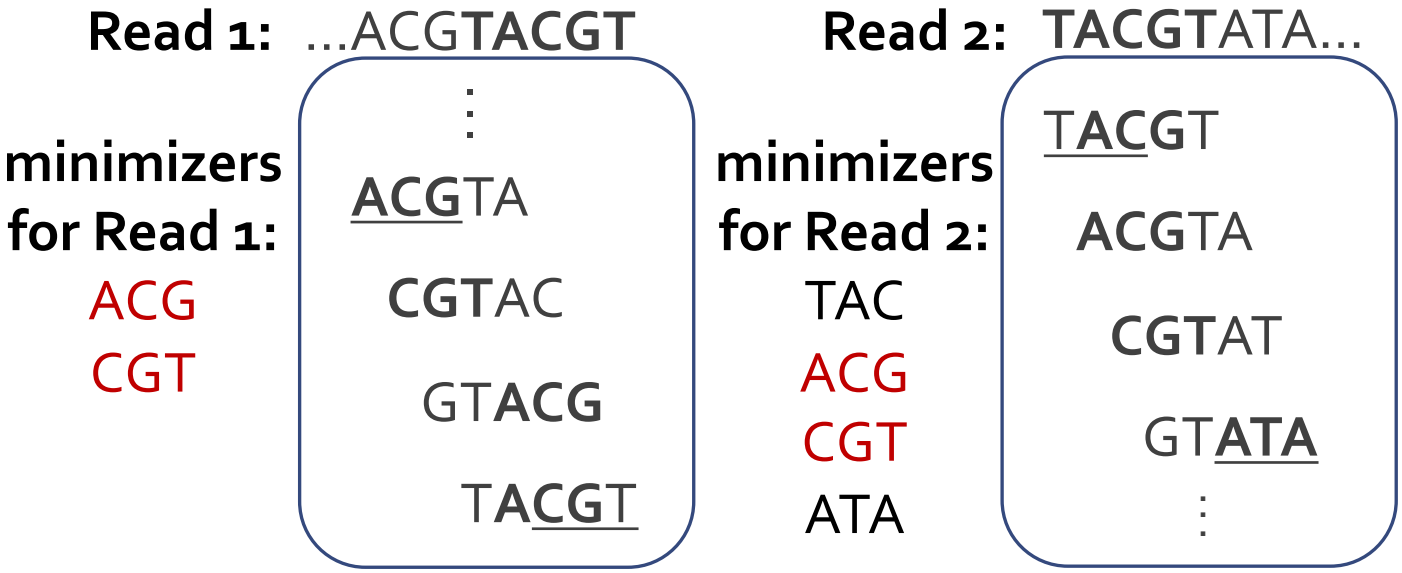
- **GraphMap**
  - Finds **k-mers** and store them in hash table with the positions.



  - Finds overlaps between two reads by **k-mer similarity**.

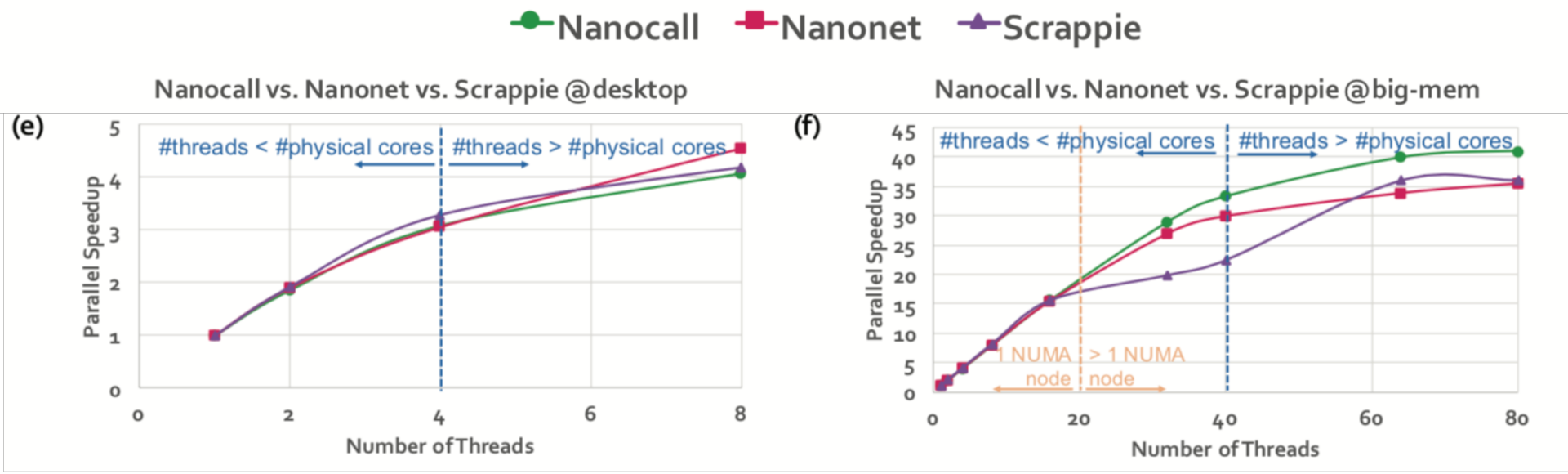
GraphMap vs. Minimap

- **Minimap**
  - Finds minimum representative set of k-mers, i.e. **minimizers** and store them in hash table, instead of storing all k-mers.



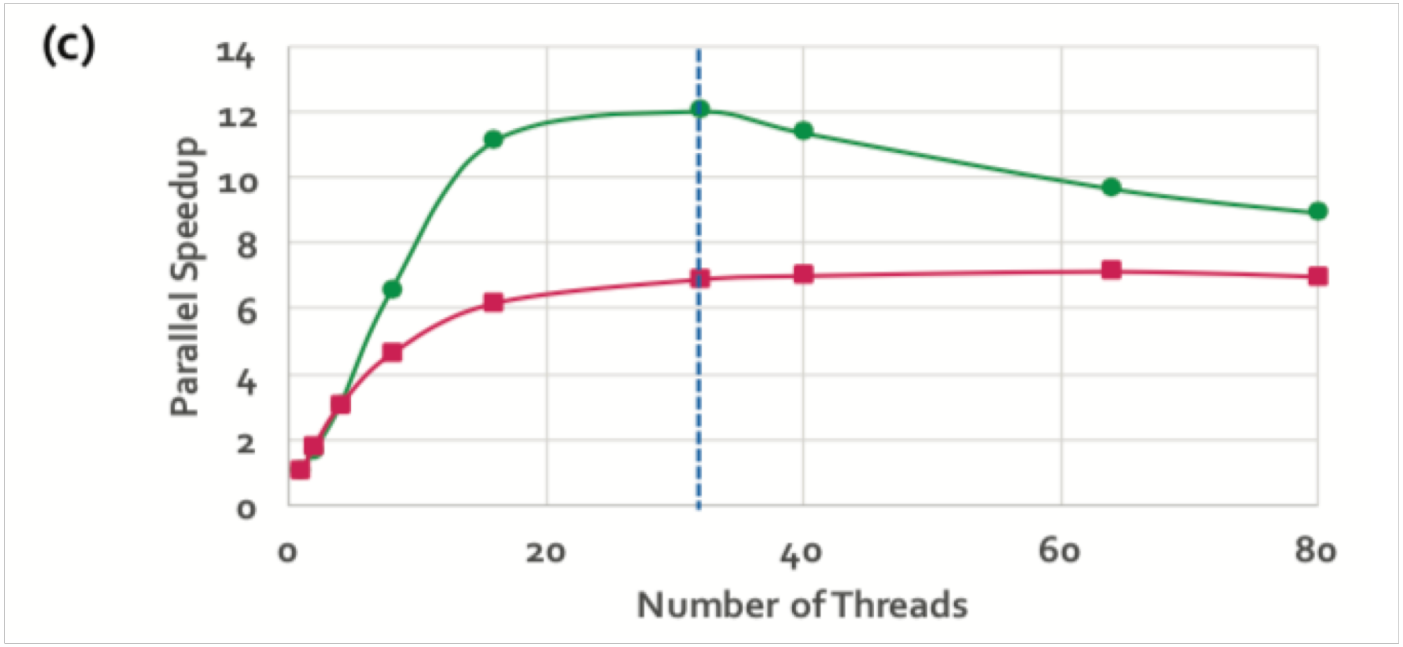
  - Finds overlaps between two reads by **minimizer similarity**.

Basecalling – Speedup



Observation 5: *When the number of threads exceeds the number of physical cores, the simultaneous multithreading overhead prevents continued linear speedup of Nanonet, Scrappie and Nanocall because of the CPU-intensive workload of these tools.*

R-to-R Overlap Finding – Speedup



Read Mapping & Polishing Tools

- **Read Mapping tools**
  - BWA-MEM
    - Commonly used long-read mapper
  - GraphMap and Minimap (from Step 2)

- **Polishing tools**
  - Nanopolish
    - HMM-based approach for polishing
  - Racon
    - Alignment graph-based approach for polishing

Read Mapping & Polishing – Accuracy

Observation 11: *Both Nanopolish and Racon significantly increase the accuracy of the draft assemblies.*

For example, Nanopolish increases the identity and coverage of the draft assembly generated with the Metrichor+Minimap+Miniasm pipeline from 87.71% and 94.85%, respectively, to 92.33% and 96.31%.
Similarly, Racon increases them to 97.70% and 99.91%, respectively.

Observation 12: *For Racon, the choice of read mapper does not affect the accuracy of the polishing step.*

Read Mapping & Polishing – Speed

Observation 13: *Nanopolish is computationally much more intensive and thus greatly slower than Racon.*

Nanopolish runs take days to complete whereas Racon runs take minutes. This is mainly because Nanopolish works on each base individually, whereas Racon works on the windows. Since each window is much longer (i.e., 20kb) than a single base, the computational workload is greatly smaller in Racon.

Observation 14: *BWA-MEM is computationally more expensive than Minimap.*

Although the choice of BWA-MEM and Minimap for the read mapping step does not affect the accuracy of the polishing step, these two tools have a significant difference in performance.