Sketch-Based Algorithmic Frameworks for Genome-Scale Mapping

Tazin Rahman (tazin.rahman@wsu.edu) | Advisor: Ananth Kalyanaraman (ananth@wsu.edu)
School of Electrical Engineering & Computer Science, Washington State University, Pullman, Washington, 99163

Motivation and Background

- Biological sequence data are expanding at a pace surpassing Moore's law, creating an urgent demand for efficient data processing.
- NCBI GenBank now holds 3+ million genomes, but ~75% are incomplete and only ~9% fully assembled (Fig 1).
- The Long-Read Sequence Read Archive (SRA) is rapidly expanding with PacBio, ONT, and HiFi data, highlighting the need for scalable processing.

The main computational bottleneck in long read-based assembly is the step of mapping the long reads against other long reads or incomplete assemblies or contigs.

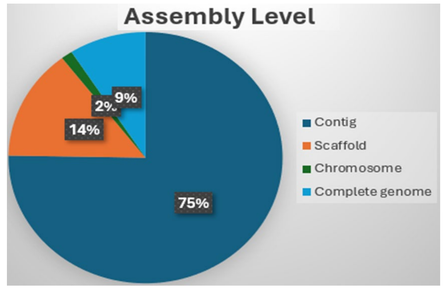

Fig. 1: NCBI GenBank assembly levels

Problem Statement

Task: Map query reads → contigs/partial assemblies (L2C) or other reads (L2L)

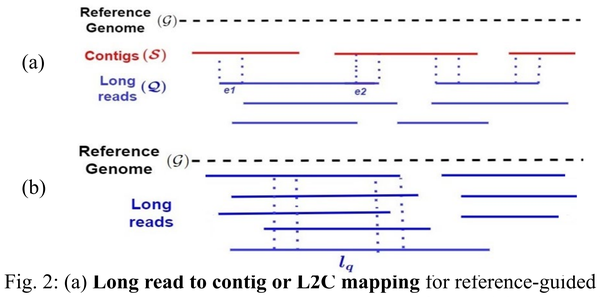

Fig. 2: (a) **Long read to contig or L2C mapping** for reference-guided and **(b) long read to long read or L2L mapping** for standalone assembly.

KEY CONTRIBUTIONS

- **Novel Sketching Algorithm:** We propose a minimizer-based Jaccard estimator (JEM) sketch-based algorithm [3] for long read mapping;
- **JEM-mapper Distributed Implementation:** We present the algorithmic workflow as JEM-mapper, capable of handling various sequence types (including both erroneous and HiFi long reads). We design and develop a distributed memory implementation of JEM-mapper [4];
- **MHsketch Library**: We introduce MHsketch, a general-purpose sketch library to support multiple sketching schemes (*e.g.* syncmers, minimizers, strobemers [2]) for long read mapping;
- **Scalable Performance**: Our method matches the mapping quality of state-of-the-art tools, achieving **96% lower memory use** and **9.3× speedup** for L2C mapping on 64 cores (MPI+OpenMP JEM-mapper vs. Minimap2).

JEM-mapper Algorithm

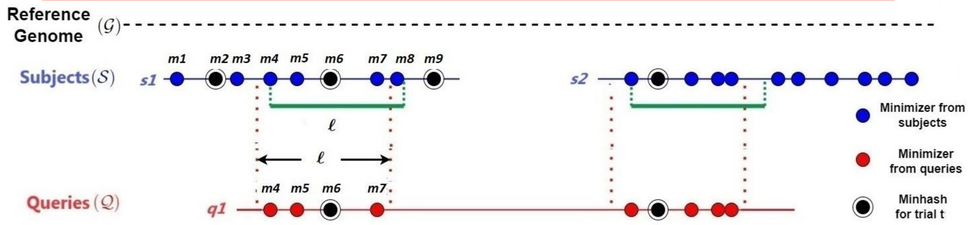

Fig 3: Example of sketch construction in JEM-mapper: it applies MinHashing on the set of minimizers generated from a sequence

Algorithmic Workflow

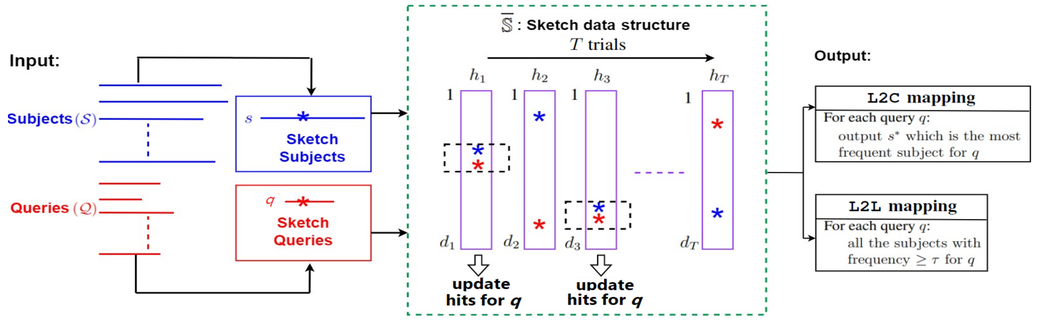

Fig. 4: Illustration of the major steps of JEM-mapper

Parallel Algorithm

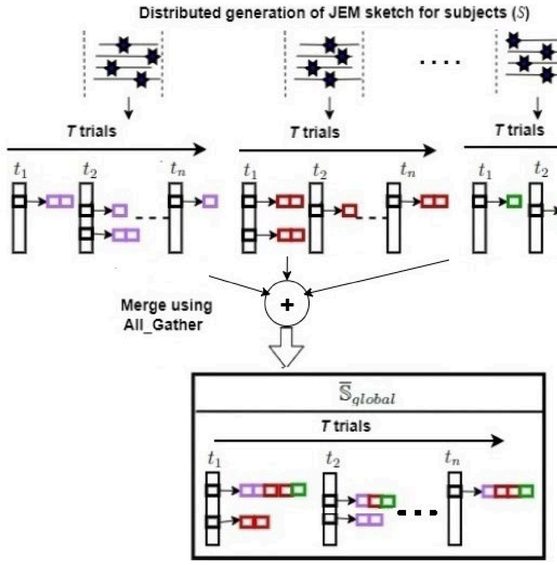

Fig. 5: Illustration of distributed global sketch data structure generation

Qualitative Evaluation

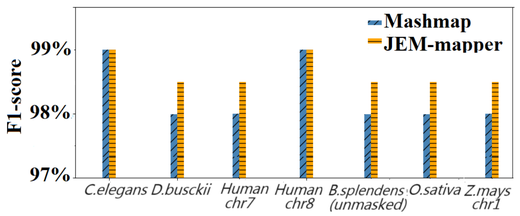

Fig. 6: **L2C:** Comparative quality evaluation (F1-score), comparing JEM-mapper and Mashmap [2]

Memory Savings

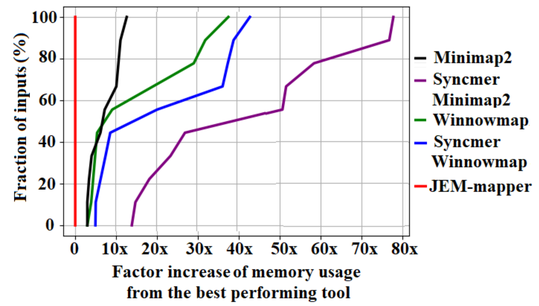

Fig. 7: Relative memory usage of different long read mapping tools compared to the best-performing tool (JEM-mapper).

Runtime Reduction

Input	JEM-mapper					Mashmap
	p=4	p=8	p=16	p=32	p=64	t=64
C. elegans	125	54	41	27	24	227
D. busckii	155	85	57	36	33	251
Human chr 7	128	75	39	27	23	260
Human chr 8	119	63	39	25	22	257
B. splendens (unmasked)	519	280	180	145	127	876
O. sativa	557	216	130	83	69	677
Z. mays chr 1	283	164	99	54	44	528
O. sativa chr 8 (real)	420	218	122	91	69	605

Table **1: L2C:** Parallel runtimes of JEM-mapper (scaling with p) vs. Mashmap (64 threads).

Input	JEM-mapper					Minimap2	MECAT
	p=4	p=8	p=16	p=32	p=64	t=64	t=64
C. elegans	485	264	154	101	79	73	498
D. busckii	589	318	218	160	102	74	595
Human chr 7	605	323	205	137	115	60	470
Human chr 8	707	295	178	119	103	71	439
B. splendens (unmasked)	2,521	1,378	760	494	388	2,886	2,931
O. sativa	1,902	1,030	579	376	302	3,008	2,184
Z. mays chr 1	1,894	945	528	350	281	4,123	2,670
O. sativa chr 8 (real)	3,171	1,659	947	608	494	7,479	3,418

Table 2**: L2L**: strong scaling of JEM-mapper vs. Minimap2 and MECAT (64 threads) [2].

Performance Evaluation on Multi-threaded Implementation

Input	(MPI-only)	JEM-mapper running in MPI + multi-threaded mode					
	p=64	p=4, t=1	p=4, t=2	p=4, t=4	p=4, t=8	p=4, t=16	p=4, t=32
C. elegans	24	125	50	28	21	14	13
D. busckii	33	168	73	53	32	20	15
Human chr 7	23	128	72	28	18	15	13
Human chr 8	22	119	53	27	22	14	13
B. splendens (unmasked)	127	519	301	178	101	83	73
O. sativa	69	557	210	120	75	32	30
Z. mays chr 1	44	283	152	89	51	31	28

Table 3 : L2C: Parallel runtimes for JEM-mapper (MPI-only vs. MPI+Threads)

Runtime Breakdown

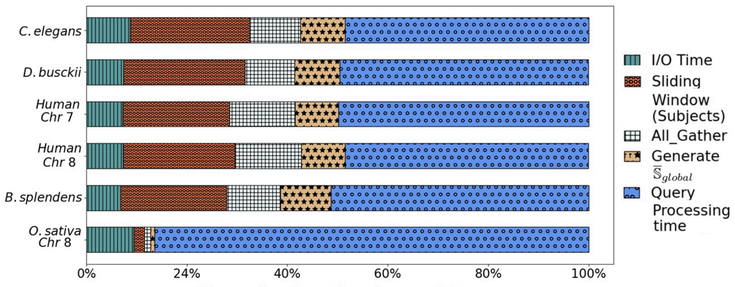

Fig. 8**: L2C:** Runtime Breakdown for steps of JEM-mapper

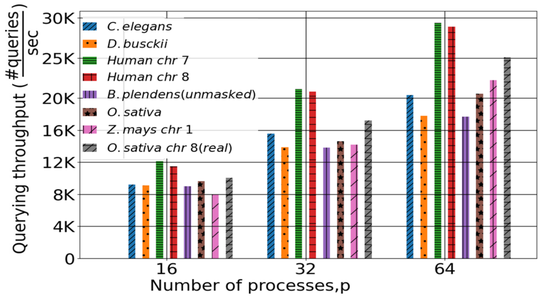

Fig. 9**: L2C:** Querying throughput for JEM-mapper

Conclusions

- Sketching enables **genome-scale mapping**.
- Our developed tools improves parallel performance by scaling on distributed memory and yields high quality results.
- **Future work:**
  - ✓ Establish theoretical guarantees
  - ✓ Build offline indexing for downstream use

Publications: **Code Repository:**

Acknowledgement: This research was supported in by NSF grants CCF 1919122 and CCF 2316160.

References:

[1] Li, H.: Minimap and miniasm: fast mapping and de novo assembly for noisy longsequences. Bioinformatics 32(14), 2103–2110 (2016)
[2] Sahlin, K., Baudeau, T., Cazaux, B., Marchet, C.: A survey of mapping algorithmsin the long-reads era. Genome Biology, (2023)
[3] Broder, A.Z.: On the resemblance and containment of documents. In: Proceedings.Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171), pp.21–29 (1997). IEEE
[4] Rahman, T., Bhowmik, O., Kalyanaraman, A.: An efficient parallel sketch-based algorithmic workflow for mapping long reads. IEEE/ACM Transactions on Computational Biology and Bioinformatics (2024)