SHARD-DB: Scalable Hash-Based Retrieval for High-Throughput Reference Database Construction

Stephanie Murray
Pacific Biosciences Research Center
University of Hawai'i at Mānoa
Honolulu, HI, USA
stephanie.murray@hawaii.edu

Lenore Pipes
Pacific Biosciences Research Center
University of Hawai'i at Mānoa
Honolulu, HI, USA
lpipes@hawaii.edu

***Abstract*—Reference databases tailored to particular metabarcoding loci are essential for environmental-DNA analysis, but constructing them can require thousands of searches against multi-terabyte nucleotide collections. CRUX-style workflows commonly expand primer-derived seed sequences using BLAST. SHARD-DB instead performs broad approximate retrieval with densified one-permutation MinHash and locality-sensitive hashing (LSH), followed by targeted locus localization.**

On a full NCBI NT MiFish workload of 10,292 queries, SHARD-DB completed retrieval and localization in 73:38 with 28.6 GiB peak RSS, while the CRUX-style merged BLAST comparison completed in 52:16 with approximately 407 GB reported peak RSS. SHARD-DB retrieved 1,606,861 unique candidate accessions compared with 133,387 for BLAST, approximately 12× more, providing a broader reference pool for downstream database construction. Historical matched-subset experiments showed large SHARD-DB runtime advantages for MarVer3 16S and PR2 18S. Results indicate strong workload dependence: broad retrieval can increase downstream localization work, while low resident memory, reduced data movement, and expanded candidate recovery remain central advantages for reference-database construction.

***Index Terms*—locality-sensitive hashing, MinHash, approximate sequence retrieval, high-performance computing, metabarcoding, reference database construction**

I. Motivation and Approach

Metabarcoding workflows depend on reference databases containing sequences from the marker being examined. Tools such as CRUX expand primer-derived seeds against large nucleotide collections using BLAST-based search [1], [2]. Repeating candidate discovery, alignment, scoring, and filtering across thousands of queries can become computationally expensive.

For reference-database construction, retrieval breadth is also important. The objective is not simply to identify a small set of best matches, but to recover a broad pool of plausible marker-specific references that can be validated and incorporated into a richer reference database.

SHARD-DB changes the retrieval representation rather than the biological scoring model. Reference sequences are divided into overlapping regions and represented by canonical 17-mer densified one-permutation MinHash sketches [4]. LSH posting lists [3] retrieve approximate sequence neighborhoods, and repeated collisions are aggregated as candidate evidence. Only retrieved regions are then examined in detail.

Targeted localization uses exact 17-mer hits, local chaining, and mismatch-tolerant k-mer comparisons to identify final locus boundaries. Overlapping jobs are grouped so reference intervals can be fetched and scanned once for multiple queries.

II. Systems Design

At NCBI NT scale, metadata traffic and repeated reference scanning are major costs. SHARD-DB separates index components for independent access, memory maps immutable posting payloads, and loads only metadata required by active queries. Candidate evidence is stored in compact numeric structures with bounded queues limiting intermediate growth.

In a controlled six-part MiFish experiment, query-directed posting-metadata access reduced metadata traffic from 251 GB to 5.84 GB while preserving byte-identical biological output, reducing matched wall time by approximately 19%.

III. Evaluation

Experiments were performed on a Dell PowerEdge R760 with an Intel Xeon Gold 6548Y+ processor and 1 TB RAM. The primary benchmark used 10,292 deduplicated MiFish 12S queries against all 14 index partitions representing the complete NCBI NT benchmark collection. SHARD-DB timing includes retrieval, localization, and final merging against a prebuilt index. The BLAST comparator represents the CRUX-style expansion/search stages rather than the complete CRUX workflow.

A. Full-Scale MiFish

For MiFish, SHARD-DB completed in 73 min 38 s, while merged BLAST completed in 52 min 16 s. SHARD-DB returned 1,606,861 unique candidate accessions compared with 133,387 for BLAST, approximately 12× as many candidates.

This broader retrieval is valuable in the reference-database construction context because it expands the pool of plausible marker-specific sequences available for downstream validation and inclusion. SHARD-only candidates were independently validated using semi-global Needleman–Wunsch alignment to assess sequence-level support.

Peak RSS was 28.6 GiB for SHARD-DB versus approximately 407 GB reported for BLAST. Thus, although BLAST

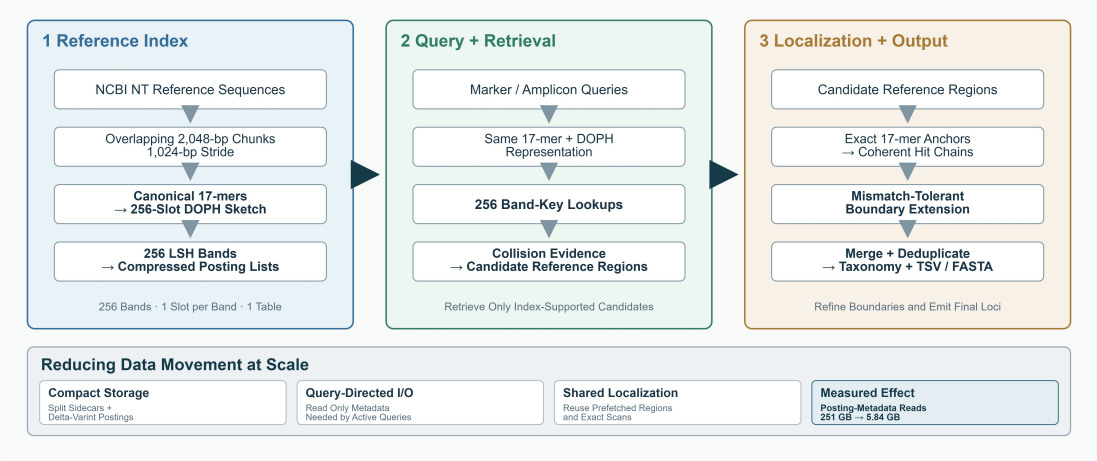


Fig. 1. SHARD-DB architecture. Reference chunks are represented with canonical 17-mer DOPH sketches and indexed using LSH posting lists. Queries retrieve candidate regions before targeted locus localization. The implementation reduces repeated data movement through query-directed metadata access, compact postings, and shared localization.

was faster in raw MiFish wall time, SHARD-DB used substantially less resident memory while retrieving a much broader candidate reference set. The MiFish result therefore reflects a tradeoff between retrieval breadth, localization work, wall time, and memory use rather than a like-for-like search returning equivalent candidate sets.

B. Cross-Locus Runtime

Historical matched 1/14-NT-subset experiments used 10,000 queries. MarVer3 16S completed in 8 min 06 s with SHARD-DB versus 2 h 09 min for the two combined BLAST searches, an observed 15.9× wall-time ratio. PR2 18S completed in 4 h 35 min with SHARD-DB versus 131 h 55 min for combined BLAST, an observed 28.8× ratio.

IV. Validation and Outlook

SHARD-DB was designed to support scalable marker-specific reference database construction by replacing repeated BLAST-based expansion with indexed approximate retrieval followed by targeted locus localization. In a separate high-recall evaluation, SHARD-DB recovered 99.95% of accessions retrieved by the CRUX-style BLAST expansion stage. SHARD-only candidates were independently validated using semi-global Needleman–Wunsch alignment, providing sequence-level support for the additional retrieved references.

Broad retrieval is an intentional feature of this design. For reference-database construction, a larger candidate pool can provide a richer set of marker-specific sequences for downstream validation and database assembly. The MiFish experiment demonstrates both sides of this tradeoff: SHARD-DB retrieved approximately 12× more unique candidate accessions than BLAST, but the resulting candidate neighborhoods also increased localization work.

Performance therefore varies across workloads. Historical MarVer3 16S and PR2 experiments showed substantial runtime reductions, while MiFish produced much broader candidate neighborhoods and shifted more work into downstream localization. Across these workloads, the indexed architecture substantially reduced resident memory and data movement.

Future work will focus on adapting sketch, LSH, and candidate-selection parameters to marker characteristics, with the goal of retaining broad validated retrieval while controlling candidate growth and localization cost.

References

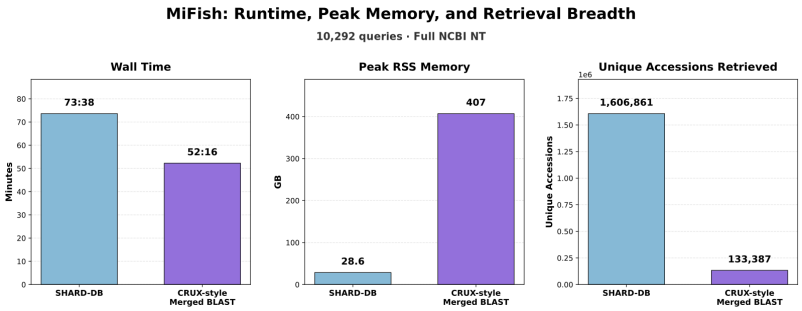
[1] E. E. Curd, Z. Gold, G. S. Kandlikar, et al., "Anacapa Toolkit: An environmental DNA toolkit for processing multilocus metabarcode datasets," *Methods in Ecology and Evolution*, vol. 10, no. 9, pp. 1469–1475, 2019, doi: 10.1111/2041-210X.13214.

[2] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

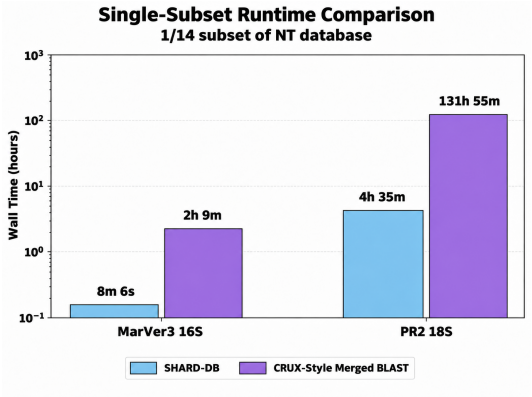
[3] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. 30th Annual ACM Symposium on Theory of Computing*, pp. 604–613, 1998, doi: 10.1145/276698.276876.

[4] A. Shrivastava and P. Li, "Densifying one permutation hashing via rotation for fast near neighbor search," in *Proc. 31st International Conference on Machine Learning*, vol. 32, no. 1, pp. 557–565, 2014.

[5] National Center for Biotechnology Information, "NCBI Nucleotide Database." [Online]. Available: https://www.ncbi.nlm.nih.gov/nucleotide



(a) Full-scale MiFish runtime, peak memory, and retrieval breadth.



(b) Historical matched 1/14-NT-subset runtime comparison.

Fig. 2. SHARD-DB performance across marker workloads. (a) For full-scale MiFish, BLAST is faster in raw wall time, while SHARD-DB uses substantially less reported peak memory and retrieves approximately 12× more unique candidate accessions, providing a broader reference pool for database construction. (b) Historical MarVer3 16S and PR2 18S experiments show substantial SHARD-DB runtime advantages on matched 1/14 NCBI NT subsets.