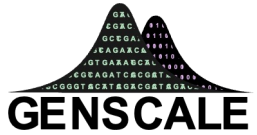
Bin Packing for Efficient Compression of Large Bacterial Genomes Collection

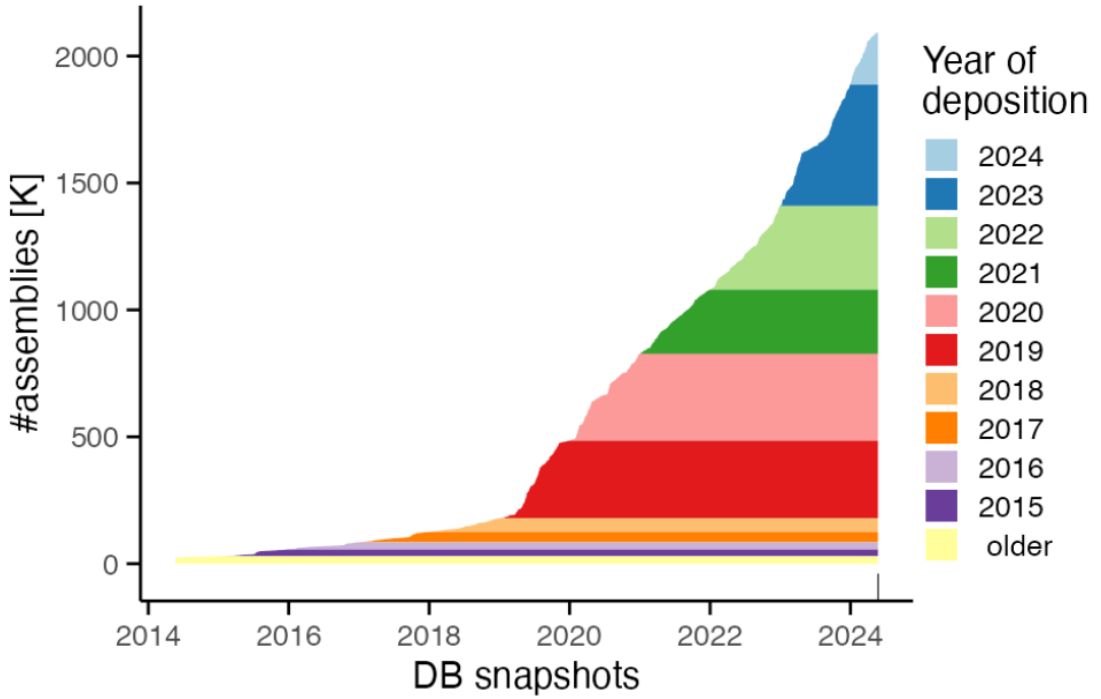
Tam TRUONG, Dominique LAVENIER, Pierre PETERLONGO, Karel BRINDA



Introduction & State Of The Art

Motivation: Rapidly Growing Bacteria Genome Data

Fast growth of bacterial genomes data[1]



Increasing Availability of Larger Bacterial Genome Collections

2021	- 661k Collection[2] ,	n = 661,405
03/2024	- AllTheBacteria[3] v0.1,	n = 1,932,812
11/2024	- AllTheBacteria[3] v0.2,	n = 2,440,377
End PhD	- Collections,	$n \geq 5 \times 10^6$

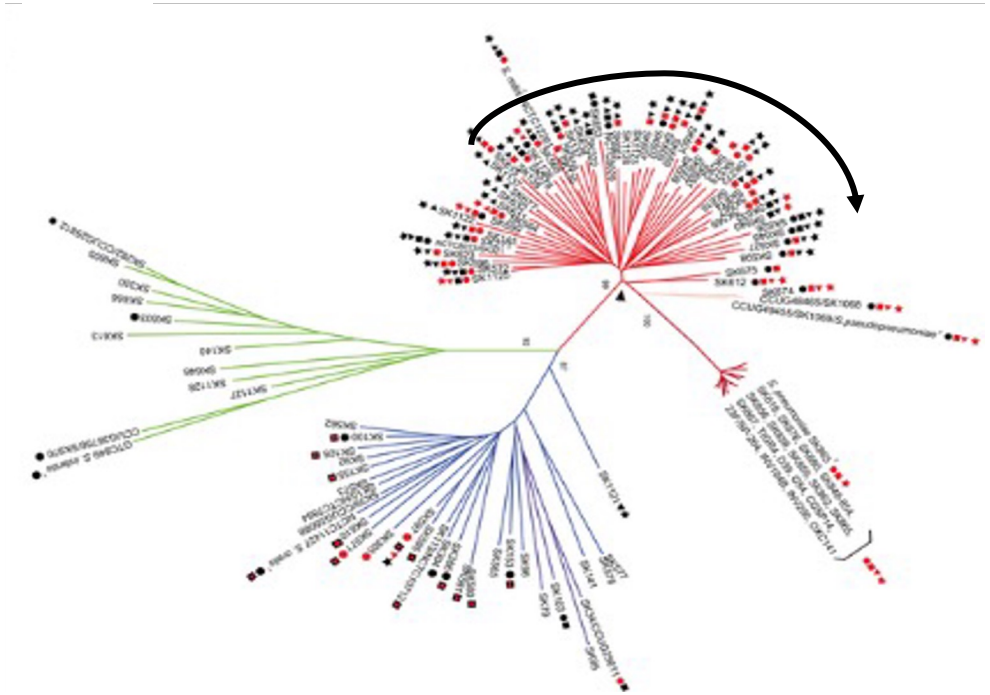
Collections will have higher diversity, metagenomes,…

➔ Enable research on bacterial adaptation, drug resistant,…

➔ Challenge: efficient compression and search within those collections

[1] Břinda et al., Efficient and Robust Search of Microbial Genomes via Phylogenetic Compression. To be appeared in *Nature Methods*. 2025
[2] Blackwell et al., Exploring bacterial diversity via a curated and searchable snapshot of archived DNA sequences. *PLOS Biology* 19, 11. 2021
[3] Hunt et a.l,. AllTheBacteria - all bacterial genomes assembled, available and searchable. *bioRxiv*. 2024

Recent Innovation: Phylogenetic Compression

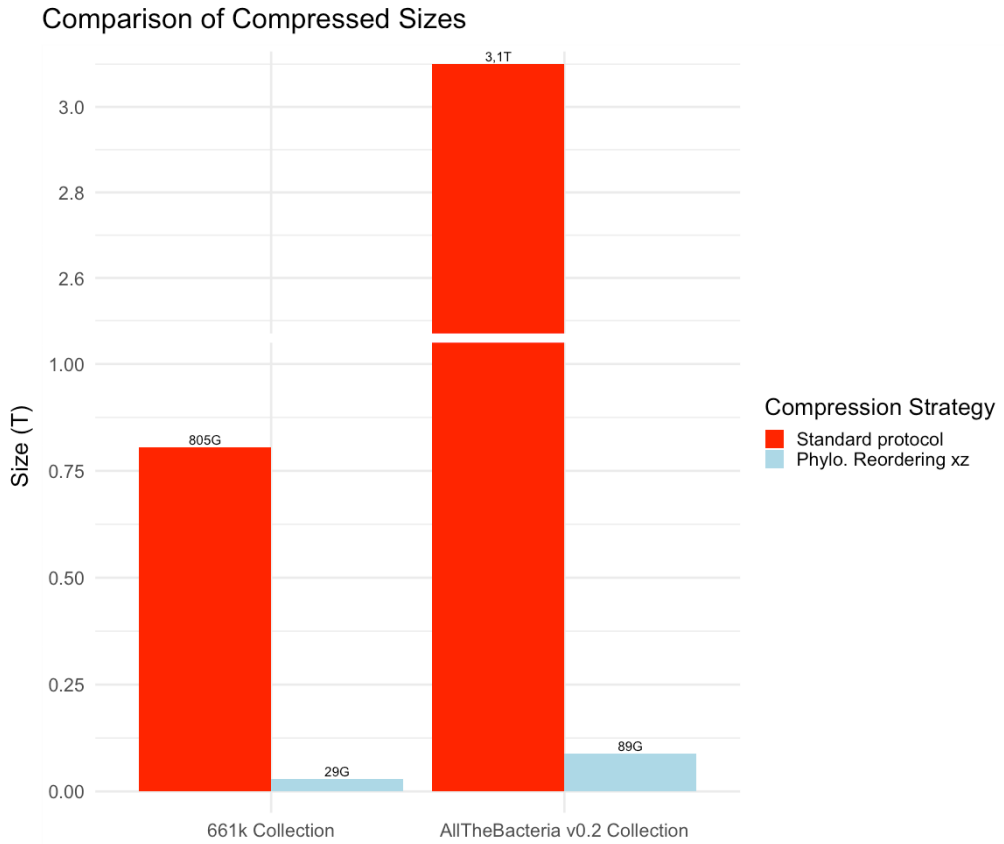


Individual genomes are not highly compressible but collections of related genomes are extremely compressible[4].

Key Idea: improves compressibility via reordering according to the evolutionary history[1]

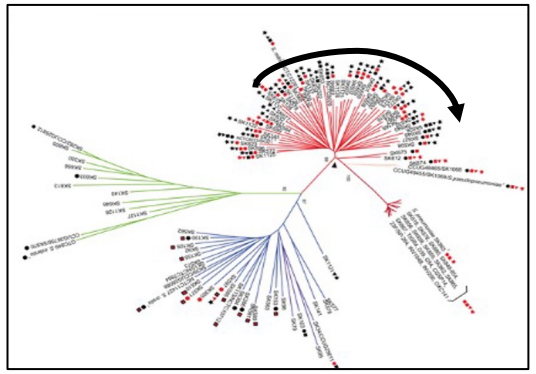
[1] Břinda et al., Efficient and Robust Search of Microbial Genomes via Phylogenetic Compression. To be appeared in *Nature Methods*. 2025
[4] Loh, PR., Baym, M. & Berger, B. Compressive genomics. *Nat Biotechnol.* 2012

Resulting compression



Lossless compression of 1-3 orders of magnitude over standard protocol

Key Step In Phylogenetic Compression: Genomes Batching



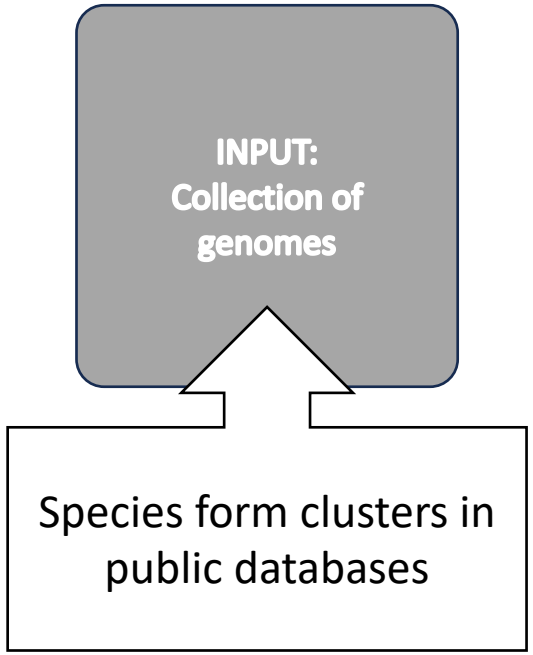
→ $\mathcal{O}(n^3)\ time$[5]

Infer a phylogenetic tree on an entire collection is infeasible

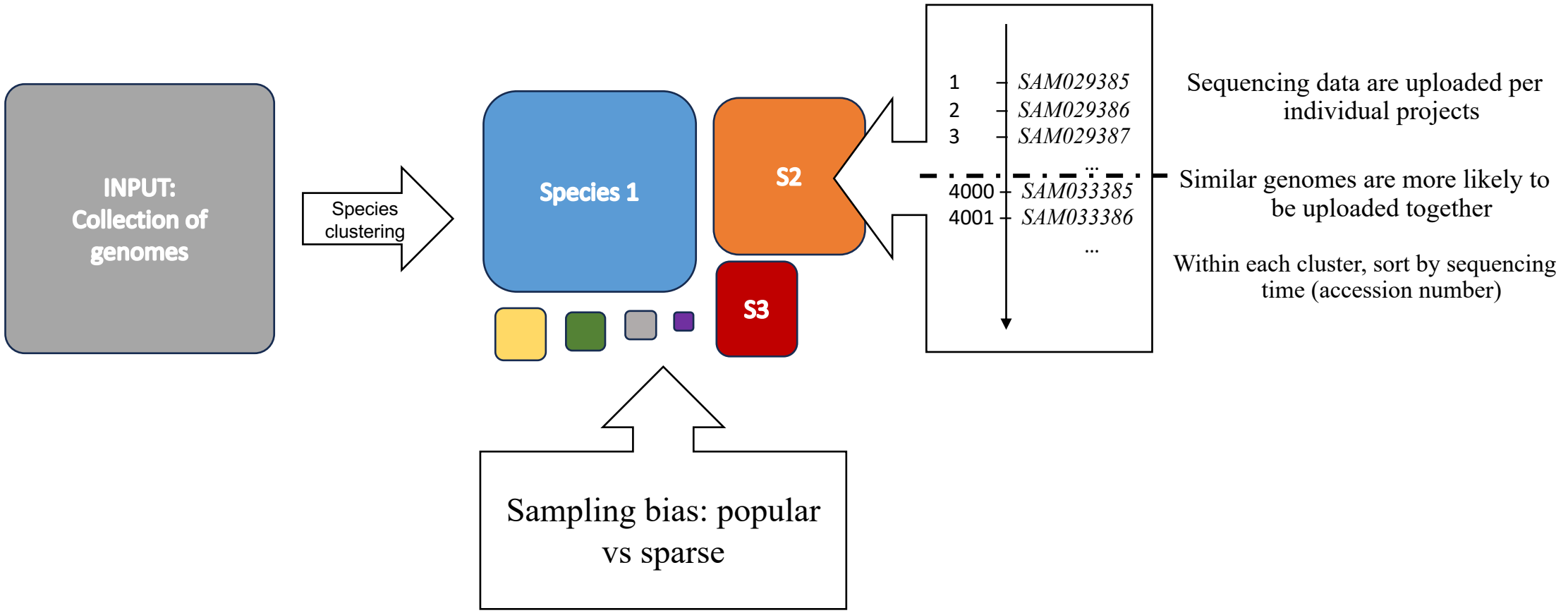
➔ Partition the collections into manageble subsets

[5] Kevin Howe, Alex Bateman, Richard Durbin, QuickTree: building huge Neighbour-Joining trees of protein sequences, *Bioinformatics*, 2002

Key Step In Phylogenetic Compression: Genomes Batching



Key Step In Phylogenetic Compression: Genomes Batching

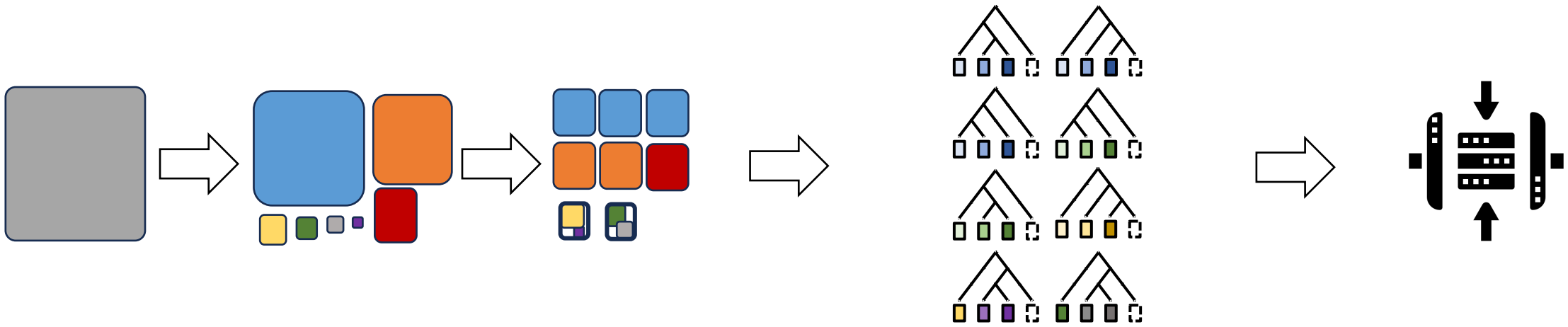


Key Step In Phylogenetic Compression: Genomes Batching

Split big clusters into smaller batches

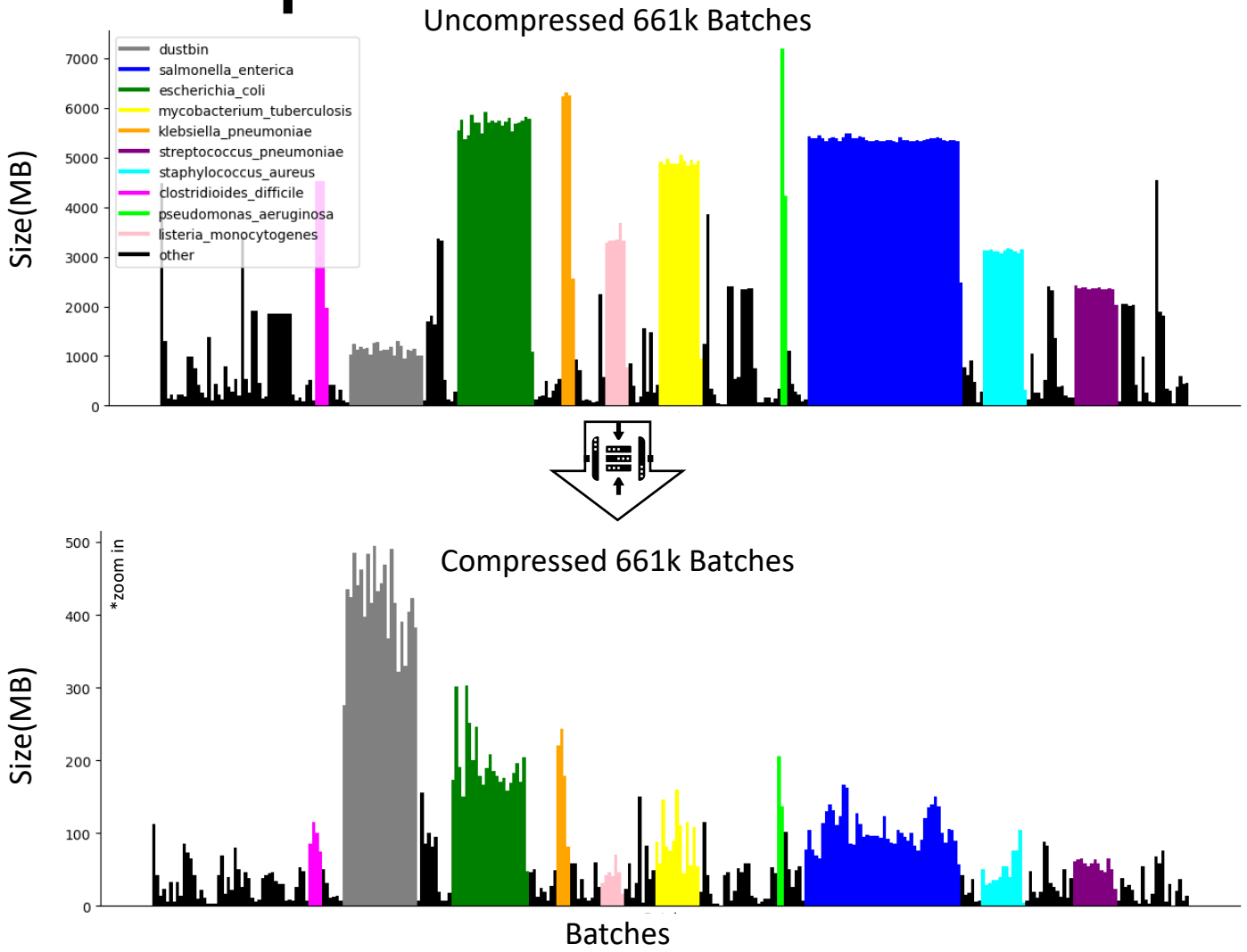
Merge small clusters together into *dustbins*

Key Step In Phylogenetic Compression: Genomes Batching



Infer evolutionary trees & reorder genomes in each batch

Current Limitation: **Non-uniform** post-compression sizes



Consequences of Non-uniformity

Unbalanced Workloads

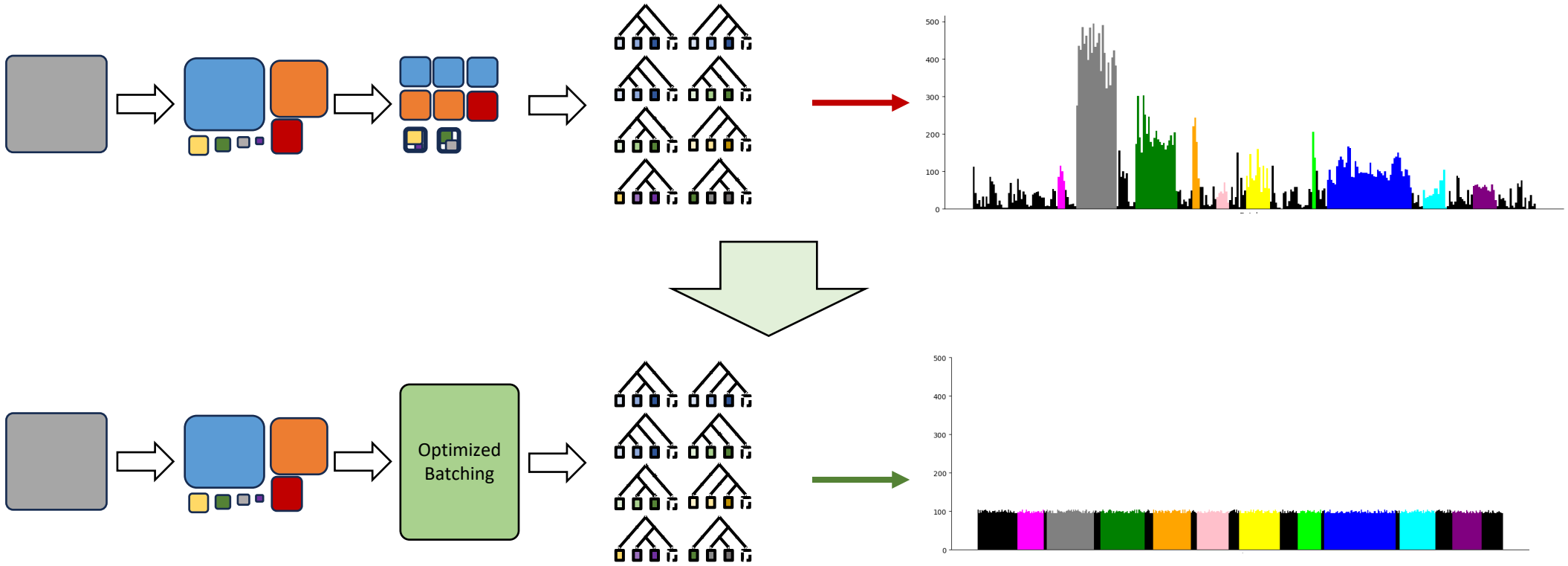
Hinder Parallelization

Inconsistent Query Time

Memory Overuse

Inefficient Transmission

Our Goal: Design A Balancing Batching Strategy



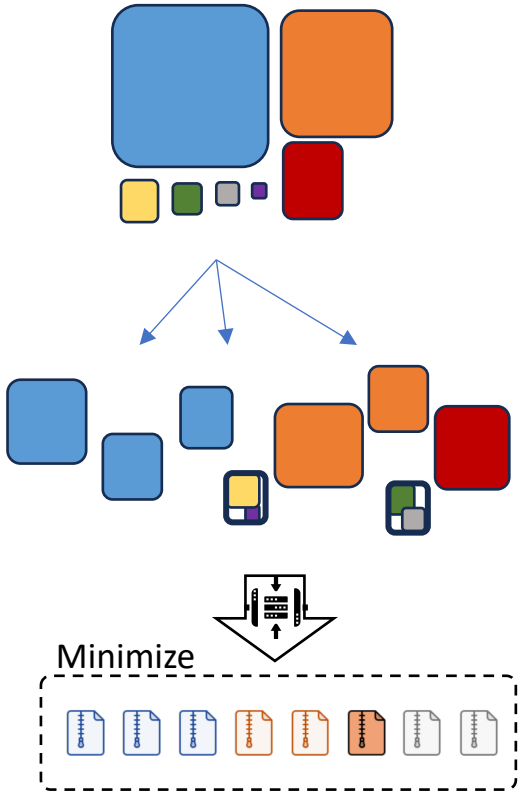
APPLICATION: Portable Devives (remote setting, field work, rapid diagnostic), Parallel Platforms (GPU, Processing-in-Memory)

Toward The First Optimization Problem

Quick Recap: What We Have So Far

- Cluster of genomes: genomes from the same species
- Put them into Batches
  - Requirements on batches: number of genomes, uncompressed size, …
  - Post-compression batch sizes must fit within a memory constraint (balance)
- Put as many genomes into the batches as possible
  - Minimize the number of batches used

➔ **Optimization Problem: Bin Packing**



Bin Packing Problem: Definition

Bin Packing Problem:
Given a list of items i = 1, . . . , n, each having a size ci ∈ Z+, and an integer value CAPACITY. Find the minimum number of bin to pack all items in such a way that the sum of the item sizes in one bin is always smaller than CAPACITY.

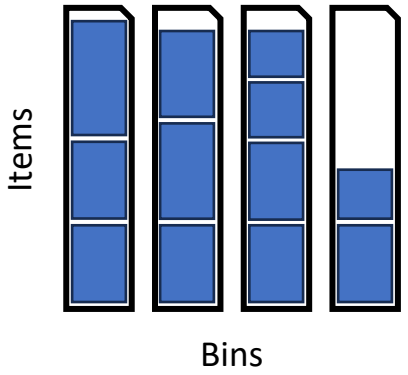
A classic combinatorial optimization problem.

The problem is NP-complete

Classical heuristics are ordered-based algorithms.

Initially, an empty bin is created. At each step, the next item is selected and packed in a bin. A new bin may be created at each step.

- First-fit: choose the first possible bin
- Best-fit: choose largest remaining CAPACITY bin
- Worst-fit: choose smallest remaining CAPACITY bin



Still a trending research topics (presentation at ROADEF 2024)

The First Optimaztion Model For Batching

- Let $G = \{g_1, g_2, \dots, g_n\}$ be the set of genomes.
- $B = \{b_1, b_2, \dots, b_m\}$ be the set of batches, where $b_j \subseteq G$. All genomes need to be assigned, one genome in one batch.

$$b_j = \begin{cases} 1 & \text{if the } j \text{ batch is used} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if genome } i \text{ is assigned to batch } j \\ 0 & \text{otherwise} \end{cases}$$

- The compression size of each batch must be less than or equal to A MB:

$$|post\_compression\_size(b_j)| \leq A, \forall j \in \{1, \dots, m\}$$

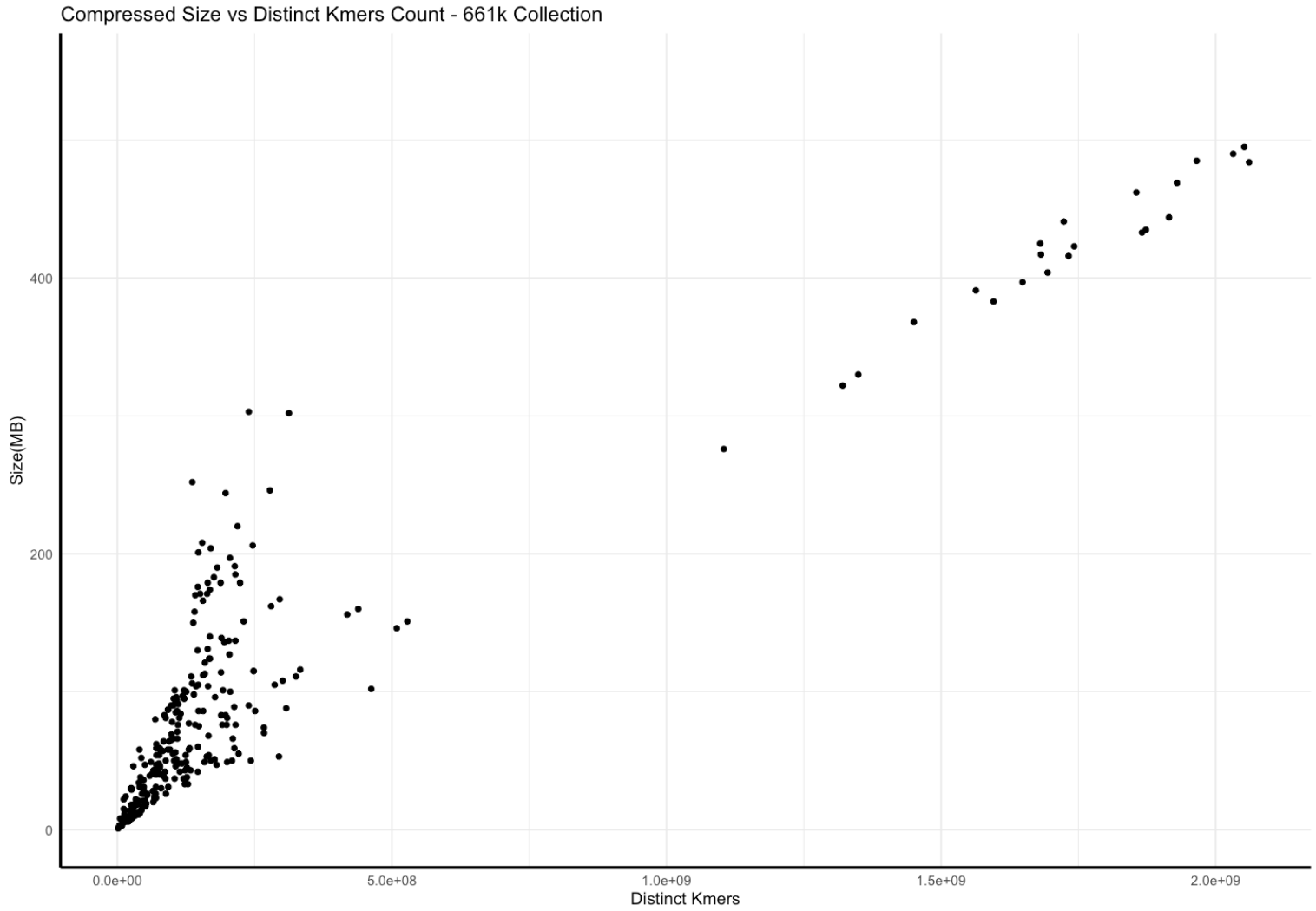
Getting the compression size is non-trivial
- ➔ xz compression speed ≈ 1 genome per sec
- ➔ 1h20m for a batch with n = 5000

Objective function: $\min \sum_{j=1}^{m} b_j$

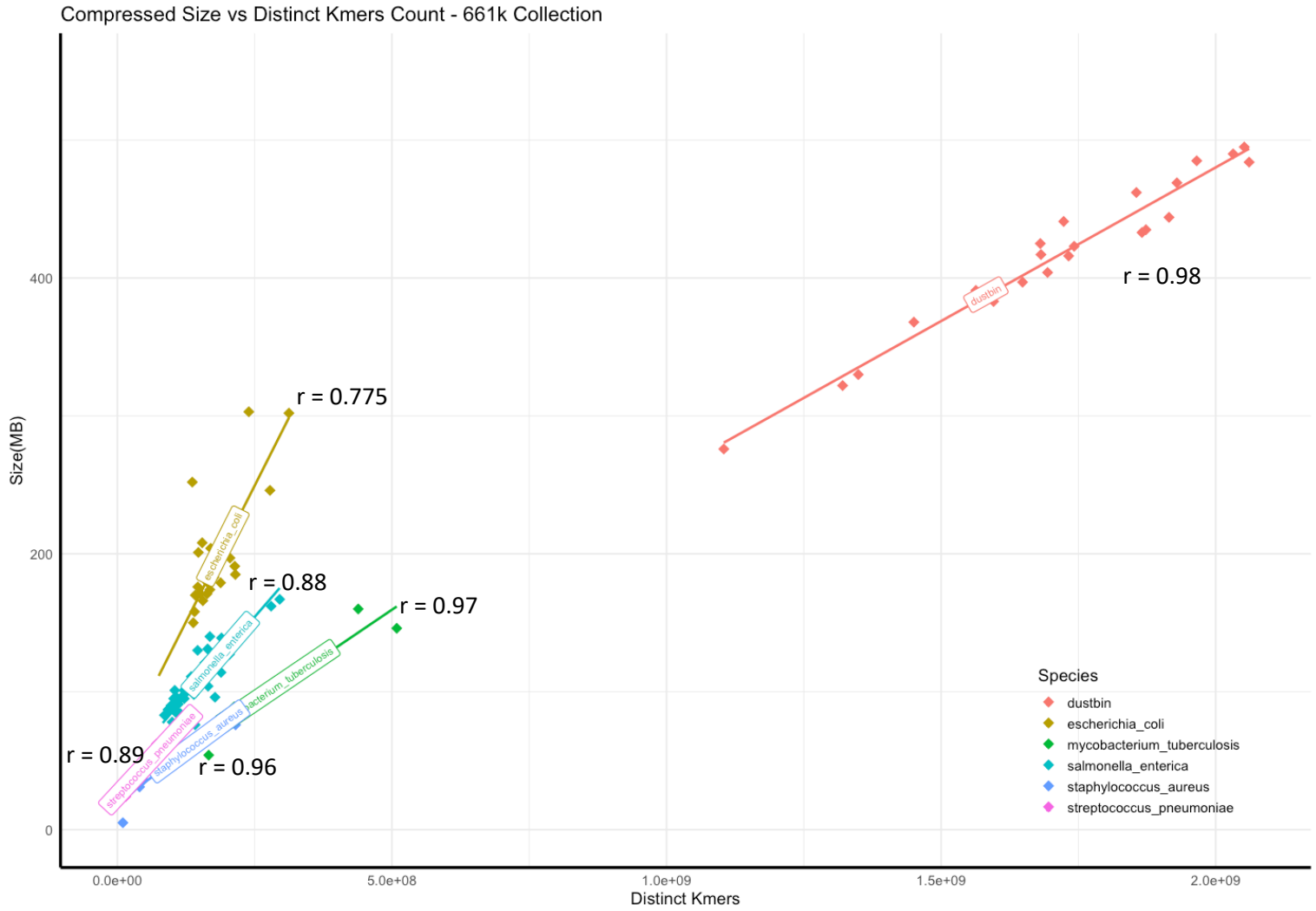
Implementation Of The Balancing Model

1. Approximation Of Post Compression Size Using Distinct Kmers Count
2. Fast Distinct Kmers Count Via Hyperloglog Sketching
3. Bin Packing Greedy Algorithms

Ingredient 1: Post-compression Sizes Vs Distinct Kmers Count In Assemblies Collections



Ingredient 1: Post-compression Sizes Vs Distinct Kmers Count In Assemblies Collections



Ingredient 2: Fast Distinct Kmers Counting via HyperLogLog

- Hash kmers into bit patterns.

- Is an algorithm for count distinct problem:
  - 0 _ _ _ _ _ : probability of getting leading 0 is 1/2
  - 0 0 _ _ _ _ : 1/4
  - 0 0 0 _ _ _ : 1/8
  - Getting k leading zeros: 1 in 2^k, i.e. to get k leading 0, we have seen $\approx 2^k$ items.

Ingredient 3: Bin Packing greedy algorithms – 2 variations

STRATEGY 1 : given unlimited batches with capacity C

$$\boldsymbol{Minimize\ nb\ of\ batch\ B}$$

s.t.

$$distinct\_kmers(bj) < C, \qquad \text{for (j} = 1, \dots, \boldsymbol{m})$$

Sort the genomes by acession number.

Initially, an empty bin is created.

At each step, the next genomes is selected and packed in the first available bin.

Create new bin as needed.

STRATEGY 2 : given a fixed number of batch n

$$T \geq distinct\_kmers(b_j), for\ j = 1, \dots, m$$

$$\boldsymbol{Minimize\ T}$$

Sort the genomes by acession number.

Create n bins.

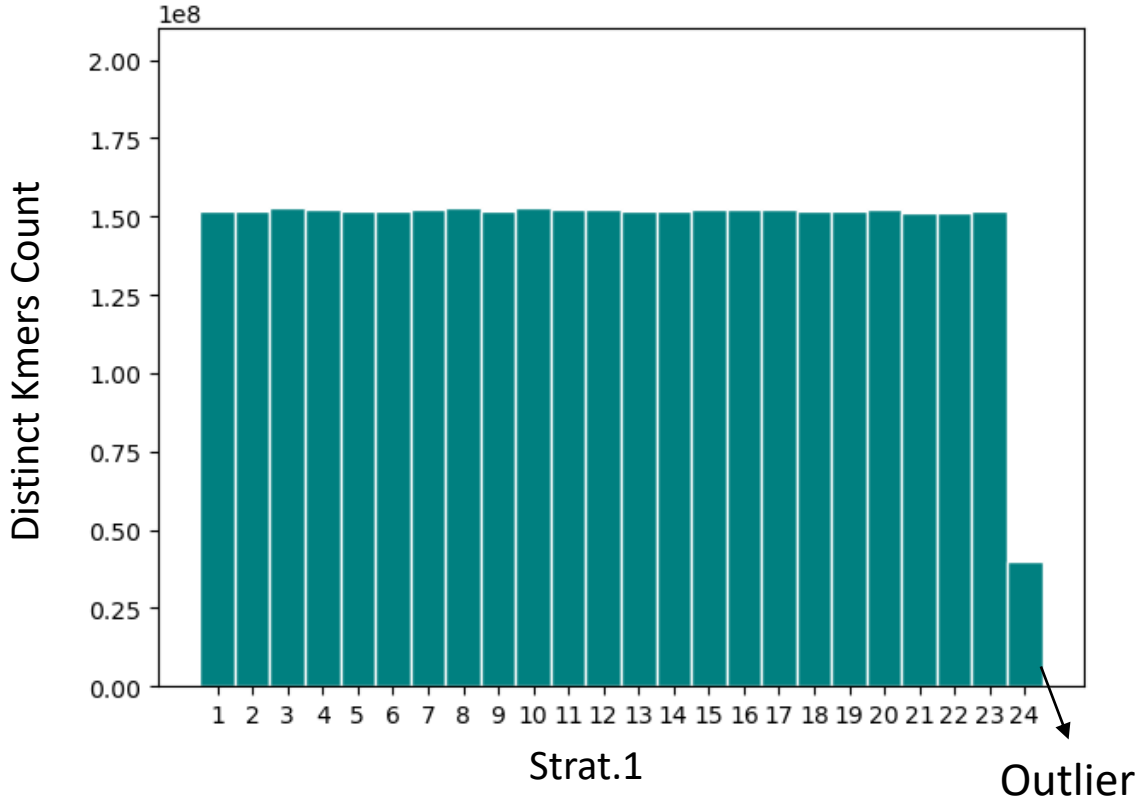
At each step, the next genomes is selected and packed in the bin with the smallest CAPACITY.

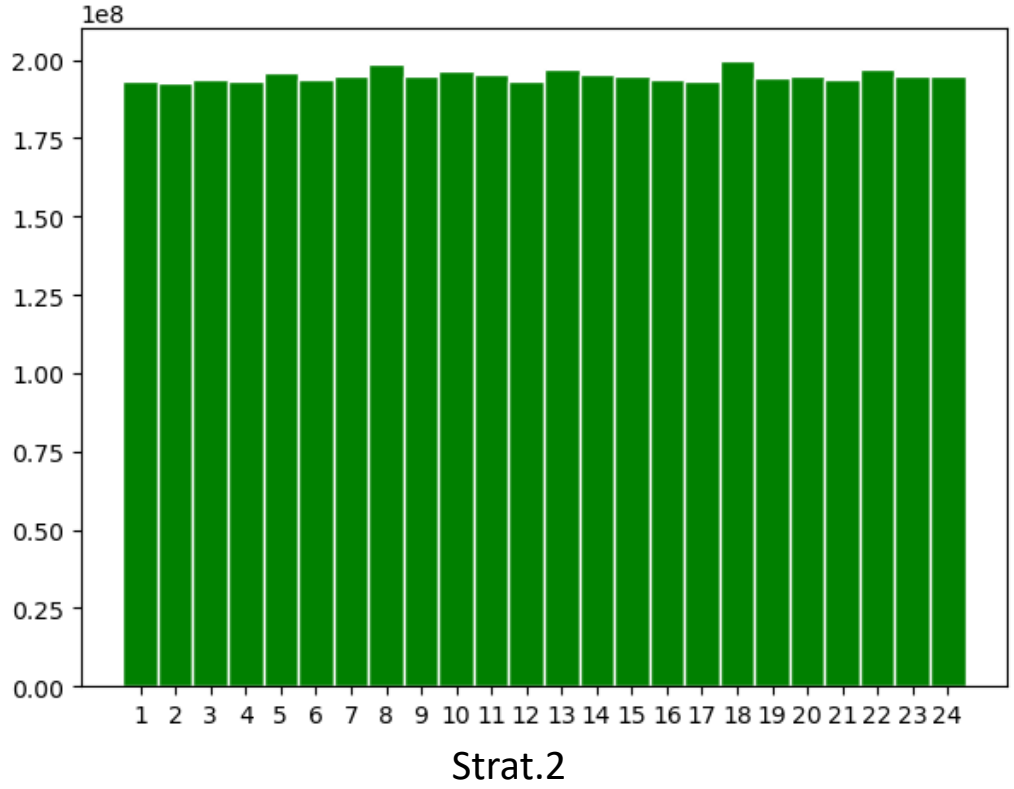
Results

Dataset:

- Asemblies of *Mycobacterium tuberculosis* from 661k
- Number of Genomes: 49000
- Uncompressed Size: 218 GB
- CAPACITY of batches: 152,000,000

Batching Results Comparison: Distinct Kmers Count



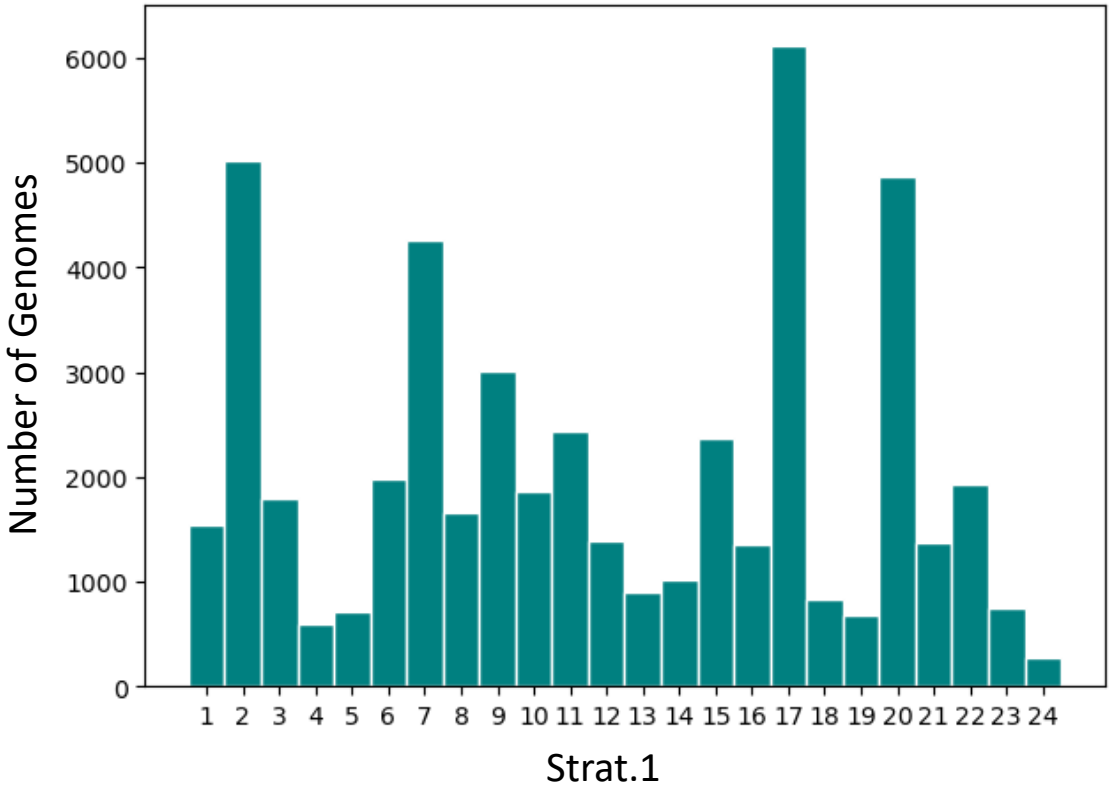


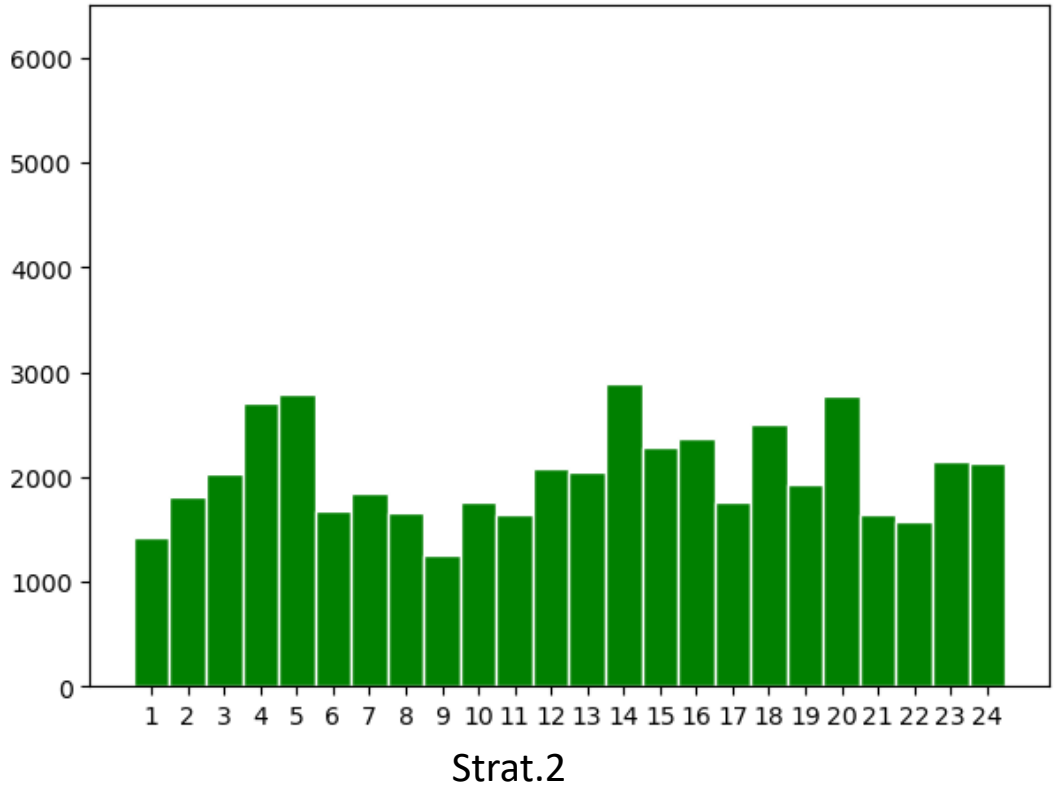
Strat.1 keep the order of accession number

➔ Better compression

➔ **Distinct Kmers Counts is balanced**

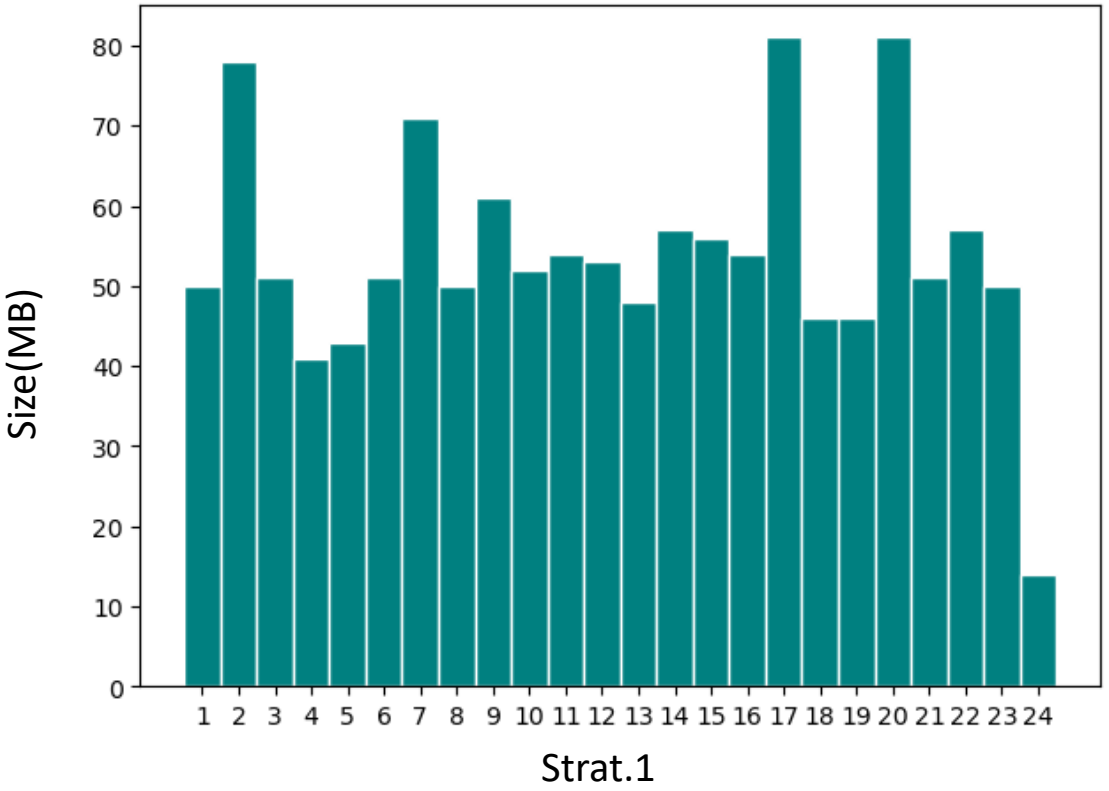
Batching Results Comparison: Number Of Genomes





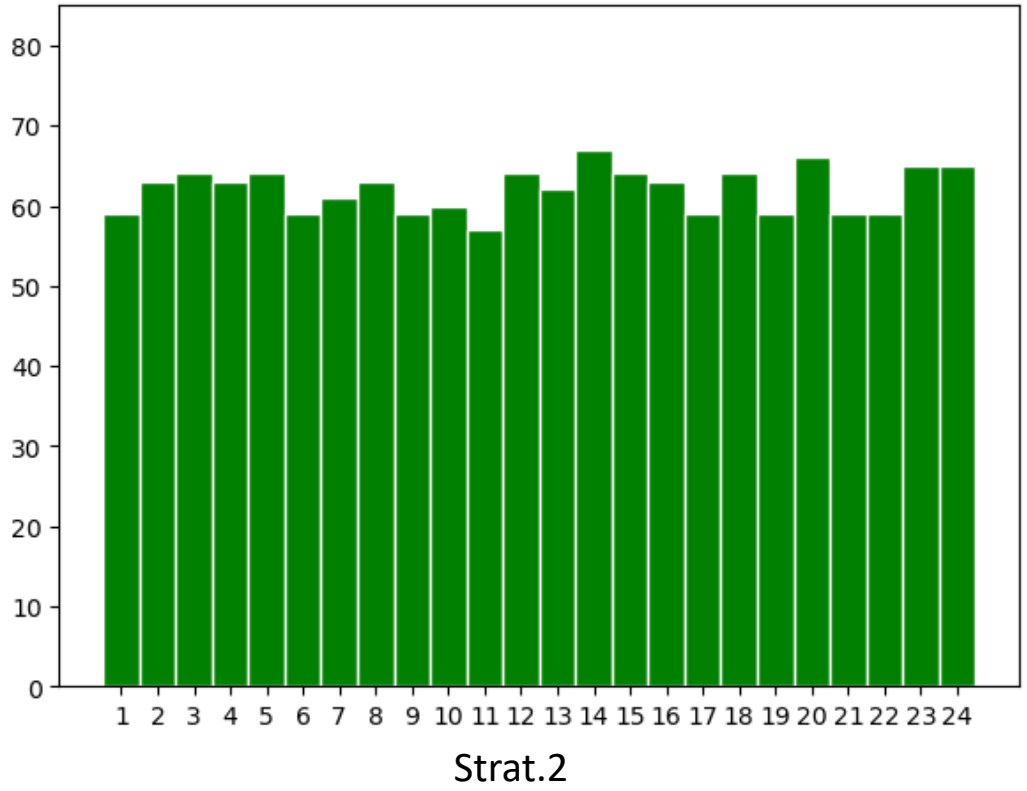
Number of genomes per batch varies, strat.2 to a lesser extent compared to strat.

Batching Results Comparison: After Compression



Post-compression size: 1,3G

Most of the batches are balanced (between 40-50MB, max size 81MB)



Post-compression size: 1,6G

All Batches are well balanced (between 59-67MB, max size 67MB)

Conclusion & Perspectives

Batching is a crucial step in Phylogenetic Compression

Batching by estimating compression size via HyperLogLog and Distinct K-mer counts improves balancing of the final compressed sizes *Mycobacterium tuberculosis.*

First results:

- First model of the Optimization Batching as a Bin Packing Problem
- Workflows of the bin packing batching strategies:
  - *https://github.com/tam-km-truong/HLL-Binning*
  - *https://github.com/tam-km-truong/HLL-Balancing*

Perspectives:

- Currently scaling up the results and methods to the 661k and the AllTheBacteria Collections
- Introducing new constraints such as Max Number of genomes per batch
- Application in other data structures such as Bloom filter, on PIM and GPU

Thank You