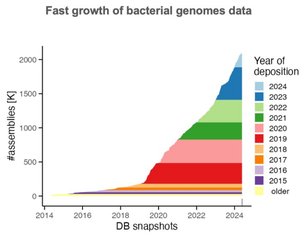
HyperLogLog-Based Load Balancing and Bin Packing for Efficient Compression and Querying of Bacterial Genomes

Motivation

Collection of bacteria genomes is growing rapidly



[Brinda et al.,bioRxiv, 2024]

Increasing Availability of Larger Bacterial Genome Collections

661k (Blackwell et al., PLOS Biology, 2021) n = 661,405

AllTheBacteria (Hunt et al., bioRxiv, 2024) n = 2,440,377

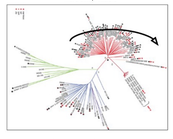
FUTURE : collection of ten of millions bacteria genomes

Phylogenetic compression allows efficient storage and query on bacteria collections

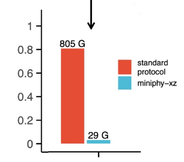
Collections of related genomes are **extremely compressible**.

Phylogenetic compression: Utilizes evolutionary relationships to guide compression and search

Genomes that are closer to each other on the phylogenetic tree are more compressible



[Brinda et al.,bioRxiv, 2024]

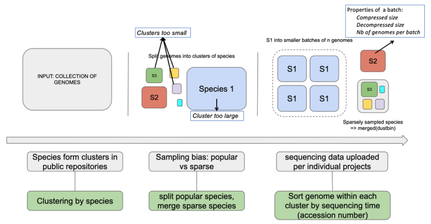


High compression ratio
Compressed using MiniPhy:
https://github.com/karel-brinda/miniphy

Phylign: tool for search of across the compressed 661 collection on a standard laptop.

https://github.com/karel-brinda/Phylign

Key idea in phylogenetic compression: clustering/batching

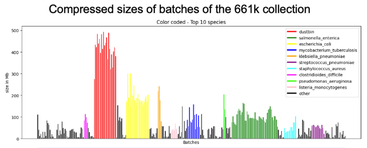


Species form clusters in public repositories → Clustering by species

Sampling bias: popular vs sparse → split popular species, merge sparse species

sequencing data uploaded per individual projects → Sort genome within each cluster by sequencing time (accession number)

Compressed sizes of batches of the 661k collection



This batching enables compression and search in a constant time (parallelization) or linear time (using a single machine)

This batching is not adapted for different hardware with different memory constraint
The compressed size of the batch is not balanced

Create balanced batches that fit within a constraint (i.e. memory) while minimizing the number of batches (maximize the number of of genomes in the batches)

Load Balancing/MultiProcessor Scheduling Optimization Problem

Instance:

Finite set I of items, a size s(i) ∈ Z + for each i ∈ I.

Given b batches.

Objective: Partition the items in set I into b batches so that we minimize the max size of the batches

- These problem are extensively studied
- NP-hard but there are efficient heuristic algorithms
- How to adapt these problem for DNA data

Bin Packing Optimization Problem

Instance:

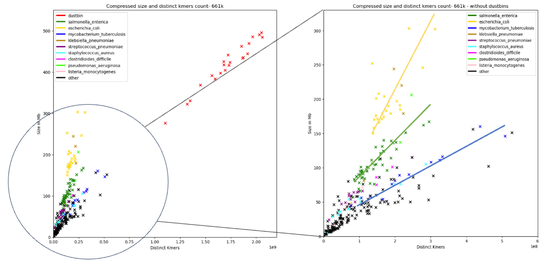
Finite set I of items, a size s(i) ∈ Z + for each i ∈ I.

Given a capacity K for each batch

Objective: Partition the items in set I into batches and minimize the number of batches used

Method

Compressed size of batches can be estimated using biological property



k-mers: substring of length k of DNA sequences

Correlation between kmer and compressed size of the batches

Different correlation for different species

➔ **Distinct kmers count can be used to set the capacity of batches**

Fast distinct kmer estimation using probabilistic counting – HyperLogLog sketching

Sketching: generating an approximate, compact summary of data (a sketch)

Probabilistic counting/approximate counting:

Sketches
Hash(item_1) = 0000... => Prob = 1 / 2^4
Hash(item_2) = 0001...
Hash(item_3) = 0010...
Hash(item_4) = 0011...
Hash(item_5) = 0100...
Hash(item_6) = 0101...
...

Longest prefix of leading zeros in the hash values is **4**
Estimated cardinality ≈ 2^4

Set cardinality estimation with Dashing:

Hyperloglog is a sketching algorithm based on advanced approximate counting.

Baker, D.N., Langmead, B. implemented it in the tool dashing.

https://github.com/dnbaker/dashing

Result

Hyperloglog based bin packing and load balancing for genomes

HLL-binning:
Put genomes into batches that fit within a user-defined memory constraint

INPUT: GENOMES SKETCHES, CAPACITY (max_distinct_kmers)
Initialize:
Sort the genomes based on their accession number.
Place Items in BATCH:
For each genome:
Try to place it in the first batch that can accommodate it (based on capacity).
If it fits, update the batch and move on to the next genome.
Create New BATCHES if Necessary:
If no batch can accommodate the current genome, create a new batch and place the genome there.

https://github.com/tam-km-truong/HLL-Binning

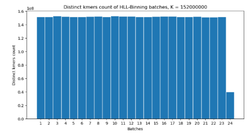
HLL-balancing:
Put genomes into balanced, user-defined number of batches

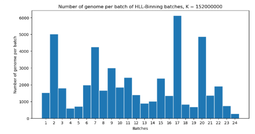
INPUT: GENOMES SKETCHES, NUMBER OF BATCH b
Initialize:
Sort the genomes based on their accession number.
Initial Assignment:
Assign the first b genomes directly to b batches.
Greedy Assignment:
For each remaining genomes, place it in the batches with the smallest number of distinct kmers.
Update the batch's contents and size.

https://github.com/tam-km-truong/HLL-Balancing

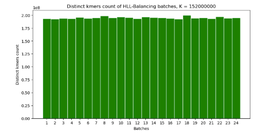
EXPERIMENT: batching of Mycobacterium Turberculosis

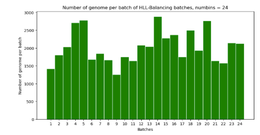
Max distinct kmers for batches: 153000000 kmers so that the compressed size stays under 64MB (caculated by using the correlation of compressed size and distinct kmers count)





Using the number off batches found by HLL-Binning, run HLL-Balancing with numbins b = 24





Eventhough the baches are balances, HLL-Balancing batches have higher distinct kmers count than the capacity set in HLL-Binning

Conclusion and perspectives

Conclusion

- Genomic data benefits greatly from data compression, especially when guided by evolutionary characteristics.
- Clustering and batching of genome collections help increase the compression ratio and facilitate parallelization.
- The compressed size of batches (using the xz compressor) is generally correlated with the distinct k-mer count.
- The distinct k-mer count can be efficiently estimated using HyperLogLog sketching (implemented in the Dasing tool).
- The Bin Packing heuristic algorithm creates batches within a user-defined capacity, i.e., distinct k-mer count.
- The Load Balancing heuristic algorithm can be used to achieve balanced batches.

Perspectives:

- Combining HLL-Binning and HLL-Balancing into one tool.
- Studying the collection as a whole instead of by species.
- Extending the scope to include different data structures (Bloom filter, graph).

Bibliography

Grace A. Blackwell, Martin Hunt, Kerri M. Malone, Leandro Lima, Gal Horesh, Blaise T. F. Alako, Nicholas R. Thomson, and Zamin Iqbal. 2021. Exploring bacterial diversity via a curated and searchable snapshot of archived DNA sequences. *PLOS Biology* 19, 11 (November 2021)

Martin Hunt, Leandro Lima, Daniel Anderson, Jane Hawkey, Wei Shen, John Lees, and Zamin Iqbal. 2024. *AllTheBacteria - all bacterial genomes assembled, available and searchable*. bioRxiv.

Karel Břinda, Leandro Lima, Simone Pignotti, Natalia Quinones-Olvera, Kamil Salikhov, Rayan Chikhi, Gregory Kucherov, Zamin Iqbal, and Michael Baym. 2024. *Efficient and Robust Search of Microbial Genomes via Phylogenetic Compression*. bioRxiv.

Po-Ru Loh, Michael Baym, and Bonnie Berger. 2012. Compressive genomics. *Nature Biotechnology* 30, 7 (July 2012)

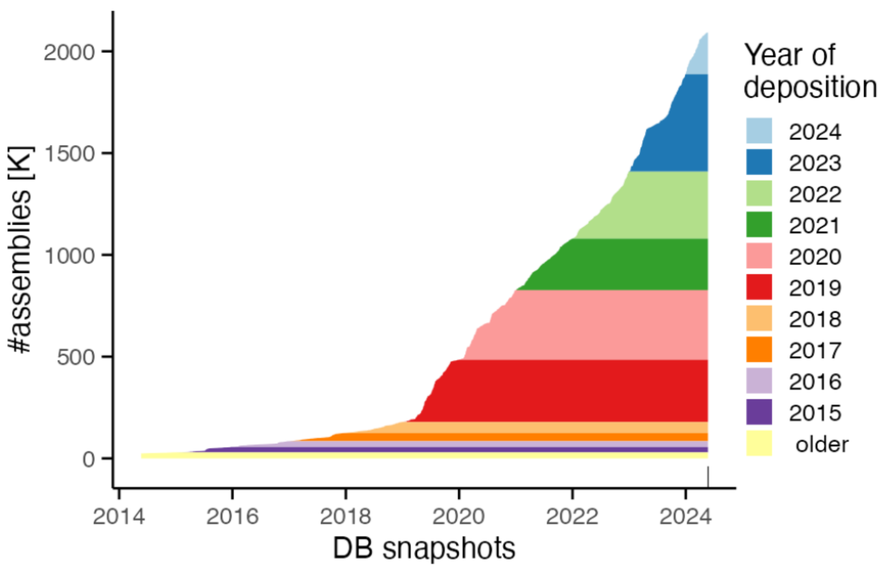
Will P. M. Rowe. 2019. When the levee breaks: a practical guide to sketching algorithms for processing the flood of genomic data. *Genome Biology* 20, 1 (September 2019), 199.

Jessica K. Bonnie, Omar Y. Ahmed, and Ben Langmead. 2024. DandD: Efficient measurement of sequence growth and similarity. *iScience* 27, 3 (March 2024).

Motivation

Collection of bacteria genomes is growing rapidly

Fast growth of bacterial genomes data



[Brinda et al.,bioRxiv, 2024]

Increasing Availability of Larger Bacterial Genome Collections

661k (Blackwell et al., PLOS Biology, 2021) n = 661,405

AllTheBacteria (Hunt et al., bioRxiv, 2024) n = 2,440,377

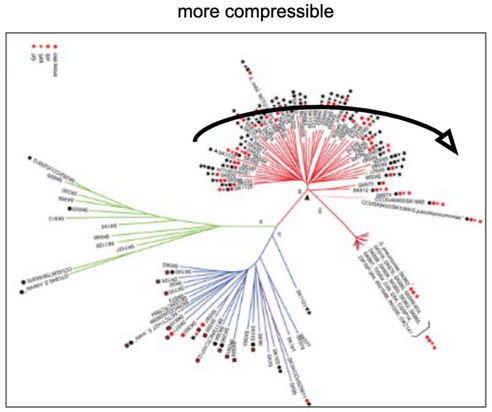
FUTURE : collection of ten of millions bacteria genomes, higer diversity

Phylogenetic compression allows efficient storage and query on bacteria collections

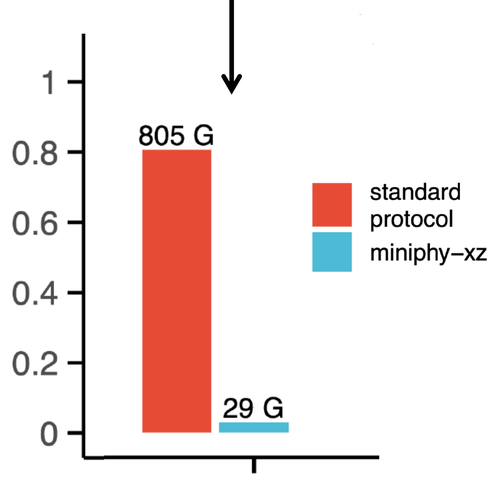
Collections of related genomes are **extremely compressible**.

Phylogenetic compression: Utilizes evolutionary relationships to guide compression and search

Genomes that are closer to each other on the phylogenetic tree are more compressible



[Brinda et al.,bioRxiv, 2024]



661k

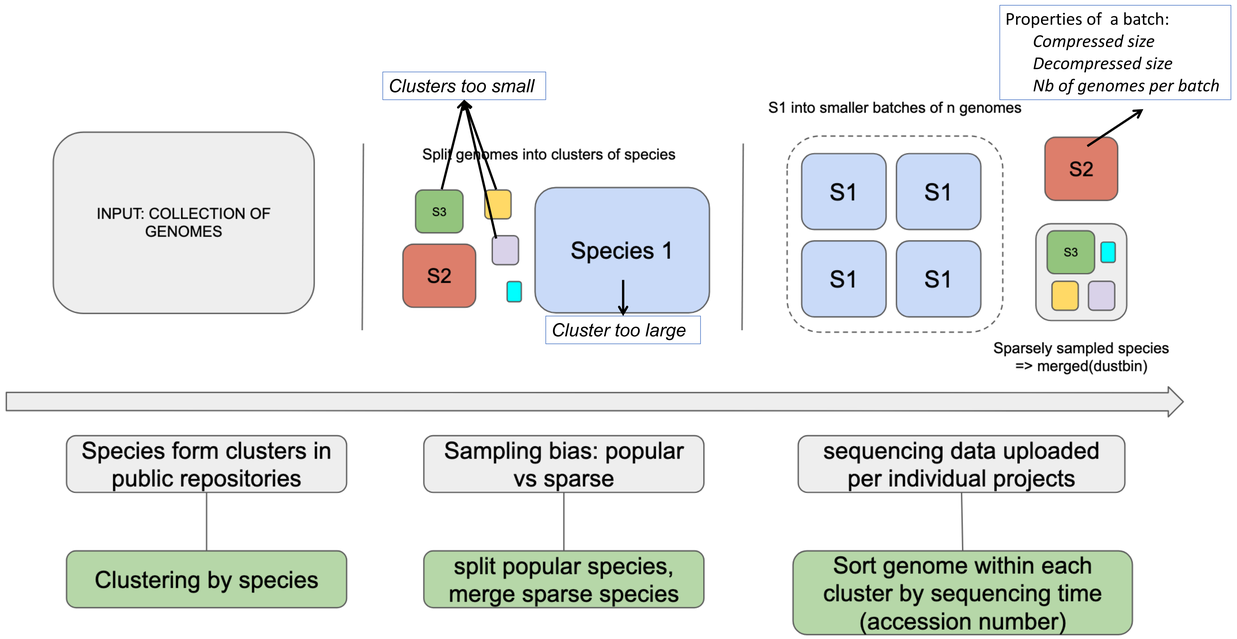
High compression ratio
Compressed using MiniPhy:
https://github.com/karel-brinda/miniphy

Phylign: tool for search of across the compressed 661 collection on a standard laptop.

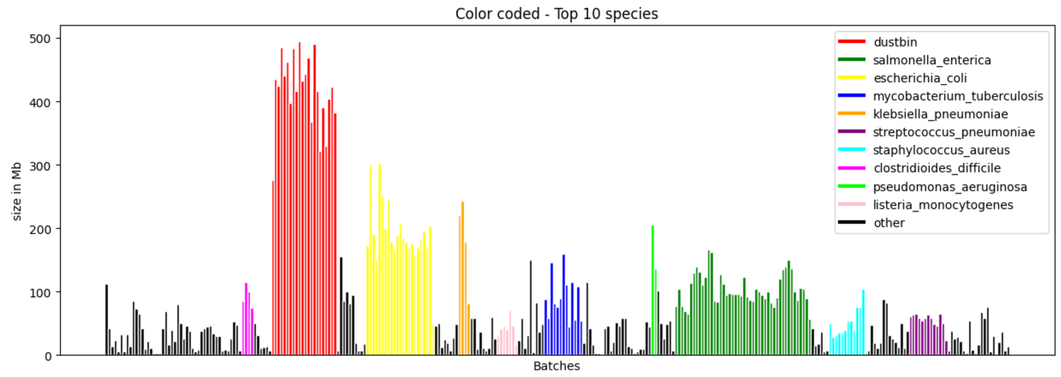
https://github.com/karel-brinda/Phylign

State of the art

Key idea in phylogenetic compression: clustering/batching



Compressed sizes of batches of the 661k collection



This batching enables compression and search in a constant time (parallelization) or linear time (using a single machine)

This batching is not adapted for different hardware with different memory constraint
The compressed size of the batch is not balanced

Create balanced batches that fit within a constraint (i.e. memory) while minimizing the number of batches (maximize the number of of genomes in the batches)

Load Balancing/MultiProcessor Scheduling Optimization Problem

Instance:

Finite set I of items, a size s(i) ∈ Z + for each i ∈ I.

Given b batches.

Objective: Partition the items in set I into b batches so that we minimize the max size of the batches

- ➔ These problem are extensively studied
- ➔ NP-hard but there are efficient heuristic algorithms
- ➔ How to adapt these problem for DNA data

Bin Packing Optimization Problem

Instance:

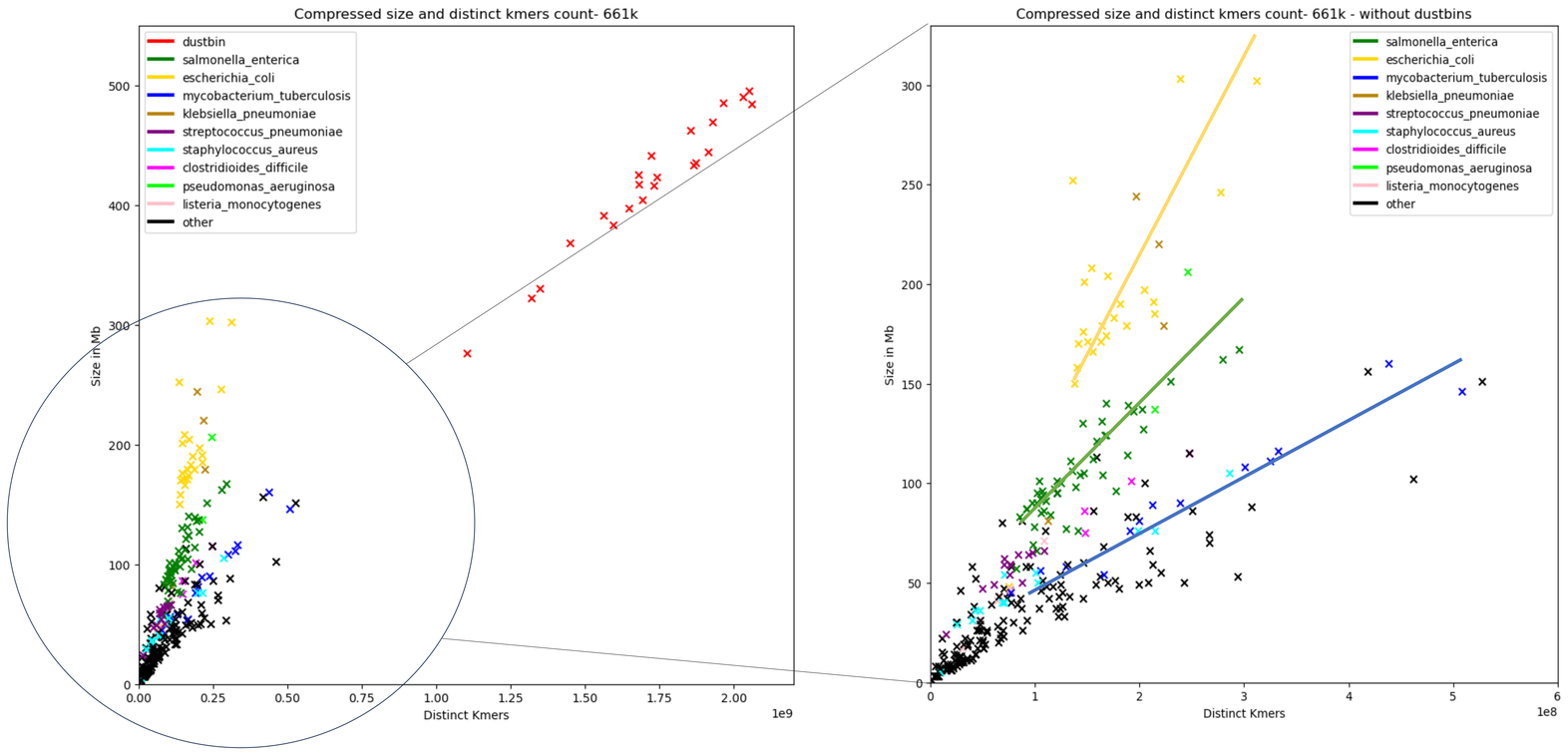
Finite set I of items, a size s(i) ∈ Z + for each i ∈ I.

Given a capacity K for each batch

Objective: Partition the items in set I into batches and minimize the number of batches used

Method

Compressed size of batches can be estimated using biological property



k-mers: substring of length k of DNA sequences

Correlation between kmer and compressed size of the batches

Different correlation for different species

➔ **Distinct kmers count can be used to set the capacity of batches**

Fast distinct kmer estimation using probabilistic counting – HyperLogLog sketching

Sketching: generating an approximate, compact summary of data (a sketch)

TCGATCGATCGATCGA

TCGAT

CGATC

GATCG

…

ATCG

TCGA

Probabilistic counting/approximate counting:

Sketches
Hash(item_1) = 0000... => Prob = 1 / 2^4
Hash(item_2) = 0001...
Hash(item_3) = 0010...
Hash(item_4) = 0011...
Hash(item_5) = 0100...
Hash(item_6) = 0101...
…

Longest prefix of leading zeros in the hash values is **4**
Estimated cardinality ≈ **2^4**

Set cardinality estimation with Dashing:

Hyperloglog is a sketching algorithm based on advanced approximate counting.

Baker, D.N., Langmead, B. implemented it in the tool dashing.

https://github.com/dnbaker/dashing

Result

Hyperloglog based bin packing and load balancing for genomes

HLL-binning:

Put genomes into batches that fit within a user-defined memory constraint

INPUT: GENOMES SKETCHES, CAPACITY (max_distinct_kmers)
Initialize:
Sort the genomes based on their accession number.
Place Items in BATCH:
For each genome:
Try to place it in the first batch that can accommodate it (based on capacity).
If it fits, update the batch and move on to the next genome.
Create New BATCHES if Necessary:
If no batch can accommodate the current genome, create a new batch and place the genome there.

https://github.com/tam-km-truong/HLL-Binning

HLL-balancing:

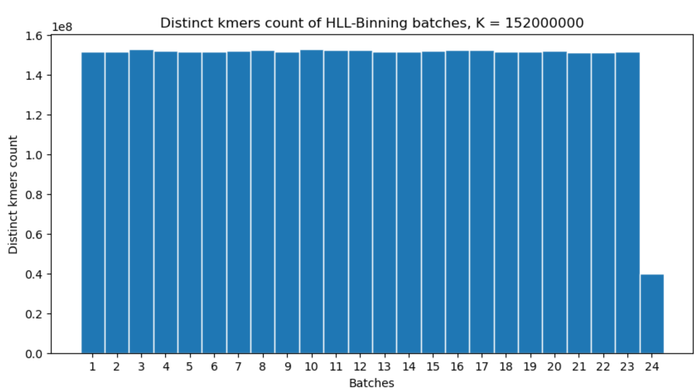
Put genomes into balanced, user-defined number of batches

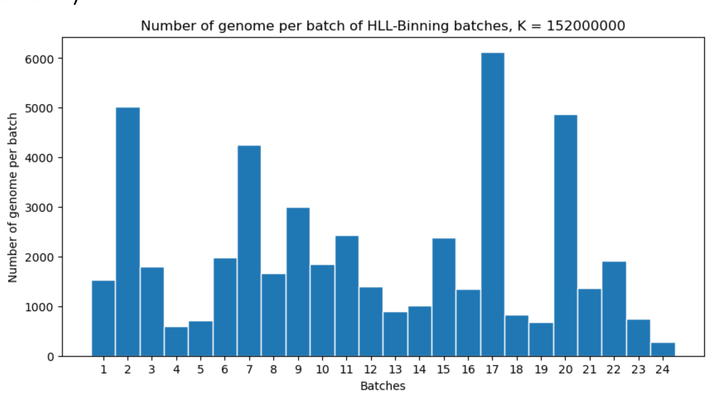
INPUT: GENOMES SKETCHES, NUMBER OF BATCH b
Initialize:
Sort the genomes based on their accession number.
Initial Assignment:
Assign the first b genomes directly to b batches.
Greedy Assignment:
For each remaining genomes, place it in the batches with the smallest number of distinct kmers.
Update the batch's contents and size.

https://github.com/tam-km-truong/HLL-Balancing

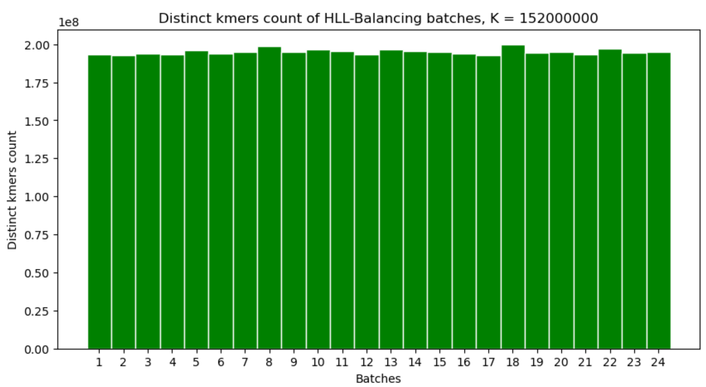
EXPERIMENT: batching of Mycobacterium Turberculosis

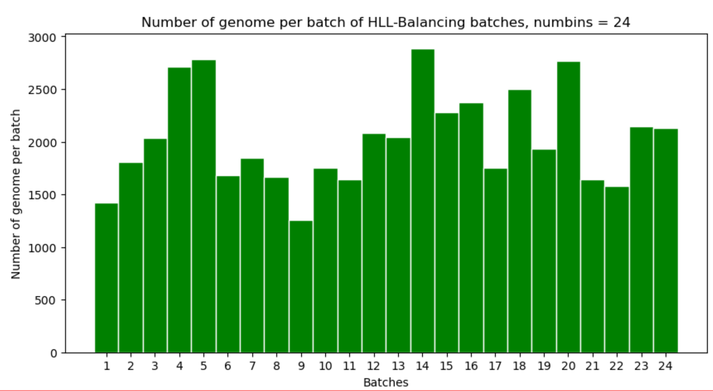
Max distinct kmers for batches: 153000000 kmers so that the compressed size stays under 64MB (caculated by using the correlation of compressed size and distinct kmers count)





Using the number off batches found by HLL-Binning, run HLL-Balancing with numbins b = 24





Eventhough the baches are balances, HLL-Balancing batches have higher distinct kmers count than the capacity set in HLL-Binning

Discussion