Mora: abundance aware metagenomic read re-assignment for disentangling similar strains*

Andrew Zheng[1], Jim Shaw[1], and Yun William Yu[1,2]

[1] Department of Mathematics, University of Toronto, Toronto, ON, M5S 2E4, Canada
[2] Computer and Mathematical Sciences, University of Toronto at Scarborough, Toronto, ON, M1C 1A4, Canada
andrewf.zheng@mail.utoronto.ca, {jshaw, ywyu}@math.toronto.edu

Abstract. Taxonomic classification of reads obtained by metagenomic sequencing is often a first step for understanding a microbial community. While species level classification has become routine, correctly assigning sequencing reads to the strain or sub-species level has remained a challenging computational problem. We introduce Mora, a MetagenOmic read Re-Assignment algorithm capable of assigning short and long metagenomic reads with high precision, even at the strain level. Mora is able to accurately re-assign reads by first estimating abundances through an expectation-maximization algorithm and then utilizing abundance information to re-assign query reads. The key idea behind Mora is to maximize read re-assignment qualities *while simultaneously* minimizing the difference from estimated abundance levels, allowing Mora to avoid over assigning reads to the same genomes. On simulated data, this allows Mora to achieve F1 scores of $> 74\%$ when assigning reads generated from three distinct E. coli strains, much higher than the 29% and 32% achieved by Pathoscope2 and Pufferfish. Furthermore, we show that the high penalty of over assigning reads to a common reference genome allows Mora to accurately identify the presence of low abundance strains and species.
Code availability: https://github.com/AfZheng126/MORA

Keywords: Metagenomics · Read Re-Assignment · Abundance Quantification.

* Supported by Natural Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2022-03074.

1 Introduction

When analyzing microbial communities through metagenomic sequencing, a fundamental task is to determine which reference genome a specific sequencing read originates from [21]. This gives information about microbial composition and allows for mapping-based analysis of genetic variation. A common first step in a processing pipeline is to use a fast taxonomic read classifier such as Kraken2 [35], CLARK [25], Centrifuge [12] and others. While such methods are extremely fast, they are not sensitive enough to assign reads to the strain level. Since strain-level resolution has important functional implications [28,6,19], sensitive methods that are able to resolve reads at the strain level are needed.

To resolve reads at the level of strains, a naive approach would be to more sensitively align the read to a set of candidate reference genomes using a read aligner such as Bowtie2 [15], Minimap2 [18], or CORA [36] (no relation to Mora) and then take the best reference genome as the correct assignment. However, strain-level reference genomes share large regions of similarity, so many ambiguous mappings are inevitable. To overcome this limitation, one can statistically calculate abundance information of the candidate set of reference strains and only use references that have high enough abundances [20,10,1,7]. Afterwards, one can re-assign reads to the "correct" references, i.e. reference strains that seem to be abundant in the metagenomic sample.

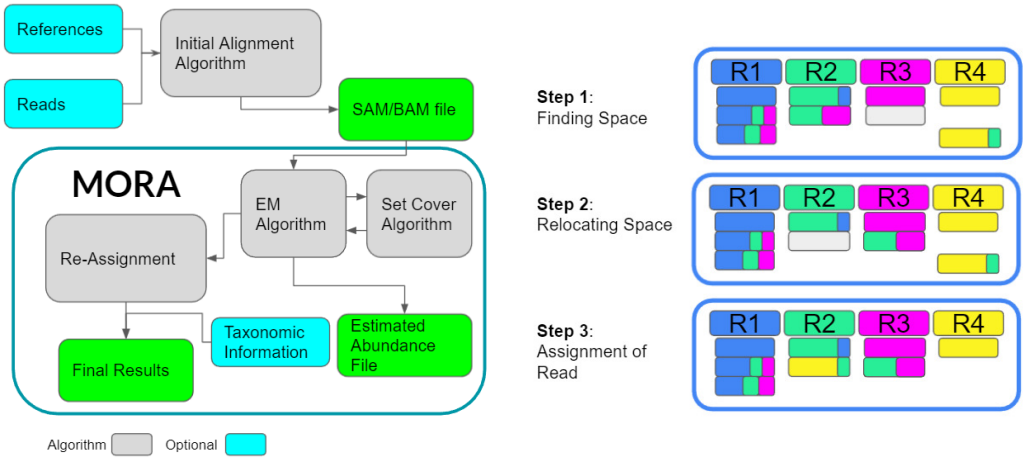
However, re-assigning reads to the correct references while maintaining abundance estimates is non-trivial. For example, if one were to assign a multi-mapped read to the most abundant reference strain with a putative mapping, all reads coming from a region of similarity between two strains will be mapped to only one strain—in this case the most abundant one. This is not an accurate assignment of the reads and will skew the abundance. Importantly, not all algorithms do abundance estimation *and* read assignment; some algorithms calculate only abundances and do not output re-assigned reads [33,2,31].

In this paper, we present Mora, a tool that allows for sensitive yet efficient metagenomic read re-assignment and abundance calculation at the strain level for both long and short reads. Given an alignment in SAM or BAM format and a set of reference strains, Mora calculates the abundance of each reference strain present in the sample and re-assigns the reads to the correct reference strain in a way such that abundance estimates are preserved. We rigorously formulate this problem as an optimization problem and give provable guarantees on our heuristic algorithm. We show that Mora is more effective than Pathoscope2 [10], a state-of-the-art read assigner, at disambiguating similar strains present in a sample on simulated data for short-reads while being an order of magnitude faster. Furthermore, we show that Mora has comparable performance to Pathoscope2 on simulated long-read data, and verify our results on real long-read data as well. Mora is available at `https://github.com/AfZheng126/MORA`.

2 Results and Discussion

Mora's pipeline consists of two main steps: abundance estimation and read re-assignment. Like other metagenomic abundance estimators [31,27], it iterates between the two steps of an expectation maximization (EM) algorithm until convergence or until a threshold to determine the most likely abundance levels of the reference genomes (Section 3.2), followed by then re-assigning reads in a manner *dependent on the calculated abundances*. Mora models the problem of maximizing correct read assignments while minimizing the difference between predicted abundance levels and final abundance levels as a non-linear minimization problem (Section 3.3). Although we use a greedy heuristic, we prove that heuristic is guaranteed to improve the minimization score. Finally, Mora outputs a (re-)assignment of each read to a reference genome that is consistent with the calculated abundances.

Comparison with other tools: We benchmarked Mora using simulated short and long reads against Pathoscope2 [10], which was the only re-assignment software we were able to get working. Pathoscope2 can



(a) Mora Pipeline

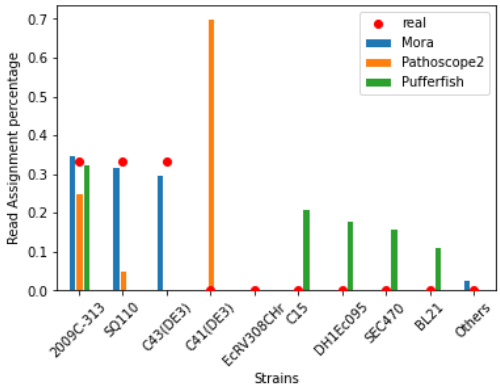
(b) Re-Assignment Algorithm

Fig. 1: (a) The pipeline for Mora to output re-assignments from query and reference FASTA files. Mora's processing steps are enclosed by the dark green rectangle. The abundance estimation step includes the Expectation Maximization (EM) algorithm and the set cover algorithm. The re-assignment step uses the estimated abundance from the previous step to re-assign reads. (b) Example of opening up space for read assignment in the Re-assignment step. R1, R2, R3, R4 represent four reference genomes and the small rectangles represent reads. The colour of the reads correspond to the mapping scores it has to each reference genome. The grey boxes below each reference represents the available capacity for that reference. The last read cannot be assigned as there is no space available in R2 and R4. The read in R4 cannot be moved as it is a uniquely mapping read. The second read in R2 is moved to R3 to open up in R2, allowing the last read to be assigned to R2.

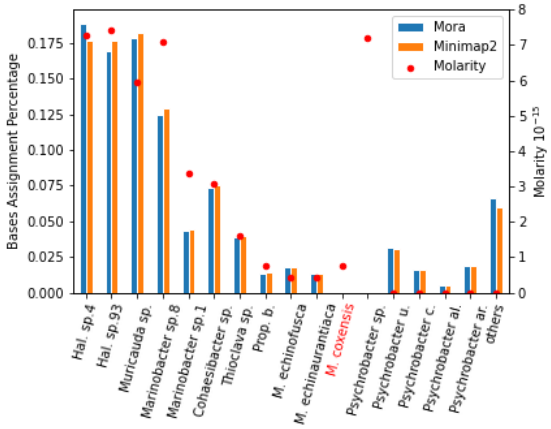
identify strains, compute abundance levels, perform re-alignments, and output informative result summaries. It has seen usage in studies in numerous studies [32,9,13,26] and its algorithm has also been incorporated into other taxonomic classification pipelines [8].

We also considered MetaMaps [7] and Sigma [1], two other re-assignment tools, but they were not included in this comparison as the software are no longer actively maintained and parts of both software are no longer functional; other studies also had issues with these two programs [30,33]. For Pathoscope2, its initial construction of the library no longer works and must be replaced with code from MetaScope [23], but given that caveat, Pathoscope2 appears to be the current practical state-of-the-art. In addition, for a fully benchmarking of Mora, we also compare against the read-mappers alone without abundance information as a baseline.

To access accuracy of the methods, we use root mean squared error (RMSE) for checking difference between the final abundance level and the predicted abundance levels. F1 score, sensitivity, and precision are used to evaluate the accuracy of the final read re-assignment at three different taxonomic ranks: strain, species, and genus (see section 3.4). For the purpose of this paper, two DNA sequences are of the same species/genus if their NCBI taxID corresponds to the same species/genus name. Two DNA sequences are of the same strain if their accession number is the same. As an important aside, in our results, we measure the sensitivity of read re-assignment component, which is our primary contribution, rather than measuring sensitivity of a full pipeline, which will be affected by the sensitivity of the initial read mapper.



(a) Assignment Percentages of Mora, Pathoscope2, and Pufferfish of simulated E. coli Short Reads



(b) Bases Assignment Percentages of Mora and Minimap2 of real long-read data of synthetic community.

Fig. 2: Figure (a): Assignment percentages of 950,000 synthetic short 150bp pair-end reads to 58 E. coli strains. The synthetic short reads were simulated from the three E. coli strains: 2009C-3133, SQ110, and C43(DE3). The strains listed are strains with at least 2000 assigned reads from at least one of the algorithms. Relative assignment percentages of the final assignments from three different algorithms are represented by the different colour bars. The real abundance levels of the strains are represented by the red dots. Assignment by Pufferfish was done by choosing the primary alignment in the SAM file, while assignment by Mora and Pathoscope2 was done using Pufferfish as the initial aligner. Figure (b): Comparison of assignment percentages of nanopore-ONT long reads [29] between Mora (using Minimap2 as initial aligner), just Minimap2 primary alignments, and real molar concentrations on a synthetic community generated from 12 strains. The reference genomes were REF-2 augmented with generation strains, except Psychrobacter s.p., which was no longer available in NCBI's Assembly database. Figure shows the generation strains and all strains that had at least 800 reads assigned. Psychrobacter s.p. reads seem to have been assigned to other related strains, whereas the lack of M. coxensis reads was hypothesized by the original authors [29] as potential human error; however, although not visible on the figure, Mora did assign 11 reads to it, whereas Minimap2 assigned none.

2.1 Re-Assignment of Short Reads at the Strain level

To test strain identification, 58 E. coli reference genomes were obtained from NCBI and 950,000 short 150bp single end reads were generated for 3 of those strains, so each strain has about 30x coverage. We aligned the reads with Pufferfish/Puffaligner [3], a very efficient read alignment method. We then used Mora and Pathoscope2 to re-assign the reads, and we report the number of assigned reads for each strain. The results are shown in Figure 2a. Pathoscope2 was programmed to perform an initial assignment by calling Bowtie2 with pathoMAP. To allow the use of other alignment algorithms, we generated the SAM file independently of Pathoscope2 and ran pathoID and pathoMAP on it.

In Figure 2a, Mora, Pathoscope2, and Pufferfish were able to identify the presence of all three strains. Mora had a F1 score of 74.03, much higher than 29.01 and 32.48, the F1 scores of Pathoscope2 and Pufferfish respectively. Pathoscope2 mapped most reads to the strain C41(DE3), a strain whose acerage nucleotide identity to C43(DE3) is 99.94% according to OrthoANI [16]. This may be caused by how Pathoscope2 incorporates reference length into its calculations of read alignment scores. The length of strain C43(DE3) was 56,061 base pairs shorter than strain C41(DE3), making the alignment scores to C41(DE3) higher than those to C43(DE3), which leads to a large number false assignments. As most high mapping scores were

Table 1: Wall clock time (s) and maximum RAM usage (GB) for algorithms on simulated short reads (SR) and long reads (LR) using 8 threads. The * indicates that we had to modify Pathoscope2's output script for it to be able to run on this dataset (otherwise it simply would not run). Other than using more RAM on short reads to REF-2, Mora consistently used fewer resources than Pathoscope2 and was substantially faster. All algorithms were run on a Ubuntu 20.04.4 LTS server with an AMD Ryzen 3970X 32-core CPU with 256 GB of RAM.

	SR to REF-1		SR to REF-2		LR to REF-1		LR to REF-2	
Program	Time	RAM	Time	RAM	Time	RAM	Time	RAM
Mora	**381.48**	**47.04**	**514.30**	46.16	**62.91**	**0.78**	**23.42**	**1.83**
Pathoscope2*	27113.49	51.58	27378.45	**15.98**	341.93	21.51	429.23	22.522
Minimap2	NA	NA	NA	NA	62.91	0.78	5578.24	100.97

equal in value, selecting the primary alignment without regard of abundance levels is likely the reason of Pufferfish's high false positive rate.

2.2 Real long-read nanopore data from synthetic community

To show applicability to real sequencing data, we ran Mora (using Minimap2 as the initial aligner) on real long nanopore data [29] from a synthetic microbial community of 12 different microbial strains and compared it with just Minimap2 alone. As shown in Figure 2b, both were able to assign more than 800 reads to each of the 12 original strain with the exception of *M. coxensis*. Unlike in the prior simulated E. coli experiment, the strains were sufficiently different that Minimap2 by itself did pretty well, and Mora's reassignment procedure did not change much for those 11 assigned strains.

However, the behavior for *M. coxensis* is instructive for the benefits of reassignment. Minimap2 assigned no reads to *M. coxensis* as primary alignments, but did include it as a secondary alignment for some reads. Mora's prediction of abundance therefore suggested the presence of some *M. coxensis*, and even with an extremely low predicted abundance, it assigned 11 reads to *M. coxensis*. This shows that Mora was able to identify the presence of all strains in the synthetic community, including very low abundance ones. Though Minimap2 had comparable assignment abundancies with Mora in bulk, it was unable to identify the presence of very low abundance strains within the microbial community.

2.3 Re-Assignment of Short Reads at Species and Genus level

We used the complex Illumina 400 dataset of 400 different microbial genomes and its corresponding simulated 26.6 million 75bp paired end short reads [22] to test the accuracy of short read alignment. The less complex Illumina 100 dataset from the same source has been commonly used for comparison studies [20,27]. We consider two cases: when there is a good guess of which genomes the reads come from, and when there is no information at all. For the first case, we built a reference library (REF-1) from the Illumina 400 dataset (400 genomes) that was used to simulate our reads. For the second case, we built a reference library (REF-2) using the complete bacterial genomes from NCBI (6487 genomes). Importantly, REF-2 does not contain all references in REF-1. REF-2 contains 36% of the strains, 72% of the species, and 87% of the genera of REF-1. Due to the lack of species and genus level taxonomic data for REF-1, the scores for species and genus are worse than the scores for strain. When using REF-2 as the reference genomes, scores for the strain rank are not reported due to the low number of strains from REF-1 included in REF-2.

Table 2: Scores between Mora and Pathoscope2 when aligning short reads. Accuracy metrics are calculated after filtering out reads that the initial aligner could not map, so metrics are not comparable across Pufferfish vs. Bowtie2. For REF-1, both Mora and Pathoscope2 were extremely good, though Pathoscope2 appears to have a slight systematic edge. For REF-2, when using Bowtie2, Mora performed substantially better.

(a) Reference = REF-1

Short reads	Strain			Species			Genus		
Algorithm	F1	Sensitivity	Precision	F1	Sensitivity	Precision	F1	Sensitivity	Precision
Mora (using Pufferfish)	97.020	97.020	97.020	92.112	87.474	97.270	99.424	99.046	99.805
Pathoscope2 (using Pufferfish)	**97.074**	**97.074**	**97.074**	**92.131**	**87.492**	**97.290**	**99.451**	**99.073**	**99.833**
Mora (using Bowtie2)	96.949	96.949	96.949	92.016	87.375	97.178	99.392	99.013	99.774
Pathoscope2 (using Bowtie2)	**97.125**	**97.125**	**97.125**	**92.221**	**87.573**	**97.391**	**99.411**	**99.032**	**99.793**

(b) Reference = REF-2

Short reads	Species			Genus		
Algorithm	F1	Sensitivity	Precision	F1	Sensitivity	Precision
Mora (using Pufferfish)	60.752	59.418	62.148	94.329	94.320	94.345
Pathoscope2 (using Pufferfish)	**62.257**	**60.890**	**63.688**	**95.246**	**95.229**	**95.263**
Mora (using Bowtie2)	**60.996**	**59.810**	**62.229**	**94.491**	**94.474**	**94.507**
Pathoscope2 (using Bowtie2)	52.703	46.328	61.114	83.491	75.609	94.282

When aligning to REF-1, Pufferfish was unable to map 10.1 million reads while Bowtie2 was unable to map 5.6 million. When aligning to REF-2, Pufferfish was unable to map 15.8 million reads while Bowtie2 was unable to map 10.1 million. These reads were filtered out of the calculations of accuracy metrics.

As shown in Table 2, Pathoscope2 performed marginally better than Mora. However, it was much slower in runtime due to the pathoREP module in Pathoscope 2. For large data sets, the pathoREP module wasn't able to output the final XML file due to lack of memory. We had to split the SAM file into 3-5 smaller files and change the XML output script in Pathoscope2 to accommodate for this problem. As seen in Table 1, Mora was over 19 times faster than Pathoscope2, while only having a 3 times increase in RAM usage. Using Bowtie2 didn't increase scores significantly for either, but did lead to a large increase in computer runtime.

2.4 Re-Assignment of Long Reads

To test Mora on long reads, we used REF-1 to simulate 1.37 million long Nanopore reads. When using Minimap2 to perform the initial alignment to REF-1, 44.3 thousand reads could not be aligned. When using Minimap2 to align to REF-2, 183 thousand reads could not be aligned.

Table 3 lists the scores evaluating assignment accuracy at the strain, species, and genus level of long reads for Mora and Pathoscope2. Mora consistently had higher scores than Pathoscope2 on long reads for REF-1, though this lead was diminished or lost when looking at the scores for REF-2. Though both algorithms had very similar scores, Table 1 shows a large difference in memory and RAM usage. Mora performs at least four times faster and uses up to 27.45 fold less RAM compared to Pathoscope2 on long reads. The main bottle neck for Pathoscope2 was again PathoREP, the algorithm to write its result files. Even after the

Table 3: Scores between Mora and Pathoscope2 when aligning long simulated reads. Accuracy metrics are calculated after filtering out reads that the initial aligner could not map. On REF-1, both Mora and Pathoscope2 performed well, but Mora had a slight edge. On REF-2, the slight edge went to Pathoscope2.

(a) Assignment to REF-1

	Strain			Species			Genus		
Algorithm	F1	Sensitivity	Precision	F1	Sensitivity	Precision	F1	Recall	Precision
Mora	**99.240**	**99.235**	**99.244**	**94.594**	**90.170**	**99.474**	**99.086**	**98.664**	**99.510**
Pathoscope2	97.708	97.644	97.773	93.013	88.605	97.883	99.005	98.520	99.494

(b) Assignment to REF-2

	Species			Genus		
Algorithm	F1	Sensitivity	Precision	F1	Sensitivity	Precision
Mora	58.176	55.667	60.922	91.193	90.842	91.547
Pathoscope2	**58.712**	**55.766**	**61.986**	**92.065**	**90.932**	**93.227**

augmentations of PathoREP to reduce its runtime and memory usage, Pathoscope's values for these fields were still higher than the values for Mora for long reads. The RMSE between percentages of final assisgned reads and estimated abundances for Mora was 0.000834, while Pathoscope2 had an RMSE of 0.00284. This is expected due to Mora's abundance aware read re-assignment method.

3 Methods

For the rest of this section, we will define *mapping score* as some positive value associated to each alignment that measures how good the alignment is. In practice, we use the AS:i secondary flag that is present in most read aligners, but our theory holds for any such score.

3.1 Initial read mapping

The input to Mora is a SAM file with an initial alignment of the reads. After filtering out unaligned reads, the remaining reads are fed into Mora. At minimum, Mora only requires a SAM file with header information to output read (re)assignments. Mora optionally allows for the inclusion of taxonomic information of reference genomes in the final output by using accession and taxonomic information from NCBI.

For convenience, Mora is also implemented with a read mapping module using Snakemake [14], allowing the user to use Mora from a set of input reads and references. The mapping modules allow the user to use Pufferfish, Bowtie2, or Minimap2. However other programs can be used as long as their generated SAM contains a header with the reference genomes and the AS:i secondary mapping flag is available in each alignment record. After the SAM file is generated, the reads that could not be assigned are filtered out. These reads are later added to the final output with the "NO ALIGNMENT" string being assigned to them.

3.2 Abundance estimation

We solve the problem of estimating the abundance levels of reference genomes by adapting the common model from calculating RNA transcript abundances used by Agamemnon [31] and RSEM [17]. In this model,

each read is generated by first selecting a reference and then a position on that reference. Assuming the same model for all the reads, we have the following likelihood function for observing a set of reads from a set of reference genomes.

$$\mathfrak{L}(\theta|\mathfrak{Q}) = \prod_{r_i \in \mathfrak{Q}} \sum_{j=1}^{M} Pr(R_j|\theta) Pr(r_i|R_j)$$

where M is the number of references, $\mathfrak{Q}$ is the set of reads, θ is the abundance estimation, $Pr(R_j|\theta)$ is the prior probability of selecting reference R_j, and $Pr(r_i|R_j)$ is the conditional probability of generating read r_i from reference R_j.

$Pr(r_i|R_j)$ is computed by normalizing the mapping score between read r_i and reference R_j over all mapping scores from r_i. The estimation of θ is done with an EM algorithm. To reduce the number of iterations needed by the EM algorithm to converge, the reads are reduced to a set of equivalent classes where two reads r_{i_1} and r_{i_2} are equivalent if they align to the same set of references.

After every 10 iterations, we implement Agamemnon's idea of using a set cover algorithm to remove redundant references with low abundances from the rest of the iteration process. This results in better estimate abundances of leftover references while also increasing convergence rate. These algorithms are implemented based on the Cedar algorithm used by Agamemnon. For more information, please consult Agamemnon's original paper.

3.3 Re-Assignment of reads

The assignment of reads based on their mapping scores while trying to stay true to the estimated abundance levels can be modeled as a variant of the Weapon-Target Assignment (WTA) problem [4]. Given different weapons that are to be fired on a set of different targets, the objective is to find which weapons should be assigned to which targets to minimize the expected remaining health of the targets. Formulating it as a non-linear problem, let $\{T_1, T_2, \ldots, T_N\}$ be the set of targets and $\{W_1, W_2, \ldots, W_M\}$ be the set of different weapons. For each weapon type W_i, there are w_i number of it and each has probability p_{ij} to destroy the target T_j. After assigning x_{ij} of weapon W_i to target T_j, the probability that T_j survives is $\prod_{i=1}^{M}(1-p_{ij})^{x_{ij}}$. Thus the WTA problem aims to minimize the following non-linear problem with constraints:

$$\min G(x) = \min\left(\sum_{j=1}^{N}\prod_{i=1}^{M}(1-p_{ij})^{x_{ij}}\right), \quad \sum_{j=1}^{N} x_{ij} \leq w_i, \quad x_{ij} \in \mathbb{Z}^+$$

Instead of selecting which weapons to assign to which targets to maximize expected damage, we are selecting which reads to assign to which references to maximize likelihood. Let R_j represent a reference genome, r_i represent a read, and p_{ij} represent the probability that r_i comes from R_j is true, calculated in the same way as for abundance estimation. To model our metagenomic problem as the WTA problem, we have three assumptions:

1. Every reference R_j appears $a_j \cdot M$ times in the set of references, where a_j is the estimated abundance of R_j and M is the total number of reads. The value of $a_j \cdot M$ is approximated to the nearest integer.
2. Every read maps to some reference genome.
3. Every read is unique, so $w_i = 1$ for $i = 1, \ldots, M$.

These three assumptions result in the re-assignment problem being formulated as the following minimization problem with constraints:

$$\min F(x) = \min\left(\sum_{j=1}^{N}\left[\sum_{k=1}^{a_j \cdot M}\prod_{i=1}^{M}(1-p_{ij})^{x_{i,(j,k)}}\right]\right) \tag{1}$$

$$\sum_{j=1}^{N} \sum_{k=1}^{a_j \cdot M} x_{i,(j,k)} \leq 1, \quad x_{ij} \in \mathbb{Z}^+$$

The second sum shows that there are $a_j \cdot M$ of reference M_j and the $x_{i,(j,k)}$ is how much of read r_i we assign to the kth copy of reference M_j. Using Equation 1, we can now use optimization methods to get exact solutions, though this is not very practical given the bad runtime scaling for large data sets. This model is more punishing against undershooting compared to overshooting as over-assigning reads to a reference does not decrease the objective function $F(x)$. This is desirable as it is better to identify all the small abundance genomes and undershoot the most abundant genome than to not identify the small abundance genomes. As leaving a read un-assigned and assigning it to a very wrong reference both contribute a value of 1 to the objective function $F(x)$, the assumption that every read maps to some reference is needed to prevent large numbers of false positives.

This is an NP hard problem [4], so Mora uses a greedy algorithm that finds a relatively good solution. Mora views each reference as a bin with a fixed space capacity. Every time a read is assigned to a reference, the available capacity of the reference decreases. Once the capacity of a reference is full, no other read can be assigned to that reference unless something is taken out. By default, the capacity C_j of a reference R_j is $C_j = a_j + 0.001$. The amount of space each read takes up is $1/M$. As reads get assigned to the references, the references R_i can be represented as a list of assignments $\mathcal{A}(R_i)$ containing the reads that have already been assigned to it.

$$\mathcal{A}(R_i) = \{(r_{i_1}, s_{i_1,k}), (r_{i_2}, s_{i_2,k}), ..., (r_{i_n}, s_{i_n,k})\} \tag{2}$$

where r_{i_k} represents a read, $s_{i_k,i}$ is the corresponding mapping score between r_{i_k} and R_i, and n is the current number of reads that have been assigned to R_i. The list is ordered using $s_{i_k,i} \geq s_{i_{(k+1)},i}$ and the capacity limitation is $\frac{n}{M} < C_i$. If $\frac{n+1}{M} \geq C_i$, we say that the reference R_i is full. Similarly, the reads r_j can be represented as a list of potential mappings $\mathcal{M}(r_j)$.

$$\mathcal{M}(r_j) = \{(R_{j_1}, s_{j,j_1}), (R_{j_2}, s_{j,j_2}), ..., (R_{j_m}, s_{j,j_m})\} \tag{3}$$

where the list is ordered such that $s_{j,j_k} \geq s_{j,j_{(k+1)}}$. The total score of a read r_j is defined to be

$$T(r_i) = \sum_{(R_a, s_{i,a}) \in \mathcal{M}(r_i)} s_{i,a}$$

which gives us, by definition, that $p_{ij} = \frac{s_{i,j}}{T(r_i)}$.

Mora assigns the reads in terms of priority. A read r_i is given priority 1 if the read maps to only one reference. A read r_i is given priority 2 if the ratio of the second best score to the best score is less than a threshold. By default, this threshold is 0.5 but can be changed. A read is given priority 3 if it doesn't satisfies the conditions of being priority 1 or priority 2. Once priority values are assigned, the priority 1 reads are assigned, followed by priority 2 reads, and then priority 3 reads.

Priority 1 reads are assigned to the unique reference they map to. For priority 2 reads, they are first sorted from highest best mapping score to lowest best mapping scores. In this order, the reads are assigned to the reference with the best mapping score if that reference has space. If the reference is at full capacity, the read is relabeled as a priority 3 read. When assigning priority 3 reads, all mappings between the priority 3 reads and references that still have space are sorted in terms of score into a list. The reads are then assigned in order of this list, or left over for a second round of assignment if of its potential references are full. After the initial assignment is done, Mora will try to "open up space" in a reference to assign leftover reads.

Definition 1. *For a read r_i and $(R_j, s_{i,j}) \in \mathcal{M}(r_i)$, R_j can open up space for r_i if it is full and there exists a $(r_k, s_{k,j}) \in \mathcal{A}(R_j)$ such that r_k can be moved to another reference $(R_l, s_{k,l}) \in \mathcal{M}(r_k)$ with the condition*

that

$$\frac{s_{i,j}}{T(r_i)} - \frac{s_{k,j}}{T(r_k)} \geq \frac{s_{k,j}}{T(r_k)} - \frac{s_{k,l}}{T(r_k)} \tag{4}$$

Using the notation in this definition, we have the following theorem.

Theorem 1. *If r_i is a read and R_j is a reference that can open up space for r_i by re-assigning r_k to another reference R_l, then doing so and then assigning r_i to R_j decreases the value of $F(x)$ from Equation 1.*

Proof. The act of performing this re-assigning r_k from R_j to R_l and then assigning r_i to R_j is equivalent to changing from $[x_{ij}, x_{kj}, x_{kl}] = [0, 1, 0]$ to $[x^*_{ij}, x^*_{kj}, x^*_{kl}] = [1, 0, 1]$. Since $p_{ij} = \frac{s_{i,j}}{T(r_i)}$, the last condition of being able to open up space gives us that

$$p_{ij} - p_{kj} \geq p_{kj} - p_{kl} \Longrightarrow (1 - p_{ij}) + (1 - p_{kl}) \leq 2(1 - p_{kj}) \leq 1 + (1 - p_{kj})$$

The right side can be written as

$$(1 - p_{ij})^{x^*_{ij}} + (1 - p_{kj})^{x^*_{kj}} + (1 - p_{kl})^{x^*_{kl}} \leq (1 - p_{ij})^{x_{ij}} + (1 - p_{kj})^{x_{kj}} + (1 - p_{kl})^{x_{kl}}$$

Since there is space in R_l, changing the three values x_{ij}, x_{kj}, x_{kl} doesn't result in the invalidation of any constraints and doesn't affect any other terms in the sum of $F(x)$. Thus, performing the re-assignment causes a decrease in the value of $F(x)$. □

If space cannot be opened up in R_{j_1}, Mora will try to open up space in R_{j_2}, and so on. If space cannot be opened for any of the references the read maps to, the read will be left to the end to be assigned randomly with weights corresponding to the mapping scores. At this stage, mapping these reads using probability is plausible as their mapping scores are relatively similar to each other (example in Figure 1b).

3.4 Evaluation metrics

Genomes are classified as the same species or genus depending on the taxonomic information listed in NCBI. Taxonomic information of the data is obtained from NCBI Taxonomy's FTP database (/pub/taxonomy/accession2taxid/) using the live and dead nucleotide sequence records. Genomes are classified as the same strain if their accession numbers are the same. The calculation of the accuracy metrics only consider reads that were successfully aligned by the first assignment algorithm. This allows us to evaluate the re-assignment algorithms without having the results be affected by the first assignment algorithms.

Read assignment accuracy on the simulated data sets are measured using F1 score, sensitivity, and precision. Let r_i be a read generated from a reference R_i. At any taxonomic rank, r_i is labeled as a true positive if it is mapped to reference R_j that agrees with R_i at that taxonomic rank. If R_i and R_j do not agree at that rank, then r_i is labeled a false positive. If r_i is not assigned to anything, it is not labeled as anything. For example, a read generated from *Escherichia coli* with accession number CP0001, assigning it to *Escherichia coli* with accession number CP0005 would be a true positive for the species and genus rank, but a false positive at the strain rank. Assigning it to *Escherichia fergusonii* with accession number CP1001 would be a true positive at the genus rank, but a false positive at the strain and species rank.

For a taxonomic rank, let TP be the total number of true positives for that rank and let FP be the total number of false positives for that rank. We define

$$\text{Sensitivity} = \frac{TP}{M}, \quad \text{Precision} = \frac{TP}{TP + FP}$$

where M is the total number of reads. The F1 score is defined to be the harmonic mean of sensitivity and precision.

To calculate the difference in the different abundance values, root mean squared error (RMSE) was used. This was used due to the lack of major outliers. The equation for root mean squared error is as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x_i})^2}{N}}$$

3.5 Data Simulation and Availability

30 E. coli genomes assemblies were downloaded from NCBI Assembly and combined to form 58 E. coli strains. The three strains 2009C-3133, SQ110, and C43(DE3) were chosen randomly to simulate short reads 1.37 million 150bp pair-end short reads. The reads were simulated according to a uniform distribution using art_illumina [11] with the default parameters for pair-end reads. The simulated 26.6 million 75bp pair-end short read data from REF-1 was obtained from [22], where it simulated using iMESS_Illumina with a skewed distribution. For the simulation of long reads, badreads [34] was used with the default parameters corresponding to mediocre Oxford Nanopore reads with quantity 20x. The read distribution is proportional to the length of the references.

The *E. coli* reads and references are available at `https://github.com/AfZheng126/MORA-data`. The full tables of scores, time, and memory usage for the different simulations are also available in the same repository as the *E. coli* reads/reference data.

4 Conclusion

In this work, we presented Mora, a new flexible algorithm and pipeline for assigning reads at the strain level. Mora takes as input an alignment and re-assigns the reads to strains by (1) estimating abundance information and (2) modelling the re-assignment problem as a discrete non-linear minimization problem for which Mora's heuristic solution has provable guarantees. We showed that Mora is more sensitive at disentangling reads on very similar sets of strains compared to Pathoscope2, yet is robust on diverse communities as well. Furthermore, even on real long-reads, where Minimap2 is already fairly good at bulk abundances using just primary alignments, Mora shows greater ability to detect low-abundance strains. Additionally, we showed that Mora is fast and practical to use, even on large datasets, with speeds and memory usage several times better than Pathoscope2.

We found that there is a surprising lack of well-engineered tools that deal with the specific problem of sensitive read re-assignment to the strain level. Thus we designed Mora using general mathematical formulations, leading to it working well on many kinds of data. Furthermore, Mora is engineered to be modular and easy to use—the minimal input required is just a single SAM file. Thus we believe that Mora will be a useful tool for researchers interested in studying strain-level read information from metagenomic sequencing data.

References

1. Ahn, T.H., Chai, J. and Pan, C.: Sigma: strain-level inference of genomes from metagenomic analysis for biosurveillance. Bioinformatics **31**(2), 170-177 (2015).
2. Albanese, D., Donati, C.: Strain profiling and epidemiology of bacterial species from metagenomic sequencing. Nat Commun. **8**, 2260 (2017).
3. Almodaresi, F., Zakeri, M., Patro, R.: PuffAligner: a fast, efficient and accurate aligner based on the Pufferfish index. Bioinformatics **37**(22), 4048-4055 (2021).
4. Andersen, A.C., Pavlikov, K., Toffolo, T.A.: Weapon-target assignment problem: Exact and approximate solution algorithms. Annals of Operations Research, 1-26 (2022).
5. Bray, N.L., Pimentel, H., Melsted, P. and Pachter, L.: Near-optimal probabilistic RNA-seq quantification. Nature biotechnology **34**(5), 525-527 (2016).
6. Brito, I.L., Alm, E.J.: Tracking Strains in the Microbiome: Insights from Metagenomics and Models. Frontiers in Microbiology. 7, (2016).
7. Dilthey, A.T., Jain, C., Koren, S., Phillippy, A.M.: Strain-level metagenomic assignment and compositional estimation for long reads with MetaMaps. Nature communications **10**(1), (2019).
8. Fan, J., Huang, S., Chorlton, S.D.: BugSeq: a highly accurate cloud platform for long-read metagenomic analyses. BMC Bioinformatics **22**, 160 (2021).
9. Fujiogi, M., Raita, Y., Pérez-Losada, M., Freishtat, R.J., Celedón, J.C., Mansbach, J.M., Piedra, P.A., Zhu, Z., Camargo, C.A. and Hasegawa, K.: Integrated relationship of nasopharyngeal airway host response and microbiome associates with bronchiolitis severity. Nature communications, **13**(1), 1-12 (2022).
10. Hong, C., Manimaran, S., Shen, Y., Perez-Rogers, J.F., Byrd, A.L., Castro-Nallar, E., Crandall, K.A. and Johnson, W.E.: PathoScope 2.0: a complete computational framework for strain identification in environmental or clinical sequencing samples. Microbiome **2**(1), 1-15, (2014).
11. Huan, W., Mysers, J.R., Marth, G.T.: ART: a next-generation sequencing read simulator. Bioinformatics **28**(4), 593-594 (2012).
12. Kim, D., Song, L., Breitwieser, F.P., Salzberg, S.L.: Centrifuge: rapid and sensitive classification of metagenomic sequences. Genome Res. **26**, 1721–1729 (2016).
13. Kinsella, C.M., Deijs, M., Becker, C., Broekhuizen, P., van Gool, T., Bart, A., Schaefer, A.S. and van der Hoek, L.: Host prediction for disease-associated gastrointestinal cressdnaviruses. Virus Evolution, (2022).
14. Koster, J., Rahmann, S.: Snakemake - a scalable bioinformatics workflow engine. Bioinformatics **34**(20), 3600 (2018).
15. Langmead, B., Salzberg, S.L.: Fast gapped-read alignment with Bowtie 2. Nature methods **9**(4), 357-359 (2012).
16. Lee, I., Ouk Kim, Y., Park, S.-C., Chun, J.: OrthoANI: An improved algorithm and software for calculating average nucleotide identity. International Journal of Systematic and Evolutionary Microbiology **66**, 1100–1103 (2016).
17. Li, B. and Dewey, C.N.: RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC bioinformatics **12**(1), 1-16 (2011).
18. Li, H.: Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics **34**(18), 3094-3100. (2018).
19. Leimbach, A., Hacker, J., Dobrindt, U.: E. coli as an all-rounder: the thin line between commensalism and pathogenicity. Curr Top Microbiol Immunol. 358, 3–32 (2013).
20. Lu, J., Breitwieser, F.P., Thielen, P. and Salzberg, S.L.: Bracken: estimating species abundance in metagenomics data. PeerJ Computer Science **3**, e104. (2017).
21. Lu, J., Rincon, N., Wood, D.E., Breitwieser, F.P., Pockrandt, C., Langmead, B., Salzberg, S.L., Steinegger, M.: Metagenome analysis using the Kraken software suite. Nat Protoc. 1–25 (2022).
22. Mende, D.R., Waller, A.S., Sunagawa, S., Javelin, A.I., Chan, M.M., Arumugam, M., Raes, J., Bork, P.: Assessment of metagenomic assembly using simulated next generation sequencing data. PloS one **7**(2), e31386 (2012).
23. Odom-Mabey A., Varki R., Johnson W.: MetaScope: Tools and functions for preprocessing 16S and metagenomic sequencing microbiome data. R package version 0.0.99, https://github.com/compbiomed/metascopehttps://compbiomed.github.io/metascope-docs/ (2022)
24. Oh, J., Byrd, A.L., Deming, C., Conlan, S., Kong, H.H. and Segre, J.A.: Biogeography and individuality shape function in the human skin metagenome. Nature **514**(7520), 59-64 (2014).

25. Ounit, R., Wanamaker, S., Close, T.J. and Lonardi, S.: CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers. BMC genomics **16**(1), 1-13 (2015).
26. Raita, Y., Pérez-Losada, M., Freishtat, R.J., Harmon, B., Mansbach, J.M., Piedra, P.A., Zhu, Z., Camargo, C.A. and Hasegawa, K.: Integrated omics endotyping of infants with respiratory syncytial virus bronchiolitis and risk of childhood asthma. Nature communications **12**(1), 1-13 (2021).
27. Schaeffer, L., Pimentel, H., Bray, N., Melsted, P., Pachter, L.: Pseudoalignment for metagenomic read assignment. Bioinformatics **33**, 2082–2088 (2017).
28. Segata, N.: On the Road to Strain-Resolved Comparative Metagenomics. mSystems. 3, (2018).
29. Sevim, V., Lee, J., Egan, R., Clum, A., Hundley, H., Lee, J., Everroad, R.C., Detweiler, A.M., Bebout, B.M., Pett-Ridge, J. and Göker, M.: Shotgun metagenome data of a defined mock community using Oxford Nanopore, PacBio and Illumina technologies. Scientific data **6**(1), 1-9 (2019).
30. Simon, H.Y., Siddle, K.J., Park, D.J., Sabeti, P.C.: Benchmarking metagenomics tools for taxonomic classification. Cell **178**(4), 779-794 (2019).
31. Skoufos, G., Almodaresi, F., Zakeri, M., Paulson, J.N., Patro, R., Hatzigeorgiou, A.G., Valchos, I.S.: AGAMEMNON: an Accurate metaGenomics and MEtatranscirptoMics quaNtificatiON analyasis suite. Genome biology **23**(1), 1-27 (2022).
32. Uzelac, M., Li, Y., Chakladar, J., Li, W.T. and Ongkeko, W.M.: Archaea Microbiome Dysregulated Genes and Pathways as Molecular Targets for Lung Adenocarcinoma and Squamous Cell Carcinoma. International Journal of Molecular Sciences **23**(19), 11566 (2022).
33. van Dijk, L.R., Walker, B.J., Straub, T.J., Worby, C.J., Grote, A., Schreiber, H.L., Anyansi, C., Pickering, A.J., Hultgren, S.J., Manson, A.L., Abeel, T., Earl, A.M.: StrainGE: a toolkit to track and characterize low-abundance strains in complex microbial communities. Genome Biology **23**, 74 (2022).
34. Wick, R.R.: Badread: simulation of error-prone long reads. Journal of Open Source Software **4**(36) 1316 (2019).
35. Wood, D.E., Lu, J. and Langmead, B.: Improved metagenomic analysis with Kraken 2. Genome biology, **20**(1), 1-13 (2019).
36. Yorukoglu, D., Yu, Y. W., Peng, J., Berger, B.: Compressive mapping for next-generation sequencing. Nature biotechnology **34**(4), 374-376 (2016).