Alice: fast and haplotype-aware assembly of high-fidelity reads based on MSR sketching

Roland Faure, Baptiste Hilaire, Jean-François Flot, Dominique Lavenier

September 29, 2025

Abstract

We introduce Mapping-friendly Sequence Reduction (MSR) sketches, a sketching method for high-fidelity (HiFi) long reads, and Alice, an assembler that operates directly on these sketches. MSR produces compact representations that (i) are alignable sequences—two sequences align if and only if their MSR sketches align—and (ii) are collision-resistant, so distinct sequences yield distinct sketches with high probability, retaining small differences between closely related strains. Alice reduces long reads to short MSR sketches, uses a classic short-read assembly method to assemble those sketches and decompresses the result to obtain the final assembly. This strategy addresses the longstanding challenge of producing a strain-resolved assembly for a low computational cost. On an *Adineta vaga* genome, a mock gut community comprising five conspecific strains, and two real metagenomes (human stool and soil), Alice is an order of magnitude faster than state-of-the-art HiFi assemblers while delivering assemblies of comparable quality and improving recovery of highly similar strains.

1 Introduction

With the rise of high-throughput sequencing, genomic experiments have been producing vast amounts of data, far outpacing the growth of computing power predicted by Moore's law [12]. It is now common for a single experiment to generate dozens or even hundreds of gigabases of data. In parallel, the length and quality of the sequencing reads haveimproved immensely. PacBio HiFi consensus reads are several thousands of basepairs long with an error rate lower than 0.1%. Oxford Nanopore Technologies (ONT) reads have also become even longer, albeit slightly less accurate.

Assembling metagenomic datasets, i.e. aligning and merging reads to obtain consensus sequences representative of the metagenome, is a taxing computational task. It can easily require several weeks of CPU hours and hundreds of gigabytes of RAM [13, 18, 37]. As dataset become larger and cheaper to produce, metagenome assembly can become a bottleneck in terms of cost, computation time and quality.

A general popular technique to diminish the size of the computations is to sketch the input data, i.e. reduce it to a smaller representation on which computations can still be made [28]. In the realm of genome assembly, sketching has long been employed for all-versus-all read mapping as a first step of the Overlap-Layout-Consensus assembly paradigm [19]. However, it has only recently been effectively integrated into the faster De Bruijn Graph assemblers, specifically for high-fidelity reads. Building on concepts from wtdbg2 [29], shasta [30], and Peregrine [8], Ekim, Berger and Chikhi introduced a method that samples a fraction δ of the k'-mers in the reads, chains the resulting series of k k'-mers into "k-mers of k'-mers" called k-min-mers, assemble those k-min-mers and subsequently transforms the resulting chain back into a genome sequence [11]. This approach demonstrated remarkable efficiency in a proof-of-concept assembler called mDBG [11], enabling human genome assemblies to be completed in minutes on a personal computer. It was further developed as a metagenomic assembler named metaMDBG [3]. However, these assemblers encounter significant limitations that arise directly from the chosen sketching method.

Metagenomic samples as well as diploid (or polyploid) genome sequences often contain strains that are genetically similar yet functionally distinct [34]. However, when (meta)mDBG sketches the reads as a chain of k-mers, differences —such as single nucleotide polymorphisms (SNPs)—between highly similar sequences is often lost. As a result, both mDBG and metaMDBG struggle to differentiate between highly similar haplotypes.

In this study, we present a novel assembler named Alice. Conceptually, Alice shares similarities with metaMDBG, as it begins by sketching reads and assembling the sketches before decompressing the obtained sequences to yield the final assembly. However, Alice introduces a significant innovation through a new sketching method called Mapping-friendly Sequence Reduction (MSR). Originally proposed to improve read mapping quality [4], the potential of MSR as a sketching technique had not been previously investigated. In our methodology, we employ a carefully parametrized MSR to sketch PacBio HiFi reads, resulting in a computationally efficient assembler that maintains the ability to reconstruct highly similar sequences. The name "Alice" is inspired by Lewis Carroll's *Alice in Wonderland* [5], where Alice uses a "drink-me potion" to pass through a small door and a "eat-me" cake to return to her original size. In this analogy, Alice represents the reads, the small door symbolizes the constraints of hardware and software capacity, and the potion corresponds to the MSR sketching technique. The assembly process is depicted in Figure 1.

We evaluated Alice on three distinct PacBio HiFi metagenomic datasets—(i) a mock community comprising five *Escherichia coli* strains, (ii) a human-gut stool sample, and (iii) a soil sample. Compared with leading assemblers such as metaMDBG [3], (meta)Flye [17, 18], and hifiasm(_meta) [6, 13], Alice assembled the data one order of magnitude faster and with lower memory consumption. Moreover, Alice reliably discriminated closely related strains and produced the most complete assemblies in several scenarios. We also examined the assembly of a genomic dataset obtained from HiFi sequencing of the bdelloid rotifer *Adineta vaga*, a rising model organism for which several genome assemblies of increasing

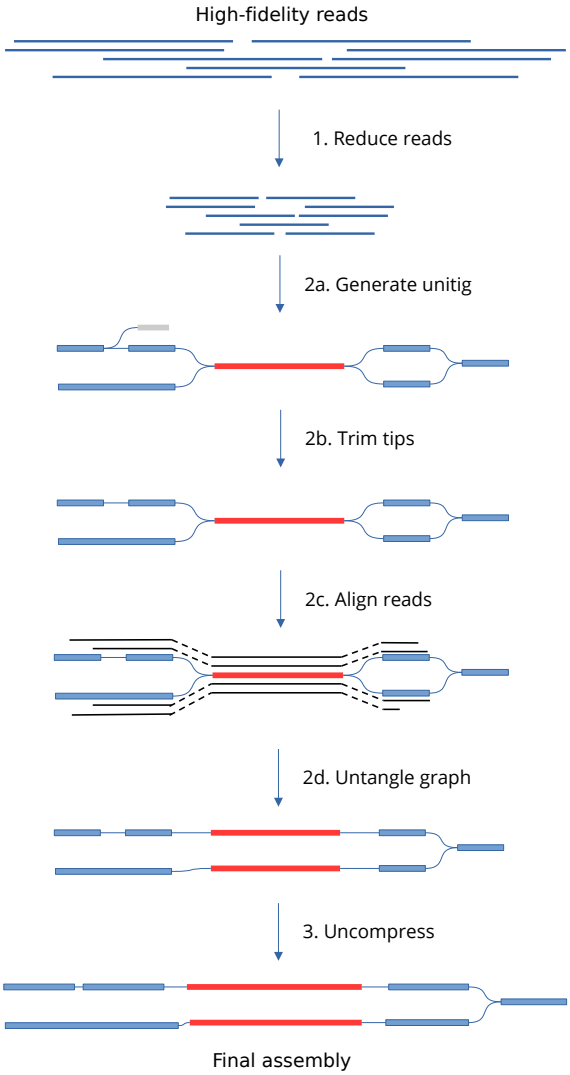


Figure 1: Assembly process of Alice. Step 2 is a very classical assembly procedure.

accuracy have been published [14, 31], albeit none based on PacBio HiFi yet. On this novel dataset, Alice generated an assembly comparable to those produced by state-of-the-art assemblers such as LJA [1], Flye [17], and hifiasm [6], but with RAM usage and run time both one order of magnitude lower than these other tools.

2 Results

The fundamental difference between (meta)MDBG and Alice is their sketching scheme. In the next two subsections, we introduce MSR sketches and their interest.

2.1 Mapping-friendly Sequence Reductions (MSR) sketches

Mapping-friendly Sequence Reductions are functions that transform a sequence of characters into a new sequence [4]. A MSR is defined by an alphabet (in this case, the DNA alphabet $\{A, C, G, T\}$), an order l and a transforming function g that maps each sequence of length l, or l-mer, to either a character in the alphabet or a special "empty" character ϵ. To ensure a a sequence and its reverse complement are reduced to reverse complement sequences (which is important for genome assembly), an extra constraint is added to g: g must map reverse-complement l-mers to reverse-complement bases.

MSRs work by taking an input sequence and breaking it down into successive overlapping l-mers, which are sequentially passed through the function g to produce a reduced sequence. If g returns a character, that character is added to the reduced sequence. If g returns the empty character ϵ, nothing is added to the reduced sequence. The pseudocode for this process is provided in Algorithm 1.

Algorithm 1 Mapping-friendly Sequence Reductions

```
Function MSR(seq, l, g)
new_seq =""
for i = 0 to len(seq) − l + 1 do
  lmer = seq[i : i + l]
  new_char = g(lmer)
  if new_char ≠ ϵ then
    new_seq = new_seq + char
  end if
end for
return new_seq
```

By design, if the length l is not too large, two highly similar sequences will share many l-mers in the same order, resulting in highly similar reduced

sequences. Consequently, the reduced versions of two sequences that align have a high probability of aligning as well; we refer to this property of the reduction as mapping-friendliness. Importantly for us, this mapping-friendly property could also be defined as assembly-friendly: the assembly of reduced reads is equivalent to the reduced assembly of the original reads (notwithstanding assembly errors). Reduced reads can thus be used as sketches of their non-reduced counterparts while being potentially much shorter.

2.2 The power of MSR sketches

While the k-min-mers used by metaMDBG tend produce identical sketches for highly similar sequences, thereby collapsing single-nucleotide polymorphism (SNP) differences between them, MSR sketches amplify the difference between highly similar sequences, hence preserving SNPs and other small differences between the haplotypes.

As an illustration, let us compare the behavior of MSR and mDBG's k-min-mers showcasing the same compression ratio. We define the compression factor c of a sketching method as the expected ratio of the number of bases in a random sequence and the number of bases in its sketch. For MSR sketching, this is equal to the inverse of the ratio of l-mers mapping to non-empty characters. For mDBG, $c = 1/\delta k'$.

Let us imagine two infinite sequences differing by a single substitution. For the sake of simplicity, let us assume that no k'-mer or l-mer is repeated around this SNP. Let c be the compression factor. In metaMBDG, a k'-mer has a probability $\delta = 1/k'c$ of being sampled, and k' k'-mers overlap the SNP. The probability that the sketches of the two sequences are different is thus

$$(1 - \frac{1}{k'c})^{2k'}$$

In MSR sketching, two sketches are identical if the l consecutive l-mers around the SNP on each sequence output the same bases in the same order. The function g employed to produce our MSR sketches is crafted to ensure that there is virtually no correlation between input k-mers and their corresponding image through g (the function is fully described in the Methods section). Therefore we can compute the probability that the sketches of the two sequences are identical by applying the law of total probability: the probability that the two sketches are identical is the probability that the two sketches have the same number of bases i (which is given by the square of the probability of choosing i items among l, if each of them has a probability $1/c$ of being chosen; i.e., the square of the binomial law) multiplied by the probability that two series of i DNA bases are identical (which is $\frac{1}{4^i}$):

$$\sum_{i=0}^{l} (\binom{l}{i} (\frac{1}{c})^i (1 - \frac{1}{c})^{l-i})^2 \cdot \frac{1}{4^i} \approx (1 - \frac{1}{c})^{2l}$$

If we use Alice's default compression factor of 20 and order l of 101, the probability that the mDBG sketches of the two sequences are different is of less

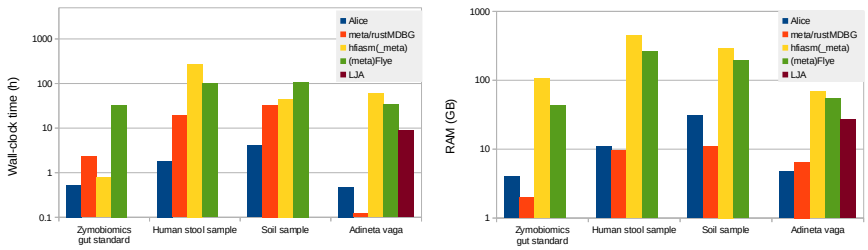


Figure 2: Wall-clock time on 8 threads and RAM usage of the assemblers on the four assemblies assemblies. Scales are logarithmic

than 10%, whereas the probability that the MSR sketches of the two sequences are different is higher than 99%.

2.3 Benchmarking setup

We conducted a benchmarking analysis of Alice, comparing its performance against the three most commonly utilized HiFi metagenomic assemblers, namely metaFlye [18], hifiasm_meta [13], and metaMDBG [3], and four genomic assemblers, namely hifiasm [6], Flye [17], LJA [1] and rust-mdbg [11]. Each assembler was executed using their recommended settings. For Alice, we used the default parameters of a compression factor of 20 and an order of 101, adding the option –single-genome for the assembly of *Adineta vaga*. A detailed discussion of these parameter choices, along with tests of alternative settings, can be found in section 4.4 of the methods.

To benchmark Alice, we first utilized the Zymobiomics Gut Microbiome Standard, a commercially available mixture of 19 bacterial strains and two yeast strains, specifically formulated to mimic the gut microbiome's composition. PacBio HiFi sequencing data for this standard were accessible under the accession number SRR13128013. The relative abundances of each organism in the mixture, along with their genomic sequences, are known. Notably, this dataset includes five closely similar strains of *Escherichia coli*, which present a challenge to assemble separately. Secondly, we assessed Alice on two true metagenomic communities, a HiFi sequencing dataset derived from a human stool sample [27] and a HiFi sequencing dataset derived from a soil sample [3]. Both datasets had previously been employed by the authors of metaMDBG to benchmark their own assembler [3]. Finally, as an exploratory endeavor, we also assembled the animal *Adineta vaga* genome to see if Alice could be applied to genomic assemblies.

2.4 Order-of-magnitude speedup

Figure 2 presents the runtime and memory consumption of each assembler across the four benchmark datasets. Alice consistently outperforms the competitors, achieving dramatic reductions in both metrics.

On the human-stool and soil samples, Alice is at least an order of magnitude faster than metaMDBG, metaFlye and hifiasm_meta. For the *Adineta vaga* dataset, Alice's speedup reaches two orders of magnitude relative to all other assemblers (aside from rust-mdbg, whose assemblies are of markedly lower quality, as discussed later).

The largest memory demand observed for Alice was $\leq$30 GB (soil assembly). Although this exceeds metaMDBG's footprint, it remains attainable even on a laptop, and represents more than a tenfold reduction compared with metaFlye and hifiasm_meta, which require several hundred gigabytes of RAM for the soil and stool samples.

Beyond saving time, money, and hardware, the modest resource demands of metaMDBG and Alice will enable much deeper sequencing of metagenomic communities in the future. Historically, the main bottleneck for deep metagenomic studies has been obtaining high-quality, high-coverage data, but recent advances now make it possible to generate HiFi datasets of hundreds of gigabases, for a rapidly decreasing price. This increased depth will allow us to detect and characterize low-abundance species that were previously missed. In contrast, assemblers such as Flye and hifiasm_meta are already approaching their practical limits in RAM consumption and runtime for these massive datasets.

2.5 Alice produces the most complete high-coverage metagenomic assemblies

We employed metaQUAST [24] to evaluate the assemblies derived from the ZymoBIOMICS gut microbiome standard. Comprehensive metrics on completeness, duplication ratios, and contiguity are reported in Supplementary Tables 1 and 2, with the full metaQUAST output provided in the supplementary data set.

Our analysis focused particularly on the assemblers' performance in separating the five closely related *Escherichia coli* strains. The assemblers displayed varied reconstruction capabilities: metaMDBG and metaFlye each reconstructed only a single strain in its entirety, whereas the remaining four strains were only partially recovered. A 27-mer–based assessment indicated that 20% (metaMDBG) and 10% (metaFlye) of strain-specific *E. coli* 27-mers were absent from the final assemblies. Conversely, both hifiasm_meta and Alice achieved high completeness across all five strains, missing merely 4.5% and 3% of the strain-specific 27-mers, respectively. Although hifiasm_meta produced longer contigs and thus attained superior completeness according to the alignment-based metaQUAST statistics relative to Alice, this came at the cost of an elevated duplication ratio. This suggests that hifiasm_meta has a propensity to "invent" spurious strains—a phenomenon also observed for the other species of

the sample.

Evaluating the assemblies of the human gut and soil metagenomes presented more challenge because the exact composition of genomes in those samples was unknown. To assess assembly completeness, we compared the 31-mer content of each assembly with the 31-mers present in the HiFi reads, which were counted using KMC [16]. We assumed that any 31-mer appearing more than five times in the HiFi reads was unlikely to be a sequencing error. Accordingly, we plotted the fraction of these high-confidence 31-mers recovered by each assembler as a function of their abundance in the reads (Figure 3a for the human stool sample and Figure 3b for the soil sample).

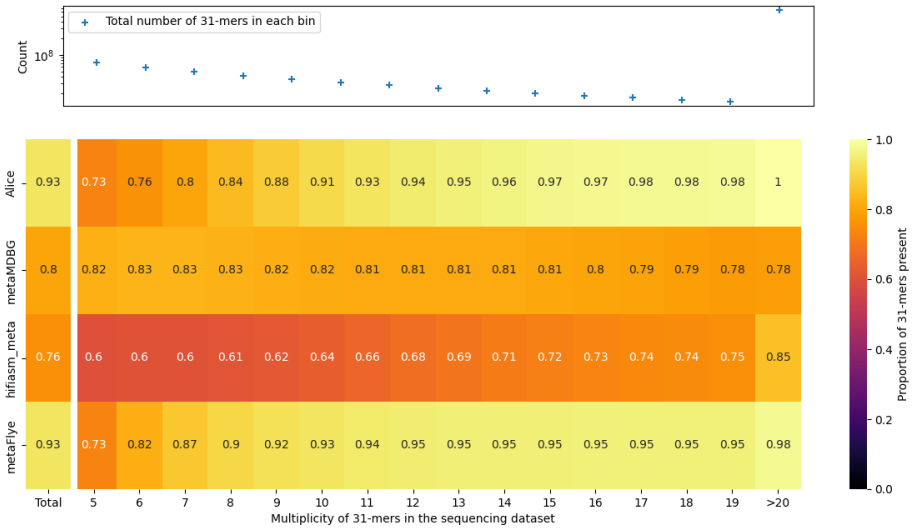
The results show that Alice yields the most complete assemblies at high coverage, whereas metaMDBG performs best at low coverage. In the human gut dataset, we were surprised to find that metaMDBG and hifiasm_meta missed 22% and 15% of the high-coverage ($\geq$20×) 31-mers, respectively. Inspection of the Alice assemblies revealed that most of the 31-mers missed by metaMDBG and hifiasm_meta reside in small bubbles or dead-ends. Indeed, these two assemblers are designed to aggressively discard such likely such short, likely artefactual sequences in order to improve overall contiguity. The low-abundance 31-mers missed by Alice were predominantly lost during the MSR compression step.

The contiguity metrics reported in Supplementary Tables 2 and 3 indicate that Alice's assemblies are generally less contiguous than those produced by the other tools. For instance, on the stool samples Alice achieved an N50 of 61 kb, whereas metaFlye, hifiasm_meta, and metaMDBG reached N50 values of 122 kb, 143 kb, and 210 kb, respectively. A similar trend appears in the soil assemblies, where Alice's N50 was 6.5 kb compared with 29 kb, 41 kb, and 17 kb for metaFlye, hifiasm_meta, and metaMDBG. This lower contiguity represents an opportunity for further optimization of Alice, acknowledging that attaining both high contiguity and high completeness remains a challenge.

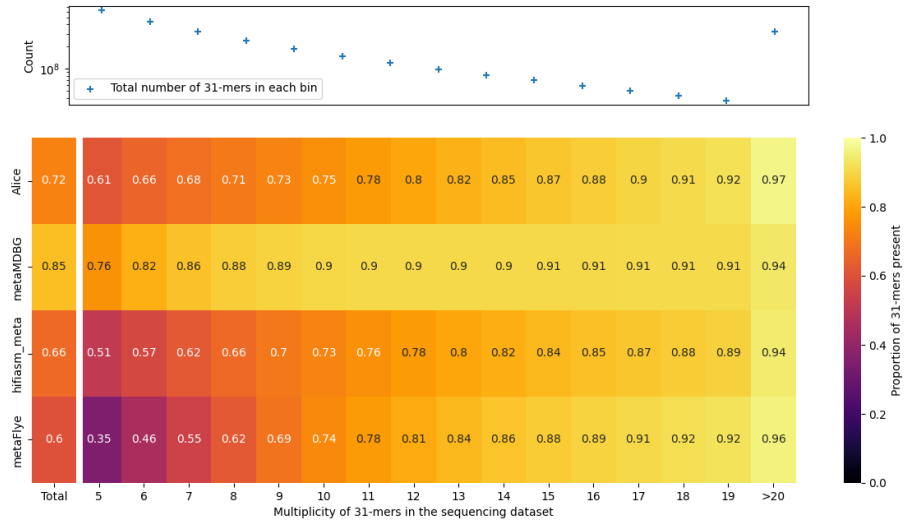
2.6 Metagenomic binners are not adapted to uncollapsed assemblies

After assembly, metagenomic contigs are typically grouped into Metagenome Assembled Genomes (MAGs), and the quality of these MAGs is commonly used to evaluate an assembler's performance [3, 13]. To assess Alice on this metric, we binned the human-gut assemblies with the popular binning program SemiBin2 [26] and evaluated the resulting bins using CheckM [9]. Across all samples, Alice's assemblies yielded fewer high-quality MAGs (>90% completeness,<5% contamination) than those produced by metaMDBG and hifiasm-meta.

A closer inspection of the *Escherichia coli* strains in the ZymoBIOMICS mock community clarified the source of this result. MetaQUAST analysis of the assemblies (Supplementary Table 1) shows that Alice and hifiasm-meta each recovered all five *E.coli* strains, whereas metaMDBG recovered only one strain in its entirety. However, SemiBin2 generated no high-quality MAG from the Alice assembly, one from the hifiasm-meta assembly, and two from the metaMDBG



(a)



(b)

Figure 3: Analysis of the 31-mer abundances of the reads vs the assemblies in (a) in the human stool sample and (b) the soil sample. The top panel displays the number of 31-mers in the reads as a function of their multiplicity in the read dataset. The bottom panel presents a heatmap in which a number of $x\%$ in bin B indicates that $x\%$ of the 31-mers of multiplicity B in the reads are present in the corresponding assembly. For example, for the stool sample, 76% of the 31-mers seen 7 times in the reads are found in the metaMDBG assembly.

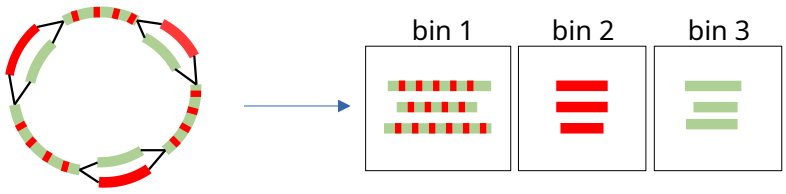


Figure 4: Typical binning problem when dealing with uncollapsed assembly. Two similar strains are assembled in an assembly graph, represented by green and red colors. However, as binners are typically heavily based on coverage, the contigs shared between the strains are not duplicated and are binned separately from strain-specific contigs.

assembly, one of which was a chimeric bin comprising fragments from multiple strains.

We hypothesize that the underlying issue is that SemiBin2 (and other binners) rely heavily on coverage profiles to assign contigs to bins. This strategy works well for long, linear contigs but fails when the assembly graph remains tangled. SemiBin2 struggles when contigs are short and some of them should be assigned to several bins (see Fig. 4). We tried using metaCoAG [21] as an alternative binning strategy to exploit more thoroughly the assembly graphs, but a similar behavior was observed.

We have seen above that the the graph-simplification steps applied by metaMDBG and hifiasm_meta can reduce assembly completeness to improve contiguity. In the light of these results, these simplifications can be seen as helping yield cleaner, more "binner-friendly" assemblies that translate into higher-quality MAGs. In contrast, Alice is more conservative in its graph simplifications to preserve the full genomic content of the sample. The trade-off is that Alice's richer, less-simplified assemblies will require the development of new binning strategies to improve over the state of the art MAGs generation.

2.7 Alice can be used to assemble genomic data

To benchmark Alice on a single-genome dataset, we sequenced the non-model diploid bdelloid rotifer *Adineta vaga* using PacBio HiFi chemistry to a depth of 140× (the reads are publicly available via BioProject PRJNA1335825). The principal challenge of this assembly is the organism's relatively high heterozygosity (1.7% [32]), whereas most assemblers are tuned for the far less heterozygous human genome.

We assessed the quality of the *A. vaga* assemblies in two ways. First, we ran a BUSCO evaluation [33, 22] against the `metazoa_odb10` reference set. This analysis showed no substantial differences in gene completeness between

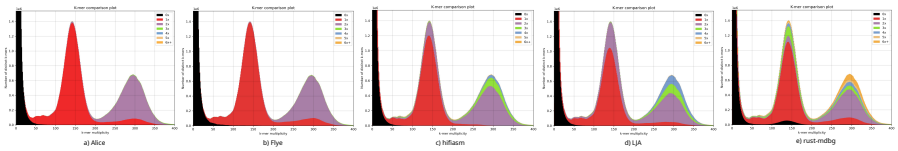


Figure 5: KAT plots of the *Adineta vaga* assemblies against the HiFi sequencing reads. The 31-mer spectra show two peaks, one corresponding to homozygous 31-mer (seen twice in the genome and on overage 280 times in the reads), and the other corresponding to heterozygous 31-mers (seen once in the genome and on average 140 times in the reads). Additionally, the peak of 31-mer with a abundance near 0 in the reads correspond to sequencing errors. The colors represent the abundance of the 31-mers in the assemblies.

the assemblers. Second, we performed a spectral analysis of 31-mer frequencies using KAT [23]; the results are displayed in Figure 5. The assembly statistics are summarized in Table 1.

Our results indicate that Alice produced a genome assembly comparable in quality to the outputs of current state-of-the-art tools such as Flye, LJA, and hifiasm, while requiring substantially less computational effort. Compared with rust-mdbg, Alice achieved similar resource usage but delivered a markedly better assembly. Spectral analysis revealed that even with a high 140× HiFi coverage, rust-mdbg lost a large fraction of 31-mers in its final contigs. Moreover, rust-mdbg, hifiasm, and LJA exhibited pronounced sequence over-duplication, whereas both Alice and Flye generated a high-fidelity assembly with only a modest amount of collapsed homozygous regions creating bubble structures in the assembly graphs. Among the tested assemblers, Flye attained the highest contiguity, giving it a slight edge over Alice, but for a much higher computational cost.

	Alice	Flye	hifiasm	LJA	rust-mdbg
31-mer completeness (%)	99.91	99.93	**99.97**	99.91	95.02
BUSCO completeness (%)	79.0	79.1	**79.9**	79.0	78.3
N50 (Mb)	1.37	9.35	**11.01**	0.08	0.05
N90 (Mb)	0.20	**0.89**	0.05	0.03	0.01
Number of contigs	1436	**165**	1429	4810	11703
CPU time (h)	1.4	253	465	62	**0.4**
Peak RAM (GB)	**4.8**	55	27	26	6.5

Table 1: Comparison of assembly statistics of *Adineta vaga* across different tools. 31-mer completeness was computed using KAT. BUSCO completeness was computed against the metazoa_odb10 database.

3 Discussion

In this study, we present a novel approach for assembling highly precise reads through the introduction of Mapping-friendly Sequence Reductions (MSR) sketches. This method is implemented in a assembler named Alice, which we evaluated on various datasets, including a diploid *Adineta vaga* genome, a challenging mock community comprising five conspecific strains of *Escherichia coli*, a human stool sample and a soil sample. Alice operated an order of magnitude faster than competing assemblers while maintaining a low memory usage. Moreover, it provided the most complete assemblies for high-coverage, strain-rich datasets.

Despite its advantages, Alice exhibits two significant limitations compared to some of its competitors. First, Alice employs conservative graph simplification strategies to preserve nodes potentially associated with strain variation, which consequently results in reduced contiguity relative to alternative assemblers and poorer downstream binning. This limitation is inherent to Alice's core assembly engine rather than the MSR sketching technique itself, therefore the assembly engine could be updated. Second, the current implementation struggles in the assembly of low-abundance strains

A promising but vast avenue for enhancing the assembler involves modifying the MSR function, which we designed to be pseudo-random. For instance, we could introduce guarantees based e.g. on syncmers [10] to ensure that at least one base is produced for all windows of length w. Another potential improvement could involve exploiting base qualities to estimate and improve the quality of the reduced sequences. The authors of [4] demonstrated that altering the function can significantly enhance results when aligning reads reduced with an MSR of order 2, indicating that the choice of the MSR function has a substantial impact on downstream applications. We hypothesize that a carefully selected MSR function could also enable Alice to effectively handle reads with higher error rates, although the challenge lies in the vast number of MSR functions available for exploration.

While this study concentrates on using MSR sketching for metagenome assembly, the technique has far-reaching potential beyond that scope. Because assembled genomes typically exhibit very low error rates, MSR sketches could be employed for tasks such as indexing or aligning assemblies—e.g., constructing pangenome graphs. Additional promising applications include SNP calling and read alignment, where the efficiency and accuracy of MSR sketching could provide substantial benefits.

4 Methods

4.1 Reducing input reads

All reads are initially reduced using a Mapping-friendly Sequence Reduction (MSR) provided by Alice. The MSR allows the user to select the order l (default

value of 101) and the compression factor c (default value of 20). The l-mers of the reads are processed through a function g. This function takes an l-mer as input and outputs either a single base, which is appended to the growing reduced read, or an "empty" base ϵ, which is not appended to the growing reduced read.

The function g of the MSR is designed as follows. The l-mer is converted into its canonical form, which is either the original l-mer or its reverse complement if the reverse complement is lexicographically smaller. g then applies to the canonical l-mer a pseudo-random hash function yielding a hash between 0 and 1 [15]. It distinguishes five cases:

- if the hash is smaller than $1/2c$ and the original l-mer is canonical, an A is outputted
- if the hash is smaller than $1/2c$ and the original l-mer is not canonical, a T is outputted
- if the hash is between $1/2c$ and $1/c$ and the original l-mer is canonical, a C is outputted
- if the hash is between $1/2c$ and $1/c$ and the original l-mer is not canonical, a G is outputted
- if the hash is between $1/c$ and 1, ϵ is outputted

This MSR is combined with a classic homopolymer compression process that occurs before all the reads are sketched, at the very beginning of the process, to reduce the error rate of the reads.

4.2 Assembling reduced reads

Many existing short-read and long-read assemblers were tested to assemble reduced reads, but they did not yield very convincing results, especially to separate haplotypes. We believe this is due to reduced reads having slightly different properties compared to regular sequencing reads of equivalent length. For example, errors tend to cluster when $c \cdot l >> 1$. Most assemblers did not manage to assemble at all the reduced reads.

To address this issue, we developed a simple custom assembler that consists of three steps.

1. Generate an unitig graph with a k-mer length of 31, discarding all k-mers seen only once. This is done with BCALM2 [7] (Figure 1a)
2. Simplify the graph by removing tips and bubbles composed of k-mers seen fewer than five times, a classic procedure in assemblers, as used for example in [11, 20, 2] (Figure 1b). If several low-coverages bubbles are situated at a distance less than $10 \cdot k$, they are deleted only when –single-genome mode is activated, as they could represent a rare haplotype.

3. The final step is to untangle the graph to improve contiguity and duplicates unitigs that are present multiple times in the genome, following the procedure of Unicycler [36]. More precisely, all reads are first aligned on the graph (Figure 1 2c). Contigs for which all reads align consensually forming a single path on both sides of the contigs are considered *single-copy contigs*. When two single-copy contigs are linked by a set of reads, the contigs on the path between the two single-copy contigs are duplicated to form a single, long, single-copy contig (Figure 1 2d).

4.3 Recovering the uncompressed assembly

The compressed assembly represents the reduced version of the final assembly. Inflating this reduced version back to the full assembly is not straightforward, as the MSR reduction function is not invertible.

Our method involves three steps:

- creating an inventory of k-mers that tile the compressed assembly, using a k-mer size of 31 by default. For example, two 3-mers that tile the sequence "ACCGTT" are "ACC" and "GTT";
- re-running the MSR on all original reads, and each time a tiling k-mer is produced, record the corresponding uncompressed sequence. For example, we can record that "ACC" corresponds to "GTCGCATGACTGAT" and "GTT" to "TCCGACTCATCAGA"; and finally
- reconstructing the full assembly by concatenating the uncompressed sequences of the tiling k-mers, which would yield in our example "GTCGCATGACTGATCCGACTCATCAGA".

4.4 Choice of parameters

We experimented with different parameter choices for the compression factor and the order of reduction on the Zymobiomics Gut Microbiome Standard dataset to understand how these parameters influence the final assembly.

We conducted two experiments: one to assess the effects of the order and another to assess the effect of the compression factor. First, we tested compression factors of 100, 50, 20, 10, and 5 with an order of 101. Second, we tested orders of 11, 21, 51, 101, and 201 with a compression factor of 10.

The variation of these parameters primarily impacted the completeness of the resulting assemblies and the run-times of the pipelines, while their accuracy, duplication ratio, and contiguity remained equivalent.

As expected, the run-time increased with the compression factor, as there was more data to assemble. This is illustrated in Figure 6.

Compressing more the data also had a positive impact on the completeness and contiguity of the five highly similar *E. coli* strains (Figure 6). When investigating the 27-mer completeness (not shown), all assemblies had a similar amount of missing 27-mers. Hence, the main difference explaining the difference

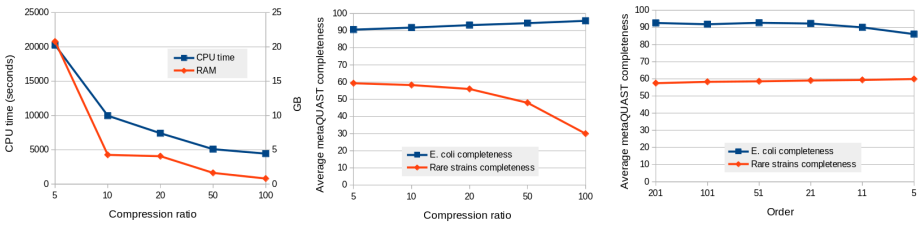


Figure 6: Variation of resource usage and metaQUAST completeness of the Zymobiomics gut microbiome standard assemblies with different compression factor and orders. "Rare strains" refer to *C. albicans*, *S. cerevisiae*, and *S. enterica*. The completeness displayed are arithmetic means of the completeness of the different genomes of the categories.

in completeness was that repeated regions were more shrunk when the data were more compressed, which helped to assemble repeated regions closer to their true multiplicity and thereby improved contiguity.

However, the results represented in Figure 6 show that compression negatively impacted the completeness of the *C. albicans*, *S. cerevisiae*, and *S. enterica* genomes, which had relatively low coverage (see Supplementary Table 1 for the coverages). This is because the assembly algorithm requires a sufficient number of error-free 31-mers in the compressed reads to produce a complete assembly. An error-free compressed 31-mer corresponds to an error-free uncompressed sequence of average length $31 * c + l$ without errors. Therefore, as the compression factor c increases, the number of correct 31-mers in the reads decreases. For genomes with low coverage, high compression can result in the loss of precious 31-mers, leading to insufficient coverage of some regions, hindering their assembly.

The order was found to have relatively little impact on the resulting assemblies. The only significant effect observed was when the order decreased to 11 and 5, where l/c approached or fell below 1. In these cases, the assembler began collapsing highly similar sequences, leading to a decrease in the completeness of the five *E. coli* strains. Despite the fact that the error rate scales approximately linearly with l, increasing l did not have a significant negative impact on the completeness of the assemblies. This is because the errors in the compressed reads cluster in increasingly large clusters, but the error-free regions between these clusters diminish in size only slowly with l.

4.5 Data access & reproducibility

Alice is freely available on github at github.com/RolandFaure/alice-asm. All the datasets used for benchmarking Alice are available publicly, under accession numbers SRR13128013 for the Zymobiomics Gut Microbiome Standard, SRR28996637 for the human gut microbiome dataset, ERR15289804 for the soil and BioProject PRJNA1335825 for *Adineta vaga*. Zymo-HiFi

mock reference genomes are available at https://s3.amazonaws.com/zymo-files/BioPool/D6331.refseq.zip

Assemblies were run with Alice-asm version 0.6.41, hifiasm 0.24.0-r702, Flye 2.9.5-b1801, metaMDBG 1.0, LJA commit 99f93262c. All assemblies and command lines used are available in Zenodo, DOI 10.5281/zenodo.17179435.

5 Acknowledgments

We acknowledge the GenOuest bioinformatics core facility (https://www.genouest.org) for providing the computing infrastructure. The programs Tablet [25] and Bandage [35] were used to visualize data while developing Alice. HiFi sequencing of *Adineta vaga* was funded by the Horizon 2020 research and innovation program of the European Union under the Marie Sklodowska-Curie grant agreement No 764840 (ITN IGNITE, www.itn-ignite.eu) to JFF. The DNA extracts used for sequencing were kindly provided by Karine Van Doninck and Julie Virgo.

We thank Rayan Chikhi for his proofreading and advice.

References

[1] Anton Bankevich et al. "Multiplex de Bruijn graphs enable genome assembly from long, high-fidelity reads". In: *Nature biotechnology* 40.7 (2022), pp. 1075–1081.

[2] Anton Bankevich et al. "SPAdes: A new genome assembly algorithm and its applications to single-cell sequencing". In: *Journal of Computational Biology* 19.5 (2012), pp. 455–477. ISSN: 10665277. DOI: 10.1089/cmb.2012.0021.

[3] Gaëtan Benoit et al. "High-quality metagenome assembly from long accurate reads with metaMDBG". In: *Nature Biotechnology* (Jan. 2024), pp. 1–6. DOI: 10.1038/s41587-023-01983-6.

[4] Luc Blassel, Paul Medvedev, and Rayan Chikhi. "Mapping-friendly sequence reductions: Going beyond homopolymer compression". In: *Iscience* 25.11 (2022).

[5] Lewis Carroll. *Alice's Adventures in Wonderland*. London: Macmillan and Co., 1865.

[6] Haoyu Cheng et al. "Haplotype-resolved *de novo* assembly using phased assembly graphs with hifiasm". In: *Nature Methods* (2021), pp. 1–6.

[7] Rayan Chikhi, Antoine Limasset, and Paul Medvedev. "Compacting de Bruijn graphs from sequencing data quickly and in low memory". In: *Bioinformatics* 32.12 (2016), pp. i201–i208.

[8] Chen-Shan Chin and Asif Khalak. "Human genome assembly in 100 minutes". In: *BioRxiv* (2019), p. 705616.

[9] Alex Chklovski et al. “CheckM2: a rapid, scalable and accurate tool for assessing microbial genome quality using machine learning”. In: *Nature Methods* 20 (July 2023), pp. 1–10. DOI: 10.1038/s41592-023-01940-w.

[10] Robert Edgar. “Syncmers are more sensitive than minimizers for selecting conserved k-mers in biological sequences”. In: *PeerJ* 9 (Feb. 2021), e10805. DOI: 10.7717/peerj.10805.

[11] Barış Ekim, Bonnie Berger, and Rayan Chikhi. “Minimizer-space de Bruijn graphs: Whole-genome assembly of long reads in minutes on a personal computer”. In: *Cell Systems* 12 (Sept. 2021). DOI: 10.1016/j.cels.2021.08.009.

[12] EMBL-EBI. *ENA website*. https://www.ebi.ac.uk/ena/browser/about/statistics. Accessed: 2025-09-15.

[13] Xiaowen Feng et al. “Metagenome assembly of high-fidelity long reads with hifiasm-meta”. In: *Nature Methods* 19 (June 2022), pp. 1–4. DOI: 10.1038/s41592-022-01478-3.

[14] Jean-François Flot et al. “Genomic evidence for ameiotic evolution in the bdelloid rotifer *Adineta vaga*”. In: *Nature* 500.7463 (2013), pp. 453–457. DOI: 10.1038/nature12326.

[15] Parham Kazemi et al. “ntHash2: recursive spaced seed hashing for nucleotide sequences”. In: *Bioinformatics* 38.20 (2022), pp. 4812–4813.

[16] Marek Kokot, Maciej Długosz, and Sebastian Deorowicz. “KMC 3: counting and manipulating k-mer statistics”. In: *Bioinformatics (Oxford, England)* 33 (Jan. 2017). DOI: 10.1093/bioinformatics/btx304.

[17] Mikhail Kolmogorov et al. “Assembly of long, error-prone reads using repeat graphs”. In: *Nature Biotechnology* 37.5 (2019), pp. 540–546. DOI: 10.1038/s41587-019-0072-8.

[18] Mikhail Kolmogorov et al. “metaFlye: scalable long-read metagenome assembly using repeat graphs”. In: *Nature Methods* 17 (Nov. 2020), pp. 1–8. DOI: 10.1038/s41592-020-00971-x.

[19] Heng Li. “Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences”. In: *Bioinformatics* 32.14 (2016), pp. 2103–2110.

[20] Antoine Limasset, Jean-François Flot, and Pierre Peterlongo. “Toward perfect reads: self-correction of short reads via mapping on de Bruijn graphs”. In: *Bioinformatics (Oxford, England)* 36 (Feb. 2019). DOI: 10.1093/bioinformatics/btz102.

[21] Vijini Mallawaarachchi and Yu Lin. “MetaCoAG: Binning Metagenomic Contigs via Composition, Coverage and Assembly Graphs”. In: *Research in Computational Molecular Biology*. Ed. by Itsik Pe’er. Cham: Springer International Publishing, 2022, pp. 70–85. ISBN: 978-3-031-04749-7.

[22] Mosè Manni et al. "BUSCO Update: Novel and Streamlined Workflows along with Broader and Deeper Phylogenetic Coverage for Scoring of Eukaryotic, Prokaryotic, and Viral Genomes". In: *Molecular Biology and Evolution* 38 (July 2021). DOI: 10.1093/molbev/msab199.

[23] Daniel Mapleson et al. "KAT: A K-mer Analysis Toolkit to quality control NGS datasets and genome assemblies". In: *Bioinformatics (Oxford, England)* 33 (Oct. 2016). DOI: 10.1093/bioinformatics/btw663.

[24] Alla Mikheenko, Vladislav Saveliev, and Alexey Gurevich. "MetaQUAST: evaluation of metagenome assemblies". In: *Bioinformatics* 32.7 (2016), pp. 1088–1090.

[25] Iain Milne et al. "Tablet - Next Generation Sequence Assembly Visualization". In: *Bioinformatics (Oxford, England)* 26 (Dec. 2009), pp. 401–2. DOI: 10.1093/bioinformatics/btp666.

[26] Shaojun Pan, Xingming Zhao, and Luis Pedro Coelho. "SemiBin2: self-supervised contrastive learning leads to better MAGs for short- and long-read sequencing". In: *Bioinformatics (Oxford, England)* 39 (June 2023), pp. i21–i29. DOI: 10.1093/bioinformatics/btad209.

[27] Daniel M Portik et al. "Highly accurate metagenome-assembled genomes from human gut microbiota using long-read assembly, binning, and consolidation methods". In: *bioRxiv* (2024), pp. 2024–05.

[28] Will P. M. Rowe. "When the levee breaks: a practical guide to sketching algorithms for processing the flood of genomic data". In: *Genome Biology* 20.1 (2019), p. 199. DOI: 10/gf8bfj. (Visited on 09/16/2019).

[29] Jue Ruan and Heng Li. "Fast and accurate long-read assembly with wtdbg2". en. In: *Nature Methods* 17.2 (2020), pp. 155–158. ISSN: 1548-7105. DOI: 10.1038/s41592-019-0669-3.

[30] Kishwar Shafin et al. "Nanopore sequencing and the Shasta toolkit enable efficient de novo assembly of eleven human genomes". In: *Nature biotechnology* 38.9 (2020), pp. 1044–1053.

[31] Paul Simion et al. "Chromosome-level genome assembly reveals homologous chromosomes and recombination in asexual rotifer *Adineta vaga*". In: *Science Advances* 7.41 (2021), eabg4216. DOI: 10.1126/sciadv.abg4216. (Visited on 10/11/2021).

[32] Paul Simion et al. "Chromosome-level genome assembly reveals homologous chromosomes and recombination in asexual rotifer Adineta vaga". In: *Science advances* 7 (Oct. 2021), eabg4216. DOI: 10.1126/sciadv.abg4216.

[33] Fredrik Tegenfeldt et al. "OrthoDB and BUSCO update: annotation of orthologs with wider sampling of genomes". In: *Nucleic acids research* 53 (Nov. 2024). DOI: 10.1093/nar/gkae987.

[34] Thea Van Rossum et al. "Diversity within species: interpreting strains in microbiomes". In: *Nature Reviews Microbiology* 18 (June 2020), pp. 1–16. DOI: 10.1038/s41579-020-0368-1.

[35] Ryan R Wick et al. “Bandage: interactive visualization of de novo genome assemblies”. In: *Bioinformatics* 31.20 (2015), pp. 3350–3352.

[36] Ryan R. Wick et al. “Unicycler: Resolving bacterial genome assemblies from short and long sequencing reads”. en. In: *PLOS Computational Biology* 13.6 (June 2017). Publisher: Public Library of Science, e1005595. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005595. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005595 (visited on 10/08/2021).

[37] Wenjuan Yu et al. “Comprehensive assessment of 11 de novo HiFi assemblers on complex eukaryotic genomes and metagenomes”. In: *Genome Research* 34 (Mar. 2024). DOI: 10.1101/gr.278232.123.