Unnamed: A new method for the production of synthetic long reads

Pierre MORISSE[1], Thierry LECROQ[1] and Aranaud LEFEBVRE[1]
Laboratory, Address, zip code, Town, Country

Corresponding author: pierre.morisse2@univ-rouen.fr

Abstract *Since a few years, long reads sequencing technologies are being developed and allow the solving of assembly problems for large and complex genomes that were, until then, unsolvable with the use of short reads sequencing technologies alone. However, despite the fact they can reach lengths of tens of kpbs, these long reads are also very noisy, and can reach an error rate as high as 30%. The vast majority of these errors being insertions and deletions, classical error correction tools developed for short reads, which mainly focus on mismatch errors, are not effective for correcting long reads. NaS, developed in 2015, uses these noisy long reads as templates to produce assemblies of related accurate short reads, thus yielding accurate synthetic long reads as corrections for the templates. We present Unnamed, a new method for the production of synthetic long reads, that gets rid of the bottleneck step from NaS. Our experiments show that, while producing comparable results both in terms of length and accuracy of the synthetic long reads, Unnamed is several orders of magnitude faster than NaS.*

Keywords NGS, correction, assembly

1 Introduction

Since a few years, long reads sequencing technologies are being developed, and allow the solving of assembly problems for large and complex genomes that were impossible with the use of short reads sequencing technologies alone. The two major actors of these long reads sequencing technologies are Pacific Biosciences and Oxford Nanopore, which, with the release of the MinION device, allowed a low-cost and easy long read sequencing.

However, even though long reads can reach lengths of tens of kbp, they also reach a very high error rate of around 15% for Pacific Biosciences, and up to 30% for Oxford Nanopore, the vast majority of these errors being insertions and deletions. Correcting these long reads before using them to solve assembly problems is therefore mandatory. Many methods are available for short reads correction, but these methods are not applicable to long reads, on the one hand because of their much higher error rate, and on the other hand, because most of the error correction tools for short reads focus on substitution errors, the dominant error type in Illumina data, whereas insertions and deletions are more common in long reads.

Recently, several methods for long read correction have been developed. These methods can be divided into two main categories: either the long reads are selfcorrected by aligning them against each other (HGAP [1], Sprai (CITE) PBcR (CITE)), or either a hybrid strategy is adopted, in which the long reads are corrected with the help of accurate short reads (LSC [2], proovread [3], CoLoRMap [4]). **De Bruijn graphs based methods, where the long reads are mapped on the graph, and erroneous regions corrected by traversing its paths, also started to develop recently**, in the hybrid case (LoRDEC [5], Jabba [6]), as well as in the non-hybrid case (LoRMA [7]).

NaS [8], however, instead of directly correcting the long reads, uses them as templates to produce synthetic long reads from an assembly of short reads. The short reads are first mapped on the long reads, and then against each other, in order to obtain a subset of related short reads for each template. A synthetic long read is thus obtained and used as the correction of a given template by assembling the subset of short reads associated to it. A complete overview of NaS is given in Section 2.

In this paper, we present a new long reads hybrid correction method that combines both the main idea from NaS to produce synthetic long reads, and a graph approach, in order to get rid of the time consuming step of aligning all the short reads against each other. We indeed focus on a seed-and-extend approach where the seeds, found by mapping the short reads on the long reads, are used as anchor points and linked together with perfectly overlapping k-mers from the short reads, found with the help of PgSA [9], which allows to simulate a hybrid structure between a de Bruijn graph [10] and an overlap graph

(CITE?). A complete overview of PgSA and of the way it allows to simulate such a structure is given in Section 3.

Our experiments show that, while producing comparable results both in terms of length and accuracy of the synthetic long reads, Unnamed is few times faster than NaS, and also yields better assembly results than others state-of-the-art error correction methods.

2 NaS Overview

NaS is a hybrid method for the error correction of long reads that, unlike other methods, uses long reads as templates rather than directly correcting them. Short reads are mapped both on these templates and against each other in order to gather different subsets of short reads, each related to one given template. Once a subset of short reads is obtained, it is assembled, and the produced contig is used as a correction for the related template. More precisely, a synthetic long read is produced as follows.

First, the short reads are aligned on the template long read using BLAT [11], **(or LAST [12] in sensitive mode?)**, in order to find seeds, which are short reads that correctly align with the template. Then, once these seeds **have been** found, all the short reads are aligned against each other, and **similar reads, which are reads that share a certain number of non-overlapping k-mers with the seeds**, are recruited with the help of Commet [13]. Finally, the obtained subset of short reads is assembled using Newbler (CITE), and a contig is produced, and used as the correction of the initial template long read.

Usually, a single contig is produced, but in repeated regions, a few bad reads can be recruited and therefore yield erroneous contigs that must not be associated with the template. **To address this issue, and produce a single contig from multiple ones**, NaS explicitly builds the contig-graph, weighted with the seeds coverage of the contigs. Once the graph is built, the path with the highest total weight is chosen with the Floyd-Warshall algorithm **(CITE?)**, and contigs along that path are assembled to generate the final synthetic long read. Finally, the consistency of the synthetic read is checked by aligning initial short reads and detecting gap of coverage.

The reads recruitment is the most crucial step of the method, as it allows to retrieve short reads corresponding to low quality regions of the template long read. However, this step is also the bottleneck of the whole NaS pipeline, as it is responsible for 70% of the total runtime on average.

NaS is able to generate synthetic long reads up to 60 kbps, that align entirely to the reference genome with no error, and that span repetitive regions. On average, the accuracy of the synthetic long reads produced by NaS reaches 99.97%, without any significant length drop compared to the input long reads.

3 PgSA Overview

PgSA, along with GkA [14] and CGkA [15] are data structures that allow the indexing of a set of reads, in order to answer the following queries, for a given string f:

1. In which reads does f occur?
2. In how many reads does f occur?
3. What are the occurrence positions of f?
4. What is the number of occurrences of f?
5. In which reads does f occur only once?
6. In how many reads does f occur only once?
7. What are the occurrence positions of f in the reads where it occurs only once?

In these queries, f can be given either as a sequence of DNA symbols, or as a couple of numbers, representing respectively a read ID, and the start position of f in that read.

As previously mentioned, in order to answer these queries, an index of the reads has to be built. To do so, PgSA first computes the overlaps between the reads, and merges the reads that do overlap, thus obtaining a pseudogenome, shorter than the naive concatenation of the whole **set of reads**. Then, an auxiliary array is built to allow the retrieval of the reads from the original set in the pseudogenome. Each record of this array associates a read ID in the original **set of reads** to a read offset in the pseudogenome, and contains a flag data that brings complementary information about the read and that will be used to handle the requests.

As the reads are overlapped during the pseudogenome computation, and the auxiliary array does not record

any information about their lengths, PgSA will only allow the indexing and querying of a set of reads of same length. However, unlike its peers GkA and CGkA, PgSA does not set the length of f at compilation time, and thus **supports querying for variable lengths of** f without any need to recompute the index, which is why we chose this data structure over the two others.

This way, indexing the reads from a given set of reads, and looping over the third request, allows PgSA to compute perfect overlaps of variable length between the reads, thus simulating an overlap graph. In the same fashion, indexing the k-mers from the set of reads, and looping over the third request, fixing the length of the queries strings as $k-1$, allows PgSA to compute perfect overlaps of length $k-1$ between the k-mers, thus simulating a de Bruijn graph. However, indexing the k-mers from a set of reads, and looping over the third request, of course, also allows PgSA to compute perfect overlaps of variable length between the different k-mers, thus simulating a hybrid structure between a de Bruijn graph and an overlap graph. To the best of our knowledge, this the first time such a structure is mentioned.

Ajouter une figure pour illustrer notre graphe ?

4 Our method

Our method, like NaS, aims to use erroneous long reads as templates, and produce synthetic long reads from an assembly of short reads related to the templates. However, our main objective is to get rid of the time consuming step of reads recruiting, that requires the mapping of all the short reads against each other. To do so, we focus on a seed-and-extend approach where k-mers from the short reads are indexed with PgSA, which, as described before, allows to simulate a hybrid structure between a de Bruijn graph and an overlap graph, and directly assembled to extend and link together the seeds, used as anchor points. The workflow of our method is summarized in Figure 1, and its four main steps are described below.

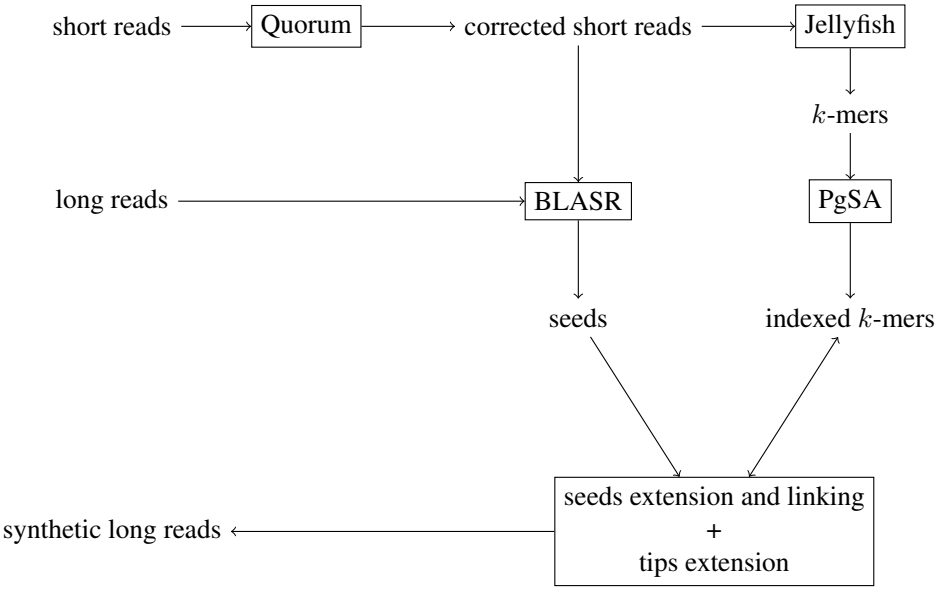

Fig. 1. Our method's workflow. First, the short reads are corrected in order to get rid of as much sequencing errors as possible. Then, all the k-mers from the corrected short reads (and from their reverse complements) are obtained with Jellyfish, and indexed with PgSA. Short reads are aligned on long reads with BLASR to find seeds, and the indexed k-mers, **simulating a graph**, are queried to extend and link together the seeds, **used as anchor points**. Finally, the sequences obtained from the seeds linkage are extended in both directions to reach the initial templates borders, and the synthetic long reads are output.

4.1 Short reads correction and indexing

Even though short reads are very accurate prior to any correction, as we seek to use their k-mers **to simulate a graph, and compute perfect overlaps to extend the seeds**, we need to get rid of as much sequencing errors

as we can in this data. We thus correct the short reads with the help of Quorum [16], which is able to provide a good raise of the accuracy in very little time. Once corrected, the k-mers from the short reads and from their reverse complements are extracted with Jellyfish [17], and indexed with PgSA, before being queried to extend and link the seeds together during the following steps.

4.2 Seeds retrieving and merging

Like with NaS, the seeds are found by mapping the corrected short reads on the long reads, used as templates. This is done with the help of BLASR [18], an alignment tool specifically designed to align long reads dominated by insertion and deletion errors. Then, for each template, two phases of analyze and merging are applied to the associated seeds. First, if the mapping positions of a given couple of seeds imply that they overlap on the template over a sufficient length, their assumed overlapping sequences are compared, and the two seeds are merged accordingly. If the mapping positions indicate the two seeds do overlap on the template, but not over a sufficient length, or if the assumed overlapping sequences do not coincide, then, only the seed with the best alignment score is kept. Then, once seeds having overlapping mapping positions have been merged or filtered out, sequence overlaps between consecutive seeds are computed. As in the previous step, if a given seed overlaps the following one over a sufficient length, the two seeds are merged.

4.3 Seeds linking

Once seeds have been found and merged for all of the templates, Unnamed processes each template separately and attempts to link together every couple of seeds mapped on the template by traversing the simulated graph. The rightmost k-mer of the left seed (source) and the leftmost k-mer of the right seed (destination) are used as anchor points, and the source is extended with perfectly overlapping k-mers from the short reads, found by following the paths of the graph, until the destination is reached. When facing branching paths, every possible path is explored with the use of backtracking, to find the one that will allow correct linking of the source to destination. However, to avoid useless important runtimes and intensive computations, as short reads from a different region of the reference genome can align on the templates and be used as seeds, due to the high error rate of long reads, a threshold on the maximum number of backtracks is set. Moreover, Unnamed traverses the graph in decreasing order of the overlaps lengths, which means that at branching paths, edges representing longer overlaps are always explored before exploring those representing shorter overlaps, if the source and the destination could not be linked and needed backtracking, for instance. Edges representing shorter overlaps can thus be explored until the defined minimum overlap length is reached.

If the previously defined threshold on the maximum number of backtracks is reached, and no path has been found to link the source to the destination, then the linking is given up. When such a situation occurs, two different cases have to be taken into account. In the first case, if no seeds have been linked so far, the current source is simply ignored, and a new linking is computed for the next couple of seeds. In the second case, if seeds have already been linked previously, the source seed remains the same, the destination seed that couldn't be reached is ignored, and the destination is defined as the next seed for the next linking iteration. As this process of skipping a seed in the middle of the template can provoke an important number of failed linking attempts, if erroneous seeds are present in great proportion on the template, a threshold on the maximum number of seeds skipping is set. Once this threshold is reached, if all the seeds could not be linked, the synthetic long read corresponding to the seeds linked so far is output, and the following seeds are ignored.

Ajouter une figure pour les deux cas de skip de seeds ?

4.4 Tips extension

Finally, it is obvious that seeds do not always map right at the beginning and until the end of the templates. Thus, in order to get as close as possible to the original template's length, once all the seeds of a given template have been linked, we keep on traversing the graph and extending the produced synthetic long read, on the left of the leftmost seed, and on the right of the rightmost seed, until we reach the template's borders, or a branching path. Indeed, in the case of tips extension, when facing a branching

path, Unnamed has no clue as to which path to chose and continue the extension with, nor any anchor points, unlike when it attempts to link two seeds. Therefore, the extension is simply stopped when such a situation occurs.

5 Results and discussion

We compare the quality of our synthetic long reads with those produced by NaS, and also with the corrected long reads produced by others state-of-the-art hybrid correction methods, namely CoLoRMap and Jabba. **We compare the results both in terms of alignment identity of the corrected reads on the reference genome, and of quality of the assembly that could be generated from these reads.**

5.1 Parameters

We ran multiple rounds of correction with Unnamed on the different datasets to experiment with the parameters, and find the combination that would produce the best results. Thereby, we found that a k-mer value of 55 for the graph construction yielded the best compromise between identity, genome coverage, and average length of the output synthetic long reads. The minimum overlap length allowed to explore an edge during the graph traversal was set to $k - 5$, as decreasing it more yielded unsatisfying results, and increasing it would make our graph closer to an actual de Bruijn graph than to the hybrid graph it's actually supposed to be. The maximum number of backtracks was set to 1,125, as decreasing it more drastically impacted the quality of the correction, and increasing it, even to very large values, barely yielded better results, but greatly increased the runtime. For the same reason, the maximum number of seeds skipping was set to 10. For the mapping of the short reads on the long reads, BLASR was used with default parameters except for bestn, that was set to 30 instead of 10. Yet again, increasing this parameter to larger values only impacted runtime, and did not improve the correction results enough to be interesting, while decreasing it induced a drop of the number of corrected reads. Finally, GNU Parallel [19] was used to allow Unnamed to run on multiple processes. All the others tools were run with default parameters, except for the number of threads / processes.

5.2 Datasets

As we mainly seek to compare our results with NaS, the only other tool that does not directly correct the input long reads, we use the same data to allow a better comparison. This data is composed of both long Oxford Nanopore and short Illumina reads for three different genomes: *Acinetobacter baylyi*, *Escherichia coli*, and *Saccharomyces cerevisae*. Details are given **in** Table 1.

Dataset	Reference genome				Oxford Nanopore data			Illumina data		
	Name	Strain	Reference sequence	Genome size	Number of reads	Average length	Coverage	Number of reads	Read length	Coverage
ADP1	*Acinetobacter baylyi*	ADP1	CR543861	3.6 Mbp	89,011	4,285	103x	900,000	250	50x
Ecoli	*Escherichia coli*	K-12 substr. MG1655	NC_000913	4.6 Mbp	22,270	5,999	29x	775,500	300	50x
Yeast	*Saccharomyces cerevisae*	W303	lala	12 Mbp	205,923	5,698	98x	2,500,000	250	50x

Tab. 1. Description of the datasets used in our experiments. Both MinION and Illumina data are available from the Genoscope website: http://www.genoscope.cns.fr/externe/nas/datasets.html **TODO : MinION coverage à revoir.**

5.3 Mapping quality

We recall that Oxford Nanopore technologies can yield both 1 direction and 2 directions, longer and more accurate, reads. **We analyze both types of reads separately, to be able to draw a better comparison of the different correction methods.** Results on the 1D reads are given **in** Table 2, and results on the 2D reads are given **in** Table 3. The raw long reads were aligned using Last [12], and the corrected long reads were aligned using BWA [20]. We discuss and analyze the results below.

We notice that the synthetic long reads produced from 1D templates by our method are longer than those produced by NaS, both in fast or sensitive mode. This is likely caused by the last extension step, when Unnamed extends the tips of the synthetic long reads as much as possible, until the borders of the template or an ambiguity is reached. As a result, the cumulative size of these reads is also greater than the one obtained with NaS.

With 2D templates, however, synthetic long reads produced by NaS are longer than those produced by our method, even in sensitive mode. We can suppose this is due to the fact that fragmented reads are present in greater proportion on the synthetic long reads produced from 2D reads (6% of the synthetic reads produced from

Dataset	Method	Number of reads	Average length	Cumulatize size	Number of aligned reads	Average identity	Error free reads	Genome coverage	Runtime
ADP1	**Original**	70,314	2,530	177,869,033	373	3.84 %	0	N.A.	N.A.
	CoLoRMap	70,314	2,538	178,467,124	2,975	15.57	0	79.01	TODO
	Jabba	2,337	9,304	21,743,004	2,332	99.67	2,286	96.86	TODO
	NaS (fast)	8,219	4,514	37,099,564	8,219	99.92 %	8,025	99.96 %	TODO
	NaS (sensitive)	12,053	6,338	76,388,104	12,053	99.89 %	11,665	100 %	TODO
	Our method	7,425 (249 fragmented)	10,250	78,739,767	7,676	99.56 %	7,081	99.93 %	TODO
Yeast	**Original**	158,896	5,525	877,844,119	1,547	2.55 %	0	N.A.	N.A.
	CoLoRMap	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx
	Jabba	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx
	NaS (fast)	44,446	5,178	230,158,404	44,363	99.63 %	37,607	98.01 %	TODO
	NaS (sensitive)	56,942	6,221	354,255,034	56,837	99.55 %	47,122	98.68 %	TODO
	Our method	TODO	TODO	TODO	TODO	TODO	TODO	TODO	TODO

Tab. 2. Results on 1D MinION reads.

Dataset	Method	Number of reads	Average length	Cumulatize size	Number of aligned reads	Average identity	Error free reads	Genome coverage	Runtime
ADP1	**Original**	18,697	10,884	203,496,742	13,636	37.07 %	0	N.A.	N.A.
	CoLoRMap	18,694	11,186	209,143,685	xxx	xxx	xxx	xxx	xxx
	Jabba	15,139	10,408	157,566,734	15,139	99.36	14,604	99.79	TODO
	NaS (fast)	15,844	11,084	175,607,625	15,844	99.78 %	14,959	100 %	TODO
	NaS (sensitive)	16,439	11,871	195,138,674	16,439	99.79 %	15,525	100 %	TODO
	Our method	15,575 (984 fragmented)	10 562	178,222,404	16,874	99.54 %	14,791	100 %	TODO
Ecoli	**Original**	22,270	5,999	133,607,392	21,318	39.08 %	0	N.A.	N.A.
	CoLoRMap	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx
	Jabba	22,065	5,794	127,848,525	22,065	99.81	21,850	99.41	TODO
	NaS (fast)	21,818	7,926	172,918,739	21,818	99.86 %	20,383	100 %	TODO
	NaS (sensitive)	22,144	8,307	183,958,832	22,144	99.86 %	20,627	100 %	TODO
	Our method	TODO	TODO	TODO	TODO	TODO	TODO	TODO	TODO
Yeast	**Original**	47,027	6,285	295,545,390	15,192	8.75 %	0	N.A.	N.A.
	CoLoRMap	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx
	Jabba	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx
	NaS (fast)	27,347	7,123	196,167,951	27,301	99.52 %	22,181	97.82 %	TODO
	NaS (sensitive)	28,490	7,866	224,096,554	28,451	99.47 %	22,693	98.61 %	TODO
	Our method	TODO	TODO	TODO	TODO	TODO	TODO	TODO	TODO

Tab. 3. Results on 2D MinION reads.

2D templates are fragmented, whereas only 3% of those produced from 1D templates are), and that therefore, they do induce a drop of the average length.

Synthetic long reads produced by our method tend to have a lower identity than those produced by NaS. This is probably caused by ???

Finally, it is interesting to mention that (here, we compare the lengths of the longest reads that were obtained with NaS / Unnamed)
Say that our reads are shorter because NaS can recruit reads outside of the template and we can't?

5.4 Assembly quality

All the corrected long reads datasets previously described were assembled using Canu [21], without the correction step, and with the following parameters: OvlMerSize=17, MhapMerSize=17, OvlMerDistinct=0.9925, OvlMerTotal=0.9925, correctedErrorRate=0.15. Results are given **in** Table 4 and discussed below.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the

Dataset	Method	Number of reads	Expected contigs	Number of contigs	Genome coverage	Identity
ADP1	**Original**	89,011	N.A.	N.A.	N.A.	N.A.
	CoLoRMap	xxx	xxx	xxx	xxx	xxx
	Jabba	xxx	xxx	xxx	xxx	xxx
	NaS (fast)	24,063	1	1	100 %	99.98 %
	NaS (sensitive)	28,492	1	1	100 %	99.99 %
	Our method	TODO	1	TODO	TODO	TODO
Ecoli	**Original**	22,270	N.A.	N.A.	N.A.	N.A.
	CoLoRMap	xxx	xxx	xxx	xxx	xxx
	Jabba	xxx	xxx	xxx	xxx	xxx
	NaS (fast)	21,818	1	1	99,90 %	99.99 %
	NaS (sensitive)	22,144	1	1	100 %	99.99 %
	Our method	TODO	1	TODO	TODO	TODO
Yeast	**Original**	205,923	N.A.	N.A.	N.A.	N.A.
	CoLoRMap	xxx	xxx	xxx	xxx	xxx
	Jabba	xxx	xxx	xxx	xxx	xxx
	NaS (fast)	71,793	30	124	97.27 %	99.78 %
	NaS (sensitive)	85,432	30	120	97.04 %	99.79 %
	Our method	TODO	30	TODO	TODO	TODO

Tab. 4. Assembly results.

original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

6 Conclusions

We developed a new hybrid strategy for the correction of long reads, that, like NaS, uses longs reads as templates and focuses on the production of synthetic long reads by an assembly of short reads related to a given template, rather than on the direct correction of the input long reads. Also, rather than aligning the short reads against each other in a recruiting step, and then assembling the obtained subset of reads with already existing tools, like NaS, we focused on a seed-and-extend approach and developed a brand new idea of simulating a hybrid structure between a de Bruijn graph and an overlap graph, with the help of PgSA, to directly assemble the k-mers from the short reads. For this aim, we used as seeds short reads that aligned perfectly on the input long reads, and used their k-mers as anchor points on the graph, to allow their extension and their linking, and thus, produce synthetic long reads.

We tested this new method and compared it with NaS, CoLoRMap and Jabba on three different genomes, namely *Acinetobacter baylyi*, *Escherichia coli*, and *Saccharomyces cerevisae*. On these three genomes, Unnamed yielded results that compared well with NaS, while being 2-3 times faster, CoLoRMap produced corrected reads of poor quality, and Jabba, while being the fastest tool, produced accurate corrected reads that however weakly covered the references genomes. As a result, only synthetic long reads produced by NaS and Unnamed were able to be assembled into a decent number of contigs.

The development of this method showed that, when having anchor points, the previously introduced hybrid graph can prove useful to produce an assembly of short reads k-mers, as it seems to yield better results than a classical de Bruijn graph. For future works, it could be interesting to focus more on this structure, and further optimize it to reduce runtime, or even try to get rid of the anchor points, in order to produce a proper assembly tool.

Acknowledgements

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

References

[1] Chen-Shan Chin, David H Alexander, Patrick Marks, Aaron A Klammer, James Drake, Cheryl Heiner, Alicia Clum, Alex Copeland, John Huddleston, Evan E Eichler, Stephen W Turner, and Jonas Korlach. Nonhybrid, finished microbial genome assemblies from long-read SMRT sequencing data. *Nature Methods*, 10(6):563–569, 2013.

[2] Kin Fai Au, Jason G. Underwood, Lawrence Lee, and Wing Hung Wong. Improving PacBio Long Read Accuracy by Short Read Alignment. *PLoS ONE*, 7(10):1–8, 2012.

[3] Thomas Hackl, Rainer Hedrich, Jörg Schultz, and Frank Förster. Proovread: Large-scale high-accuracy PacBio correction through iterative short read consensus. *Bioinformatics*, 30(21):3004–3011, 2014.

[4] Ehsan Haghshenas, Faraz Hach, S Cenk Sahinalp, and Cedric Chauve. CoLoRMap: Correcting Long Reads by Mapping short reads. *Bioinformatics*, 32(17):i545–i551, 2016.

[5] Leena Salmela and Eric Rivals. LoRDEC: Accurate and efficient long read error correction. *Bioinformatics*, 30(24):3506–3514, 2014.

[6] G Miclotte, M Heydari, P Demeester, S Rombauts, Y Van de Peer, P Audenaert, and J Fostier. Jabba: hybrid error correction for long sequencing reads. *Algorithms Mol Biol*, 11:10, 2016.

[7] Leena Salmela, Riku Walve, Eric Rivals, and Esko Ukkonen. Accurate selfcorrection of errors in long reads using de Bruijn graphs. *Bioinformatics*, page btw321, 2016.

[8] Mohammed-Amin Madoui, Stefan Engelen, Corinne Cruaud, Caroline Belser, Laurie Bertrand, Adriana Alberti, Arnaud Lemainque, Patrick Wincker, and Jean-Marc Aury. Genome assembly using Nanopore-guided long and error-free DNA reads. *BMC Genomics*, 16:327, 2015.

[9] Tomasz Kowalski, Szymon Grabowski, and Sebastian Deorowicz. Indexing arbitrary-length k-mers in sequencing reads. *PLoS ONE*, 10(7):1–14, 2015.

[10] N.G. de Bruijn. A combinatorial problem. *Proceedings of the Section of Sciences of the Koninklijke Nederlandse Akademie van Wetenschappen te Amsterdam*, 49(1946)(7):758–764, 1946.

[11] W James Kent. BLAT — The BLAST -Like Alignment Tool. *Genome research*, 12:656–664, 2002.

[12] Szymon M Kielbasa, Raymond Wan, Kengo Sato, Szymon M Kiebasa, Paul Horton, and Martin C Frith. Adaptive seeds tame genomic sequence comparison. *Genome Research*, pages 487–493, 2011.

[13] Nicolas Maillet, Guillaume Collet, Thomas Vannier, Dominique Lavenier, and Pierre Peterlongo. Commet: Comparing and combining multiple metagenomic datasets. *Proceedings - 2014 IEEE International Conference on Bioinformatics and Biomedicine*, IEEE BIBM(November):94–98, 2014.

[14] N Philippe, M Salson, T Lecroq, M Leonard, T Commes, and E Rivals. Querying large read collections in main memory: a versatile data structure. *BMC bioinformatics*, 12(1):242, 2011.

[15] V Niko. Scalable and Versatile k -mer Indexing for High-Throughput Sequencing Data. (250345):237–248, 2013.

[16] Guillaume Marçais, James A Yorke, and Aleksey Zimin. QuorUM: An Error Corrector for Illumina Reads. pages 1–13, 2015.

[17] Guillaume Marçais and Carl Kingsford. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, 27(6):764–770, 2011.

[18] Mark J Chaisson and Glenn Tesler. Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory. *BMC bioinformatics*, 13:238, 2012.

[19] Ole Tange. GNU Parallel: The Command-Line Power Tool. 3(1):42–47.

[20] Ruiqiang Li, Chang Yu, Yingrui Li, Tak Wah Lam, Siu Ming Yiu, Karsten Kristiansen, and Jun Wang. SOAP2: An improved ultrafast tool for short read alignment. *Bioinformatics*, 25(15):1966–1967, 2009.

[21] Sergey Koren, Brian P. Walenz, Konstantin Berlin, Jason R. Miller, and Adam M. Phillippy. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *bioRxiv*, page 071282, 2016.