e-RGA: enhanced Reference Guided Assembly of Complex Genomes

Francesco Vezzi[1,2], Federica Cattonaro[2], Alberto Policriti[1,2]

[1]DIMI department of Informatics and Mathematics University of Udine, Udine, Italy, [2]IGA institute of applied genomics, Udine, Italy,

Abstract

Next Generation Sequencing has totally changed genomics: we are able to produce huge amounts of data at an incredibly low cost compared to Sanger sequencing. Despite this, some old problems have become even more difficult, *de novo* assembly being on top of this list. Despite efforts to design tools able to assemble, *de novo*, an organism sequenced with short reads, the results are still far from those achievable with long reads. In this paper, we propose a novel method that aims to improve *de novo* assembly in the presence of a closely related reference. The idea is to combine *de novo* and reference-guided assembly in order to obtain enhanced results.

Introduction

DNA sequencing is becoming cheaper every day. Instruments like the Illumina HiSeq 2000 or Solid 4 System are able to produce higher than 30X coverage of a human genome for less than $10,000. Next Generation Sequencing (NGS) allows sequencing at low cost and at a fraction of the time with respect to the Sanger Method [1]. NGS technologies (Illumina, Roche and Solid, to mention just a few) are capable of producing an enormous amount of raw data (for a complete review, see [2]). The throughput of such instruments is increasing so fast that descriptions of their performance became obsolete in just a few months.

Even though many papers have presented high-quality assemblies based on NGS data (see [3,4]), *de novo* assembly, especially for large-genome eukaryota, is still a ‘holy grail’ [5].

When a completely new organism is to be sequenced, the basic assembly strategy is still the celebrated Whole-Genome Shotgun (WGS) method. A number of tools that aim to perform *de novo* assembly using NGS data have been proposed. Among the most popular, we mention SOAPdenovo [6] and ABySS [7] (see [8] for an updated list). The assemblies produced by tools designed for NGS data are, in general, not comparable in quality with the assemblies produced by instruments like Arachne [9] and those designed for capillary sequencing data. The reason for this is that while, with NGS instruments, coverage is no more a bottleneck, read length, as well as the reduced size of the insertion between paired reads, makes correct assembly and positioning of repeats much more of an issue. *De novo* assembly with short reads is still very difficult [5] and, when assembling complex and repeated genomes, reasonably conservative *de novo* assembly programs are likely to produce collections of highly fragmented contigs. An interesting strategy to improve *de novo* assemblies has been termed “assembly reconciliation”. The goal of assembly reconciliation is to merge the assemblies produced by different tools and to detect possible mis-assemblies [10].

The number of organisms whose genomes have been completely sequenced has been increasing rapidly each year and, for this reason, it is becoming viable to sequence an organism and then to align the sequence against a closely related genome. This strategy goes by the name of *Reference Guided Assembly* (RGA), its main advantage being the fact that, in general, even low coverage is sufficient to yield useful results. RGA consists of two phases: first, all the reads are aligned against the reference genome; then, a consensus sequence is extrapolated. Everywhere the coverage drops to zero, a sequence of Ns is placed. This problem has already been studied in the context of Sanger sequencing. In [11] and [12], two methods were proposed that use reference sequences to assist the assembly of new organisms. The challenge is becoming even more interesting with the advent of next generation sequencers, beginning with the technicalities and the practical considerations involved in the alignment phase.

As a matter of fact, many tools capable of rapidly aligning millions of short reads against a reference genome have been proposed recently. Tools like SOAP2 [13] and rNA [14] are essential for performing reference-guided assembly. The main problem with RGA and NGS is that (essentially for efficiency reasons) mapping algorithms are highly conservative: it is possible to align reads with only a low number of errors and usually without gaps. In other words, we are able to reconstruct the conserved regions, while we cannot reconstruct areas that are divergent and (usually) more interesting. While, for example, there are techniques that use the paired-reads information to identify insertions/deletions (structural variations) [15,16], there is no clear way to reconstruct them.

In [17], a tool (MAIA) has been proposed to integrate multiple *de novo* and reference-guided assemblies. This tool uses the output of different assemblers, and of different reference-guided assemblies obtained with several reference sequences, to improve the final assembly result. MAIA constructs an overlap graph from the pairwise alignments of all the contigs. In large and repetitive genomes, like plant genomes, this step is computationally expensive and could easily lead to a large number of ambiguous or false overlaps.

Velvet's Columbus module tries to improve the assembly results using a reference sequence. In particular, the Columbus module aims to reconstruct candidate structural variations. Again, in this case, as a consequence of the repeats, this approach cannot be used on large, repetitive genomes.

In this paper, we briefly discuss and present results of a novel strategy to assemble genomes in the presence of a related sequence. In particular, we study how to merge the *de novo* and reference-guided assembly strategies, in order to assemble new organisms and improve the result achievable using only either one of the two. We will show how, by applying some of the ideas of assembly reconciliation, we can obtain an *enhanced reference assembly* of a new organism. All the alignments are guided by the reference sequence, in this way avoiding mis-assemblies and ambiguous overlaps. In [18], we illustrated the results obtained by our pipeline on a set of small organisms (chloroplasts and microbial); in this work, we show how the same pipeline can be effectively used for the assembly of large, highly repetitive and heterozygous genomes (plant genomes).

Reference-guided assembly

When a reference sequence *A* and a set of reads *R* are given, there are essentially two possible ways to perform reference assembly. The standard way simply involves aligning all the reads in *R* against the reference *A*, and then obtaining some consensus sequences. In the text that follows, we refer to this method as standard-RGA (*s-RGA*), and similarly, we call the consensus sequence produced *s-A*. An alternative approach is to perform de novo assembly on *R* first, and then to align the resulting contigs against the reference *A*. We call this second method de novo-RGA (*dn-RGA*), and the consensus sequence produced from it *dn-A*. Both the output sequences have "N" everywhere the coverage drops to 0. In order to simplify the discussion, we suppose *A* to be a single sequence (and therefore *s-A* and *dn-A* are also single sequences). It will be clear that this is not a limitation.

In the presence of NGS data, we have to use aligners like SOAP2 [13] and rNA [14] to obtain *s-A*. These aligners are highly conservative, they allow alignment of reads with a low number of errors, usually without gaps. For this reason, the length of *s-A* is the same of *A*. The sequence *dn-A* is obtained in three phases: first, reads are assembled using a short-read assembler ([6,7]); the resulting contigs are then aligned against *A*; after this, the consensus is generated. The challenge is to find an order for the contigs generated through the *de novo* assembly procedure. Several tools have been proposed to address this task. OSLay [19] computes a synthetic layout of the contigs using a reference sequence to anchor the *de novo* sequences; the Mauve aligner [20] gives as output an ordered version of the *de novo* contigs; and PGA [21] is able to layout the contigs with more than one reference genome at a time using global searches. All these tools implement or use a BLAST-like [22] search to align contigs against the reference. This alignment technique allows us to place reads on a reference with low similarity constraints. In particular, the contigs can be aligned against the reference sequence allowing partial hits and gaps.

This situation is similar to the already studied situation of assembly reconciliation. Casagrande *et al*. [10] proposed a method capable of merging two draft assemblies without performing global alignment. In particular, they proposed the use of one of the two assemblies as an "anchor", in order to resolve conflicts (*Master Assembly*).

Given the two assemblies *s-A* and *dn-A*, a suitable adaptation of this idea can be applied in the current context to obtain an enhanced reference assembly.

Methods

The Merge Graph: Definition

In this section, we briefly sketch the formal steps necessary to define the Merge Graph, at the base of our technique. More details can be found in [18].

With *S[i,j]* (*S* being either Δ or Γ), we identify the so-called "slice" of a string *S*, namely the substring of S from position *i* to *j*. If *S[i,j]* belongs to *{a,c,g,t}**, we call it a *(pure) contig*, while if *S[i,j]* belongs to *{N}**, it is named *gap*. In this context, when a (pure) contig is maximally extended, we say that it is a *max-contig*, and we define, in an analogous way, a *max-gap* (in general, we speak of max-area). Given two strings δ and γ, the function $D(\delta,\gamma,d)$ returns a value between 0 and 1, representing a percentage difference between the two strings. This value is naturally computed using a distance metric d (e.g., Hamming). The merge graph $MG(\Delta,\Gamma)$ is a directed graph such that V is contained in $I_\Delta \times I_\Gamma$ (I_Δ and I_Γ being all possible intervals in Δ and Γ, respectively) and can be partitioned into four sets: *gap-nodes* (V_g, gap against gap), *delta-nodes* (V_δ, a Δ-contig against a Γ-gap), *gamma-nodes* (V_γ, a Δ-gap against a Γ-contig) and *merge-nodes* (V_m, contig against contig). We also fix *t* and *s*, two thresholds that express bounds on the distance (alignment similarity) and the absolute position (within the relative string), respectively.

Edges are defined in such a way as to connect pairs of intervals (one on Δ and one on Γ) that can subsequently be put in the output as-

Table 1. Table of symbols and definitions.

A	Reference sequence
s-A	Consensus sequence obtained aligning short reads against reference A
dn-A	Consensus sequences obtained aligning de novo contigs against reference A
e-A	Enhanced reference-guided assembly obtained through the e-RGA pipeline
Δ, Γ	Generic sequences
MG(Δ,Γ)	Merge graph for the sequences Δ and Γ
MGA	Merging Global Alignment

```
         0          11         21         31         41         51         61
  Δ:    NNNNNNNaag gtttaaggtc ctacaNNNNa ctcatcataa aaacccNNNN NNNNNNctct aggtaaaa
  Γ:    NNNNNNNNNN NNNNggccta cacNNNNNac tcatcatcaa aaacccNNaa aaaaNNNNNN NNNNaaaaa
```

Figure 1 (a). The strings Δ and Γ ($|\Delta|=68$, $|\Gamma|=69$).

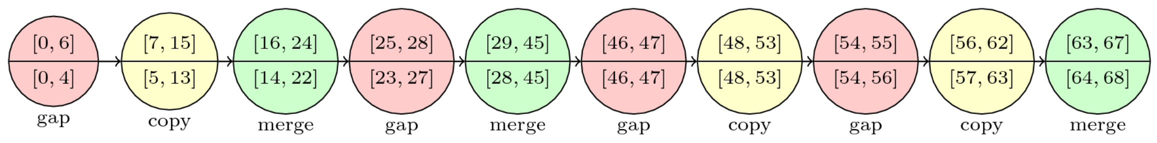

Figure 1 (b). A possible Merge Graph G for Δ and Γ with s=2 and t=2.

```
  Δ':   NNNNNNN AAGGTTTAA GGTCCTACA- NNNN- ACTCATCATAAAAA-CCC NN NNNNNN NN- CTCTAGG TAAAA
  Γ':   NNNNN-- NNNNNNNNN GG-CCTACAC NNNNN ACTCATCATCAAAAACCC NN AAAAAA NNN NNNNNNN AAAAA
  Λ:    MMMMMII MMMMMMMMM MMIMMMMMMD MMMMD MMMMMMMMMSMMMMDMMM MM MMMMMM MMD MMMMMMM SMMMM
```

Figure 1 (c). A possible Global Alignment obtained from the Merge Graph G (M means match, S means substitution, I means insertion, while D means deletion).

```
  Δ':   NNNNNNN AAGGTT- TAAGGTCCTACA- NNNN- ACTCATCAT-AAAAACCC NN NNNNNN NN- CTCTAGG TAAAA
  Γ':   NNNNNNN NNNNNNN ---GG-CCTACAC NNNNN ACTCATCATCAAAAACCC NN AAAAAA NNN NNNNNNN AAAAA
                          constrain
                          violated
```

Figure 1 (d). A Global Alignment that does not allow the creation of an MGA.

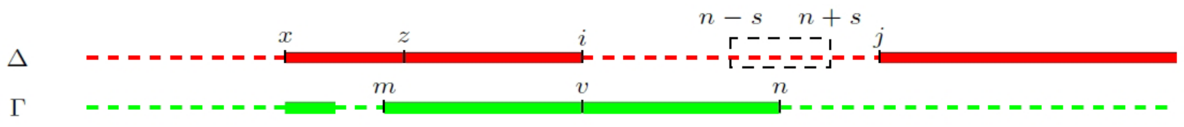

Figure 2. MGA construction. A possible scenario.

sembly. The resulting graph must be acyclic. In Figure 1B, an example of a merge graph is given.

In order to clarify the notation, we have summarised the symbols used throughout the paper in Table 1.

Merge Graph and Global Alignment

Given the strings Δ and Γ, there is a deep connection between the merge graph *MG(Δ,Γ)* and a global alignment between them. Building global alignments turns out to be equivalent to building a merge graph. In following text, we refer to the global alignment ensuing from a merge graph as *Merging Global Alignment (MGA)*.

The merge graph *MG(Δ,Γ)* can be used to extract a family of edit strings. From each node, we can produce an edited version of the two substrings that are represented. From delta and gamma nodes, we simply extract the contigs, while from the merge and gap nodes we can produce an edited version of one of the two strings. In the case of gap nodes, the edited version will contain only insertions and deletions (see Figures 1A, 1B and 1C). Once the edit strings are computed, the corresponding *MGA* can be calculated.

Given an *MGA*, the construction of a merge graph is more complicated. An *MGA* is a global alignment with two kinds of local properties: *locality* and *similarity*. In general, a global alignment does not guarantee these local properties, hence we can easily construct a global alignment that violates a local constraint. It can be proved that the global alignments we are seeking must respect the following properties: if *Δ[i,j]* and *Γ[k,l]* are aligned one against the other, then $|i\text{-}k|\text{-}1<s$ and $|j\text{-}l|\text{-}1<s$ (locality); if *Δ[i,j]* and *Γ[k,l]* are aligned and at least one is a max-contig or *Δ[i-1]=Γ[l+1]=N* (or *Δ[j+1]=Γ[k-1]=N*), then *D(Δ[i,j],Γ[k,l],d)* (similarity).

If such an alignment exists, calling *Δ´* and *Γ´* the two strings over the alphabet *{a,c,g,t,N,-}* returned by the global alignment between *Δ* and *Γ*, we can build *MG(Δ,Γ)* by simply reading from left to right *Δ´* and *Γ´*. For each position, we have to judge if a new node is beginning, or if we can continue extending the current one.

The determination of this global alignment can be computationally cumbersome. We cannot simply use an algorithm that calculates an optimal sequence alignment because a choice that can create an optimal global alignment will not necessarily lead to an alignment that respects all the local constraints (see Figure 1(d)). We will now sketch a complete algorithm that, given the strings *Δ* and *Γ*, generates all possible *MG(Δ,Γ)*.

The algorithm starts by reading the two sequences from left to right. For every contig in *Δ* and *Γ*, we can recursively compute all the possible alignments that satisfy the locality, and possibly the similarity, constraints. More detail is given in Figure 2. Let us assume that the last generated node was *<[z,i],[m,v]>*. From the merge graph definition, we have that at least one of i or v must be the end of a max-area, in this case i. At this point, we have to calculate the nearest (from *i*) max area end, *n* in the case shown in Figure 2. So the node we are going to create is *<[i,k],[v,n]>*, with *n-a<k<n+s* (paying attention to some special case, we can reduce the search space). In order to generate all the possible graphs, we have to recursively generate all the nodes. In case we are generating a merge node, we have also to check if the similarity constraint is respected. The algorithm terminates because at every step it proceeds forward along both strings.

Minimal Merge Graph

Given two strings Δ and Γ, it is clear that the existence of an *MG(Δ,Γ)* depends on the two thresholds *s* (the bound on the relative distance between intervals involved in the same node) and *t* (the bound on the similarity distance between strings involved in the same merge node). By setting s to be large enough, we can easily go towards computing a merge graph composed only by gap, delta and gamma nodes. Another trivial solution can be found when *t* is set in such a way that merge nodes accept very low levels of similarity.

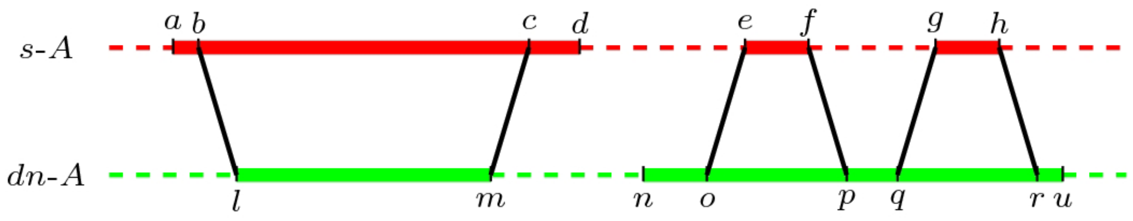

Figure 3. Practical identification of Merge-nodes for the Merge Graph Construction.

It is clear that a merge graph between a pair of sequences is interesting when *s* is a small constant when compared to Δ and Γ lengths and when *t* is sufficiently strict. These two constraints can strongly help in designing a better performing algorithm.

Merge graph and reference-guided assembly

A merge graph is a data-structure able to describe a global alignment between two strings, with further constraints on local alignment and similarity. This data-structure can be used both to describe the relations between two strings, and to extract a consensus.

When working with *s-A* and *dn-A*, we elect one of the two sequences to be the *Master Assembly (MA)*, that is the assembly we believe to be correct. In practice, in presence of a merge node, instead of calculating a consensus, we simply keep the sequence from the *MA*. Usually, even though two choices are possible, the *MA* will almost always be *dn-A*, with regions present in the sequenced organism and absent in the reference. If the merge graph *MG(s-A,dn-A)* is available, it can be used to extract a new assembly. Each node *p* in *MG(s-A,dn-A)* is characterised by two intervals, *[i,j]* and *[k,l]*. For *p=<[i,j],[k,l]>*, we extract the sequence *dn-A[k,l]*, if *p* is a gamma or a merge node, *s-A[i,j]*, if *p* is a delta-node, or the shortest between *s-A[i,j]* and *dn-A[k,l]*, if *p* is a gap node. This assembly is named *e-A* (*enhanced Reference Guided Assembly*).

The difficult part is the *MG(s-A,dn-A)* construction. The correct and complete algorithm presented in section 3.2 takes a time proportional to $O(s(j-i)^2)$ for each max-contig in the worst case. Additionally, the parameters *s* and *t* are unknown and, in general, there is no clear way to estimate them in advance, or to at least sensibly approximate them.

When working with *s-A* and *dn-A*, we have that the *MG(s-A,dn-A)* merge graph must exist for some *s* belonging to the set *{0,...,|dn-A|-|s-A|}*. This follows directly from the construction of the two strings. It is more difficult to limit *t*. A good working approximation is the percentage difference allowed in the *de novo* contig alignment.

The particular context provided by *s-A* and *dn-A* allows us to further improve the construction algorithm, concentrating only on a significant subset of all the global alignments associated to *MG(s-A,dn-A)*. Thanks to some intrinsic properties of *s-A* and *dn-A*, the brute-force algorithm can be improved, avoiding the generation of all possible global alignments. See Figure 3 for a graphical representation, and [18] for further details.

Enhanced-rga: implementation details

The pipeline represented in Figure 4 was implemented using several third-party tools and a set of Perl scripts implemented by the authors.

In order to construct *s-A*, first, a short-string aligner is used to align all the reads against the reference sequence, and a consensus is then extracted. In all cases in which a read is found in multiple occurrences, we randomly choose one of the alignments. We used the short-string aligner rNA [14] and the "pileup" command provided by samtools [23] to extract the consensus sequence.

dn-A was obtained by first performing *de novo* assembly with ABySS [7] and the CLC assembler Cell 3.0 [24]. Together with SOAPdenovo, [6] these are the only two assemblers able to assemble complex genomes using a reasonable amount of time and RAM memory. We noticed that, using contigs from both assemblies, the amount of genome reconstructed in *dn-A* greatly improves. The delicate phase of mapping contigs against

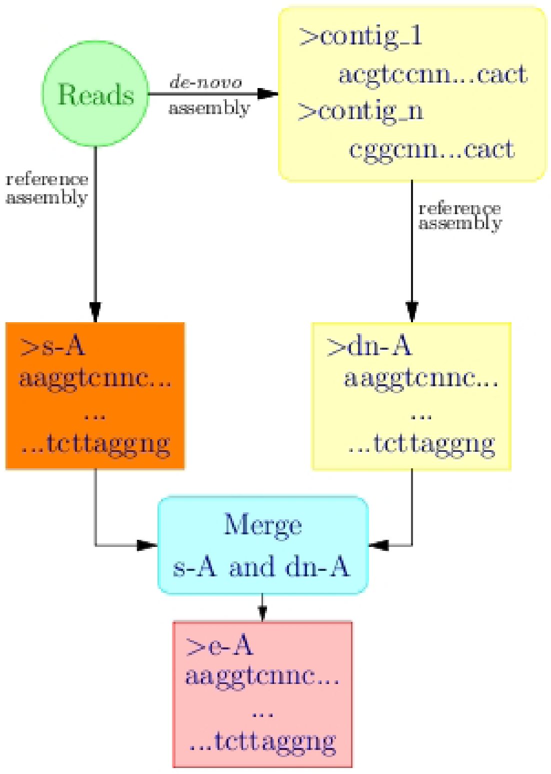

Figure 4. eRGA implementation

the reference sequence was accomplished with the CLC-Workbench[1].

Although we used these specific tools, clearly the production of *s-A* and *dn-A* can be carried out using different software without significant modification of the pipeline.

The core of *e-RGA* is the *MG(s-A,dn-A)* construction and *e-A* generation. These crucial phases are implemented within a Perl script that uses BLAST [22] to perform the approximate alignment. The program first memorises both *s-A* and *dn-A*, and localises all the *max-areas* (*max-contigs* and *max-gaps*). The *MG(s-A,dn-A)* construction proceeds as outlined in Section 4, with all the alignments performed by BLAST.

The software, together with a small example, can be downloaded at http://sole.dimi.uniud.it/~francesco.vezzi/software.php.

1 www.clcbio.com

Experiments and results

The Datasets

In [18], the *e-RGA* pipeline was tested on small genomes, demonstrating the effectiveness of our pipeline. We have further improved our pipeline and successfully used *e-RGA* on two large and complex plant genomes. The first dataset, named Sangiovese, comprises 5 Illumina lanes used to re-sequence a grapevine variety (Sangiovese, Rauscedo clone R24) with 100bp paired-end reads for a 90X total raw coverage. For this dataset, we used as reference sequence the genome of the highly homozygous grape clone, PN40024, used as reference genotype from the French-Italian Consortium for grape genome characterisation [25]. The second dataset, named Poplar, comprises 6 Illumina lanes used to sequence a Poplar individual belonging to the *Populus nigra* species, with 100bp paired-end reads for an 85X total raw coverage. In this case, we used as reference sequence the *Populus trichocarpa* genome [26]. While in the Sangiovese dataset we used a reference sequence belonging to the grapevine species, in the Poplar dataset the reference belongs to a different species. This difference is important in order to understand the differences obtained in the results. In both cases, before assembling and aligning, all the reads were filtered for quality, and we eliminated all sequences belonging to chloroplasts and mitochondria.

Both grapevine and poplar are characterised by long, repetitive genomes (480Mbp and 417Mbp respectively); moreover, both the sequenced individuals are highly heterozygous. These three conditions (length, repetitiveness and heterozygosity), together with the presence of two reference genomes, are perfect for our pipeline.

Results Discussion

Tables 2 and 3 summarise the results from the Sangiovese and Poplar datasets. As a measure of the assembly quality and correctness, we report the percentage of aligned reads (the same reads used to perform reference and *de novo* assembly), the number of contigs reconstructed, the mean contig length, the L50g (the length of the longest contig such that the sum of all the contigs greater than it represents half the expected genome length) and, in brackets, the L50c (the length of the longest contig such that the sum of all the contigs greater than it represents half the total contig length), and the percentage of Ns in the sequence. The L50g gives us a normalised value that describes the connectivity level of the assembly.

Those statistics have been computed for the reference sequence A, for the *s-RGA* output *s-A*, for the *de novo* assembly output *dn*, for the *dn-RGA* output *dn-A*, and finally, for the *e-RGA* output *e-A*.

Statistics such as the mean contig length, contig number, L50c and L50g, give us an idea of the quality of the assembly. In the Sangiovese case, we can see how the mean length obtained through *e-RGA* is longer than the other approaches, and although the *dn-A* mean length has a close value, we must consider the fact that these contigs cover only half the genome length as described by the high percentage of unknown characters. In the Sangiovese dataset, the most impressive results are the L50g and L50c improvements. Both *e-A*'s L50g and L50c are better than those of *s-A*, and they largely improve the results achievable with *de novo* assembly alone. This shows that our pipeline can effectively improve the final assembly result.

Similar results are summarised in Table 2 for the Poplar dataset. Owing to the distance between the sequenced organism (*Populus nigra*) and the reference genome (*Populus trichocarpa*), the Poplar results can look less promising than those of Sangiovese. However, the number of mapped reads against *e-A* is higher than the number of reads mapped against both *s-A* and *dn-A*. The fact that we are able to map a higher number of reads against *dn* should also be a consequence of the distance between the reference and the sequenced genome. As far as the standard assembly statistics are concerned (L50g, L50c and mean contig length), we can again see how the *e-A* results are better than those achievable by simply mapping reads or contigs back to the reference. Despite the *de novo* assembly result looking much better than the other approaches, we must stress the fact that *de novo* assembly alone gives us a set of 289,854 unordered contigs, with no information about their position in the final genome. It would be interesting, but outside the scope of this work, to understand the composition of the contigs not used to construct *dn-A*. These contigs, if correctly assembled, represent the areas belonging to the sequenced organism exclusively.

A further measure of the improvements introduced by the use of *e-RGA* is the number of successfully aligned paired reads (*i.e.*, paired reads

Table 2. Results obtained for the Sangiovese dataset.
For all the techniques used, we show the percentage of aligned reads, the number of contigs, the mean contig length, the L50 length computed both on the expected genome length and on the total contig length, and the number of unknown characters "N".

	Sangiovese				
	% aligned reads	Contigs number	Mean contig length	L50g (L50c)	% Ns
A	80.21%	-	-	-	3.00%
s-A	80.99%	246752	1758 bp	8514 bp (9901 bp)	7.64%
dn	53.10%	289854	1942 bp	1753 bp (3328 bp)	0.70%
dn-A	50.71%	109833	2246 bp	600 bp (3947 bp)	47.70%
e-A	81.77%	198194	2282 bp	12494 bp (14219 bp)	6.40%

Table 2. Results obtained for the Poplar dataset.
For all the techniques used, we show the percentage of aligned reads, the number of contigs, the mean contig length, the L50 length computed both on the expected genome length and on the total contig length, and the number of unknown characters "N".

	Poplar				
	% aligned reads	Contigs number	Mean contig length	L50g (L50c)	% Ns
A	55.00%	-	-	-	2.14%
s-A	58.00%	778065	365 bp	525 bp (1105 bp)	25.22%
dn	67.84%	116683	2728 bp	2906 bp (4487 bp)	0.40%
dn-A	37.00%	77370	1335 bp	0 bp (2085 bp)	62.46%
e-A	59.00%	558762	482 bp	957 bp (1959 bp)	18.56%

that align on the sequence at the expected distance and orientation). In both datasets, *e-A* is the sequence on which the largest number of constraints is respected.

More Applications

A possible *e-RGA* application, not explored in this work, is the identification and, more importantly, the reconstruction of structural variation. Two different steps of the *e-RGA* pipeline can be instrumental to this purpose. First is in the construction of *dn-A*. We can identify contigs that are aligned with gaps: alignments that introduce gaps in the reference sequence represent a putative insertion in the sequenced genome; conversely, alignments that introduce gaps in contigs reveal a putative deletion in the sequenced genome. The second step in which we might be able to identify structural variations is in *e-A* construction from the *MG(s-A,dn-A)* graph. In this case, a gamma-node <*[i,j],[k,l]*> (*dn-A*-contig against a *s-A*-gap) in which the interval *[i,j]* (the *s-A*-gap) is shorter than *[k,l]* (the *dn-A*-contig) is witness to an insertion in the sequenced genome; conversely, if the interval *[i,j]* is longer than *[k,l]* then the node is witness to a deletion in the sequenced genome. In the case of *delta-nodes*, the situation is symmetric.

Conclusions

The *e-RGA* pipeline was successfully applied to small organisms in [18]; in this paper, we have shown how the same approach can easily scale to large datasets with high coverage over complex (large and highly repetitive) genomes like the Grapevine and Poplar genomes.

e-RGA needs a reference genome belonging to a closely related organism. With the number of available genomes growing at a speed believed impossible only few years ago, this requirement is becoming standard.

Several research efforts are ongoing to design tools to identify and study *structural variations* (SV) [16], in particular within individual genomes. One major stumbling block is that it is still unclear how the identified or putative SV can be reconstructed. A future development of *e-RGA* will be to output putative SV identified during *e-A* construction. In this way, our pipeline, coupled with a tool able to identify/verify SV, could be used to reconstruct sequences that are specific to a particular individual.

Competing interest statement

None declared

References

1. Metzker ML (2009) Sequencing technologies — the next generation. Nat Rev Genet 11: 31-46.
2. Mardis ER (2008) The impact of next-generation sequencing technology on genetics. Trends Genet: 24: 133-141.
3. Li R, Fan W, Tian G et al. (2009) The sequence and de novo assembly of the giant panda genome. Nature 463: 311-317.
4. Velasco R, Zharkikh A, Affourtit J et al. (2010) The genome of the domesticated apple (Malus × domestica Borkh.). Nat Genet: 42: 833-839.
5. Nagarajan N, Pop M (2009) Parametric complexity of sequence assembly: Theory and applications to next generation sequencing. J Comput Biol 16: 897-908.
6. Li R, Zhu H, Ruan J al. (2010) De novo assembly of human genomes with massively parallel short read sequencing. Genome Res 20: 265-2727.
7. Simpson J, Wong K, Jackman S et al. (2009) ABySS: A parallel assembler for short read sequence data. Genome Res 19: 1117-1123.
8. Miller J, Koren S, Sutton G (2010) Assembly algorithms for next-generation sequencing data. Genomics 95(6): 315-327.
9. Batzoglou S, Jaffe DB, Stanley K et al. (2002) ARACHNE: A Whole-Genome Shotgun Assembler. Genome Res 12: 1100-1105.
10. Casagrande A, Del Fabbro C, Scalabrin S et al. (2009) GAM: Genomic Assemblies Merger: A Graph Based Method to Integrate Different Assemblies. Proc IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 321-326.
11. Pop M, Phillippy A, Delcher AL et al. (2004) Comparative genome assembly. Brief Bioinform 5:237-48
12. Gnerre S, Lander ES, Lindblad-Toh K et al. (2009) Assisted assembly: how to improve a de novo genome assembly by using related species. Genome Biol 10: R88.
13. Li R, Yu C, Li Y et al. (2009) SOAP2: an improved ultrafast tool for short read alignment. Bioinformatics 25: 1966-1967.

14. Policriti A, Tomescu A, Vezzi F (2010) A Randomized Numerical Aligner (rNA). Proc Language and Automata Theory and Applications (LATA) 6031: 512–523.
15. LeeS, Hormozdiari F, Alkan C et al. (2009) MoDIL: detecting small indels from clone-end sequencing with mixtures of distributions. Nat Methods 6: 473–474.
16. Kerstens HH, Crooijmans RP, Dibbits BW et al. (2011) Structural variation in the chicken genome identified by paired-end next-generation DNA sequencing of reduced representation libraries. BMC Genomics 12: 94-110.
17. Nijkamp J, Winterbach W, Broek M et al. (2010) Integrating genome assemblies with MAIA. Bioinformatics 26: i433-i439.
18. Cattonaro F, Policriti A, Vezzi F (2010) Enhanced reference guided assembly. Proc IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 77-80.
19. Richter DC, Schuster SC, Huson DH (2007) OSLay: optimal syntenic layout of unfinished assemblies. Bioinformatics 23: 1573-1579.
20. Rissman AI, Mau B, Biehl BS et al. (2009) Reordering contigs of draft genomes using the Mauve aligner. Bioinformatics 25: 2071-2073.
21. Zhao F, Zhao F, Li T et al. (2008) A new pheromone trail-based genetic algorithm for comparative genome assembly. Nucleic Acids Res. 36: 3455-3462.
22. Altschul S. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25: 3389-3402.
23. Li H, Handsaker B, Wysoker A et al. (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25: 2078-2079.
24. CLC Team (2010) De novo assembly on the CLC Assembly Cell. (White Paper) http://www.clcbio.com/white-paper/
25. Jaillon O, Aury JM, Noel B et al. (2007) The grapevine genome sequence suggests ancestral hexaploidization in major angiosperm phyla. Nature 449: 463-467.
26. Tuskan GA, Difazio S, Jansson S et al. (2006) The genome of black cottonwood, Populus trichocarpa (Torr. & Gray). Science 313: 1596-1604.