BIOINFORMATICS REVIEW
Vol. 29 no. 23 2013, pages 2959–2963
doi:10.1093/bioinformatics/btt525

Genome analysis
Advance Access publication September 10, 2013

Toward a statistically explicit understanding of *de novo* sequence assembly

Mark Howison[1,*], Felipe Zapata[2] and Casey W. Dunn[2]
[1]Center for Computation and Visualization and [2]Department of Ecology and Evolutionary Biology, Brown University, Providence, RI 02912, USA
Associate Editor: Jonathan Wren

ABSTRACT
Motivation: Draft *de novo* genome assemblies are now available for many organisms. These assemblies are point estimates of the true genome sequences. Each is a specific hypothesis, drawn from among many alternative hypotheses, of the sequence of a genome. Assembly uncertainty, the inability to distinguish between multiple alternative assembly hypotheses, can be due to real variation between copies of the genome in the sample, errors and ambiguities in the sequenced data and assumptions and heuristics of the assemblers. Most assemblers select a single assembly according to *ad hoc* criteria, and do not yet report and quantify the uncertainty of their outputs. Those assemblers that do report uncertainty take different approaches to describing multiple assembly hypotheses and the support for each.
Results: Here we review and examine the problem of representing and measuring uncertainty in assemblies. A promising recent development is the implementation of assemblers that are built according to explicit statistical models. Some new assembly methods, for example, estimate and maximize assembly likelihood. These advances, combined with technical advances in the representation of alternative assembly hypotheses, will lead to a more complete and biologically relevant understanding of assembly uncertainty. This will in turn facilitate the interpretation of downstream analyses and tests of specific biological hypotheses.
Contact: mhowison@brown.edu

Received on June 20, 2013; revised on July 25, 2013; accepted on September 2, 2013

1 INTRODUCTION

The low cost and increasing availability of next-generation sequencing data have driven a growing interest in methods and software tools for *de novo* genome assembly of short read sequences. Recent surveys of assembly tools (Finotello *et al.*, 2012; Miller *et al.*, 2010), practical guides (Nagarajan and Pop, 2013; Paszkiewicz and Studholme, 2010), competitions like the Assemblathon (Bradnam *et al.*, 2013; Earl *et al.*, 2011) and benchmarking tools like GAGE (Salzberg *et al.*, 2012) highlight the diverse ecosystem of available assemblers. New data structures, algorithms and software tools for assembly continue to be published every month.

Many investigators have claimed that it is now possible to assemble high quality genomes from next-generation sequencing data when using appropriate protocols and assembly methods (Gnerre *et al.*, 2011; Li *et al.*, 2010; Schatz *et al.*, 2010). Yet, others have expressed concern over the integrity of publicly available draft genomes assembled from such data. Some have described errors and shortcomings in specific draft assemblies (Alkan *et al.*, 2011; Ricker *et al.*, 2012; Salzberg and Yorke, 2005), whereas others have questioned the quality of publicly available draft assemblies in general, and advocated better quality standards for the community (Chain *et al.*, 2009; Mardis *et al.*, 2002). In particular, the Assemblathon 2 competition (Bradnam *et al.*, 2013) found large-scale inconsistencies among current assembly methods, suggesting they are not robust to changes in parameters and input data, and that there is a need for unambiguous measures of assembly uncertainty.

A genome assembly is a hypothesis consisting of a collection of *contigs* (contiguous sequences) and *scaffolds* (groups of contigs with gaps of known length between them) that typically cover 90% or more of the genome (Chain *et al.*, 2009), but are often fragmented and unordered. Current *de novo* assemblers use various heuristics and algorithms to select an assembly that optimizes some criteria, such as path length or graph complexity (Miller *et al.*, 2010); however, these optimization criteria are typically *ad hoc*. This is largely because of the computational difficulty of performing assembly on short reads, and a primary goal for existing assembly methods has been computational tractability and efficiency. As a result, assemblers choose a single point estimate as their final output with sparse information about the quality, certainty or validity of the chosen assembly, or of alternative assembly hypotheses (many of which may have almost as much support). In most cases, it is difficult, if not impossible, to answer even basic questions like, 'How well is this contig supported by the read sequences?' or 'Are there alternative assemblies that have similar support from the data?'

Downstream analysis tools use assemblies to make their own point estimates of other aspects of biology, such as multiple sequence alignments, differential gene expression analyses or phylogenetic trees. In the end, there is no accounting for how the uncertainty is compounded at each stage. Existing tools cannot be integrated into pipelines that propagate uncertainty through a large multistep analysis, for example integrating assembly uncertainty with tree uncertainty when constructing phylogenies. The ability to propagate uncertainty about point estimates or, preferably, to propagate entire sets of multiple alternative hypotheses will become increasingly important as analyses grow in complexity.

*To whom correspondence should be addressed.

Thanks to the progress on computational efficiency of genome assembly, it is now possible to tackle the difficult goal of placing *de novo* sequence assembly within an explicit statistical framework. In such a framework, single assembly hypotheses selected according to *ad hoc* optimality criteria are replaced by sets of hypotheses accompanied by statistics that summarize confidence in each.

2 VARIANTS IN ASSEMBLIES

Alternate assembly hypotheses are called variants. There are many types of variants, but they fall into two broad categories that we refer to as hard and soft:

Hard variants correspond to real differences present in the sample. Hard variation can include heterozygosity, somatic polymorphism (as in the case of cancer), polymorphism across multiple individuals when they are pooled for sequencing or variation across individuals when they are sequenced and assembled separately but data are then combined across assemblies. Hard variants reflect aspects of organism biology that may or may not be of direct interest to the investigator.

Soft variants are uncertainties that are introduced during the sequencing and assembly process, and include library preparation artifacts and sequencing errors. They persist when there is not enough information to resolve conflicts and identify the true assembly. Soft variants are nuisances that investigators seek to reduce or work around.

Discerning between hard and soft variants presents difficult statistical and computational challenges, and is a fundamental difficulty for metagenome assembly in particular (Charuvaka and Rangwala, 2011). Although hard and soft variants have different origins, they can both be described within a common statistical framework, as they both result in multiple assembly hypotheses. After this common framework is in place, the next challenge will be to differentiate between hard and soft variants, either by eliminating soft variation, or by learning to identify each. However this is ultimately addressed, the very existence of hard variation is a direct challenge to the expectation that there is a single true assembly that accurately represents an organism's genome.

One of the best-studied types of hard variation is heterozygosity in diploid individuals. Provided enough depth of coverage, existing statistical methods can accurately identify alleles (Nielsen *et al*., 2011). In the absence of enough coverage, though, it becomes difficult to differentiate true alleles from sequencing errors. The identification of alleles from different loci that are colocated on the same chromosome is called haplotype *phasing*. Phasing can be achieved computationally or experimentally (Browning and Browning, 2011). Computational phasing requires population level sampling, which is uncommon in most studies of *de novo* genome assembly. Experimental phasing relies on laboratory techniques that are applied during data generation, such as developing fosmid libraries or separation of chromosomes. This approach incurs higher costs, and it usually involves additional computational phasing when phased haplotype fragments must be pieced together into larger haplotypes (Browning and Browning, 2011). At present, phase information from sequencing reads is not sufficient to fully determine haplotype phase.

3 RECORDING VARIANTS

Some assemblers report variants in their output, though without any accompanying statistical interpretation or distinction between hard and soft variants. Although much of the focus has been on tools for single nucleotide polymorphism (SNP) detection, there is interest in larger-scale structural variants as well. Preserving and reporting ambiguities in the assembly is an important step toward assessing assembly uncertainty, especially if future computational methods can incorporate alternative assemblies. Assemblers that report variants include:

ALLPATHS-LG (Gnerre *et al*., 2011) has a custom intermediate output format for SNPs or homopolymers. For example, the output sequence `TC{A,T}GG` represents an SNP, and `TT{,T,TT}AC` represents a homopolymer. The authors note that making use of this information in downstream analyses is an important challenge for the field.

The String Graph Assembler (SGA) (Simpson and Durbin, 2012) retains variants that are not selected by the assembly algorithm, but instead of storing them in a custom format, writes them to a separate FASTA file that can be inspected after assembly.

ABySS (Simpson *et al*., 2009) similarly writes multiple variants and organizes them into two FASTA files, one for SNPs and the other for insertions–deletions.

Cortex (Iqbal *et al*., 2012) and **fermi** (Li, 2012) are both designed to discover variants during assembly. Both show that structural variant detection can be improved by discovering variants during assembly rather than through simply mapping the assembly to a reference genome.

In addition to advances in the assemblers themselves, there have also been improvements in data formats. The FASTG (Jaffe *et al*., 2012) specification addresses the problem of storing complex polymorphisms and variants by using a graph representation for assembly output. Most assemblers' final output uses a linear FASTA representation, with a record for each contig or scaffold sequence. Although this format is compact, human-readable and a suitable representation of a correct unambiguous assembly, in practice most assemblies include ambiguities that cannot be represented linearly. At the opposite extreme of the linear FASTA representation is the intermediate output provided by most assemblers that dumps out the complete unresolved graph structure produced during assembly. For most downstream applications, this output is too verbose and too raw: it might not even include the graph traversals chosen by the assembler's heuristics or algorithms as the final assembly.

FASTG attempts a balance between these two extremes. It is an extension of the approach taken by ALLPATHS-LG, and specifies 'constructs' enclosed in brackets that can be inserted into a typical FASTA sequence to represent local non-linear features like gaps, alleles, tandem repeats or haplotypes. For example, the sequence `GANNNNNN[5:gap:size=(5,4..6)] CAGGC[1:alt:allele—C,G]` includes constructs for both a gap of 4–6 bases and an SNP with a similar proportion of C and G bases, which can therefore be interpreted as an allele.

Another example of a richer description for assembly output is the 'gene graph', introduced by the GeneStitch (Wu *et al*., 2012) method for reconciling and improving metagenomic assemblies. Using alignments against a reference genome, GeneStitch identifies clusters of gene fragments that are highly similar across the individual genomes within the metagenome. Instead of trying to separate the individual genes, GeneStitch merges them into a structure called a gene graph, which is a subgraph of the assembly graph that connects all the similar gene fragments. The gene graph is a condensed representation of the similar genes, and individual genes can be reconstructed by traversing paths through the gene graph.

Another middle ground between linear representation and full assembly graph output is to simply enumerate the full set of possible assemblies, the approach that the SGA assembler takes when it writes alternative contigs to an auxiliary file. This is analogous to the approach taken by phylogenetic inference tools that generate sets of phylogenetic trees. An investigator will typically construct and report a *consensus* tree, which is a lossy summary of the full set of trees according to some statistical justification (Holder *et al*., 2008). Similarly, an assembler could output the full set, but construct a consensus assembly for each contig. The full set of assemblies is inherently redundant, and could be compressed with generic text compression tools, like `gzip`.

Although these representations are better suited to storing the variation in assembly output than FASTA, they do not address the statistical or computational problem of how to quantify the uncertainty of a given assembly hypothesis. We discuss existing approaches to these problems below.

4 MIS-ASSEMBLY APPROACHES

Earlier efforts to automate assembly validation successfully applied statistical tests to identify 'mis-assemblies,' or regions of an assembly hypothesis that violate specific statistical assumptions. For instance, the *amosvalidate* tool (Phillippy *et al*., 2008) uses the *compression-expansion* statistic (Zimin *et al*., 2008) to identify regions of an assembly where paired-end reads align with insert sizes that deviate from an expected normal distribution. It also calculates statistics based on the overall read coverage, the distribution of *k*-mers and the presence of fragmented read alignments.

More recently, the Recognition of Errors in Assemblies using Paired Reads (REAPR) tool (Hunt *et al*., 2013) applied similar metrics of fragment coverage and insert-size distribution to identify mis-assembled regions, and introduced the ability to call errors at specific bases in an assembly hypothesis. Computationally, it decides which individual bases are 'error-free,' meaning that the base is supported by a specified number (by default 5) of perfectly and uniquely aligned reads, and that the difference between the theoretical and observed fragment coverage falls below a dynamically inferred threshold. Regions with erroneous bases are reported as mis-assemblies. The algorithm also distinguishes between contig and scaffolding errors, and can produce a new assembly where erroneous scaffolds are broken into separate contigs.

5 LIKELIHOOD APPROACHES

In statistics, likelihood is the probability of the data if the data were generated according to a specified hypothesis. In the context of assembly, it is the probability of sequencing the observed reads under a specified assembly hypothesis and model of read generation.

Maximum likelihood estimation attempts to identify the hypothesis that has the highest probability of producing the observed data. A maximum likelihood assembly is the assembly that has the highest likelihood. Maximum likelihood estimation does not itself provide a confidence interval on any particular hypothesis; it simply provides a way to find the hypothesis that maximizes the probability of the data. The assembly with the maximum likelihood may do a much better job than any other assembly at explaining the data, or there may be millions of other assemblies that are almost as likely. Even though a likelihood approach does not directly quantify assembly uncertainty, it provides an explicit framework with a clear statistical interpretation for optimizing and evaluating alternative assembly hypotheses.

The Computing Genome Assembly Likelihoods (CGAL) tool (Rahman and Pachter, 2013) approximates the likelihood of an assembly given the sequence reads and a generative model. To reduce computational burden, read generation is considered only in the region of the assembly where each read maps. The generative model incorporates separate terms for the length of a read pair and its aligned site on the genome, and an error model for SNPs, insertions and deletions. The generative model has to be learned from the data. Because the distribution of insert sizes for read pairs depends on both the sequencer and library preparation, CGAL uses the empirical distribution for the read pair lengths. For the distribution of sites, it assumes uniform sampling of read pairs across the genome. For the error model, it assumes sequencing errors are independent events and learns the substitution rates for each position and for each substitution combination (because there are known biases for some sequencing technologies), and the insertion and deletion rates for each position in a read sequence. The aggregate CGAL score for an assembly is the log of the product of the probabilities that each individual read could have been generated from the assembly.

Although CGAL is not an assembler, it could be applied to optimizing assembly by using the annotated likelihood score to iteratively guide the selection of assemblies and parameter values. In fact, a maximum likelihood genome assembler was already proposed based on similar principles (Medvedev *et al*., 2009). Like CGAL, it calculates likelihood based on the depth of read coverage, but it does not incorporate paired-end information at this stage. Instead, it takes the approach typical of many genome assemblers of first assembling the contigs, then resolving conflicts by looking for contigs that agree with the orientation and insert size of the paired reads. Also, it requires as a parameter the accurate size of the target genome, which is not available in all *de novo* assembly projects. A related design for maximum likelihood assembly (Varma *et al*., 2011) uses a different formulation that starts from an approximate size and estimates the actual size during the optimization.

One of the limitations of the maximum likelihood approach is that it relies on complex optimizations that are polynomial time in the number of read sequences, compared with the linear time algorithms used by most de Bruijn graph assemblers. Also, unlike most assembly methods described in the literature, neither the maximum likelihood methods by Medvedev *et al*. nor Varma *et al*. provide an open-source reference implementation.

6 BAYESIAN APPROACHES

Instead of the probability that a given assembly hypothesis could have generated the sequenced data (i.e. the likelihood), an investigator may be more interested in the conditional probability of the assembly hypothesis after taking into account the sequenced data. This is the posterior probability, $P(H|D)$, of the assembly hypothesis, and it is related to the likelihood, $P(D|H)$, by the Bayes' theorem:

$$P(H|D) = \frac{P(D|H)(P(H))}{P(D)} \tag{1}$$

where $P(D)$ and $P(H)$ are the prior probabilities on the data and the assembly hypothesis, respectively. The priors are the probability distributions that express the uncertainty before the data are taken into account.

There is already at least one tool that considers posterior probabilities on assemblies, the Assembly Likelihood Evaluation (ALE) framework (Clark *et al.*, 2013). ALE implements an expression for the probability that an assembly is correct, and also reveals the contribution of local regions of the assembly to this score. This is an important advance toward assessing the uncertainty of assemblies, especially because it is made in the context of an explicit statistical framework rather than *ad hoc* optimality criteria. ALE estimates the posterior probability of an assembly (their $P(S|R)$) by estimating the prior probabilities (their $P(S)$) directly from the data (i.e. an empirical Bayes approach) in conjunction with an approximation of the assembly likelihood (their $P(R|S)$) in a similar fashion to CGAL. One of the most difficult aspects of calculating a posterior probability is deriving the prior probability of the read data, $P(R)$ (that they denote as Z). They address this challenge with a rough but efficient approximation of Z. They then refer to the approximated posterior probability as the ALE score.

The ALE score is a comparative measure of assembly correctness and should be compared among assemblies of the same genome from the same sequenced data. The ALE score cannot be calculated for different datasets because of the possible inaccuracy in approximating the prior probability of the data, which cancels out when computing a comparative score between difference assembly hypotheses of the same data. In contrast, CGAL could conceivably be used to compare the likelihood of an assembly hypothesis against different datasets (for instance, from different sequencing technologies) because it does not calculate the prior probabilities of the data.

Markov chain Monte Carlo (MCMC) is an alternative approach to approximating posterior probabilities. Rather than approximate the posterior probability of a particular assembly as ALE does, an MCMC approach would generate a set of alternative assembly hypotheses. This provides a natural way to deal with assembly uncertainty. The frequency of a particular attribute of the assembly in this set is an approximation of the posterior probability of that attribute. In addition to deriving this probability, the investigator can also examine the other alternative hypotheses. An investigator could ask, for example, 'What are the most probable hypotheses for gene order that together account for 90% of the posterior probability?'

To overcome the challenges of estimating the prior probability on the data, MCMC uses the ratios of posterior probabilities so that the prior probability on the data cancels out and does need to be calculated (for an introduction to MCMC, see Gilks *et al.*, 1996). MCMC methods have been applied to related problems, such as assembling the haplotype of resequenced human genomes (Bansal *et al.*, 2008). However, we do not know of a *de novo* assembly method that has used MCMC to generate a set of assembly hypotheses. Like maximum likelihood assembly, MCMC assembly will have significant technical challenges with computational cost and scalability because of the many samples needed to construct a stable posterior distribution.

7 CONCLUSION

The pieces are now falling in place for assembly to move away from point estimates that are selected according to *ad hoc* criteria, toward a statistically explicit framework that provides not only biologically relevant measures of certainty but also sets of alternative hypotheses. This will greatly facilitate the evaluation of assemblies, their application to specific biological questions, improvements in assembly algorithms and integration with downstream analyses that can then take assembly uncertainty into account. Bioinformatics workflow frameworks, such as the web-based framework Galaxy (Giardine *et al.*, 2005) and the lightweight command-line framework BioLite (Howison *et al.*, 2012), already provide biologists with functionality for establishing provenance and reproducibility for computational analyses. These workflow frameworks are the logical foundation for implementing pipelines that propagate uncertainty through complex multistage analyses.

ACKNOWLEDGEMENTS

The authors thank C. Titus Brown and an anonymous reviewer for their helpful comments.

Conflict of Interest: none declared.

REFERENCES

Alkan,C. *et al.* (2011) Limitations of next-generation genome sequence assembly. *Nat. Methods*, **8**, 61–65.

Bansal,V. *et al.* (2008) An MCMC algorithm for haplotype assembly from whole-genome sequence data. *Genome Res.*, **18**, 1336–1346.

Bradnam,K.R. *et al.* (2013) Assemblathon 2: evaluating *de novo* methods of genome assembly in three vertebrate species. *Gigascience*, **2**, 10.

Browning,S.R. and Browning,B.L. (2011) Haplotype phasing: existing methods and new developments. *Nat. Rev. Genet.*, **12**, 703–714.

Chain,P.S. *et al.* (2009) Genomics. Genome project standards in a new era of sequencing. *Science*, **326**, 236–237.

Charuvaka,A. and Rangwala,H. (2011) Evaluation of short read metagenomic assembly. *BMC Genomics*, **12**, S8.

Clark,S.C. *et al.* (2013) ALE: a generic assembly likelihood evaluation framework for assessing the accuracy of genome and metagenome assemblies. *Bioinformatics*, **29**, 435–443.

Earl,D. *et al.* (2011) Assemblathon 1: a competitive assessment of *de novo* short read assembly methods. *Genome Res.*, **21**, 2224–2241.

Finotello,F. *et al.* (2012) Comparative analysis of algorithms for whole-genome assembly of pyrosequencing data. *Brief. Bioinform.*, **13**, 269–280.

Giardine,B. *et al.* (2005) Galaxy: a platform for interactive large-scale genome analysis. *Genome Res.*, **15**, 1451–1455.

Gilks,W.R. *et al.* (1995) *Markov Chain Monte Carlo in Practice*. Chapman and Hall/CRC, London.

Gnerre,S. *et al.* (2011) High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc. Natl Acad. Sci.* USA, **108**, 1513–1518.

Holder,M.T. *et al.* (2008) A justification for reporting the majority-rule consensus tree in Bayesian phylogenetics. *Syst. Biol.*, **57**, 814–821.

Howison,M. *et al.* (2012) BioLite, a lightweight bioinformatics framework with automated tracking of diagnostics and provenance. In: *Proceedings of the 4th USENIX Workshop on the Theory and Practice of Provenance (TaPP'12)*. Boston, MA, USA.

Hunt,M. *et al.* (2013) REAPR: a universal tool for genome assembly evaluation. *Genome Biol.*, **14**, R47.

Iqbal,Z. *et al.* (2012) *De novo* assembly and genotyping of variants using colored de Bruijn graphs. *Nat. Genet.*, **44**, 226–232.

Jaffe,D.B. *et al.* (2012) The FASTG Format Specification (v1.00). http://fastg.sourceforge.net/FASTG_Spec_v1.00.pdf. (27 February 2013, date last accessed).

Li,Y. *et al.* (2010) State of the art *de novo* assembly of human genomes from massively parallel sequencing data. *Hum. Genomics*, **4**, 271–277.

Li,H. (2012) Exploring single-sample SNP and INDEL calling with whole-genome *de novo* assembly. *Bioinformatics*, **28**, 1838–1844.

Mardis,E. *et al.* (2002) What is finished, and why does it matter. *Genome Res.*, **12**, 669–671.

Medvedev,P. and Brudno,M. (2009) Maximum likelihood genome assembly. *J. Comput. Biol.*, **16**, 1101–1116.

Miller,J.R. *et al.* (2010) Assembly algorithms for next-generation sequencing data. *Genomics*, **95**, 315–327.

Nagarajan,N. and Pop,M. (2013) Sequence assembly demystified. *Nat. Rev. Genet.*, **14**, 157–167.

Nielsen,R. *et al.* (2011) Genotype and SNP calling from next-generation sequencing data. *Nat. Rev. Genet.*, **12**, 443–451.

Paszkiewicz,K. and Studholme,D.J. (2010) *De novo* assembly of short sequence reads. *Brief. Bioinform.*, **11**, 457–472.

Phillippy,A. *et al.* (2008) Genome assembly forensics: finding the elusive mis-assembly. *Genome Biol.*, **9**, R55.

Rahman,A. and Pachter,L. (2013) CGAL: computing genome assembly likelihoods. *Genome Biol.*, **14**, R8.

Ricker,N. *et al.* (2012) The limitations of draft assemblies for understanding prokaryotic adaptation and evolution. *Genomics*, **100**, 167–175.

Salzberg,S.L. and Yorke,J.A. (2005) Beware of mis-assembled genomes. *Bioinformatics*, **21**, 4320–4321.

Salzberg,S.L. *et al.* (2012) GAGE: a critical evaluation of genome assemblies and assembly algorithms. *Genome Res.*, **22**, 557–567.

Schatz,M.C. *et al.* (2010) Assembly of large genomes using second-generation sequencing. *Genome Res.*, **20**, 1165–1173.

Simpson,J.T. and Durbin,R. (2012) Efficient *de novo* assembly of large genomes using compressed data structures. *Genome Res.*, **22**, 549–556.

Simpson,J.T. *et al.* (2009) ABySS: a parallel assembler for short read sequence data. *Genome Res.*, **19**, 1117–1123.

Varma,A. *et al.* (2011) An improved maximum likelihood formulation for accurate genome assembly. In: *Proceedings of the 1st IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*. Orlando, FL, USA, pp. 165–170.

Wu,Y.-W. *et al.* (2012) Stitching gene fragments with a network matching algorithm improves gene assembly for metagenomics. *Bioinformatics*, **28**, i363–i369.

Zimin,A.V. *et al.* (2008) Assembly reconciliation. *Bioinformatics*, **24**, 42–45.