OMGS: Optical Map-Based Genome Scaffolding

Weihua Pan, Tao Jiang, and Stefano Lonardi(✉)

Department of Computer Science and Engineering, University of California,
Riverside, CA 92521, USA
stelo@cs.ucr.edu

Abstract. Due to the current limitations of sequencing technologies, *de novo* genome assembly is typically carried out in two stages, namely contig (sequence) assembly and scaffolding. While scaffolding is computationally easier than sequence assembly, the scaffolding problem can be challenging due to the high repetitive content of eukaryotic genomes, possible mis-joins in assembled contigs and inaccuracies in the linkage information. Genome scaffolding tools either use paired-end/mate-pair/linked/Hi-C reads or genome-wide maps (optical, physical or genetic) as linkage information. Optical maps (in particular Bionano Genomics maps) have been extensively used in many recent large-scale genome assembly projects (e.g., goat, apple, barley, maize, quinoa, sea bass, among others). However, the most commonly used scaffolding tools have a serious limitation: they can only deal with one optical map at a time, forcing users to alternate or iterate over multiple maps. In this paper, we introduce a novel scaffolding algorithm called OMGS that for the first time can take advantages of multiple optical maps. OMGS solves several optimization problems to generate scaffolds with optimal contiguity and correctness. Extensive experimental results demonstrate that our tool outperforms existing methods when multiple optical maps are available, and produces comparable scaffolds using a single optical map. OMGS can be obtained from https://github.com/ucrbioinfo/OMGS.

Keywords: *De novo* genome assembly · Scaffolding · Optical maps · Combinatorial optimization

1 Introduction

Genome assembly is a fundamental problem in genomics and computational biology. Due to the current limitations of sequencing technologies, the assembly is typically carried out in two stages, namely contig (sequence) assembly and scaffolding. Scaffolds are arrangements of oriented contigs with gaps representing the estimated distance separating them. The scaffolding process can vastly improve the assembly contiguity and can produce chromosome-level assemblies. Despite significant algorithmic progress, the scaffolding problem can be challenging due to the high repetitive content of eukaryotic genomes, possible mis-joins in assembled contigs and the inaccuracies of the linkage information.

L. J. Cowen (Ed.): RECOMB 2019, LNBI 11467, pp. 190–207, 2019.
https://doi.org/10.1007/978-3-030-17083-7_12

Genome scaffolding tools either use paired-end/mate-pair/linked/Hi-C reads or genome-wide maps. The first group includes scaffolding tools for second generation sequencing data, such as Bambus [17,29], GRASS [13], MIP [31], Opera [12], SCARPA [11], SOPRA [8] and SSPACE [5] and the scaffolding modules from assemblers ABySS [35], SGA [34] and SOAPdenovo2 [22]. Since the relative orientation and approximate distance between paired-end/mate-pair/linked/Hi-C reads are known, the consistent alignment of a sufficient number of reads to two contigs can indicate their relative order, their orientation and the distance between them. An extensive comparison of scaffolding methods in this first group of tools can be found in [14].

The second group uses genome-wide maps such as genetic maps [37], physical maps, or optical maps. According to the markers provided by these maps, contigs can be anchored to specific positions so that their order and orientations can be determined. The distance between contigs can also be estimated with varying degree of accuracy depending on the density of the map.

The optical mapping technologies currently on the market (e.g., BioNano Genomics Irys systems, OpGen Argus) allow computational biologists to produce genome-wide maps by fingerprinting long DNA molecules (up to 1 Mb), via nicking restriction enzymes [32]. Linear DNA fragments are stretched on a glass surface or in a nano-channel array, then the locations of restriction sites are identified with the help of dyes or fluorescent labels. The results are imaged and aligned to each other to map the locations of the restriction sites relative to each other. While the assembly process for optical molecules is highly reliable, there is clear evidence that a small fraction of the optical molecules is chimeric [15].

A few scaffolding algorithms that use optical maps are available. SOMA appears to be the first published tool that can take advantage of optical maps but it can only deal with a non-fragmented optical map [25]. The scaffolding tool proposed in [30] was used for two bacterial genomes *Yersinia pestis* and *Yersinia enterocolitica*, but the software is no longer publicly available. In the last few years, Bionano optical maps have become very popular, and have been used to improve the assembly contiguity in many large-scale *de novo* genome assembly projects (e.g., goat, apple, barley, maize, quinoa, sea bass [4,7,23,28]). To the best of our knowledge, the main tools used to generate scaffolds using Bionano optical maps are SewingMachine from KSU [33] and HybridScaffold from Bionano Genomics (unpublished, 2016). SewingMachine seems to be favored by practitioners over HybridScaffold.

Both HybridScaffold and SewingMachine have, however, a serious limitation: they can only deal with one optical map at a time, forcing users to alternate or iterate over optical maps when multiple maps are available. In this paper, we introduce a novel scaffolding algorithm called OMGS that for the first time can take advantage of any number of optical maps. OMGS solves several optimization problems to generate scaffolds with optimal contiguity and correctness.

2 Problem Definition

The input to the problem is the genome assembly to be scaffolded (represented by a set of assembled contigs), and one or more optical maps (represented by a set of sets of genomic distances). We use $C = \{c_i | i = 1, \ldots, l\}$ to denote the set of contigs in the genome assembly, where each c_i is a string over the alphabet $\{A, C, G, T\}$. Henceforth, we assume that the contigs in C are chimera-free.

An optical map is composed by a set of optical molecules, each of which is represented by an ordered set of positions for the restriction enzyme sites. As said, optical molecules are obtained by an assembly process similar to sequence assembly, but we will reserve the term *contig* for sequenced contigs. We use $M = \{m_i | i = 1, \ldots, n\}$ to denote the optical map, where each optical molecule m_i is an ordered set of integers, corresponding to the distances in base pairs between two adjacent restriction enzyme sites on molecule m_i. By digesting *in silico* the contigs in C using the same restriction enzyme used to produce the optical map and matching the sequence of adjacent distances between sites, one can align the contigs in C to the optical map M. If one is given multiple optical maps obtained using different restriction enzymes, M will be the union of the molecules from all optical maps. In this case, each genomic location is expected to be covered by multiple molecules in M. As said, high quality alignments allows one to anchor and orient contigs to specific coordinates on the optical map. When multiple contigs align to the same optical map molecule, one can order them and estimate the distance between them. By filling these gaps with a number of N's equal to the estimated distance, longer DNA sequences called *scaffolds* can be obtained.

A series of practical factors make the problem of scaffolding non-trivial. These factors include imprecisions in optical maps (e.g., mis-joins introduced during the assembly of the optical map [15]), unreliable alignments between contigs and optical molecules, and multiple inconsistent anchoring positions for the same contigs. As a consequence, it is appropriate to frame this scaffolding problem as an optimization problem.

We are now ready to define the problem. We are given an assembly represented by a set of contigs C, a set of optical map molecules M and a set of alignments $A = \{a_{1,1}, a_{1,2}, \ldots a_{l,n}\}$ of C to M, where $a_{i,j}$ is the alignment of contig c_i to optical map molecule o_j. The problem is to obtain a set of scaffolds $S = \{s_1, s_2, \ldots s_k\}$ where each s_i is a string over the alphabet $\{A, C, G, T, N\}$, such that (i) each contig c_i is contained/assigned to exactly one scaffold, (ii) the *contiguity* of S is maximized and (iii) the conflicts of S with respect to A are minimized. This optimization problem is not rigorously defined unless one defines precisely the concepts of *contiguity* and *conflict*, but this description captures the spirit of what we want to accomplish. In genome assembly, the assembly contiguity is usually captured by statistical measures like the N50/L50 or the NG50/LG50. The notion of conflict is not easily quantified, and even if it was made precise, this multi-objective optimization problem would be hard to solve. We decompose this problem into two separate steps, namely (a) scaffold detection and (b) gap estimation, as explained below.

3 Method

As said, our proposed method is composed of two phases: scaffold detection and gap estimation. In the first phase, contigs are grouped into scaffolds and the order of contigs in each scaffold is determined. In the second phase, distances between neighboring contigs assigned to scaffolds are estimated. The pipeline of the proposed algorithm is illustrated in Fig. 1.

3.1 Phase 1: Detecting Scaffolds

Phase 1 has three major steps. In Step 1, we align *in silico*-digested chimeric-free contigs to the optical maps (e.g., for a Bionano optical map, we use RefAligner), but not all alignments are used in Step 2. We only consider alignments that (i) exceed a minimum confidence level (e.g., confidence 15 in the case of RefAligner); (ii) do not overlap each other more than a given genomic distance (e.g., 20 kbp) and (iii) do not create conflict with each other. The method we use here to select conflict-free alignments was introduced in our previous work [27]. In Step 2, we compute candidate scaffolds by building the *order graph* and formulating an optimization problem on it. In Step 3, either the exhaustive algorithm or a $\log n$-approximation algorithm is used to solve the optimization problem (depending on the size of the graph) and produce the final scaffolds.

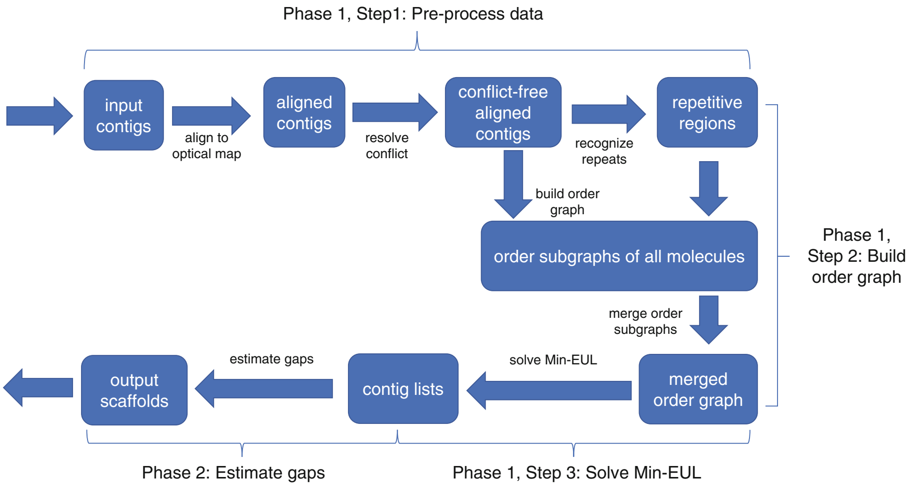

Fig. 1. Pipeline of the proposed algorithm

3.1.1 Building the Order Graph

The order graph O is a directed weighted graph in which each vertex represents a contig. Given two contigs c_i and c_j aligned to an optical molecule o with alignments a_i and a_j, we create a directed edge (c_i, c_j) in O if (i) the starting coordinate of alignment a_i (that we call $a_i.start$ henceforth) is smaller than the starting coordinate of alignment a_j (that we call $a_j.start$ henceforth) and (ii) there is no other alignment a_k such that $a_k.start$ is between $a_i.start$ and $a_j.start$ and (iii) there are no conflict sites between $a_i.end$ and $a_j.start$ on the optical molecule, as defined below. For each alignment a between optical molecule o and contig c, we compute the left overhang l_o and right overhang r_o from o and the left overhang l_c and right overhang r_c from c. The left-end of alignment a is declared a *conflict site* if (i) both l_o and l_c are longer than some minimum length (e.g., 50 kbp) and (ii) at least one restriction enzyme sites appear in both l_o and l_c. A symmetric argument applies to the right-end of the alignment, which determines the values for r_o and r_c.

Directed edge (c_i, c_j) is assigned a weight equal to $\text{qual}(o, a_i.end, a_j.start) * (\text{conf}(a_i)+\text{conf}(a_j))$, where (i) $\text{qual}(o, a_i.end, a_j.start)$ is the *quality* of the region between $a_i.end$ and $a_j.start$ on molecule o (higher is better, defined next) and (ii) $\text{conf}(a)$ is the confidence score provided by RefAligner alignment a (higher is better). The quantity $\text{qual}(o, s, t)$ is defined based on the length of a repetitive region between coordinates (s, t). Based on our experience, assembly mis-joins on optical molecule almost always happen in repetitive regions [15]. Given the length of repetitive region $\text{len\_rep}(o, s, t)$ in base pairs (defined below), we define the quality of o in the interval (s, t) as $\text{qual}(o, s, t) = e^{-\text{len\_rep}(o,s,t)/100000}$. When a_i and a_j have a small overlap (e.g., shorter than 20 kbp), we set $\text{len\_rep}(o, s, t) = 0$.

We recognize repetitive regions in optical molecules based on the distribution of restriction enzyme sites. For a molecule o with n sites, let m_i be the coordinate of the i-th site for $i = 1, \ldots, n$. As said, molecule o can be represented as a list of positions $\{m_i | i = 1, \ldots, n\}$. In order to determine the repetitive regions in o, we slide a window that covers k sites (e.g., $k = 10$ sites). At each position $j = 1, \ldots, n-k+1$, we select window $w_j = \{m_j, \ldots, m_{j+k-1}\}$. While repetitive regions in genome can be highly complex (see, e.g., [40]), we observed only two types of repetitive regions in optical molecules, namely single-site repetitive region (see Fig. 2-A) and two-site repetitive region (see Fig. 2-B). It is entirely possible that more complex repetitive regions exist: if they do, they seem rare. Based on this observation, in order to decide whether window w_j is repetitive, we first compute two lists of pairwise distances between sites, namely $D_{j,1} = \{m_{j+l} - m_{j+l-1} | l = 1, \ldots, k-1\}$ and $D_{j,2} = \{m_{j+l+1} - m_{j+l-1} | l = 1, \ldots, k-2\}$ that we call *distance lists*, then we apply the statistical test described next.

In our statistical test we assume that the values in the distance lists that belong to repetitive regions are independent and identically distributed as a Gaussian. We further assume that each specific distance list ($D_{j,1}$ or $D_{j,2}$) is associated with a Gaussian with a specific mean $\mu_{j,q}$ ($q \in \{1, 2\}$). Finally, we assume that the variance σ^2 is globally shared by all molecules. An estimator of the mean is $\mu_{j,q}$ is $\hat{\mu}_{j,q} = \sum_{i=1}^{k-q} d_i/(k-q)$, where $d_i \in D_{j,q}$ and k is the window size. To

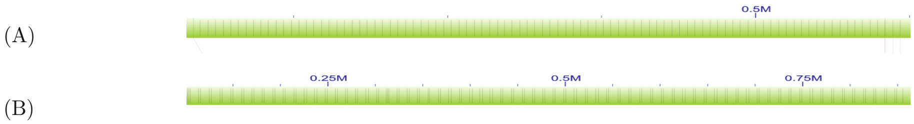

Fig. 2. Examples of single-site repetitive region (A) and two-site repetitive region (B) in optical maps. Observe the small variations in the repetitive patterns in (B)

estimate σ^2, we first get an initial (rough) estimate of the repetitive regions on all molecules. Given a particular $D_{j,q}$, let d_{max} and d_{min} be the maximum and minimum distance in $D_{j,q}$. We declare a distance list $D_{j,q}$ to be *estimated repetitive* if $d_{max} - d_{min}$ is smaller than a given distance (e.g., 1.5 kbp). We collect all estimated repetitive lists in set $R = \{D_p \text{ is estimated repetitive} | p = 1, \ldots, P\}$ and the estimated mean $\hat{\mu}_p$ for each distance list D_p in the set R, where P is the total number of estimated repetitive lists. Then, we define the log likelihood function L as follows (additional details can be found in Appendix, Sect. B)

$$\log L(\sigma^2) = -\frac{\log \sigma^2}{2} \sum_{p=1}^{P} |D_p| - \frac{1}{2\sigma^2} \sum_{p=1}^{P} \sum_{d_i \in D_p} (d_i - \hat{\mu}_p)^2.$$

By maximizing $\log L(\sigma^2)$, the estimator for the variance becomes

$$\hat{\sigma}^2 = \sum_{p=1}^{P} \sum_{d_i \in D_p} (d_i - \hat{\mu}_p)^2 / \sum_{p=1}^{P} |D_p|.$$

Then, we carry out the test on the statistic $d_{max} - d_{min}$ for each $D_{j,q}$. The joint density function of (d_{max}, d_{min}) is

$$f_{d_{max}, d_{min}}(u, v) = n(n-1) f_{d_i}(u) f_{d_i}(v) [F_{d_i}(v) - F_{d_i}(u)]^{n-2}$$

for $-\infty < u < v < +\infty$, where F_{d_i} and f_{d_i} are the distribution function and density function of $d_i \sim N(\hat{\mu}_{j,q}, \hat{\sigma}^2)$, respectively. The density function of $d_{max} - d_{min}$ is

$$f_{d_{max} - d_{min}}(x) = \int_{-\infty}^{+\infty} n(n-1) f_{d_i}(y) f_{d_i}(x+y) [F_{d_i}(x+y) - F_{d_i}(y)]^{n-2} dy,$$

defined when $x \geq 0$ (additional details can be found in Appendix, Sect. C). Let now X be a random variable associated with the distribution $f_{d_{max} - d_{min}}$. If the p-value $P(X > d_{max} - d_{min})$ is greater than a predefined threshold (e.g., 0.001), we accept the null hypothesis and declare that window w_j is repetitive. The repetitive regions for the entire molecule o is the union of all the windows w_j's recognized as repetitive according to the test above.

Once the order graph of each optical molecule is built, we connect all the order graphs which share the same contigs using the association graph introduced in [27]. The association graph is an undirected graph in which each vertex represents an optical molecule and an edge indicates that the two molecules share at least one contig aligned to both of them. We use depth first search (DFS) to first build a spanning forest of the association graph. Then, we traverse each spanning tree and connect the corresponding order subgraph to the final order graph. Every time we add a new graph, new vertices and new edges might be added. If an edge already exist, the weights of the new edges are added to the weights of existing edges.

3.1.2 Generating Scaffolds

Once the order graph O is finalized, we generate the ordered sequence of contigs in each scaffold. In the ideal case, each connected component O_i of O is a directed acyclic graph (DAG) because the genome is one-dimensional and the order of any pair of contigs is unique. In practice however, O_i may contain cycles caused by the inaccuracy of the alignments and mis-joins in optical molecules. To convert each cyclic component O_i into a DAG, we solve the Minimum Feedback Arc Set problem on O_i. In this problem, the objective is to find the minimum subset of edges (called *feedback arc set*) containing at least one edge of every cycle in the input graph. Since the minimum feedback edge set problem is APX-hard, we use the greedy local heuristics introduced in [2] to solve it.

We then break each DAG G_i of connected component O_i into subgraphs as follows. In each subgraph, we require that the order of every pair of vertices to be uniquely determined by the directed edges. This allows us to uniquely determine the order of the contigs for each scaffold. The formal definition of this optimization problem is as follows.

Definition 1 (Minimum Edge Unique Linearization problem)**.** Input: A weighted directed acyclic graph $G = (V, E)$. Output: A subset of edges $E' \subseteq E$ such that (i) in each connected component G'_i of the graph $G' = (V, E - E')$ obtained after removing E', the order of all vertices can be uniquely determined, and (ii) the total weights of the edges in E' is the minimum among all the subset of edges satisfying (i).

In Theorem 1 below, we show that the Minimum Edge Unique Linearization problem (Min-EUL) is NP-hard by proving that it is equivalent to the Minimum Edge Clique Partition problem (Min-ECP), which is know to be NP-hard [10]. In Min-ECP, we are given a general undirected graph, and we need to partition its vertices into disjoint clusters such that each cluster forms a clique and the total weight of the edges between clusters is minimized.

Theorem 1. Min-EUL *is equivalent to* Min-ECP*.*

Proof. First, we show that Min-EUL polynomially reduces to Min-ECP. Given an instance $G = (V, E)$ of Min-EUL, we build an instance $G' = (V', E')$ of Min-ECP as follows. Let $V' = V$. For each pair of vertices $u, v \in V'$ where v is reachable from u, define an undirected edge between u and v in E'. For each directed edge $(u, v) \in E$, set the weight of the corresponding undirected edge

$(u, v) \in E'$ as 1. Set the weights of the other edges in E' as 0. Then it is easy to see that a Min-EUL solution to G' is equivalent to a Min-ECP solution to G and vice versa.

Now we show that Min-ECP polynomially reduces to Min-EUL. Given an instance $G' = (V', E')$ (assuming G' is connected) of Min-ECP, we build an instance $G = (V, E)$ of Min-EUL as follows. Let $V = V'$. Pick any total linear order O of all vertices in V'. For each undirected edge $(u, v) \in E'$ where $\text{rank}(u) < \text{rank}(v)$ in O, define a directed edge from u to v in E and set its weight to be the same as its corresponding undirected edge in E'. For any two vertices $u, v \in V$, where $\text{rank}(u) < \text{rank}(v)$ and $(u, v) \notin E'$, add a new vertex $x_{uv} \in V$ with $\text{rank}(x_{uv}) = \text{rank}(v)$ and a directed edge u to x_{uv} of weight 1 in E. Now for each pair of vertices $u, v \in V$ where $\text{rank}(u) < \text{rank}(v)$ and $(u, v) \notin E$, add a directed edge u to v with weight zero in E. Then it is easy to see that a Min-EUL solution to G corresponds to a Min-ECP solution to G' and vice versa. □

Given the complexity of Min-EUL, we propose an exponential time exact algorithm and a polynomial time $\log n$-approximation algorithm for solving it. To describe the exact algorithm, we need to introduce some notations. A *conjunction* vertex in a DAG is a vertex which has more than one incoming edge or outgoing edge. A *candidate* edge is an edge which connects at least one conjunction vertex. In Theorem 2 below, we prove that the optimal solution E' of Min-EUL must only contain candidate edges. Let E_c be the set of all candidate edges in the DAG G, for each subset E'_j of E_c, we check whether the graph $G' = (V, E - E'_j)$ satisfies requirement (i) in Definition 1 after removing E'_j from G. Among all the feasible E'_j, we produce the set of edges with minimum total weights. To check whether E'_j is feasible, we use a variant of topological sorting which requires one to produce a unique topological ordering. To do so, we require that in every iteration of topological sorting, the candidate node to be added to sorted graph is always unique. Details of this algorithm are shown as Algorithm 1 in Appendix A.

Theorem 2. *The optimal solution E' of* Min-EUL *only contains candidate edges.*

Proof. For sake of contradiction, we assume that E' contains a non-candidate edges (u, v). Since E' is optimal, $G' = (V, E - E')$ satisfies condition (i) in Definition 1. Since both u and v are conjunction vertices, u has only one incoming edge and v has only one outgoing edge. Therefore, by adding (u, v) to $G' = (V, E - E')$, we still satisfy condition (i) in Definition 1. Since the weight of (u, v) is positive, the total weight of $E - E' + \{(u, v)\}$ is larger than $E - E'$. Therefore $E' - \{(u, v)\}$ is optimal, contradicting the optimality of E'. □

As said, Min-EUL is equivalent to Min-ECP (Theorem 1). In addition, the authors of [10] showed that for any instance of Min-ECP one can find an equivalent instance of the Minimum Disagreement Correlation Clustering problem. As a consequence, any algorithm for the Minimum Disagreement Correlation Clustering problem could be used to solve Min-EUL. In our tool OMGS, we implemented a $O(\log n)$-approximation algorithm based on linear programming, originally proposed in [9]. Standard linear programming packages (e.g., GLPK or CPLEX) are used to solve the linear program. We use the exact algorithm for DAGs with no more than twenty candidate edges, and the approximation algorithm for larger DAGs.

3.2 Phase 2: Estimating Gaps

Let $s = \{c_i | i = 1, \ldots, h\}$ be one of the scaffold generated in Phase 1 where each c_i is a contig. In Phase 2, we estimate the length l_i of the gap between each pair c_i and c_{i+1} of adjacent contigs. We estimate all gap lengths $L = \{l_i | i = 1, \ldots, h-1\}$ at the same time using the distances between the contigs provided by the alignments and the corresponding order subgraphs. We assume that each l_i is chi-square distributed with α_i degrees of freedom. The choice of chi-square distribution is due to its additive properties, namely the sum of independent chi-squared variables is also chi-squared distributed. Recall that each order subgraph O_k provides an unique ordering $x_k = \{c_j | j = 1, \ldots, r\}$ of the contigs aligned to molecule o_k, while the coordinates of the alignment provide the distances between all pairs of adjacent contigs c_j and c_{j+1} as $y_k = \{d_j | j = 1, \ldots, r-1\}$. We use the distances d_j as samples to estimate gap lengths l_i. If edge (c_j, c_{j+1}) in O_k is removed in the order graph O when solving MIN-EUL in Phase 1, d_j will be considered not reliable and removed from y_k.

In the ideal case, d_j should be a sample of a single l_i (i.e., $c_j c_{j+1}$ in x_k corresponds to $c_p c_{p+1}$ in s). In practice however, $c_j c_{j+1}$ in x_k will corresponds to a different pair $c_p c_q$ in s where $q > p + 1$ (i.e., $c_{p+1} \ldots c_{q-1}$ are missing from the order subgraph because some alignments with low confidence were removed in Step 1 of Phase 1). In this situation, after subtracting the length of missing contigs from d_j, $d_j - \sum_{c=c_{p+1}}^{c_{q-1}} |c|$ is a sample of $\sum_{i=p}^{q-1} l_i$ where $|c|$ represents the length of contig c. Since $l_p, \ldots, l_{q-1}$ are independent chi-square random variables, $\sum_{i=p}^{q-1} l_i$ is chi-square distributed with degree of freedom $\sum_{i=p}^{q-1} \alpha_i$, so that the log likelihood of this sample is

$$\log l = (\beta - 1) \log \gamma - \frac{\gamma}{2} - \beta \log 2 - \log \Gamma(\beta).$$

where $\beta = \sum_{i=p}^{q-1} \frac{\alpha_i}{2}$, $\gamma = d_j - \sum_{c=c_{p+1}}^{c_{q-1}} |c|$ and Γ is the gamma function (additional details can be found in Appendix, Sect. D). The total log likelihood is the sum of the log likelihoods across all samples. To find the α_i maximizing the total log likelihood, we use the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [1]. Since the mean of a chi-square distribution equals its degree of freedom, we obtain the estimated gaps $\hat{l}_i = \hat{\alpha}_i$. For the case in which the l_i are pre-estimated as negative in the first step, the second and third steps are ignored and the pre-estimated distances are used as final estimates.

Finally, we add $\lceil \hat{l}_i \rceil$ nucleotides (represented by `N`s) between each pair of contigs c_i and c_{i+1}. When $\hat{l}_i < 0$, we add exactly 100 `N`s between c_i and c_{i+1}, which is the convention for a gap of unknown length.

4 Experimental Results

We compared OMGS against KSU SewingMachine (version 1.0.6, released in 2015) and Bionano HybridScaffold (version 4741, released in 2016) which, to the best of our knowledge, are the only available scaffolding tools for Bionano Genomics optical maps. All tools were run with default parameters, unless otherwise specified. We collected experimental results on scaffolds of (i) cowpea (*Vigna unguiculata*) and (ii) fruit fly (*Drosophila melanogaster*).

4.1 Experimental Results on Cowpea

Cowpea is a diploid with a chromosome number $2n = 22$ and an estimated genome size of 620 Mb. We sequenced the cowpea genome using single-molecule real-time sequencing (Pacific Biosciences RSII). A total of 87 SMRT cells yielded about 6M reads for a total of 56.84 Gbp (91.7x genome equivalent). We tested the three scaffolding tool on a high-quality assembly produced by Canu [3,18] with parameters `corMhapSensitivity=high` and `corOutCoverage=100`, then polished it with Quiver. We used Chimericognizer to detect and break chimeric contigs, using seven other assemblies generated by Canu, Falcon [6] and ABruijn [20] as explained in [26].

In addition to standard contiguity statistics (N50[1], L50[2]), total assembled size and scaffold length distribution, we determined incorrect/chimeric scaffolds by comparing them against the high-density genetic map available from [24]. We BLASTed 121bp-long design sequence for the 51,128 genome-wide SNPs described in [24] against each assembly, then we identified which contigs had SNPs mapped to them, and what linkage group (chromosome) of the genetic map those mapped SNPs belonged to. Chimeric contigs were revealed when their mapped SNPs belonged to more than one linkage group. The last line of Tables 1 and 2 report the total size of contigs in each assembly for which (i) they have at least one SNPs mapped to it and (ii) all SNPs belong to the same linkage group (i.e., likely to be non-chimeric).

As said, the three scaffolding tools were run on a chimera-free assembly of cowpea described above using two available Bionano Genomics optical maps (the first obtained using the BspQI nicking enzyme, and the second obtained with the BssSI nicking enzyme). Since SewingMachine can only use a single optical map, we alternated the optical maps in input (BspQI map first, then BssSI and vice versa). SewingMachine provides two outputs depending on the minimum allowed alignment confidence, namely ‘default’ and ‘relax’. Mode ‘relax’ considers more alignments than ‘default’, but it has a higher chance of introducing mis-joins. HybridScaffold failed on the BssSI map, so we could not test it on alternating maps.

Table 1 shows that when using a single optical map, OMGS can generate comparable or better scaffolds than SewingMachine and HybridScaffold.

[1] Length for which the set of contigs/scaffolds of that length or longer accounts for at least half of the assembly size.

[2] Minimum number of contigs/scaffolds accounting for at least half of the assembly.

With two optical maps, OMGS' correctness ("contigs/scaffolds with 100% consistent LG") and contiguity (N50) are significantly better than other two tools. Observe that OMGS' correctness ("contigs/scaffolds with 100% consistent LG") is even better than the input assembly. This can happen when contigs with SNPs belonging to same linkage group are scaffolded with contigs that have no SNP.

We also compared the performance of OMGS, SewingMachine and HybridScaffold when using optical maps corrected by Chimericognizer (on the same cowpea assembly). Observe in Table 2 that OMGS, SewingMachine and HybridScaffold increased the correctness but decreased the contiguity when the corrected BspQI optical map was used. The results on the corrected BssSI optical map or both corrected optical maps did not change significantly. But again, OMGS produced better scaffolds than SewingMachine and HybridScaffold.

4.2 Experimental Results on *D. Melanogaster*

D. melanogaster has four pairs of chromosomes: three autosomes, and one pair of sex chromosomes. The fruit fly's genome is about 139.5 Mb. We downloaded three *D. melanogaster* assemblies generated in [36] (https://github.com/danrdanny/Nanopore_ISO1). The first assembly (295 contigs, total size 141 Mb, N50 = 3 Mb) was generated using Canu [3,18] on Oxford Nanopore (ONT) reads longer than 1 kb. The second assembly (208 contigs, total size 132 Mb, N50 = 3.9 Mb) was generated using MiniMap and MiniAsm [19] using only ONT reads. The third assembly (339 contigs, total size 134 Mb, N50 = 10 Mb) was generated by Platanus [16] and Dbg2Olc [39] using 67.4x of Illumina paired-end reads and the longest 30x ONT reads. The first and third assemblies were polished using nanopolish [21] and Pilon [38]. The Bionano optical for *D. melanogaster* map was provided by the authors of [36]. This BspQI optical map (363 molecules, total size = 246 Mb, N50 = 841 kb) was created using IrysSolve 2.1 from 78,397 raw Bionano molecules (19.9 Gb of data with a mean read length 253 kb).

As said, all tools were run with default parameters, with the exception of OMGS' minimum confidence, which was set at 20 (default is 15). To evaluate the performance of OMGS, HybridScaffold and SewingMachine, we compared their output scaffolds to the high-quality reference genome of *D. melanogaster* (release 6.21, downloaded from FlyBase). We reported the total length of correct/non-chimeric scaffolds as a measure of the overall correctness. To determine which scaffolds were incorrect/chimeric we first selected BLAST alignments of the scaffolds against the reference genome which had an e-value lower than 1e-50 and an alignment length higher than 30 kbp. We defined a scaffold S to be *chimeric* if S had at least two high-quality alignments which satisfied one or more of the following conditions: (i) S aligned to different chromosomes; (ii) the orientation of S's alignments were different; or (iii) the difference between the distance of alignments on the scaffold and the distance of alignments on the reference sequence was larger than 100 Kbp.

Table 1. Comparing OMGS, SewingMachine (SM) and HybridScaffold (HS) on a cowpea assembly using one or two optical maps. Numbers in boldface highlight the best N50 and scaffold consistency with the genetic map for one map (BspQI and BssSI) or two maps ('A + B' refers to the use of map A followed by map B, 'A&B' refers to the use of both maps at the same time).

		One optical map							
		BspQI				BssSI			
	Input	SM (default)	SM (relax)	HS	OMGS	SM (default)	SM (relax)	HS	OMGS
contig/scaffold N50 (bp)	5,633,882	13,154,336	13,154,336	12,211,658	**14,339,314**	10,620,326	10,886,079	N/A	**11,536,649**
contig/scaffold L50	28	15	15	17	14	18	17	N/A	15
total assembled (bp)	511,101,122	521,209,608	521,210,640	516,455,893	518,265,608	518,987,660	518,945,404	N/A	518,252,638
# contigs/scaffolds	948	863	863	877	847	849	846	N/A	832
# contigs/scaffolds ≥100kbp	269	185	185	198	170	177	174	N/A	165
# contigs/scaffolds ≥1Mbp	94	59	59	63	56	63	62	N/A	59
# contigs/scaffolds ≥10Mbp	10	20	20	21	20	18	18	N/A	17
contigs/scaffolds with consistent LG (bp)	425,812,490	404,408,642	404,409,674	381,974,417	**410,552,582**	**425,572,265**	425,530,009	N/A	424,143,108

		Two optical maps				
		BspQI+BssSI	BspQI+BssSI	BssSI+BspQI	BssSI+BspQI	BspQI&BssSI
	Input	SM (default)	SM (relax)	SM (default)	SM (relax)	OMGS
contig/scaffold N50 (bp)	5,633,882	14,892,230	14,892,230	13,527,997	14,892,235	**16,364,046**
contig/scaffold L50	28	13	13	14	13	12
total assembled (bp)	511,101,122	525,577,823	525,198,231	525,827,900	525,105,345	521,324,385
# contigs/scaffolds	948	822	823	816	814	802
# contigs/scaffolds ≥100kbp	269	149	150	145	143	137
# contigs/scaffolds ≥1Mbp	94	46	46	48	46	44
# contigs/scaffolds ≥10Mbp	10	21	21	22	22	21
contigs/scaffolds with consistent LG (bp)	425,812,490	385,449,577	385,069,985	425,678,421	403,637,207	**432,639,234**

Table 2. Comparing OMGS, SewingMachine (SM) and HybridScaffold (HS) on a cowpea assembly using optical maps corrected by Chimericognizer. Numbers in boldface highlight the best N50 and scaffold consistency with the genetic map for one map (BspQI and BssSI) or two maps ('A + B' refers to the use of map A followed by map B, 'A&B' refers to the use of both maps at the same time).

		One optical map							
		BspQI				BssSI			
	Input	SM (default)	SM (relax)	HS	OMGS	SM (default)	SM (relax)	HS	OMGS
contig/scaffold N50 (bp)	5,633,882	12,487,373	12,487,373	12,495,655	**13,505,314**	9,420,899	10,886,079	N/A	**11,256,770**
contig/scaffold L50	28	16	16	15	14	19	17	N/A	16
total assembled (bp)	511,101,122	519,785,777	519,785,777	515,519,585	518,405,022	517,678,278	517,636,022	N/A	517,318,151
# contigs/scaffolds	948	863	863	871	849	854	851	N/A	837
# contigs/scaffolds ≥100kbp	269	185	185	192	172	182	179	N/A	169
# contigs/scaffolds ≥1Mbp	94	60	60	60	58	66	65	N/A	62
# contigs/scaffolds ≥10Mbp	10	19	19	19	19	17	17	N/A	17
contigs/scaffolds with consistent LG (bp)	425,812,490	413,819,557	413,819,557	402,840,302	**421,466,164**	**424,262,883**	424,220,627	N/A	423,117,331

		Two optical maps				
		BspQI+BssSI	BspQI+BssSI	BssSI+BspQI	BssSI+BspQI	BspQI&BssSI
	Input	SM (default)	SM (relax)	SM (default)	SM (relax)	OMGS
contig/scaffold N50 (bp)	5,633,882	14,354,752	14,354,752	13,527,997	14,892,235	**16,364,046**
contig/scaffold L50	28	14	14	14	13	12
total assembled (bp)	511,101,122	523,520,329	523,139,705	521,540,185	525,105,345	520,697,623
# contigs/scaffolds	948	823	824	817	814	805
# contigs/scaffolds ≥100kbp	269	150	151	146	143	139
# contigs/scaffolds ≥1Mbp	94	48	48	48	46	46
# contigs/scaffolds ≥10Mbp	10	21	21	21	22	21
contigs/scaffolds with consistent LG (bp)	425,812,490	402,344,751	401,964,127	420,269,616	403,637,207	**431,921,182**

Table 3 reports the main statistics for the three *D. melanogaster* scaffolded assemblies. Even with one map, OMGS' scaffolds are better than SewingMachine and HybridScaffold.

Table 3. Comparing OMGS, SewingMachine (SM) and HybridScaffold (HS) on three *D. melanogaster* assemblies (produced by MiniAsm, Canu, and Dbg2Olc) using the BspQI optical map. Numbers in boldface highlight the best N50 and the best scaffold consistency with the reference genome

	MiniAsm assembly				
	Input	SM (default)	SM (relax)	HS	OMGS
contig/scaffold N50 (bp)	3,866,686	4,494,241	**4,906,224**	3,866,686	**4,906,224**
contig/scaffold L50	9	8	8	9	8
total assembled (bp)	131,856,353	132,480,826	133,233,999	132,138,056	132,838,677
# contigs/scaffolds	208	205	203	206	206
# contigs/scaffolds ≥100kbp	85	82	80	83	83
# contigs/scaffolds ≥1Mbp	26	26	25	26	25
# contigs/scaffolds ≥10Mbp	2	2	2	2	2
non-chimeric contigs/scaffolds (bp)	131,317,873	125,305,638	**132,695,519**	131,174,201	132,300,197

	Canu assembly				
	Input	SM (default)	SM (relax)	HS	OMGS
contig/scaffold N50 (bp)	3,004,953	3,004,953	3,004,953	3,918,649	**5,336,340**
contig/scaffold L50	11	11	11	10	7
total assembled (bp)	140,720,404	140,923,974	140,923,974	140,867,226	140,960,395
# contigs/scaffolds	295	291	291	286	280
# contigs/scaffolds ≥100kbp	111	107	107	102	96
# contigs/scaffolds ≥1Mbp	31	31	31	29	27
# contigs/scaffolds ≥10Mbp	1	1	1	1	5
non-chimeric contigs/scaffolds (bp)	140,720,404	140,923,974	140,923,974	140,867,226	**140,960,395**

	Dbg2Olc assembly				
	Input	SM (default)	SM (relax)	HS	OMGS
contig/scaffold N50 (bp)	10,113,899	11,223,142	11,223,142	12,785,467	**12,928,771**
contig/scaffold L50	6	5	5	5	4
total assembled (bp)	134,109,164	134,164,629	134,164,629	134,162,857	134,208,377
# contigs/scaffolds	339	337	337	331	327
# contigs/scaffolds ≥100kbp	78	76	76	70	66
# contigs/scaffolds ≥1Mbp	22	22	22	17	16
# contigs/scaffolds ≥10Mbp	6	6	6	5	7
non-chimeric contigs/scaffolds (bp)	134,109,164	134,164,629	134,164,629	134,162,857	**134,208,377**

5 Conclusions

We presented a scaffolding tool called OMGS for improving the contiguity of *de novo* genome assembly using one or multiple optical maps. OMGS solves several optimization problems to generate scaffolds with optimal contiguity and correctness. Experimental results on *V. unguiculata* and *D. melanogaster* clearly demonstrate that OMGS outperforms SewingMachine and HybridScaffold both in contiguity and correctness using multiple optical maps.

Acknowledgements. This work was supported in part by National Science Foundation grants IIS-1814359, IOS-1543963, IIS-1526742 and IIS-1646333, the Natural Science Foundation of China grant 61772197 and the National Key Research and Development Program of China grant 2018YFC0910404.

Appendix

A DAG Unique Ordering

Algorithm 1. Sketch of the algorithm for checking whether a DAG provides an unique ordering

```
procedure ORDER_UNIQUENESS_CHECK($G = (V, E)$)
    $S$ = nodes with no incoming edges
    while $S \neq \emptyset$ do
        if $|S| > 1$ then
            return $False$
        remove a node $n$ from $S$
        for each node $m$ with an edge $e = (n, m)$ do
            remove edge $e$ from the $E$
            if $m$ has no other incoming edges then
                insert $m$ into $S$
    return $True$
```

B Statistical Test for Repetitive Regions

Here we provide additional details for the estimation of σ^2 during the analysis of repetitive regions. Recall that we collect all estimated repetitive lists in set $R = \{D_p \text{ is estimated repetitive} | p = 1, \ldots, P\}$ and the estimated mean $\hat{\mu}_p$ for each distance list D_p in the set R, where P is the total number of estimated repetitive lists. For each D_p, the distances d_i's are distributed as a Gaussian with mean $\hat{\mu}_p$ and variance σ^2. According to the density function of Gaussian distribution, the log likelihood of one D_p is

$$-\frac{|D_p|}{2}\log(2\pi) - \frac{|D_p|}{2}\log\sigma^2 - \frac{1}{2\sigma^2}\sum_{d_i \in D_p}(d_i - \hat{\mu}_p)^2.$$

The total log likelihood is the sum of the log likelihoods across all D_p's in R, which is

$$\log L(\sigma^2) = -\frac{\sum_{p=1}^{P}|D_p|}{2}\log\sigma^2 - \frac{1}{2\sigma^2}\sum_{p=1}^{P}\sum_{d_i \in D_p}(d_i - \hat{\mu}_p)^2,$$

after ignoring all terms not related to σ^2. To maximize $\log L(\sigma^2)$, we require that the derivative of total log likelihood

$$\frac{\partial \log L(\sigma^2)}{\partial \sigma^2} = 0,$$

that is,

$$-\frac{\sum_{p=1}^{P}|D_p|}{2\sigma^2}+\frac{1}{2(\sigma^2)^2}\sum_{p=1}^{P}\sum_{d_i\in D_p}(d_i-\hat{\mu}_p)^2=0.$$

After some simplification, the estimator for variance becomes

$$\hat{\sigma}^2=\frac{\sum_{p=1}^{P}\sum_{d_i\in D_p}(d_i-\hat{\mu}_p)^2}{\sum_{p=1}^{P}|D_p|}.$$

C Density Function of $d_{max}-d_{min}$

Here we provide additional details for calculating the density function of $d_{max}-d_{min}$. It is well-known that the joint density function of order statistics is

$$f_{X(i),X(j)}(u,v)=\frac{n!}{(i-1)!(j-1-i)!(n-j)!}f_x(u)f_x(v)[F_x(u)]^{i-1}[F_x(v)-F_x(u)]^{j-1-i}[1-F_x(v)]^{n-j} \tag{1}$$

for $-\infty<u<v<+\infty$, where $X(i)$ and $X(j)$ are the i-th and j-th order statistics in $X_1,\ldots,X_n$ and F_x and f_x are the distribution function and density function of each X_i, respectively. Using (1), the joint density function of (d_{max},d_{min}) can be expressed as

$$f_{d_{max},d_{min}}(u,v)=n(n-1)f_{d_i}(u)f_{d_i}(v)[F_{d_i}(v)-F_{d_i}(u)]^{n-2}$$

for $-\infty<u<v<+\infty$, where F_{d_i} and f_{d_i} are the distribution function and density function of $d_i\sim N(\hat{\mu}_{j,q},\hat{\sigma}^2)$, respectively.

Now, let $X=d_{max}-d_{min}$ and $Y=d_{min}$. Then $d_{max}=X+Y$ and $d_{min}=Y$, and the corresponding Jacobian determinant is

$$J=\begin{vmatrix}\partial d_{\max}/\partial X & \partial d_{\max}/\partial Y\\ \partial d_{\min}/\partial X & \partial d_{\min}/\partial Y\end{vmatrix}=\begin{vmatrix}1&1\\0&1\end{vmatrix}=1.$$

Thus, the joint density function of $(X,\,Y)$ is given by

$$f_{X,Y}(x,y)=f_{d_{max},d_{min}}(x+y,y)|J|=n(n-1)f_{d_i}(y)f_{d_i}(x+y)[F_{d_i}(x+y)-F_{d_i}(y)]^{n-2},$$

where $x\geq 0$ and $-\infty<y<+\infty$. By integrating over Y, the density function of $X=d_{max}-d_{min}$ becomes

$$f_{d_{max}-d_{min}}(x)=\int_{-\infty}^{+\infty}n(n-1)f_{d_i}(y)f_{d_i}(x+y)[F_{d_i}(x+y)-F_{d_i}(y)]^{n-2}dy,x\geq 0.$$

D Gap Estimation

Here we provide additional details for calculating the log likelihood function when estimating gaps. Recall that $l_p, \ldots, l_{q-1}$ are independent chi-square random variables, and $\sum_{i=p}^{q-1} l_i$ is chi-square distributed with degree of freedom $\sum_{i=p}^{q-1} \alpha_i$. Since the density function of a chi-square random variable X with degree of freedom k is

$$f_X(x) = \frac{1}{2^{k/2}\Gamma(k/2)} x^{k/2-1} e^{-x/2}$$

where Γ is the gamma function, the likelihood of $\sum_{i=p}^{q-1} l_i$ with observation

$$\gamma = d_j - \sum_{c=c_{p+1}}^{c_{q-1}} |c|$$

is

$$\frac{1}{2^{\beta}\Gamma(\beta)} \gamma^{\beta-1} e^{-\gamma/2},$$

where $\beta = \sum_{i=p}^{q-1} \frac{\alpha_i}{2}$. Therefore, the log likelihood function for one sample is

$$\log l = (\beta - 1)\log\gamma - \frac{\gamma}{2} - \beta\log 2 - \log\Gamma(\beta).$$

The total log likelihood is the sum of the log likelihoods across all samples.

References

1. Avriel, M.: Nonlinear Programming: Analysis and Methods. Courier Corporation, New York (2003)
2. Baharev, A., Schichl, H., Neumaier, A., Achterberg, T.: An exact method for the minimum feedback arc set problem, vol. 10, pp. 35–60. University of Vienna (2015)
3. Berlin, K., Koren, S., Chin, C.-S., Drake, J.P., Landolin, J.M., Phillippy, A.M.: Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. Nature Biotechnol. **33**(6), 623 (2015)
4. Bickhart, D.M., et al.: Single-molecule sequencing and chromatin conformation capture enable de novo reference assembly of the domestic goat genome. Nature Genet. **49**(4), 643 (2017)
5. Boetzer, M., Henkel, C.V., Jansen, H.J., Butler, D., Pirovano, W.: Scaffolding pre-assembled contigs using sspace. Bioinformatics **27**(4), 578–579 (2010)
6. Chin, C.-S., et al.: Phased diploid genome assembly with single-molecule real-time sequencing. Nature Methods **13**(12), 1050 (2016)
7. Daccord, N., et al.: High-quality de novo assembly of the apple genome and methylome dynamics of early fruit development. Nature Genet. **49**(7), 1099 (2017)
8. Dayarian, A., Michael, T.P., Sengupta, A.M.: SOPRA: scaffolding algorithm for paired reads via statistical optimization. BMC Bioinform. **11**(1), 345 (2010)

9. Demaine, E.D., Immorlica, N.: Correlation clustering with partial information. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) APPROX/RANDOM -2003. LNCS, vol. 2764, pp. 1–13. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45198-3_1
10. Dessmark, A., Jansson, J., Lingas, A., Lundell, E.-M., Persson, M.: On the approximability of maximum and minimum edge clique partition problems. Int. J. Found. Comput. Sci. **18**(02), 217–226 (2007)
11. Donmez, N., Brudno, M.: SCARPA: scaffolding reads with practical algorithms. Bioinformatics **29**(4), 428–434 (2012)
12. Gao, S., Nagarajan, N., Sung, W.-K.: Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences. In: Bafna, V., Sahinalp, S.C. (eds.) RECOMB 2011. LNCS, vol. 6577, pp. 437–451. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20036-6_40
13. Gritsenko, A.A., Nijkamp, J.F., Reinders, M.J.T., de Ridder, D.: GRASS: a generic algorithm for scaffolding next-generation sequencing assemblies. Bioinformatics **28**(11), 1429–1437 (2012)
14. Hunt, M., Newbold, C., Berriman, M., Otto, T.D.: A comprehensive evaluation of assembly scaffolding tools. Genome Biol. **15**(3), R42 (2014)
15. Jiao, W.-B., et al.: Improving and correcting the contiguity of long-read genome assemblies of three plant species using optical mapping and chromosome conformation capture data. Genome Res. **27**(5), 778–786 (2017)
16. Kajitani, R., et al.: Efficient de novo assembly of highly heterozygous genomes from whole-genome shotgun short reads. Genome Res. **24**(8), 1384–1395 (2014). https://doi.org/10.1101/gr.170720.113
17. Koren, S., Treangen, T.J., Pop, M.: Bambus 2: scaffolding metagenomes. Bioinformatics **27**(21), 2964–2971 (2011)
18. Koren, S., Walenz, B.P., Berlin, K., Miller, J.R., Bergman, N.H., Phillippy, A.M.: Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. Genome Res. **27**(5), 722–736 (2017). https://doi.org/10.1101/gr.215087.116
19. Li, H.: Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. Bioinformatics **32**(14), 2103–2110 (2016)
20. Lin, Y., Yuan, J., Kolmogorov, M., Shen, M.W., Chaisson, M., Pevzner, P.A.: Assembly of long error-prone reads using de Bruijn graphs. Proc. National Acad. Sci. **113**(52), E8396–E8405 (2016)
21. Loman, N.J., Quick, J., Simpson, J.T.: A complete bacterial genome assembled de novo using only nanopore sequencing data. Nature Methods **12**(8), 733 (2015)
22. Luo, R., et al.: SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler. GigaScience **1**(1), 18 (2012)
23. Mascher, M., et al.: A chromosome conformation capture ordered sequence of the barley genome. Nature **544**(7651), 427 (2017)
24. Muñoz-Amatriaín, M., et al.: Genome resources for climate-resilient cowpea, an essential crop for food security. Plant J. **89**(5), 1042–1054 (2017)
25. Nagarajan, N., Read, T.D., Pop, M.: Scaffolding and validation of bacterial genome assemblies using optical restriction maps. Bioinformatics **24**(10), 1229–1235 (2008)
26. Pan, W., Lonardi, S.: Accurate detection of chimeric contigs via bionano optical maps. Bioinformatics (2018, in press)
27. Pan, W., Wanamaker, S.I., Ah-Fong, A.M.V., Judelson, H.S., Lonardi, S.: Novo&stitch: accurate reconciliation of genome assemblies via optical maps. Bioinformatics **34**(13), i43–i51 (2018)

28. Pendleton, M., et al.: Assembly and diploid architecture of an individual human genome via single-molecule technologies. Nature Methods **12**(8), 780 (2015)
29. Pop, M., Kosack, D.S., Salzberg, S.L.: Hierarchical scaffolding with Bambus. Genome Res. **14**(1), 149–159 (2004)
30. Saha, S., Rajasekaran, S.: Efficient and scalable scaffolding using optical restriction maps. BMC Genomics **15**(5), S5 (2014)
31. Salmela, L., Mäkinen, V., Välimäki, N., Ylinen, J., Ukkonen, E.: Fast scaffolding with small independent mixed integer programs. Bioinformatics **27**(23), 3259–3265 (2011)
32. Samad, A., Huff, E.F., Cai, W., Schwartz, D.C.: Optical mapping: a novel, single-molecule approach to genomic analysis. Genome Res. **5**(1), 1–4 (1995)
33. Shelton, J.M., et al.: Tools and pipelines for BioNano data: molecule assembly pipeline and FASTA super scaffolding tool. BMC Genomics **16**(1), 734 (2015)
34. Simpson, J.T., Durbin, R.: Efficient de novo assembly of large genomes using compressed data structures. Genome Res. **22**(3), 549–556 (2012)
35. Simpson, J.T., Wong, K., Jackman, S.D., Schein, J.E., Jones, S.J.M., Birol, I.: ABySS: a parallel assembler for short read sequence data. Genome Res. **19**(6), 1117–1123 (2009). https://doi.org/10.1101/gr.089532.108
36. Solares, E.A., et al.: Rapid low-cost assembly of the Drosophila melanogaster reference genome using low-coverage, long-read sequencing. G3: Genes Genomes Genet. **8**(10), 3143–3154 (2018)
37. Tang, H., et al.: ALLMAPS: robust scaffold ordering based on multiple maps. Genome Biol. **16**(1), 3 (2015)
38. Walker, B.J., et al.: Pilon: an integrated tool for comprehensive microbial variant detection and genome assembly improvement. PloS One **9**(11), e112963 (2014)
39. Ye, C., Hill, C.M., Wu, S., Ruan, J., Ma, Z.S.: DBG2OLC: efficient assembly of large genomes using long erroneous reads of the third generation sequencing technologies. Sci. Rep. **6** (2016). Article number: 31900
40. Zheng, J., Lonardi, S.: Discovery of repetitive patterns in DNA with accurate boundaries. In: Fifth IEEE Symposium on Bioinformatics and Bioengineering (BIBE 2005), pp. 105–112, October 2005