A Graph-Theoretic Barcode Ordering Model for Linked-Reads

Yoann Dufresne
Department of Computational Biology, C3BI USR 3756 CNRS, Institut Pasteur, Paris, France
yoann.dufresne@pasteur.fr

Chen Sun
Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA
chensunx@gmail.com

Pierre Marijon
Center for Bioinformatics, Saarland University, Saarland Informatics Campus,
Saarbrücken, Germany
pmarijon@mmci.uni-saarland.de

Dominique Lavenier
IRISA, Inria, Université de Rennes, France
lavenier@irisa.fr

Cedric Chauve
Department of Mathematics, Simon Fraser University, Burnaby, Canada
LaBRI, Université de Bordeaux, France
cedric.chauve@sfu.ca

Rayan Chikhi
Department of Computational Biology, C3BI USR 3756 CNRS, Institut Pasteur, Paris, France
rayan.chikhi@pasteur.fr

Abstract

Considering a set of intervals on the real line, an interval graph records these intervals as nodes and their intersections as edges. Identifying (i.e. merging) pairs of nodes in an interval graph results in a multiple-interval graph. Given only the nodes and the edges of the multiple-interval graph without knowing the underlying intervals, we are interested in the following questions. Can one determine how many intervals correspond to each node? Can one compute a walk over the multiple-interval graph nodes that reflects the ordering of the original intervals? These questions are closely related to linked-read DNA sequencing, where barcodes are assigned to long molecules whose intersection graph forms an interval graph. Each barcode may correspond to multiple molecules, which complicates downstream analysis, and corresponds to the identification of nodes of the corresponding interval graph. Resolving the above graph-theoretic problems would facilitate analyses of linked-reads sequencing data, through enabling the conceptual separation of barcodes into molecules and providing, through the molecules order, a skeleton for accurately assembling the genome. Here, we propose a framework that takes as input an arbitrary intersection graph (such as an overlap graph of barcodes) and constructs a heuristic approximation of the ordering of the original intervals.

2012 ACM Subject Classification Applied computing → Bioinformatics

Keywords and phrases DNA sequencing, graph algorithms, linked-reads, interval graphs, cliques

Digital Object Identifier 10.4230/LIPIcs.WABI.2020.11

Supplementary Material `https://gitlab.pasteur.fr/ydufresne/linkedreadsmoleculeordering`

Funding *Cedric Chauve*: Natural Sciences and Engineering Research Council of Canada Discovery Grant RGPIN-03986.
Rayan Chikhi: INCEPTION project (PIA/ANR-16-CONV-0005).

20th International Workshop on Algorithms in Bioinformatics (WABI 2020).
Editors: Carl Kingsford and Nadia Pisanti; Article No. 11; pp. 11:1–11:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Acknowledgements The authors are grateful to Paul Medvedev and Jean-Stéphane Varré for preliminary work and discussions and to Marthe Bonamy for discussions about interval graphs models.

1 Introduction

A well-known limitation of short-read sequencing is that it does not provide long-range information, which is crucial to many biological endeavors, such as genome assembly and structural variant identification. There have been several sequencing technologies developed to overcome this limitation, such as matepair libraries, Hi-C, and long reads (PacBio & Oxford Nanopore). Another family of approaches is linked-read sequencing, which includes 10XGenomics Chromium, stLFR [28], CPTv2-seq [32] and TELL-seq [8]. In these approaches, DNA is cloned and cut into large molecules (10-100 kbp), which are then isolated (physically in 10X, or virtually using beads) and sheared into shorter fragments. A barcode is attached to each short fragment for identification of its originating molecule. Importantly, barcodes do not uniquely identify molecules: several molecules are typically labeled with the same barcode. The number of different barcodes differ from 150k for CPTv2 to 2 billions for TELL-seq. Fragments are then sequenced using a standard short-read protocol (e.g Illumina).

Linked-reads have been used to assemble genomes [29], detect complex structural variants [16], and more recently assemble metagenomes [4]. A common challenge faced by most linked-read methods is that in order to make use of the linking information, the reads within each barcode should be first separated into their constituent molecules. More formally, for each read r, we would like to find the identifier $mi(r)$ of its originating molecule, given as input an observed identifier $b(mi(r))$, where $b(x)$ associates a barcode identifier to a molecule x. Note that the image of b (all barcodes) is significantly smaller than its domain (all molecules), hence b can be viewed as a non-invertible hash function. Currently, this problem is being tackled, one way or another, as part of any method using linked-read data. Switching from a read-centric view to a molecule-centric view opens the possibility of using methodology similar to long-read overlap graphs. Finding an ordering of barcodes that reflects the underlying order of molecules would indeed greatly facilitate and decrease errors during the scaffolding stage of genome assembly. As noted by the authors of the ARCS scaffolder [31], different molecules having the same barcode can induce false joins in a scaffolding algorithm, resulting in misassemblies.

Linked-read mapping tools such as `longranger` or `ema` [25] are able to infer molecules by clustering mapping locations of reads from the same barcode. While such reference-based algorithms are often applicable, they do not replace the need for *de novo* algorithms. The quality of reference-based algorithms is related to the quality of the assembly, since clusters cannot be identified across different contigs. When the genome or metagenome references are in a draft state, molecules will frequently span multiple contigs, preventing their identification. Moreover, in many situations the reference is simply unavailable.

To the best of our knowledge, the barcode ordering problem has not been previously studied, and the assignment of molecule identifiers to reads without a reference has only been previously studied in [11], where it was referred to as *barcode deconvolution*. The authors first constructed a bipartite graph between reads and barcodes. An edge (r, b) was added when a k-mer of read r was found in another read of barcode b. Then a second graph was constructed with reads as nodes, and edges indicating whether two reads were connected to sufficiently many common barcodes in the bipartite graph. Finally, the second graph was clustered and each cluster reflected reads from the same molecule. This algorithm was implemented in a software called `Minerva`. We note that Minerva only assigns molecules identifiers to a fraction of the reads. In our tests on a simulated *E. coli* dataset, Minerva reported results for 12% of the reads, inferring around 50% of the true number of molecules.

This raises the question of whether reference-free inference of molecules is fundamentally unsolvable in the setting of linked-read data, or whether an adequate technique has just not yet been found. Surely there exist corner cases where the problem is either impossible, e.g. in an hypothetical situation where all molecules are assigned to the same barcode, or trivial, when each molecule is assigned to a different barcode. As we will see in Section 2, while there exist previous works in graph theory (e.g. in a setting corresponding to all barcodes containing 2 molecules each), the general setting does not appear to have previously been studied.

In this paper, we establish theoretical grounds for studying the feasibility of inferring molecule without a reference genome. We will not directly tackle the problem of assigning molecule identifiers to reads (as Minerva does), but instead we look at two problems which can be reduced, in the complexity sense, to molecule inference:

1. **Molecule counting**: count the number of molecules assigned to each barcode
2. **Molecule ordering**: reconstruct a total (or partial) order of molecules as a sequence of barcodes

Both problems, if solved accurately, can provide useful information for barcode deconvolution (molecule counting) and genome scaffolding (molecule ordering). Staying at the level of barcodes and molecules instead of reads will allow to thoroughly establish expectations on whether molecule inference is at all feasible, and how various parameters (e.g. number of molecules, how many molecules per barcode, etc) influence its difficulty.

We first present the commononalities between the barcodes ordering problem, and the previously-known concepts of interval graphs and multiple-interval graphs. We then introduce the notion of barcode graph, which models overlaps between molecules across different barcodes. We discuss its link with well-known graph classes leading to the conclusion that solving the molecules ordering problem for a barcode graph is likely difficult. Next we introduce another graph structure, the local clique-pairs graph, inspired of approaches used to realise an interval graph. By identifying maximal cliques in the barcode graph, which are then paired into structures that we call local clique-pairs, we show that the local clique-pairs graph captures a strong signal related to the ordering of the barcodes according to their underlying molecules. We apply this technique to synthetic interval graphs, as well as barcode graphs constructed from simulated molecules from a real genome, and show that on synthetic interval graphs we are able to accurately count the number of molecules per barcode, and reconstruct an approximate but accurate molecule ordering on barcodes. Finally, we demonstrate how to construct a barcode graph directly from linked-read sequencing data.

2 Models and Methods

We consider the problem of sequencing a single long DNA molecule (e.g. a chromosome) using linked reads. We assume that the sequencing data were obtained by sequencing n fragments (called *molecules* from now) from the chromosome, each molecule being assigned a barcode, where several molecules can be assigned the same barcode; for a molecule m we denote by $b(m)$ its barcode. We denote by $\mathcal{B}$ the barcode alphabet and by $|\mathcal{B}| = \mu$ its size, i.e. the total number of observed barcodes; for a barcode b we denote by $m(b)$ the molecules it labels (the barcode size). Let $F = \max_{b \in \mathcal{B}} |m(b)|$. Finally, we assume that no two molecules do start at the same coordinate, which implies that molecules can be totally ordered by their start coordinates.

2.1 Barcode graphs and families of interval graphs.

The sequenced molecules can be seen as intervals along the real line if the sequenced chromosome is linear, or arcs around a circle if it is circular; their *intersection graph* is the graph whose vertices are the n molecules and two vertices are linked by an edge if the corresponding intervals do intersect. Intersection graphs of intervals on the real line (resp. arcs around a circle) form the class of *interval graphs* (resp. *circular-arc graphs*). It is well-known that deciding if a graph is an interval graph or an arc-circular graph can be done in linear time [6, 23], and many algorithmic problems that are computationally hard in general graphs are tractable in these graph classes [15].

However, the result of the sequencing experiment with linked reads does not provide direct knowledge of the sequenced molecules and of their intersections, as the reads originating from molecules having the same barcode b are all labeled by b and, as discussed in introduction, the problem of separating reads with the same barcodes into clusters corresponding to molecules is non-trivial. Nevertheless, we assume here first that it is possible to infer, from the barcoded reads if, for a given pair of barcodes b_1, b_2 there exists molecules m_1 and m_2 such that $b(m_1) = b_1$, $b(m_2) = b_2$ and and m_1 and m_2 do intersect: we then say that barcodes b_1 and b_2 do *intersect.* We assume here moreover that we do not observe two intersecting molecules m_1 and m_2 such that $b(m_1) = b(m_2)$[1].

▶ **Definition 1.** *The exact barcode graph of a set of barcoded molecules is the graph with vertex set $\mathcal{B}$ and edges between pairs of intersecting barcodes.*

In the case of a linear chromosome, exact barcode graphs generalize the class of interval graphs and form another well-studied graphs class, *multiple-interval graphs* [12]. Moreover if we assume that each barcode labels exactly f molecules, exact barcode graphs form the class of f-interval graphs; finally, under the additional assumption that all sequenced molecules have exactly the same length, exact barcode graphs are equivalent to the class of *unit f-interval graph.* We are not aware of any study of the equivalent graph classes for circular chromosomes, i.e. arcs around a circle, and from now on we concentrate on the case of linear chromosomes. We describe below the formulation of several algorithmic problems related to barcode graphs and how they translate into problems on the aforementioned graph classes. Note that an exact barcode graph can be a multi-graph (a graph where multiple edges may have the same endpoints) in the case where there exist molecules m_1, m_2, m_3, m_4 with $b(m_1) = b(m_3)$, $b(m_2) = b(m_4)$ and m_1, m_2 (resp. m_3, m_4) do intersect.

Recognizing exact barcode graphs. The link with unit f-interval graph, although it assumes an unrealistic uniformity in the sequencing process (uniform molecules length and uniform number of molecules per barcode) sheds a light on the computational hardness of analyzing barcoded sequencing data. Indeed, recognizing 2-interval graphs is NP-complete [30], while the complexity of recognizing unit f-interval graphs is still open, the only positive recognition result being for depth-2 unit f-interval graphs [18], corresponding to the case where no chromosome base is covered by more than two molecules, an unrealistic assumption for sequencing experiments. To the best of our knowledge, given a graph on a barcode alphabet whose edges represent possible molecules intersections, deciding if it is an exact barcode graph, even in the setting of molecules of uniform length and barcodes of uniform size, is open.

[1] We justify this assumption as such molecules could be seen as a single molecule defined by the union of m_1 and m_2; moreover, simulations with realistic sequencing parameters show that this situation occurs rarely and most often with molecules that share a small intersection.

Realizing an exact barcode graph. A barcode sequence is a sequence $b_1 \dots b_n$ over the barcode alphabet. Given a barcode graph BG, a barcode sequence *realizes* BG if every edge of BG can be assigned to two barcodes of the sequence in such a way that if b_j is covered by an edge between b_i and b_k (i.e. $i < j < k$) then there are also edges between b_i and b_j and between b_j and b_k. The molecules ordering problem applied to an exact barcode graph BG is then equivalent to finding a barcode sequence realizing BG. This problem is tractable in the case of interval graphs ($F = 1$); note that if intervals lengths are also fixed, then the problem becomes NP-complete [24], while it solvable in polynomial time if additionally the intersection lengths are provided [19]. We are not aware of similar tractability results for multiple-interval graphs. However, existing algorithms to realize interval graphs are mainly based on the property that such a realization can be obtained by a sequence of overlapping maximal cliques. While maximal cliques are easy to find in an interval graph, it is not the case in multiple-interval graph, as the problem of finding the maximum clique in multiple-interval graphs is NP-complete, even for unit 2-interval graphs [13], although approximation and parameterized algorithms do exist [7, 12]. Moreover a structural property of interval graphs that is important toward the realization through maximal cliques, the existence of a vertex whose neighbourhood is a clique, does not hold for multiple interval graphs [2]. Finally, it is easy to see that a maximal clique of size c in an exact barcode graph might not correspond to a set of c pairwise intersecting molecules. This leads us to conjecture that realizing an exact barcode graph is difficult.

Handling inexact barcode graphs. Constructing an exact barcode graph implies to detect intersecting barcodes from sequenced barcoded reads and it is thus likely unrealistic to expect perfectly obtaining such a graph from sequencing data. It follows that solving the molecules ordering problem would then implicitly assumes to solve a graph modification problem, aimed at transforming a graph into a multiple-interval graph, with additional constraints about the number of occurrences of barcodes in a realization. Graph modification problems that aim to minimize the number of modifications are generally hard, even in the case of interval graph, [10], and so for multiple-interval graphs; note however that it was recently shown to be fixed-parameter tractable [27, 5, 10]. Such problems naturally translates into vertex ordering optimization problems (also known as graph layout problems) that can, in principle, be addressed with combinatorial optimization techniques such as Integer Linear Programming (ILP). However, ILP approaches to vertex ordering currently do not scale to the size of instances corresponding to sequencing experiments [9].

From the link we described above between barcode graphs and multiple-interval graphs, and the current state-of-the art in multiple-interval graphs algorithms, it does not appear that the problem of realizing a barcode graph can be addressed by existing algorithms, and we actually conjecture that this problem is difficult, whether the provided barcode graph is exact or not. Nevertheless, toward application to real sequencing data, additional assumptions about the sought realization, such as the expected length of intervals or the expected size of the barcodes, lead to specific open problems of interest in the field of multiple-interval graphs algorithms that deserves further research.

2.2 The Local Clique-Pairs Graph

In this section, we assume that we are given a barcode graph BG. The barcode graph needs not be perfect: it might contain additional (wrong) edges that do not correspond to true overlaps between molecules of two barcodes, or even have missing edges. We will describe the construction of another graph based on the BG: the local clique-pairs (lcp) graph. We will then use the lcp graph to identify a sequence of barcodes that reflects the true order of molecules.

The idea behind the lcp graph will be that, similarly to interval graphs, a realization of an exact barcode graph can be described as a succession of overlapping maximal cliques. Intuitively, these maximal cliques correspond to a set of barcodes that each contain at least one molecule coming from a common genomic region. The task is made more difficult by the fact that not all maximal cliques in a barcode graph satisfy this property. We observed that one can identify and skip such 'wrong' maximal cliques by instead considering a slightly more advanced object: pairs of co-localized maximal cliques, that we name local clique-pairs.

▶ **Definition 2.** *Let c be a vertex of a barcode graph BG. A neighbour of c is a vertex adjacent to c. The neighbourhood of c is the subgraph induced by the set of neighbours of c. A local clique-pair (lcp) is a triplet $(c; C_1, C_2)$ where C_1 and C_2 are maximal cliques in the neighbourhood of c. If there are k edges between vertices of C_1 and vertices of C_2 and d is the maximum number of vertices in either C_1 or C_2, the weight of the lcp (c, C_1, C_2) is defined by $w(c; C_1, C_2) = |d(d-1)/2 - k|$.* [2]

The definition of the weight of an lcp follows from the following observation: when molecules are all of the same length and are evenly spaced along the chromosome, if both cliques C_1 and C_2 are of size d and do indeed correspond to the d barcodes of the molecules preceding (resp. following) the molecule of barcode c, then one expects to observe $d(d-1)/2$ edges between them in the barcode graph. So the weight measures the divergence between the observed number of edges between C_1 and C_2 and the expected number of edges in the case of uniform sequencing (see Fig. 1).

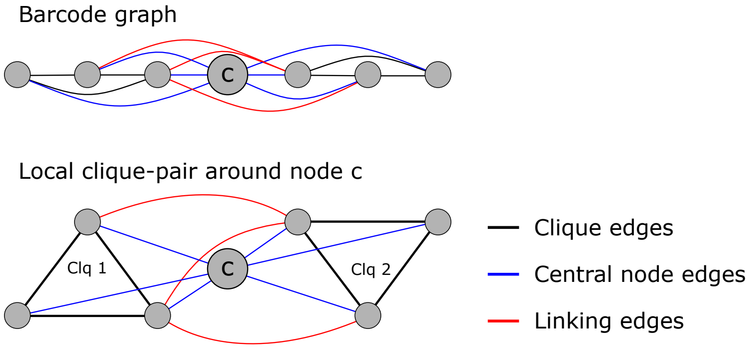

Figure 1 (Top) linear representation of a barcode graph obtained from 7 molecules of uniform length. (Bottom) The local clique-pair associated to c. In black, the edges from the side cliques of the unit 3-graph. In blue, the edges between the central nodes and the other nodes. In red, the edges between the cliques.

To motivate the introduction of lcps, we ran an experiment described in Appendix 5.1, showing that the rate of lcps that actually encode the barcodes of consecutive molecules is higher than the rate of maximal cliques having the same property (Table A1).

We now present our algorithm to compute lcps. Given a barcode c, there can be many maximal cliques among nodes in its neighbourhood, especially cliques that involve the two barcodes that respectively precede and follow c in the true barcode sequence. Given the

[2] The weight is presented for an ideal case where no node is shared between the cliques. If C_1 and C_2 share nodes, there are two modifications. For each node shared, 1 is added to the weight because the shared node is due to a barcode collision. Each shared edge between C_1 and C_2 counts for 2 additional points in the score instead of 1, because 2 edges can be merged inside.

set of all maximal cliques C in the neighbourhood of c, we thus need to extract a matching defining pairs of cliques C_1 and C_2 forming lcps. To do so, we consider the complete graph of size $|C|$ whose vertices are maximal cliques and edges are putative lcps. Edges are weighted by the previously-defined lcp weights. Let W be the maximum observed edge weight. We replace the weight w of each lcp by $W - w$ and apply a maximum-weight matching to clique pairs in order to obtain the set of lcps associated to c (Algorithm 1, illustrated in Fig. 2).

Algorithm 1 Determination of a set of lcps centered at a barcode c.

```
procedure COMPUTE_LCP($c, BG$)                    ▷ c: barcode, BG: barcode graph
    $LCP \leftarrow \emptyset$
    $ngbs \leftarrow BG.neighbours(c)$                       ▷ Neighbours of $c$
    $subgraph \leftarrow BG.induced\_subgraph(ngbs)$          ▷ Neighbourhood of $c$
    $cliques \leftarrow subgraph.max\_cliques()$           ▷ Enumerate maximal cliques
    $CG \leftarrow clique\_graph(cliques)$
    $m = CG.maximum\_weight\_matching()$
    for $(C_1, C_2) \in m$ do
        $LCP \leftarrow LCP \cup new\_lcp(c, C_1, C_2)$          ▷ Add the new lcp
    return $LCP$
```

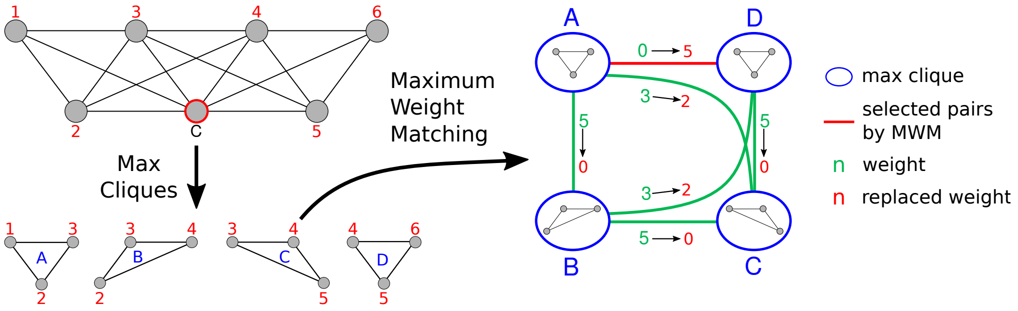

Figure 2 Top left: barcode graph; bottom left: max-cliques of the barcode graph; right: max-clique graph construction and maximum weight matching to construct lcps. The resulting maximum-weight matching is the edge A-D, yielding a single lcp with clique-pair (A, D).

The time complexity of enumerating all maximal cliques of a graph is exponential [26], while computing a maximum-weight matching is polynomial-time solvable [14]. We implemented Algorithm 1 in Python using the output-sensitive cliques enumeration and maximum-weight matching methods implemented in the Networkx library [17]. Its complexity is $O(\max(C^3, M(n)C))$ with n the number of graph nodes, C the number of maximal cliques in the graph, and $M(n)$ the cost of multiplying two $n \times n$ matrices.

Local search for linked cliques are akin to local graph community detection. Soft clustering is being performed with maximal clique detection, i.e. a node may belong to multiple communities. This property leads to a lcp detection algorithm that, intuitively, is resilient to the situation where a barcode corresponds to two or more molecules. Yet it is not perfect: some of the generated lcps may not reflect a collection of overlapping molecules (due to additional artifactual maximal cliques); and also, missing edges in the barcode graph may lead to missing lcps. In the ideal case, lcps can be totally ordered according to their overlaps. But because of artefactual and missing lcps, a total order is not always self-evident.

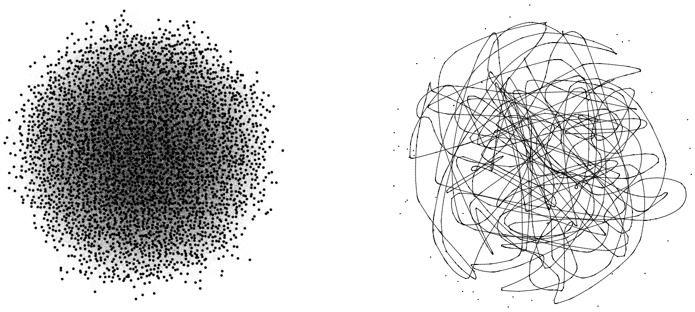

Figure 3 Left: barcode graph (3k nodes and 98k edges) of a simulated interval graph. Right: Resulting lcp graph (13k nodes and 23k edges). Graphs are drawn using Gephi, ForceAtlas 2 layout [3].

▶ **Definition 3.** *Let BG be a barcode graph and V a set of lcps obtained from BG. The lcp graph $lcp(BG)$ is the weighted graph with V for vertex set and where there is an edge between two lcps $(b; B_1, B_2)$ and $(c; C_1, C_2)$ such that some barcode belongs to both one of the C_i cliques and one of the B_i cliques. The weight of an edge is the size of the symmetric differences of the barcodes content of $(b; B_1, B_2)$ and $(c; C_1, C_2)$.*

The lcp graph is a framework for determining which lcps are consecutive, also enabling to identify lcps that are not overlapping with others. Figure 3 shows a simulated barcode graph where the corresponding lcp graph has a linear structure, similar to the original interval graph among molecules. This makes the task of finding a suitable path within the lcp graph, which reflects the ordering of molecules, easier than in the barcode graph. In the following, we will describe how we determine a barcode ordering based on finding a path in the lcp graph.

2.3 Finding a suitable path in the lcp graph

Recall that the molecule counting problem amounts to finding how many molecules were merged in each barcode. The molecule ordering problem asks for a sequence of barcodes that reflects the order of molecules. As these two problems are centered on barcodes and not lcps, we need a way to convert a lcp path into an ordered list of barcodes. We do this as follows: i) each lcp in the path is replaced by its central barcode, ii) an edge reduction step is applied to the lcp graph, and finally iii) a path is found using a branch-and-bound algorithm. Formally, the algorithmic problem we address heuristically in this section is to find a path in the lcp graph that maximizes the sum of the weight of the selected lcps and of the selected edges between lcps, under the constraint that the union of the selected lcps covering sets contains all edges of the initial barcode graphs.

lcp graph simplification

We simplify the lcp graph by performing transitive reduction over triplets. Given an edge (a, b) of weight w_{ab}, we remove this edge from the graph if there exist 2 edges (a, c) of weight w_{ac} and (b, c) of weight w_{bc} such that $w_{ab} \leqslant w_{ac} + w_{bc}$. This operation does not change the node set (lcps) but reduces the number of possible paths to explore. Intuitively, requiring to go through lcp c when going from lcp a to lcp b forces to select two higher-confidence lcp overlaps instead of one lower-confidence overlap between two lcps.

lcp path construction

Assuming the barcode graph has been obtained by merging nodes of an exact interval graph defined by the molecules intersections, every edge of the barcode graph corresponds to one (or potentially several) edges of the interval graph; we show in Section 3.3 that barcode graphs created from reads have nearly all correct edges corresponding to such molecules intersections. This observation motivates to require that a walk in the lcp graph that reflects the true order of molecules should be composed of lcps that contain most of the edges of the original barcode graph. Each lcp is an induced subgraph of the barcode graph, and we associate to it a *covering set* defined as the set of edges of the barcode graph it contains. We will seek a path such that the union of covering sets over all its constituent lcps is as close as possible to the set of all edges of the barcode graph.

Our lcp path construction strategy is a local branch & bound algorithm. Assuming we have already constructed a path of lcps $p = l_1, \ldots, l_i$, we consider as candidates for l_{i+1} all the neighbours of l_i in the lcp graph such that $l_{i+1} \notin p$. Those neighbours are sorted by priority over three criteria: first if one or more lcp(s) cover at least one uncovered edge of the original barcode graph, we prioritize those lcps. For the second sorting criterion, we sort the candidate l_{i+1}'s by increasing lcp weight (Def. 3). Last, if multiple candidates have equal clique pair weights, we sort them by increasing $l_i \to l_{i+1}$ edge weight in the lcp graph. Selecting the first element in the sorted neighbours at each step defines a greedy heuristic for the path computation.

The above algorithm might result in a short path due to tips in the lcp graph, i.e. nodes of degree 1. In order to address this issue, we use a local branch and bound algorithm and backtrack a few nodes when a dead end is reached. This can result in several paths and we use the size of the union of covering sets in the path as a score to keep only the best solutions according to that score.

The last part of the algorithm is the selection of the first node l_1 of the path. We initially select a l_1 at random among all lcps, and compute a path using the above procedure which ends at some node l_e. We then discard this path and restart again our algorithm from $l'_1 = l_e$ to create a new path, where l_e has a higher chance to be an endpoint of the true lcp path than l_1.

We will show in the next section that despite this heuristic being very simple and likely leaving room for improvement, it does work very well on simulated data, suggesting the lcp graph does actually capture a robust signal toward recovering the correct barcode sequence.

3 Results

3.1 Overview

Simulated data

We will examine three types of barcode graphs ordered by increasing level of realism. They will be generated from either:

1. entirely synthetic sets of intervals (i.e. interval graphs) with randomly identified vertices,
2. intersections of molecules sampled from a genome,
3. directly from simulated linked-read sequencing data.

Analysis pipeline

Our complete analysis pipeline performs the following steps:

1. Generate all the lcps from the barcode graph (Algorithm 1)
2. Generate the lcp graph
3. Simplify the graph by transitive reduction of the triplets (Section 2.3)
4. Generate the lcp path using the hybrid greedy/branch and bound algorithm (Section 2.3)
5. Replace all the lcp by their central barcodes
6. Evaluate the accuracy of the resulting barcode sequence

In the remaining of the Results sections, all the graphs and paths are generated by the above pipeline, implemented using `Snakemake` [20] and available at `https://gitlab.pasteur.fr/ydufresne/linkedreadsmoleculeordering`.

Quality metrics

We design quality metrics that are applicable to both barcode graphs and lcp graphs. To do so, in lcp graphs we identify each lcp to its central barcode. We consider three metrics over the graphs: accuracy, sensitivity and longest correct path. The first two metrics are estimated by randomly sampling paths having $l \in \{2, 4, 10, 100\}$ edges from the graph. To measure accuracy, a path having barcodes $(b_0, b_1, \dots, b_l)$ is considered to be correct if there exists $m_0, m_1, \dots, m_l$ overlapping (but not necessarily consecutive) molecules such that $m_i \in m(b_i)$, $0 \leq i \leq l$. Accuracy is then defined as the number of correct paths over the total number of sampled paths. To measure sensitivity, we determined for all $(l+1)$-tuples $m_i, m_{i+1}, \dots, m_{i+l}$ of consecutive molecules in the genome, whether there exist a path $b(m_i), b(m_{i+1}), \dots, b(m_{i+l})$ in the graph. Sensitivity is then the ratio of such paths that are found in the graph. Finally, the Longest Correct path (LC) metric is defined as the longest path that can be found in the lcp graph that is correct, i.e. corresponding to a barcode sequence equal to the barcodes of a sequence of overlapping molecules. This measure is not informative on barcode graphs; it measures the conservation of molecule overlap information in a lcp graph.

Two additional quality metrics are defined on lcp paths found by our branch-and-bound algorithm: Undercounted/Overcounted (U/O) molecules and Longest Common Subsequence (LCS). The U/O metric is computed by recording two counters, U and O initialized at 0. Given each barcode b that appears within a lcp path, we compare the number of occurrences of b to M_b, the true number of molecules having barcode b. If b occurs in the lcp path strictly more (resp. less) than M_b times, U (resp. O) is incremented by the absolute difference. U and O should both be as close to zero as possible, and they indicate how well we solve the molecule counting problem. For the LCS metric, we compute the longest common subsequence between central nodes of the lcp path and the molecule path where each molecule is replaced by its barcode. The LCS reflects how well we solve the molecule ordering problem.

3.2 Simulated data from interval graphs

Dataset generation

At first we focus on purely synthetic interval graphs, where a genome is conceptually a real line and molecules are intervals on this line. We make the simplifying assumption that molecules all have the same size, and are evenly distributed along the genome. To simulate barcode graphs, we start from an intersection graph of molecules and perform so-called *merges* of molecules. A merge is defined as follows: given two nodes a and b that will be merged, create a new node c; for all neighbours v of either a or b, create edges (c, v), and finally delete a and b. Merging two nodes in the graph is equivalent to replacing two molecules by one barcode corresponding to those two molecules. A succession of merge operation creates an exact barcode graph as defined in Section 2.

We created 8 synthetic test datasets, using the following grid of parameters: $5,000$ or $10,000$ molecules, average number of merges (i.e. molecules per barcode) of 2 or 3, standard deviation in the number of merges of 0 or 1.

Quality of lcp graphs

Table 1 shows the accuracy and sensitivity of barcode graphs and their corresponding lcp graphs. Recall that accuracy measures whether a random path in the graph has a correct order of barcodes. As expected, paths in the barcode graph are mostly inaccurate, as one may jump from one genome location to another due to barcode merges. Conversely paths in the lcp graph are very accurate (100% for nearly all $l = 10$ paths), with a slight decrease at $l = 100$ ($95\% - 100\%$). The sensitivity metric measures how much of the true barcode ordering is present in short paths of the graph. It is (unsurprisingly) high for barcode graphs, as they indeed record all overlaps between molecules. Note that some merges collapse consecutive molecules by chance, hence the sensitivity of barcode graphs can sometimes be lower than 1. On lcp graphs, sensitivity is high for short paths ($> 93\%$ for $l = 10$) and drops for long ones ($54\% - 98\%$ for $l = 100$). Nevertheless, this shows that at least partial molecule order can be inferred through looking at central nodes of lcps in the lcp graph, and that the lcp graph shows a better balance between accuracy and sensitivity than the barcode graph. Note that central nodes are not the only way to infer molecule order, as one could also extract information from clique-pairs, yet we leave this direction for future work.

Overall, lcp graphs are clearly more informative than barcode graphs for reconstructing accurate barcode orderings. The hardest instances, in terms of accuracy and sensitivity on lcp graphs, are when the number of molecules is low and the number of merges is high.

Quality of lcp paths

Table 2 reports additional metrics on lcp graphs and lcp paths constructed using the branch-and-bound algorithm, over the same 8 datasets. All lcp graphs have a high longest correct path (LC), confirming the theoretical possibility of reconstructing over 99% of the true barcode order, through central nodes of a suitable path of lcps. The last two metrics of Table 2 are computed on lcp paths found by the algorithm described in Section 2.3. On 10,000 molecules graphs, the longest common subsequence (LCS) of the computed lcp path is 90% of the true barcode order, indicating that we nearly recovered the correct barcode order. The 5,000 molecules graphs appear to be more challenging to process as, smaller graphs are more sensitive to information loss by the merging process, yet LCS values remain above 79%. The U/O metric reports the ability to count the number of molecules that are present in each barcode, though counting the number of times each barcode occurs in the computed lcp path. Overall, lcp paths tend to undercount molecules (higher U metric than O), yet both U and O metrics are around or below 10% of the number of molecules, indicating that lcp path provides a reliable estimation of the number of molecules per barcode.

Table 1 Accuracy and sensitivity of randomly sampled paths of lengths 2, 4, 10 and 100 edges in lcp graphs generated from merged interval graphs, compared to sampled paths of the same lengths in barcode graphs (G_b) as a base-line.

Graph			**l=2**		**l=4**		**l=10**		**l=100**	
# mols	Merges	Type	Acc	Sens	Acc	Sens	Acc	Sens	Acc	Sens
5,000	2 ± 0	G_b	0.48	1.00	0.09	1.00	0.00	0.99	0.00	0.94
		lcp	1.00	1.00	1.00	0.99	1.00	0.98	1.00	0.88
5,000	2 ± 1	G_b	0.46	1.00	0.09	1.00	0.00	1.00	0.00	0.98
		lcp	1.00	1.00	1.00	0.99	1.00	0.98	0.99	0.84
5,000	3 ± 0	G_b	0.31	1.00	0.03	1.00	0.00	0.99	0.00	0.88
		lcp	1.00	0.99	1.00	0.98	0.99	0.95	0.99	0.60
5,000	3 ± 1	G_b	0.33	1.00	0.03	1.00	0.00	1.00	0.00	0.96
		lcp	1.00	0.99	1.00	0.97	0.99	0.93	0.95	0.54
10,000	2 ± 0	G_b	0.48	1.00	0.10	1.00	0.00	1.00	0.00	1.00
		lcp	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98
10,000	2 ± 1	G_b	0.47	1.00	0.09	1.00	0.00	1.00	0.00	0.97
		lcp	1.00	1.00	1.00	1.00	1.00	0.99	0.99	0.93
10,000	3 ± 0	G_b	0.31	1.00	0.02	1.00	0.00	1.00	0.00	1.00
		lcp	1.00	1.00	1.00	0.99	1.00	0.99	1.00	0.87
10,000	3 ± 1	G_b	0.31	1.00	0.03	1.00	0.00	1.00	0.00	0.97
		lcp	1.00	0.99	1.00	0.99	1.00	0.97	0.99	0.78

Table 2 Experiments on synthetic barcode graphs. The dataset is described on the first part of the columns (Number of molecules in the molecule graph, number of merges, resulting number of barcodes in the barcode graph). The LC column is the length of the longest correct path in the lcp graph. The U/C column is the number of undercounted and overcounted molecules per barcode in our computed lcp path, and the LCS column is the length of the longest common subsequence between the lcp path and the correct barcode order.

# mols	Merges	# barcodes	LC	U/O Counts	LCS
5,000	2 ± 0	2500	4990	227/56	4748
5,000	2 ± 1	2428	4991	405/109	4512
5,000	3 ± 0	1667	4985	549/240	4282
5,000	3 ± 1	1682	4975	498/665	3972
10,000	2 ± 0	5000	9992	268/68	9667
10,000	2 ± 1	4889	9993	418/129	9531
10,000	3 ± 0	3334	9981	593/184	9309
10,000	3 ± 1	3341	9987	753/201	9140

Table 3 Accuracy and sensitivity of randomly sampled paths of lengths 2, 4, 10 and 100 edges in lcp graphs, compared to sampled paths of the same lengths in barcode graphs (G_b) as a base-line, with 15 kbp *E. coli* molecules, 50X coverage, minimal molecule overlap lengths of 7000.

	l=2		**l=4**		**l=10**		**l=100**	
Graph	Acc	Sens	Acc	Sens	Acc	Sens	Acc	Sens
$\boldsymbol{G_b, m = 1}$	1	1	1	1	1	1	1	1
$\boldsymbol{lcp, m = 1}$	1	1	1	0.99	1	0.99	1	0.84
$\boldsymbol{G_b, m = 2}$	0.50	1	0.12	0.99	0.001	0.99	0	0.99
$\boldsymbol{lcp, m = 2}$	0.99	0.99	0.99	0.99	0.99	0.99	0.94	0.84
$\boldsymbol{G_b, m = 3}$	0.34	1	0.04	1	0	1	0	1
$\boldsymbol{lcp, m = 3}$	0.99	0.99	0.99	0.99	0.98	0.99	0.88	0.88
$\boldsymbol{G_b, m = 4}$	0.26	1	0.02	0.99	0	0.99	0	0.99
$\boldsymbol{lcp, m = 4}$	0.99	0.99	0.99	0.99	0.98	0.99	0.83	0.87

3.3 Genome graphs

Quality of genome LCP graphs

We designed experiments to evaluate the quality of lcp graphs constructed from the barcode graphs that originate from real molecules. We created a synthetic *E.coli* molecule graph by simulating molecules of length 15 kbp using `wgsim`, corresponding to sequences of the *E. coli* genome, at 50x coverage of the genome and with no sequencing errors. Overlaps between all pairs of molecules were computed using `minimap2` using default parameters, and we selected overlaps of lengths greater than 7000 using `fpa` [22].

Table 3 shows the accuracy and sensitivity on our constructed lcp graphs versus the average number of merges, i.e. average number of molecules per barcode. As in Section 3.2, barcode graphs have poor accuracy, which is expected due to the glueing of molecules, and near-perfect sensitivity as all molecule overlaps are found. In contrast, lcp-graphs manage to keep both near-perfect accuracy and sensitivity (> 0.98) for short paths (< 10) and have a decrease in accuracy ($0.83 - 0.94$) and sensitivity ($0.84 - 0.87$) for paths of length 100.

Construction of genome barcode graphs from reads

In this section we describe a method that constructs an accurate barcode graph directly from linked-read data. This closes a gap between our theoretical results, that required to already have a barcode graph, and experimental data which only consist of sequencing reads.

We simulated reads from the *E. coli* genome at 50X coverage using LRSIM[21]. We assembled these reads using SPAdes[1] version 3.12.0 without using linked-read information (only using paired-end information), in order to generate contigs to which linked-reads can be mapped to using the EMA aligner[25]. We designed an algorithm[3] to infer molecule overlaps given the set of contigs and the EMA alignments. In brief, the algorithm proceeds as follows.

For each barcode, and within each contig, we collect and sort the mapping positions of all reads associated that barcode. We define a molecule interval to be the first and last mapping positions of a group of mapping positions that are all within a distance $< M_d$ than each other. A barcode can be associated to multiple molecule intervals even within the same contig. We construct the barcode graph by looking at overlapping molecule intervals from different barcodes. If two intervals share an overlap larger than a parameter M_o, we add an edge between the two associated barcodes.

[3] Available at `https://github.com/natir/mapping2barcode`

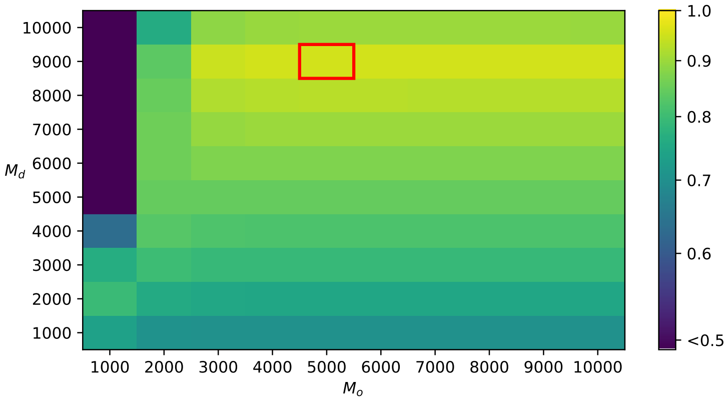

Figure 4 Quality of barcode graph construction from a set of reads and corresponding paired-end assembly. Each square represents the F1-score given parameters M_d (read mapping distance) and M_o (minimal molecule overlap length), from purple (low F1-score) to yellow (high F1-score).

The algorithm has two key parameters: M_d, the maximal distance between two reads in an inferred molecule interval, and M_o, the minimal overlap length between molecules. As we used simulated data, we were able to generate a ground-truth barcode graph given that molecule intervals in the underlying genome are known for each barcode. Figure 4 shows the performance of the algorithm in terms of F1-score (combining both sensitivity and precision, computed by comparing the edge set of the inferred barcode graph versus the edge set of the ground truth). We observe that the best F1-score (0.953) is reached for $(M_o, M_d) = (5000, 9000)$, with otherwise consistently high F1-scores (≥ 0.9) whenever $M_o > 2000$ and $M_d > 7000$.

4 Conclusion

In this paper, we introduced novel approaches to analyze linked-reads sequencing data. We introduced the problem of recovering a barcode sequence from the barcode graph, and described its link with natural algorithmic problems on multiple-interval graphs; we believe that the potential applications in sequencing data analysis motivate further research on these algorithmic questions. Moreover, motivated by classic algorithmic techniques in interval graph realization, we introduced the concept of local clique pairs (lcp) and lcp graph. Our experiments on simulated data suggests that the lcp graph exhibits a much more linear structure than the barcode graph and is likely a relevant intermediate structure between the barcode graph and the barcode sequence.

This work casts a spotlight on a couple open problems in graph theory, for which linked-reads bring an additional practical application: recognizing perfect barcode graphs, realizing multiple-interval graphs, and the complexity of recognizing unit f-interval graphs. We suspect also that more effective algorithms than the ones proposed here may exist for constructing lcp graphs and finding lcp paths.

While our treatment of synthetic barcode graphs demonstrates the feasibility of recovering barcode orders, we encountered difficulties going further in our analyses of realistic instances (e.g. real *E. coli* reads). First, a more advanced path(s) discovery procedure will be needed to deal with lcp graphs constructed from real molecule intersections (Section 3.3), which have inferior accuracy than those in Section 3.2. Second, refinements to Algorithm 1, potentially in the form of post-processing, will be needed to avoid outputting too many artefactual lcps in barcode graphs of high-coverage molecules, such as the one produced in Section 3.3.

Finally, the proposed barcode graph construction approach has potential to be applied to larger instances, but so far we only tested it on simulated data and is merely a proof of concept. The current method relies on having sufficient assembly contiguity (longer contigs than molecules). A potential direction that we leave for future work is to determine molecule intervals using the structure of an assembly graph instead of contigs.

References

1 Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A Gurevich, Mikhail Dvorkin, Alexander S Kulikov, Valery M Lesin, Sergey I Nikolenko, Son Pham, Andrey D Prjibelski, et al. SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J. Comput. Biol.*, 19(5):455–477, 2012. doi:10.1089/cmb.2012.0021.

2 Reuven Bar-Yehuda, Magnús M. Halldórsson, Joseph Naor, Hadas Shachnai, and Irina Shapira. Scheduling split intervals. *SIAM J. Comput.*, 36(1):1–15, 2006. doi:10.1137/S0097539703437843.

3 Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. In *Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009, San Jose, California, USA, May 17-20, 2009*. The AAAI Press, 2009. URL: http://aaai.org/ocs/index.php/ICWSM/09/paper/view/154.

4 Alex Bishara, Eli L Moss, Mikhail Kolmogorov, Alma E Parada, Ziming Weng, Arend Sidow, Anne E Dekas, Serafim Batzoglou, and Ami S Bhatt. High-quality genome sequences of uncultured microbes by assembly of read clouds. *Nat. Biotechnol.*, 36:1067–1075, 2018. doi:10.1038/nbt.4266.

5 Ivan Bliznets, Fedor V. Fomin, Marcin Pilipczuk, and Michal Pilipczuk. Subexponential parameterized algorithm for interval completion. *ACM Trans. Algorithms*, 14(3):35:1–35:62, 2018. doi:10.1145/3186896.

6 Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. Syst. Sci.*, 13(3):335–379, 1976. doi:10.1016/S0022-0000(76)80045-1.

7 Ayelet Butman, Danny Hermelin, Moshe Lewenstein, and Dror Rawitz. Optimization problems in multiple-interval graphs. *ACM Trans. Algorithms*, 6:268–277, 2007. doi:10.1145/1721837.1721856.

8 Zhoutao Chen, Long Pham, Tsai-Chin Wu, Guoya Mo, Yu Xia, Peter Chang, Devin Porter, Tan Phan, Huu Che, Hao Tran, Vikas Bansal, Justin Shaffer, Pedro Belda-Ferre, Greg Humphrey, Rob Knight, Pavel Pevzner, Son Pham, Yong Wang, and Ming Lei. Ultra-low input single tube linked-read library method enables short-read NGS systems to generate highly accurate and economical long-range sequencing information for de novo genome assembly and haplotype phasing. *bioRxiv*, page 852947, 2019. doi:10.1101/852947.

9 David Coudert. A note on integer linear programming formulations for linear ordering problems on graphs. Research Report hal-01271838, INRIA, I3S, Université Nice Sophia, 2016. URL: https://hal.inria.fr/hal-01271838.

10 Christophe Crespelle, Paal Gronaas Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification. *ArXiv*, abs/2001.06867, 2020. URL: https://arxiv.org/abs/2001.06867.

11 David C Danko, Dmitry Meleshko, Daniela Bezdan, Christopher Mason, and Iman Hajirasouliha. Minerva: an alignment and reference free approach to deconvolve linked-reads for metagenomics. *Genome Res.*, 29:116–124, 2019. doi:10.1101/gr.235499.118.

12 Michael R Fellows, Danny Hermelin, Frances A Rosamond, and Stéphane Vialette. On the parameterized complexity of multiple-interval graph problems. *Theor. Comput. Sci.*, 410(1):53–61, 2009.

13 Mathew C. Francis, Daniel Gonçalves, and Pascal Ochem. The maximum clique problem in multiple interval graphs. *Algorithmica*, 71(4):812–836, 2015. doi:10.1007/s00453-013-9828-6.

14 Zvi Galil. Efficient algorithms for finding maximum matching in graphs. *ACM Comput. Surv.*, 18(1):23–38, 1986. doi:10.1145/6462.6502.

15 Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57).* North-Holland Publishing Co., 2004.

16 Stephanie U Greer, Lincoln D Nadauld, Billy T Lau, Jiamin Chen, Christina Wood-Bouwens, James M Ford, Calvin J Kuo, and Hanlee P Ji. Linked read sequencing resolves complex genomic rearrangements in gastric cancer metastases. *Genome Med.*, 9(1):57, 2017. doi:10.1186/s13073-017-0447-8.

17 Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using NetworkX. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11–15, Pasadena, CA, 2008.

18 Minghui Jiang. Recognizing d-interval graphs and d-track interval graphs. *Algorithmica*, 66(3):541–563, 2013. doi:10.1007/s00453-012-9651-5.

19 Johannes Köbler, Sebastian Kuhnert, and Osamu Watanabe. Interval graph representation with given interval and intersection lengths. *J. Discrete Algorithms*, 34:108–117, 2015. doi:10.1016/j.jda.2015.05.011.

20 Johannes Köster and Sven Rahmann. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, 28(19):2520–2522, 2012.

21 Ruibang Luo, Fritz J Sedlazeck, Charlotte A Darby, Stephen M Kelly, and Michael C Schatz. LRSim: a linked-reads simulator generating insights for better genome partitioning. *Comput Struct Biotechnol J.*, 15:478–484, 2017. doi:10.1016/j.csbj.2017.10.002.

22 Pierre Marijon, Rayan Chikhi, and Jean-Stéphane Varré. yacrd and fpa: upstream tools for long-read genome assembly. *Bioinformatics*, advance access:btaa262, 2020. doi:10.1093/bioinformatics/btaa262.

23 Ross M. McConnell. Linear-time recognition of circular-arc graphs. *Algorithmica*, 37(2):93–147, 2003. doi:10.1007/s00453-003-1032-7.

24 Itsik Pe'er and Ron Shamir. Realizing interval graphs with size and distance constraints. *SIAM J. Discret. Math.*, 10(4):662–687, 1997. doi:10.1137/S0895480196306373.

25 Ariya Shajii, Ibrahim Numanagić, and Bonnie Berger. Latent variable model for aligning barcoded short-reads improves downstream analyses. In *Research in Computational Molecular Biology - 22nd Annual International Conference, RECOMB 2018*, volume 10812 of *Lecture Notes Comput. Sci.*, pages 280–282. Springer, 2018. doi:10.1007/978-3-319-89929-9.

26 Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theor. Comput. Sci.*, 363(1):28–42, 2006. doi:10.1016/j.tcs.2006.06.015.

27 Yngve Villanger, Pinar Heggernes, Christophe Paul, and Jan Arne Telle. Interval completion is fixed parameter tractable. *SIAM J. Comput.*, 38(5):2007–2020, 2009. doi:10.1137/070710913.

28 Ou Wang, Robert Chin, Xiaofang Cheng, Michelle Ka Yan Wu, Qing Mao, Jingbo Tang, Yuhui Sun, Ellis Anderson, Han K. Lam, Dan Chen, Yujun Zhou, Linying Wang, Fei Fan, Yan Zou, Yinlong Xie, Rebecca Yu Zhang, Snezana Drmanac, Darlene Nguyen, Chongjun Xu, Christian Villarosa, Scott Gablenz, Nina Barua, Staci Nguyen, Wenlan Tian, Jia Sophie Liu, Jingwan Wang, Xiao Liu, Xiaojuan Qi, Ao Chen, He Wang, Yuliang Dong, Wenwei Zhang, Andrei Alexeev, Huanming Yang, Jian Wang, Karsten Kristiansen, Xun Xu, Radoje Drmanac, and Brock A. Peters. Efficient and unique cobarcoding of second-generation sequencing reads from long DNA molecules enabling cost-effective and accurate sequencing, haplotyping, and de novo assembly. *Genome Res.*, 29(5):798–808, 2019. doi:10.1101/gr.245126.118.

29 Neil I Weisenfeld, Vijay Kumar, Preyas Shah, Deanna M Church, and David B Jaffe. Direct determination of diploid genome sequences. *Genome Res.*, 27, 2017. doi:10.1101/gr.214874.116.

30 Douglas B. West and David B. Shmoys. Recognizing graphs with fixed interval number is NP-complete. *Discret. Appl. Math.*, 8(3):295–305, 1984. doi:10.1016/0166-218X(84)90127-6.

31 Sarah Yeo, Lauren Coombe, René L Warren, Justin Chu, and Inanç Birol. ARCS: scaffolding genome drafts with linked reads. *Bioinformatics*, 34(5):725–731, 2017. doi:10.1093/bioinformatics/btx675.

32 Fan Zhang, Lena Christiansen, Jerushah Thomas, Dmitry Pokholok, Ros Jackson, Natalie Morrell, Yannan Zhao, Melissa Wiley, Emily Welch, Erich Jaeger, Ana Granat, Steven J. Norberg, Aaron Halpern, Maria C Rogert, Mostafa Ronaghi, Jay Shendure, Niall Gormley, Kevin L. Gunderson, and Frank J. Steemers. Haplotype phasing of whole human genomes using bead-based barcode partitioning in a single tube. *Nat. Biotechnol.*, 35(9):852–857, September 2017. doi:10.1038/nbt.3897.

5 Appendix

5.1 Experiment on perfect cliques versus perfect lcps

Given an interval graph with m vertices, and an integer parameter f, we repeated for each vertex x the following process f times: pick another unmerged node y at random and merge vertices x and y. This generates a simulated barcode graph where all barcodes correspond to exactly f molecules. Then on this graph we computed all maximal cliques and all lcps and call such a set of vertices *perfect* if it corresponds to a set of consecutive intervals in the original interval graph.

Table A1 Average number of perfect maximal clique vs perfect lcps, averaged over 10 runs for each setting defined by m and f.

m	f	cliques	perfect cliques	lcp	perfect lcp
5000	2	5318.5	54645.2 (9.73%)	4600.2	12763 (36.04%)
5000	3	6361.8	76569.4 (8.31%)	4054.3	29924.8 (13.55%)
10000	2	10344.8	104817.5 (9.87%)	9586.2	18958.4 (50,56%)
10000	3	12070.6	130092.4 (9.28%)	9031.2	44276 (20.40%)