Consistency of Sequence-based Gene Clusters*

Roland Wittler[1,2,†], Ján Maňuch[1,3], Murray Patterson[3], Jens Stoye[2]

Keywords: whole-genome comparison, gene cluster, gene order, ancestral reconstruction

Abstract

In comparative genomics, differences or similarities of gene orders are determined to predict functional relations of genes or phylogenetic relations of genomes. For this purpose, various combinatorial models can be used to specify gene clusters — groups of genes that are co-located in a set of genomes. Several approaches have been proposed to reconstruct putative ancestral gene clusters based on the gene order of contemporary species. One prevalent and natural reconstruction criterion is *consistency*: For a set of reconstructed gene clusters, there should exist a gene order that comprises all given clusters. For permutation-based gene cluster models, efficient methods exist to verify this condition.

In this paper, we discuss the consistency problem for different gene cluster models on sequences with restricted gene multiplicities. Our results range from linear-time algorithms for the simple model of *adjacencies* to NP-completeness proofs for more complex models like *common intervals*.

*To appear in J. Comp. Biol., version of April 8, 2011.

†corresponding author

[1]Department of Mathematics, Simon Fraser University, 8888 University Drive, Burnaby, B.C., V5A 1S6, Canada,

[2]Technische Fakultät, Universität Bielefeld, 33594 Bielefeld, Germany, {rwittler, stoye}@techfak.uni-bielefeld.de, phone: +49 521 106-6882, fax: +49 521 106-6495

[3]Department of Computer Science, University of British Columbia, 201-2366 Main Mall, Vancouver, B.C., V6T 1Z4, Canada, {jmanuch, murrayp}@cs.ubc.ca, phone:+1 778 549 4587, fax: +1 604 822 5485

1 Introduction

The exploration of the ancestral history of different species can give valuable information about their evolution. In whole-genome comparison, one commonly considers the order of the genes or other markers within the genome to study changes and similarities in the structure of different genomes.

One approach for the reconstruction of phylogenetic scenarios or for the comparison of genomes is the examination of the genetic material on the level of the DNA, RNA or protein sequences. Another possibility is to study the genomic structure. On this higher level, one commonly considers the order of the genes or other markers within the genome. Genes belonging to the same gene family are represented by the same identifier. To simplify matters, the term 'gene' will be used to refer to the corresponding gene family identifier. One simple way to model genomes is to use permutations. However, this approach includes the assumption that every gene occurs exactly once in each considered genome. To allow for duplications and deletions, a relaxation to sequences of genes is necessary. A convenient way to account for the orientation of a gene within the genome is to use signed permutations or signed sequences, respectively.

Evolutionary processes can rearrange a gene order. The gene composition of some regions, however, is preserved and can be found in several related genomes. These segments, denoted as *gene clusters*, often contain functionally or evolutionarily associated genes (Lawrence and Roth, 1996; Overbeek et al., 1999). Hence, the analysis of gene clusters can give clues about the function of genes and valuable insights into evolutionary processes like rearrangement processes or lateral gene transfer. Various formal definitions of gene clusters based on different models of gene order have been discussed and analyzed in the literature. See Hoberman and Durand (2005) and Bergeron et al. (2008) for surveys of different concepts of gene clusters. Whenever the genomes of several species comprise the same gene cluster, it was presumably inherited from a common ancestor. Recent studies (Bergeron et al., 2004; Adam et al., 2007; Chauve and Tannier, 2008; Stoye and Wittler, 2009) build on this idea to reconstruct ancient gene clusters and to infer ancient gene orders. More precisely, the internal nodes of a given phylogenetic tree are labeled with sets of gene clusters, based on the gene orders of contemporary species at the leaves of the tree. Besides the pure identification of gene clusters, such reconstructed scenarios for the origin of the clusters and the development of the gene order can give valuable information about underlying evolutionary processes, the ancestral history of the species, and functional and evolutionary relations of genes.

Proposed reconstruction approaches differ in the underlying models for gene order and gene clusters, and in the applied methodology. However, a general aim is to ensure *consistency*: For a set of putative ancient gene clusters, there should exist at least one gene order that comprises all given clusters. Otherwise, the reconstruction result would be inconsistent with respect to the genome model.

The goal of reconstructing consistent labelings was first introduced by Bergeron et al. (2004) who presented an algorithm that reconstructs sets of framed common

intervals on permutations. Adam et al. (2007) applied the parsimony principle as an objective function to reconstruct common intervals on permutations. A heuristic is used to reach consistency. Recently, Chauve and Tannier (2008) proposed a methodology to reconstruct the gene order of the amniote genome, based on consistent labelings of common intervals and adjacencies. In our previous work (Stoye and Wittler, 2009), we introduced an algorithmical framework that is not restricted to a specific model but instead follows an oracle-based approach to compute most parsimonious consistent labelings for various models.

All of the above methods have been successfully applied to real data and proven to yield reasonable and valuable results. They all rely on permutation-based models, which enable efficient algorithms and data structures. In particular, the verification of consistency can be translated to the well-studied *Consecutive Ones Problem* and be solved in polynomial time and space using data structures like PQ-trees or PC-trees (Booth and Lueker, 1976; Habib et al., 2000; Hsu, 2002; Hsu and McConnell, 2004). Some reconstruction approaches could be easily adapted to the model of *sequences without duplications* which allows genes to be missing in some genomes but still requires each gene to occur at most once in each genome.

In this paper, we discuss consistency for *sequence-based* gene cluster models. Particularly, we consider the simple model of *adjacencies*, the classical model of *common intervals* (Uno and Yagiura, 2000), and two variants of the latter. For each of these models we address the problem: Given a set of gene clusters and a maximum copy number for each gene, decide whether there exists a valid gene order that contains all of the clusters. Our results range from algorithms that verify consistency for adjacencies in linear time to the confirmation of NP-completeness for the more complex models.

This paper extends a conference version presented at the RECOMB satellite workshop on Comparative Genomics (Wittler and Stoye, 2010). Here, we present much stronger NP-hardness results, emphasizing the strict complexity border between adjacencies and only slightly more relaxed models. In fact, for two of the models discussed, we rule out fixed-parameter-tractability in any natural parameter, and for a third model, only a small gap remains.

The paper has been organized in the following way. First, we formally introduce the *Consistency Problem* in Section 2. Then, in Section 3, we give an efficient solution for the gene cluster model of both signed and unsigned adjacencies. In Section 4, we present NP-completeness results for the model of common intervals and its variants before we finish with some discussions and conclusions in Section 5.

2 The Consistency Problem

Assume a set of putative gene clusters, assigned to an ancestral node in a given tree. These ancient clusters in turn imply a set of putative ancient genomes: all those which contain all of the given clusters. Depending on the gene cluster model used, this set of genomes can be empty if some of the clusters derived from different contemporary species are in contradiction with others. For example, when we model gene order as permutations, there is no valid gene order comprising the three adjacencies $\{a, b\}$, $\{a, c\}$ and $\{a, d\}$, because, according to the model, gene a can only occur once and thus only be the neighbor of two other genes.

In a more general case, we represent genomes as *sequences* of genes or other genomic markers. In a sequence, any element can occur multiple times or not at all, which, in the context of gene order comparison, corresponds to paralogous genes and gene deletions, respectively.

If we allow each gene to appear arbitrarily often in any genome, the question of consistency would become redundant: Any set of gene clusters is consistent since there is a valid gene order containing all assigned clusters. For instance, we can simply create a short sequence of genes according to each cluster separately and then concatenate these sequences to an absurd yet valid gene order. Such a construction is possible for any gene cluster model. As a consequence, consistency always holds and does not contribute to a specification of reasonable reconstruction results.

Even if we replace the naive concatenation approach and instead construct preferably compact valid gene orders, we cannot avoid to include some genes multiple times. In some cases, this causes side effects. In the example given in Figure 1, the classical parsimony principle is applied to assign gene clusters to the inner nodes of a given tree, minimizing the number of gains and losses of clusters. Although a gene is contained in all input genomes only once, it is reconstructed to occur multiple times for ancestral nodes. In this simple example, we consider a subsequence of only three genes in each input genome and obtain a segment of five genes for the examined internal node. In general, such artifacts imply unnaturally long genomes for higher levels in the tree.

To preclude this unwanted effect, we refine the concept of consistency. Instead of simply restricting the total length of a genome, we limit the multiplicity of each individual gene.

In the following problem definition, we intentionally refrain from specifying a concrete model of gene clusters and instead use the imprecise notion of a *sequence containing a cluster*. For instance, in the simple model of gene adjacencies, a sequence g *contains* a gene cluster $\{a, b\}$ if and only if the genes a and b occur adjacently in g.

Definition 1 (Consistency Problem) *Let $\mathcal{G}_N := \{1, \ldots, N\}$ be the set of genes and let $m : \mathcal{G}_N \to \mathbb{N}$ assign a maximum copy number to each gene. Further, let C be a set of gene clusters. The* consistency problem *is to decide if C is* consistent

with respect to m, i.e., whether there exists a sequence s over $\mathcal{G}_N$ for which the following properties hold:

(i) s contains each gene g at most $m(g)$ times, and

(ii) s contains all gene clusters $c \in C$.

Whenever we want to consider consistency as a reconstruction criterion, we have to provide a solution for the above problem. As we will see in the following sections, the problem complexity highly depends on the specific cluster definition.

In our framework, we assume that the gene multiplicities are given. Nevertheless, we want to sketch some ways to specify $m(g)$ for the internal nodes in the phylogenetic tree.

The threshold m_l for a leaf l is obviously given by the input: We simply count the occurrences of each gene g in the gene order assigned to l to define $m_l(g)$. However, we have to determine m_v for the internal nodes.

For some specific datasets, we can rely on knowledge about the genomic history. For instance, several studies suggest two whole genome duplications in the evolution of the Chordate genome in the teleost fishes lineage (Jaillon et al., 2004). Such information can be used to deduce the ancestral number of genes.

Otherwise, the most accurate but also elaborate approach would be to deploy *gene-tree species-tree reconciliation* (Page and Charleston, 1997) to reconstruct the history of the genes in terms of speciation events, gene duplications and gene losses. While less extensive and more suitable for our needs, one could also utilize approaches which do not require any further data or pre-knowledge. Probability-based methods (Csűrös and Miklós, 2009) could be applied to effectively and reliably infer ancestral gene multiplicities $m_v(g)$ for all internal nodes v, given the copy number at the leaves. Or, we could apply the concept of parsimony and minimize the amount of copy number differences. A less restrictive solution is to define the multiplicity of a gene g for node u in a bottom-up fashion as the maximum over the multiplicities of its child nodes $v_1, \ldots, v_k$: $m_u(g) := \max_{i=1,\ldots,k} \left(m_{v_i}(g)\right)$.

Instead of performing a separate preprocessing step to fix the thresholds in advance, one could also try to include the gene multiplicity into the overall objective of the reconstruction. However, in general, optimizing for a combination including an original objective, consistency, and the gene copy number would be an intricate task due to the strong interdependencies of the sub-criteria.

3 An Efficient Solution for Adjacencies

Probably the simplest formalization of co-localization of genes is the concept of *adjacencies*, i.e., pairs of directly neighboring genes. This elementary pattern of gene order conservation, also known as *gene pairs* or *neighboring genes*, has been widely used in whole genome comparison. Especially in the field of gene order

reconstruction, this model is one of the most prevalent concepts (Ma et al., 2006; Bhutkar et al., 2007; Chauve and Tannier, 2008).

3.1 Unsigned Adjacencies

In the following, we formalize the concept of adjacencies and present a method to efficiently solve the consistency problem for adjacencies on sequences, i.e., to decide if there exists a sequence that contains a set of given adjacencies while considering each gene g at most $m(g)$ times. To model the problem, we use a graph theoretic approach.

Definition 2 (Unsigned Adjacencies on Sequences) *Let $\mathcal{G}_N := \{1, \ldots, N\}$ be a set of genes. An adjacency $\{a, b\}$ of the genes $a, b \in \mathcal{G}_N$ is contained in a sequence s over $\mathcal{G}_N$ if and only if a and b occur adjacently at least once in s.*

Definition 3 (Gene Order Graph) *Let $\mathcal{G}_N = \{1, \ldots, N\}$ be a set of genes and C be a set of pairs $\{a, b\}$ with $a, b \in \mathcal{G}_N$. Then, the* gene order graph of C*, denoted by $G_N(C)$, is the graph with the vertex set $\{v_g \mid g \in \mathcal{G}_N\}$ and the edge set $\big\{\{v_a, v_b\} \mid \{a, b\} \in C\big\}$.*

The gene order graph of a set of adjacencies C can be constructed in $\mathcal{O}(N + |C|)$ time and space. In this process, we keep track of the degree of each node v_g, denoted by $\deg(v_g)$. Then, the following lemma allows us to test for consistency of C in $\mathcal{O}(N + |C|)$ steps and thus in a total running time and with a space requirement of $\mathcal{O}(N + |C|)$.

Lemma 1 *Let $\mathcal{G}_N = \{1, \ldots, N\}$ be a set of genes and let $m : \mathcal{G}_N \to \mathbb{N}$ assign a maximum copy number to each gene. Further, let C be a set of pairs $\{a, b\}$ with $a, b \in \mathcal{G}_N$ and $G_N(C) = (V, E)$ be the gene order graph of C. Then, C is consistent with respect to m if and only if the following conditions hold:*

(i) $\deg(v_g) \leq 2m(g)$ *for all vertices $v_g \in V$, and*

(ii) $\sum_{v_g \in c} \big(2m(g) - \deg(v_g)\big) > 0$ *for each connected component c in $G_N(C)$.*

Proof. Assume we have given $\mathcal{G}_N$, m, C and $G_N(C)$ as required by the lemma. We extend the gene order graph $G_N = (V, E)$ to a multigraph $H_N = (V', E')$, where the new vertex set contains one additional node v_0, i.e., $V' = V \cup \{v_0\}$. The multiset of edges E' contains all edges in E with multiplicity one and further auxiliary edges: For each vertex $v_g \neq v_0$ with $deg(v_g) < 2m(g)$ we add the edge $\{v_0, v_g\}$ with multiplicity $2m(g) - deg(v_g)$ to E'.

If condition (i) of the lemma holds, then all nodes in the obtained extended graph have even degree: All vertices $v_g \neq v_0$ are filled up to a degree of $2m(g)$ and v_0 is incident to $\sum_{v_g \in V} \big(2m(g) - deg(v_g)\big) = \sum_{v_g \in V} 2m(g) - 2|C|$ edges.

Further, condition (ii) implies that for each connected component of G_N, in the extended graph, at least one edge connects this subgraph to v_0. Hence, H_N is connected.

Conditions (i) and (ii) imply that H_N is Eulerian. That means, there is a closed walk (Eulerian cycle) which contains all edges, especially the edges of the original gene order graph, exactly once. Since each node $v_g \neq v_0$ has a degree of $2m(g)$, it is traversed exactly $m(g)$ times. Each such Eulerian path corresponds to a sequence of genes that contains all adjacencies in C and each gene g exactly $m(g)$ times, as exemplified in Figure 2. Thus, C is consistent with respect to m.

On the contrary, if condition (i) is not satisfied, there is at least one gene g that is contained in more given adjacencies than its multiplicity $m(g)$ allows. And, if condition (ii) does not hold for any connected component c, the maximum number of adjacencies of all genes in c is exhausted and the genes cannot be put into a linear order, i.e., a cycle containing v_0, with the remaining genes. In both cases, the existence of a valid gene order is precluded and, thus, consistency is disproven. □

3.2 Signed Adjacencies

A slightly more sophisticated variant of the adjacency model is motivated by the observation that the orientation of genes can play a role in co-expression and also in gene order conservation (Huynen et al., 2001).

Definition 4 (Signed Adjacencies on Signed Sequences) *Let $\mathcal{G}_N := \{1, \ldots, N\}$ be a set of genes. A signed adjacency $\{a, b\}$ of the genes $a, b \in \{g, -g \mid g \in \mathcal{G}_N\}$ is contained in a sequence s over $\mathcal{G}_N$ if and only if a is directly followed by $-b$, or b by $-a$ at least once in s.*

Note that the representation of a signed adjacency as an unordered pair is accurate since the definition of containedness does not depend on the actual assignment of a and b.

Example 1 *Consider the model of signed adjacencies for $N = 4$. The signed adjacency $\{2, -3\}$ is contained in both sequences $s_1 = (1, 2, 3, 4)$ and $s_2 = (4, 1, -3, -2)$. No other signed adjacencies of the genes 2 and 3 are contained in any of the two sequences.*

We transfer the general idea from the unsigned to the signed case. To this end, we adjust the definition of the gene order graph. Now, each gene g is represented by two nodes in the graph, where each such pair is connected by $m(g)$ edges.

Definition 5 (Signed Gene Order Graph) *Let $\mathcal{G}_N = \{1, \ldots, N\}$ be a set of genes and let $m : \mathcal{G}_N \to \mathbb{N}$ assign a maximum copy number to each gene. Further, let C be a set of pairs $\{a, b\}$ with $a, b \in \{g, -g \mid g \in \mathcal{G}_N\}$. Then, the* signed gene order graph of C*, denoted by $G_N^s(C)$, is the multigraph with the vertex set $\{v_g, v_{-g} \mid g \in \mathcal{G}_N\}$ and the multiset of edges $\big\{\{v_g, v_{-g}\}$ with multiplicity $m(g) \mid g \in \mathcal{G}_N\big\} \cup \big\{\{v_a, v_b\}$ with multiplicity one $\mid \{a, b\} \in C\big\}$.*

Similarly to the unsigned case, we can construct the graph in $\mathcal{O}(N + |C|)$ time.

Lemma 2 *Let $\mathcal{G}_N = \{1, \ldots, N\}$ be a set of genes and let $m : \mathcal{G}_N \to \mathbb{N}$ assign a maximum copy number to each gene. Further, let C be a set of signed adjacencies $\{a, b\}$ with $a, b \in \{g, -g \mid g \in \mathcal{G}_N\}$ and $G_N^s(C) = (V, E)$ be the signed gene order graph of C. Then, C is consistent with respect to m if and only if the following conditions hold:*

(i) $deg(v_g) \leq 2m(|g|)$ *for all vertices* $v_g \in V$*, and*

(ii) $\sum_{v_g \in c} \big(2m(|g|) - deg(v_g)\big) > 0$ *for each connected component* c *in* $G_N^s(C)$.

Proof. We proceed analogously to the unsigned case described in the proof of Lemma 1: We extend the signed gene order graph $G_N^s = (V, E)$ to a multigraph $H_N^s = (V', E')$, where the new vertex set contains one additional node v_0, i.e., $V' = V \cup \{v_0\}$. The multiset of edges E' contains all edges in E with multiplicity one and further auxiliary edges: For each vertex $v_g \neq v_0$ with $deg(v_g) < 2m(|g|)$ we add the edge $\{v_0, v_g\}$ with multiplicity $2m(|g|) - deg(v_g)$ to E'.

Then, again, the conditions (i) and (ii) of Lemma 2 imply the existence of an Eulerian path in $H_N^s(C)$. But in this case, the correspondence of such a path to a valid gene order is not trivial. When the pair of nodes representing gene g is traversed by a path $(\ldots, v_{-g}, v_g, \ldots)$, this relates to a signed gene order $(\ldots, g, \ldots)$, whereas a path $(\ldots, v_g, v_{-g}, \ldots)$ correlates to a signed gene order $(\ldots, -g, \ldots)$. By definition, $H_N^s(C)$ includes $m(|g|)$ edges $\{v_g, v_{-g}\}$. An Eulerian cycle passes each of these edges, but not necessarily in the above mentioned way. It might also be of the form $(v_0, \ldots, v_f, v_{-g}, v_g, v_{-g}, v_h, \ldots, v_0)$ with $f \neq g \neq h$, which does not represent a signed gene order. In this case, $m(|g|) \geq 2$ and due to the construction of the extended graph, there are $m(|g|)$ edges $\{v_g, v_{-g}\}$ and at least $m(|g|)$ edges $\{v_g, v_h\}$ with $h \neq -g$. Hence, the considered Eulerian cycle has to pass node v_g again in the form $\ldots, v_i, v_g, v_j, \ldots$ with $i \neq -g \neq j$, as shown in Figure 3(a). Without loss of generality assume that $\ldots, v_f, v_{-g}, v_g, v_{-g}, v_h, \ldots$ occurs before $\ldots, v_i, v_g, v_j, \ldots$. However, whenever this situation arises, it is always possible to construct an alternative Eulerian cycle $(v_0, \ldots, v_f, v_{-g}, v_g, v_i, \ldots, v_h, v_{-g}, v_g, v_j, \ldots, v_0)$, as depicted in Figure 3(b). If these modifications are performed for all such improperties, the obtained Eulerian cycle is proper in the sense that it represents a signed gene order $(\ldots, g, \ldots, g, \ldots)$. Thus, conditions (i) and (ii) imply not only the existence of an Eulerian path but also the existence of a valid signed gene order and hence consistency of C with respect to m. The reverse direction of the lemma holds analogously to Lemma 1. □

Based on the definition of a gene order graph, Lemmas 1 and 2 provide algorithms to solve the consistency problem on adjacencies on sequences in time and space linear in the number of genes and in the number of given adjacencies. Both the models and the lemmas can easily be modified to allow one circular gene order

or even several circular chromosomes. Only the connectivity requirement has to be relaxed correspondingly.

4 NP-Completeness for Common Intervals

To find larger conserved regions, we now address a model for gene clusters that, in contrast to adjacencies, generally spans more than two genes: *common intervals*, segments of the genome containing the same set of genes in an arbitrary order but not interrupted by other genes.

The term *interval* stems from the original, mathematical problem statement. There, a common interval is defined on a set of permutations which is, without loss of generality, assumed to include the identity $(1, \ldots, N)$ (otherwise all genes can be renamed appropriately). In this case, a set of genes contiguous in all genomes has to appear contiguously in the identity permutation as well and thus be of the form $\{g, g+1, \ldots, g+l\}$, which corresponds to an interval $[g, g+l]$. In our framework however, a common interval is represented as an arbitrary subset of genes.

The detection of common intervals conserved among several gene orders is a well studied problem. For details, we refer to the recent review of Bergeron et al. (2008).

As already detailed in Introduction, common intervals were successfully applied in ancestral gene order reconstruction (Adam et al., 2007; Chauve and Tannier, 2008; Stoye and Wittler, 2009).

4.1 Basic Common Intervals

In line with other studies, we base our definition on the notion of *character sets*, which enables us to formalize the cluster model in a straightforward way. Since we utilize this term for models on signed sequences later on, we directly define it for the general, signed case. Although, in our framework, a common interval is defined on a *single* gene order, we stick to the term *common* to not confuse the reader familiar with this gene cluster model by redefining the same concept under a different name.

Definition 6 (Character Set) *Let $s = (a_1, \ldots, a_{|s|})$ be a signed sequence. Then, the* character set *of s, denoted $\mathcal{CS}(s)$, is the set of all elements in s: $\mathcal{CS}(s) := \left\{|a| \mid a \in \{a_1, \ldots, a_{|s|}\}\right\}$.*

Definition 7 (Common Intervals on Sequences) *Let $\mathcal{G}_N := \{1, \ldots, N\}$ be a set of genes. Then, a common interval $c \subseteq \mathcal{G}_N$ with $|c| > 1$ is contained in a sequence s over $\mathcal{G}_N$ if and only if s contains a substring s' such that $\mathcal{CS}(s') = c$.*

A common interval can occur multiple times in one genome. Furthermore, one occurrence of a common interval in a genome may contain several occurrences of the same gene, as illustrated by the following example.

Example 2 *Consider the above model for $N = 6$ and sequence $s = (5, 4, 2, 1, 2, 3, 6)$. Then, the common interval $\{1, 2, 3, 4\}$ is contained in s as illustrated below, where the corresponding substring is underlined:*

$$(5, \underline{4, 2, 1, 2, 3}, 6)\,.$$

Recall that we want to find an answer to the question: Given a set of common intervals C and a multiplicity threshold function m, is there a valid gene order that contains all elements of C and meets the restrictions imposed by m? As we will show now, this problem is NP-complete.

Theorem 1 *The consistency problem for common intervals on sequences is NP-complete, even if $\max\{m(g)\} = 2$ and $\max\{|c| \mid c \in C\} = 3$.*

Before giving the proof, we would like to emphasize that this is the strongest possible result. If the maximum multiplicity would be one, the problem becomes the polynomially solvable Consecutive Ones Problem (Booth and Lueker, 1976). If the maximum cluster size is restricted to two, this corresponds to the model of adjacencies, for which we gave a polynomial algorithm in the previous section.

Proof. One can easily formulate an algorithm that verifies a given solution, i.e., a proper gene order, for correctness in polynomial time, which shows that the problem belongs to the complexity class NP.

We will show NP-hardness of the consistency problem on common intervals by reduction from 3SAT(3), which has been proven to be NP-complete by Papadimitriou (1994). 3SAT(3) is a restricted version of 3SAT in which every variable has exactly two positive and one negative occurrence in the clauses.[1] The general technique of the reduction is similar to that used in Maňuch and Patterson (2010) to prove NP-hardness for a generalized variant of the Consecutive Ones Problem. Please note that, in the rather abstract ambience of this proof, we will use the term *object* instead of *gene*.

Given a 3SAT(3) formula ϕ with variables $X = \{x_1, \ldots, x_n\}$ and clauses $C = \{c_1, \ldots, c_m\}$, we construct an instance $\mathcal{C}_\phi$ of the consistency problem for common intervals on sequences consisting of at most $5n+2m$ objects of multiplicity at most two and at most $4n + 2m$ common intervals of size two or three for which a valid sequence s exists if and only if ϕ is satisfiable.

For this instance ϕ of 3SAT(3), we say that a clause *selects* one of its literals in a truth assignment of ϕ if this literal has value *true* in this assignment. Obviously, a truth assignment of ϕ is a satisfying truth assignment if and only if every clause selects at least one literal and for every $x \in X$, at most one of x and $\neg x$ is selected.

[1] We remark that the exact formulation of 3SAT(3) in Papadimitriou (1994) allows also variables with one negated and two positive occurrences, but these can easily be converted to the other type of variables by replacing them with their negations in all clauses. Clearly, this does not affect the complexity of the problem.

We design an instance $\mathcal{C}_\phi$ of the consistency problem composed of clause gadgets which will guarantee the first condition and variable gadgets which will ensure the second condition.

For each 2-clause c_i with literals c_i^1 and c_i^2, we add to $\mathcal{C}_\phi$ the two objects c_i^1 and c_i^2, each of multiplicity two, and the two objects c_i^* and c_i^{**}, each of multiplicity one, and the subsets $S_i^1 = \{c_i^1, c_i^2, c_i^*\}$ and $S_i^2 = \{c_i^*, c_i^{**}\}$. This is referred to as the *2-clause gadget*.

For each 3-clause c_i with literals c_i^1, c_i^2 and c_i^3, we add to $\mathcal{C}_\phi$ the three objects c_i^1, c_i^2 and c_i^3, each with multiplicity 2, and the subset $S_i = \{c_i^1, c_i^2, c_i^3\}$. This is referred to as the *3-clause gadget*.

Figure 4 shows graphical representations of these gadgets, which also highlights that an instance of the consistency problem for common intervals on sequences can be viewed as a hypergraph with a vertex for each object and a hyperedge for each common interval. A sequence that is consistent with this instance is then a collection of walks on this hypergraph that *covers* each hyperedge, that is, for each hyperedge e there is a connected subwalk containing all and only vertices in e, such that no vertex v is visited more than $m(v)$ times.

We say that a literal object c_i^j is *selected* in string s, if in s, c_i^j is enclosed on both the left and right side by at least one object of the clause gadget for clause c_i. Note that in both clause gadgets, at least one of its literal objects is selected in any valid string s. For the 2-clause gadget, a valid string s has to contain one of the substrings $c_i^1, c_i^2, c_i^*, c_i^{**}$ or $c_i^2 c_i^1, c_i^*, c_i^{**}$, or one of their reversals. For the 3-clause gadget, a valid string s has to contain one of the substrings c_i^1, c_i^2, c_i^3 or c_i^2, c_i^1, c_i^3, or c_i^1, c_i^3, c_i^2, or one of their reversals. Clearly, for each clause and each of these substrings, at least one literal of the clause is selected.

Now, all $3n$ literal objects c_i^j from the set of clause gadgets for C will appear in n variable gadgets described below. For each variable x_ℓ with the two positive occurrences c_i^α and c_j^β and the negative occurrence c_k^γ, we already added to $\mathcal{C}_\phi$ the objects c_i^α, c_j^β and c_k^γ, each of multiplicity two in the corresponding clause gadgets for the clauses containing x_ℓ and $\neg x_\ell$. We further add to $\mathcal{C}_\phi$ the two objects x'_ℓ and x''_ℓ, each of multiplicity one, and the four subsets $P_\ell^1 = \{c_i^\alpha, c_k^\gamma, x'_\ell\}$, $P_\ell^2 = \{x'_\ell, c_j^\beta\}$, $P_\ell^3 = \{c_j^\beta, c_k^\gamma\}$, $P_\ell^4 = \{c_i^\alpha, x''_\ell\}$. This is referred to as the *variable gadget* for x_ℓ, depicted in Figure 5.

We will show that the variable gadget for x_ℓ ensures that in a valid string literal c_k^γ is never selected together with c_i^α or c_j^β. Consider a valid string s where the literal c_k^γ is selected. Since one copy of c_k^γ is completely used up by its clause gadget, s must contain the substring $c_j^\beta, c_k^\gamma, c_i^\alpha, x'_\ell, c_j^\beta$ or its reversal because it is the only way to ensure consistency for subsets P_ℓ^1, P_ℓ^2 and P_ℓ^3 with the one remaining copy of c_k^γ. Since the above substring contains two occurrences of c_j^β, literal c_j^β cannot be selected. If literal c_i^α is also selected, then there is no way that s can be consistent with P_ℓ^4, a contradiction to the fact that s is valid. It follows that if $\mathcal{C}_\phi$ has a valid string s, then ϕ is satisfiable.

We now show that the converse holds, namely if ϕ has a satisfying truth assignment τ, then $\mathcal{C}_\phi$ has a valid string s. Given τ, we construct s as follows.

For each clause c_i, we pick one literal c_i^j with value *true* in τ. If c_i is a 2-clause, we create a substring $c_i^{j-1}, c_i^j, c_i^*, c_i^{**}$ satisfying S_i^1 and S_i^2, and if it is a 3-clause, a substring $c_i^{j-1}, c_i^j, c_i^{j+1}$ satisfying S_i, where the upper indices are taken modulo 3. For any variable x_ℓ with the two positive occurrences c_i^α and c_j^β and the negative occurrence c_k^γ:

1. if $\tau(x_\ell) =$ *false*, we create the substrings $c_j^\beta, c_k^\gamma, c_i^\alpha, x'_\ell, c_j^\beta$ and c_i^α, x''_ℓ, fulfilling all $P_\ell^{1,2,3,4}$; and

2. if $\tau(x_\ell) =$ *true*, we create the substring $x''_\ell, c_i^\alpha, c_k^\gamma, x'_\ell, c_j^\beta, c_k^\gamma$, fulfilling all $P_\ell^{1,2,3,4}$.

The requirements imposed by all given common intervals are fulfilled. It remains to be shown that the substrings can be merged to one string s that satisfies the multiplicity restrictions as well. First, we should note that each of the objects with multiplicity one, $c_i^*, c_i^{**}, x'_\ell, x''_\ell$, appears only in one of the created substrings, and thus satisfies the multiplicity condition (no matter how the substrings are concatenated). The objects with multiplicity two are only the literal objects. Since each literal object is contained in exactly one clause gadget and exactly one variable gadget, it appears exactly once in the clause substrings and at most twice in the variable substrings. If a literal object appears only once in the variable substrings, then the multiplicity restriction for this object is satisfied.

Consider a literal object o that appears twice in the variable substrings. It is either c_i^α or c_j^β, where c_i^α and c_j^β are positive occurrences of a variable x_ℓ and $\tau(x_\ell) =$ *false*, or c_k^γ, where c_k^γ is the negative occurrence of a variable x_ℓ and $\tau(x_\ell) =$ *true*. In any case, (a) one occurrence of o in the variable substrings is the first or the last element of the variable substring; and (b) o cannot be picked in its clause (as it is *false*), and thus appears as the first or the last element in the clause substring that contains it. Consequently, the two substrings (one clause and one variable) can be merged to one by reusing the literal object o, and thus only two copies of o are used. Since each of these merges connects one clause substring and one variable substring and each variable substring is used in at most one merge, it follows that these merges cannot create a cycle.

Eventually, any concatenation of the remaining substrings yields a string s that is consistent with $\mathcal{C}_\phi$. Thus if ϕ has a satisfying assignment τ, then $\mathcal{C}_\phi$ has a valid string s.

Since the number of objects used in the construction is at most $5n + 2m$, the number of common intervals is at most $4n + 2m$, and each common interval is of size at most three, i.e., the construction is linear in the size of ϕ, it can be built in linear time, and hence the consistency problem on common intervals is NP-hard.

□

4.2 Variants of Common Intervals

Besides its classical definition, there are different generalizations of common intervals on sequences discussed in the literature, such as r-window clusters (Friedman and Hughes, 2001; Durand and Sankoff, 2002), and max-gap clusters (He and Goldwasser, 2005; Hoberman and Durand, 2005; Pasek et al., 2005), or approximate gene clusters (Rahmann and Klau, 2006; Böcker et al., 2009). Since the consistency problem is NP-complete for basic common intervals, any generalization is NP-hard as well.

In contrast to generalizations, there are also other cluster models which are restricted variants of common intervals. In the following, we will discuss such models, in particular framed and nested common intervals.

Framed Common Intervals

This gene cluster model, common intervals framed by two genes whose orientations have to be conserved, was first introduced on permutations as *conserved intervals* (Bergeron and Stoye, 2006). In gene order reconstruction, framed common intervals on permutations was the first model to formally state the problem of finding putative ancestral sets of gene clusters preserving consistency (Bergeron et al., 2004).

Definition 8 (Framed Common Intervals on Signed Sequences) *Let $\mathcal{G}_N := \{1, \ldots, N\}$ be a set of genes. A framed common interval $[a\, I\, b]$ consists of two* extremities *a and b with $|a|, |b| \in \mathcal{G}_N$, and a set of* inner elements *$I \subseteq \mathcal{G}_N$. We say that $[a\, I\, b]$ is contained in a signed sequence s, if and only if, in s, a is followed by b or $-b$ is followed by $-a$, and the character set of the substring between the extremities is equal to I.*

According to this definition, a gene can be an extremity and an inner element, or even a left and right extremity at the same time. Apart from that, analogously to basic common intervals, a cluster can occur multiple times in one genome, and one gene can be contained several times in one cluster occurrence, as illustrated by the following example.

Example 3 *Consider the model of framed common intervals for $N = 6$ and sequence $s = (5, 4, -2, -1, 2, -3, 6)$. Besides others, the framed common interval $[4\,\{1, 2\}\,{-3}]$, is contained in s as illustrated by the* box diagram *below, where the occurrences of the extremities and the inner elements are surrounded by rectangles:*

$$s = (\ +5,\ \boxed{+4,}\ \boxed{-2,\ -1,\ +2,}\ \boxed{-3,}\ +6\ ).$$

The obvious relationship of basic and framed common intervals allows us to infer an important correlation of these models with respect to the consistency problem: Any instance of this problem for common intervals can be reduced to an

instance of framed common intervals. Based on this, we can deduce the following statement.

Theorem 2 *The consistency problem for framed common intervals on signed sequences is NP-complete, even if* $\max\{m(g)\} = 2$ *and* $\max\{|I| \mid [a\, I\, b] \in C\} = 6$.

Proof. Again, one can easily formulate an algorithm that verifies a given solution for correctness in polynomial time, which shows that the problem belongs to the complexity class NP.

NP-hardness is shown by reducing the basic common intervals used in the proof of Theorem 1 to framed common intervals.

The basic idea is to replace each common interval $B = \{e_1, \ldots, e_m\}$ by a framed common interval $\mathbb{B} = [\tilde{B}\, \{e_1, \ldots, e_m, \ldots\}\, \bar{B}]$ containing, besides others, the basic common interval as inner elements. Then, if the framed common intervals allow for a valid gene order s, there is a valid gene order s' for the original instance on basic common intervals: We simply remove all newly introduced objects from s such that only the objects contained in the basic common intervals are left in s'. Because the inner elements of all framed common intervals have to occur contiguously in s, the objects of the basic common intervals occur contiguously in s'.

Since the given basic common intervals used in the proof of Theorem 1 overlap, the framing elements have to be included into the set of inner elements of overlapping intervals. We use the following technique to ensure that, if there is a valid gene order for the basic common intervals, there is a valid gene order for the constructed set of framed common intervals. Together with the argument in the previous paragraph, this will yield equivalence of the two instances of the consistency problem.

For each basic common interval $B = \{e_1, \ldots, e_m\}$ overlapping with intervals $B_1, \ldots, B_k$, we create a framed common interval $\mathbb{B} = [\tilde{B}\, \{e_1, \ldots, e_m, \tilde{B}_1, \ldots, \tilde{B}_k, \bar{B}_1, \ldots, \bar{B}_k\}\, \bar{B}]$ containing the framing elements of $\mathbb{B}_1, \ldots, \mathbb{B}_k$, the framed common intervals constructed for $B_1, \ldots, B_k$. Note that this means that the framing elements $\tilde{B}$ and $\bar{B}$ also appear as inner elements to $\mathbb{B}_1, \ldots, \mathbb{B}_k$. All basic common intervals used in the proof of Theorem 1 have the property that no common interval is included in anther, and furthermore, in a valid gene order, an occurrence of a given basic common interval can overlap with the occurrence of only one other basic common interval on each side. Assume, the occurrence of some B overlaps with B_l in $e_1, \ldots, e_l$ on one side and with B_r in $e_r, \ldots, e_m$ on the other side. Then, we can extend the substring that fulfills B to $\tilde{B}, e_1, \ldots, e_l, \bar{B}_l, \ldots, \tilde{B}_r, e_r, \ldots, e_m, \bar{B}$. Between $\bar{B}_l$ and $\tilde{B}_r$ we include all remaining inner elements of $\mathbb{B}$ in an arbitrary order. The resulting substring fulfills $\mathbb{B}$ and also allows a realization of the framed common intervals created for B_l and B_r. We can choose the orientation (the sign) of the new objects in the string extension in accordance to the definition of the framed common intervals, e.g., positive in both. This extension can be performed for all common intervals such that, finally, all framed common intervals are contained in the extended, overall string.

What remains to be shown is that such a construction is possible using at most six inner elements in each framed common interval, as well as that a maximum multiplicity of two is sufficient.

To minimize the number of inner elements, we do not always add both framing elements to all overlapping intervals. The structure of the common intervals used in the gadgets of the proof of Theorem 1 restricts the possible overlaps of their occurrences in a valid gene order. As can be seen in the proof of Theorem 1, if there is a valid sequence, we can construct one using the following orders (or their reversals) of interval occurrences within the gadgets:

$$P_\ell^3, P_\ell^1, P_\ell^2 \text{ or } P_\ell^4, P_\ell^1, P_\ell^2, P_\ell^3, \text{ and } S_i^1, S_i^2 .$$

Within the gadget, P_ℓ^3 can only be followed by P_ℓ^1. We thus add $\bar{P}_\ell^3$ (but not $\tilde{P}_\ell^3$) to the inner elements of $\mathbb{P}_\ell^1$, and $\tilde{P}_\ell^1$ (but not $\bar{P}_\ell^1$) to those of $\mathbb{P}_\ell^3$. Analogously, we add $\bar{P}_\ell^1$ to $\mathbb{P}_\ell^2$ and $\tilde{P}_\ell^2$ to $\mathbb{P}_\ell^1$, $\bar{P}_\ell^4$ to $\mathbb{P}_\ell^1$ and $\tilde{P}_\ell^1$ to $\mathbb{P}_\ell^4$, $\bar{P}_\ell^2$ to $\mathbb{P}_\ell^3$ and $\tilde{P}_\ell^3$ to $\mathbb{P}_\ell^2$, and $\bar{S}_i^1$ to $\mathbb{S}_i^2$ and $\tilde{S}_i^2$ to $\mathbb{S}_i^1$.

A 2-clause gadget c_i overlaps the gadgets of two variables, say x_j and x_k. As can be seen in the proof of Theorem 1, if there is a valid sequence, we can construct one with one of the following orders (or their reversals) of interval occurrences:

$$P_j^p, S_i^1, S_i^2, \text{ or } P_k^q, S_i^1, S_i^2$$

where $p, q \in \{2, 3, 4\}$, depending on where it overlaps the variable gadgets. Thus, we add $\tilde{S}_i^1$ to the inner elements of $\mathbb{P}_j^p$ and $\mathbb{P}_k^q$.

A 3-clause gadget c_i overlaps the gadgets of three variables, say x_j, x_k and x_ℓ in the element c_i^1, c_i^2 and c_i^3, respectively. As can be seen in the proof of Theorem 1, if there is a valid sequence, we can construct one with one of the following orders (or their reversals) of interval occurrences:

$$P_j^p, S_i, P_k^q \quad \text{or} \quad P_j^p, S_i, P_\ell^r \quad \text{or} \quad P_\ell^r, S_i, P_k^q,$$

where $p, q, r \in \{2, 3, 4\}$, depending on where the variable gadgets are overlapped by S_i. We add $\tilde{S}_i$ to the inner elements of $\mathbb{P}_j^p$, $\bar{S}_i$ to the inner elements of $\mathbb{P}_k^q$, and $\tilde{S}_i$ and $\bar{S}_i$ to the inner elements of $\mathbb{P}_\ell^r$. This way, in any of the three cases, there is at least one copy of each framing element of $\mathbb{S}_i$ available on both sides.

In summary, we reduce a given set of common intervals as used in the proof of Theorem 1 to a set of framed common intervals with at most six inner elements as follows:

For the basic common intervals $P_\ell^{1,2,3,4}$ used in the variable gadget for x_ℓ with positive occurrences c_i^α and c_j^β and negative occurrence c_k^γ, we create the framed

common intervals

$$\mathbb{P}^1_\ell = [\tilde{P}^1_\ell \, \{c^\alpha_i, c^\gamma_k, x'_\ell, \tilde{P}^2_\ell, \bar{P}^3_\ell, \bar{P}^4_\ell\} \, \bar{P}^1_\ell],$$
$$\mathbb{P}^2_\ell = [\tilde{P}^2_\ell \, \{x'_\ell, c^\beta_j, \bar{P}^1_\ell, \tilde{P}^3_\ell\} \cup I^\beta_j \, \bar{P}^2_\ell],$$
$$\mathbb{P}^3_\ell = [\tilde{P}^3_\ell \, \{c^\beta_j, c^\gamma_k, \tilde{P}^1_\ell, \bar{P}^2_\ell\} \cup I^\gamma_k \, \bar{P}^3_\ell] \text{ and}$$
$$\mathbb{P}^4_\ell = [\tilde{P}^4_\ell \, \{c^\alpha_i, x''_\ell, \tilde{P}^1_\ell\} \cup I^\alpha_i \, \bar{P}^4_\ell], \text{ where}$$
$$I^\mu_t = \begin{cases} \{\tilde{S}_t\} & \text{if } \mu = 1 \text{ and } |c_t| = 3, \\ \{\bar{S}_t\} & \text{if } \mu = 2 \text{ and } |c_t| = 3, \\ \{\tilde{S}_t, \bar{S}_t\} & \text{if } \mu = 3 \text{ and } |c_t| = 3, \\ \{\tilde{S}^1_t\} & \text{if } |c_t| = 2. \end{cases}$$

For the basic common intervals $S^{1,2}_i$ used in the 2-clause gadget for c_i, we create the framed common intervals

$$\mathbb{S}^1_i = [\tilde{S}^1_i \, \{c^1_i, c^2_i, c^*_i, \tilde{S}^2_i, \bar{P}(c^1_i), \bar{P}(c^2_i)\} \, \bar{S}^1_i] \text{ and}$$
$$\mathbb{S}^2_i = [\tilde{S}^2_i \, \{c^*_i, c^{**}_i, \bar{S}^1_i\} \, \bar{S}^2_i],$$

where we define $\bar{P}(c^j_i)$ to be the right framing element of the (unique) $\mathbb{P}^m_\ell$ that contains c^j_i and $\tilde{S}^1_i$ as inner elements (m can only be 2, 3 or 4).

For the basic common interval S_i used as in the 3-clause gadget for c_i, we create the framed common interval

$$\mathbb{S}_i = [\tilde{S}_i \, \{c^1_i, c^2_i, c^3_i, \bar{P}(c^1_i), \bar{P}(c^2_i), \bar{P}(c^3_i)\} \, \bar{S}_i],$$

where we define $\bar{P}(c^j_i)$ to be the right framing element of the (unique) $\mathbb{P}^m_\ell$ that contains c^j_i, and $\tilde{S}_i$ and/or $\bar{S}_i$ as inner elements (m can only be 2, 3 or 4).

It remains to be shown that a maximum multiplicity of two for all newly added elements suffices. This is true, because each new element is included in the inner elements of at most two intervals. In fact, we can assign a multiplicity of one to some of the objects. We define: $m(\bar{P}^1) = m(\tilde{P}^{2,3,4}) = m(\tilde{S}^2_i) = m(\bar{S}^2_i) = 1$ and $m(\tilde{P}^1) = m(\bar{P}^{2,3,4}) = m(\tilde{S}_i) = m(\bar{S}_i) = 2$.

Since the number of objects used in the construction is at most $6n + 8m$, the number of framed common intervals is at most $4n + 2m$, and each framed common interval contains at most six inner elements, i.e., the construction is linear in the size of ϕ, it can be built in linear time, and hence the consistency problem on common intervals is NP-hard. □

Please note that, again, for a maximum multiplicity of one, polynomial solutions exist. Framed common intervals with no inner elements are equivalent to signed adjacencies, for which we gave an efficient solution. However, there is a gap left for framed common intervals with one to five inner elements. For these, the complexity is still open.

Nested Common Intervals

Hoberman and Durand (2005) discussed nestedness as a desired property of gene clusters and proposed a first algorithm to identify respective clusters. Recently, *nested common intervals* were formally defined and studied in Blin et al. (2010), and gave efficient algorithms to detect them in genomes modeled both as permutations and as sequences.

Definition 9 (Nested Common Intervals on Sequences) *Let $\mathcal{G}_N := \{1, \ldots, N\}$ be a set of genes. The structure of a nested common interval is defined recursively. A* nested common interval *is either*

(i) an unordered pair of genes $\{a, b\}$ with $a \neq b$, which is contained in a sequence s over $\mathcal{G}_N$ if and only if a and b are adjacent in s, or

(ii) an unordered pair $\{c, a\}$ of a nested common interval c and a gene a, which is contained in a sequence s if and only if, in s, a is adjacent to a substring s' of s such that $\mathcal{CS}(c) = \mathcal{CS}(s')$ and c is contained in s',

where the character set of a nested common interval is the set of all contained genes: $\mathcal{CS}(\{a, b\}) := \{a, b\}$ and $\mathcal{CS}(\{c, a\}) := \mathcal{CS}(c) \cup \{a\}$. Further, we define the size of a cluster being $|\{a, b\}| := 2$ and $|\{c, a\}| := |c| + 1$, respectively.

Similar to the other cluster models discussed above, any nested common interval may occur multiple times in one genome and one gene may be contained multiple times in the occurrence of a cluster in one genome. Analogously to framed common intervals, one gene may be incorporated in the definition of one cluster several times.

Example 4 *Consider the model of nested common intervals for $N = 6$ and sequence $s = (5, 4, 2, 1, 2, 3, 6)$. Then, besides others, the nested common interval $\{\{\{2, 3\}, 1\}, 4\}$ is contained in s as illustrated below, where the occurrences of the subclusters are indicated by lines:*

$$(5, \underline{4}, \underline{2, 1, \underline{2, 3}}, 6)\,.$$

In contrast, $\{\{\{1, 3\}, 2\}, 4\}$ is not contained in s since, although gene 4 is adjacent to a substring with character set $\{1, 2, 3\}$, none of the occurrences of gene 2 is adjacent to a substring with character set $\{1, 3\}$.

Note that cluster $\{\{2, 3\}, 3\}$ is not contained in g, because 3 is not adjacent to a substring with character set $\{2, 3\}$, whereas cluster $\{\{1, 2\}, 2\}$ is contained in s:

$$(5, 4, \underline{2, 1}, \underline{2}, 3, 6)\,.$$

Even the strict assumption of nestedness is not strong enough to allow an efficient verification of consistency. In fact, similar to basic common intervals, there is no gap left for fixed-parameter tractability in the considered parameters.

Theorem 3 *The consistency problem for nested common intervals on sequences is NP-complete, even if $\max\{m(g)\} = 2$ and $\max\{|c| \mid c \in C\} = 3$.*

Proof. NP-hardness is proven by reduction from 3SAT(3) using a construction very similar to that of Theorem 1. Given 3SAT(3) formula ϕ, we will again design an instance $\mathcal{C}_\phi$ of the consistency problem on *nested* common intervals on sequences comprising of clause gadgets and a variable gadget, and then argue why they simulate exactly this instance ϕ.

For each 2-clause c_i with literals c_i^1 and c_i^2, we add to $\mathcal{C}_\phi$ the two objects c_i^1 and c_i^2, each of multiplicity two, and the object c_i^* of multiplicity one, and the *nested* common interval $S_i^1 = \{\{c_i^1, c_i^2\}, c_i^*\}$. The 2-clause gadget is depicted in Figure 6(a).

For each 3-clause c_i with literals c_i^1, c_i^2 and c_i^3, we add to $\mathcal{C}_\phi$ the three objects c_i^1, c_i^2 and c_i^3, each with multiplicity two, the three objects $\tilde{c}_i^1, \tilde{c}_i^2$ and $\tilde{c}_i^3$, each with multiplicity one, the three objects $\bar{c}_i^1, \bar{c}_i^2$ and $\bar{c}_i^3$, each with multiplicity two and the six nested common intervals $S_i^1 = \{\{c_i^1, \bar{c}_i^1\}, \tilde{c}_i^1\}, S_i^2 = \{\{c_i^2, \bar{c}_i^2\}, \tilde{c}_i^2\}, S_i^3 = \{\{c_i^3, \bar{c}_i^3\}, \tilde{c}_i^3\}, S_i^4 = \{\bar{c}_i^1, \bar{c}_i^2\}, S_i^5 = \{\bar{c}_i^2, \bar{c}_i^3\}, S_i^6 = \{\bar{c}_i^3, \bar{c}_i^1\}$. The 3-clause gadget is depicted in Figure 6(b).

Note again that in both clause gadgets, at least one of the literal objects is selected in any valid string s. For the 2-clause gadget, string s has to contain one of the substrings c_i^1, c_i^2, c_i^* or c_i^2, c_i^1, c_i^*, or one of their reversals, thus a literal object is always selected in this case. In the 3-clause gadget, if no literal object is selected in string s, i.e., s contains substrings $\tilde{c}_i^q, \bar{c}_i^q, c_i^q$ (or their reversals) for every $q \in \{1, 2, 3\}$, there is only one remaining copy of $\bar{c}_i^q$ for $q \in \{1, 2, 3\}$ and hence there is no way that s can be consistent with all of $S_i^{\{4,5,6\}}$ simultaneously without creating a cycle, a contradiction. Therefore at least one literal object is selected in this case as well.

For each variable x_ℓ, we will use the same construction as in the proof of Theorem 1 with one exception that instead of the basic common interval $P_\ell^1 = \{c_i^\alpha, c_k^\gamma, x'_\ell\}$, we use the nested common interval $P_\ell^1 = \{\{c_i^\alpha, c_k^\gamma\}, x'_\ell\}$, cf. Figure 7. It follows by the same argument as in the proof of Theorem 1 that in a valid string c_k^γ is never selected together with c_i^α or c_j^β. It follows that if $\mathcal{C}_\phi$ has a valid string s, then ϕ is satisfiable.

We now show that the converse holds, namely if ϕ has a satisfying truth assignment τ, then $\mathcal{C}_\phi$ has a valid string s. Given τ, we construct s as follows.

For each clause c_i, we pick one literal c_i^j with value *true* in τ. If c_i is a 2-clause, we create a substring c_i^{j-1}, c_i^j, c_i^* satisfying S_i^1, and if it is a 3-clause, substrings (i) $\tilde{c}_i^j, c_i^j, \bar{c}_i^j, \bar{c}_i^{i+1}, \bar{c}_i^{j+2}, \bar{c}_i^j$, (ii) $c_i^{j+1}, \bar{c}_i^{j+1}, \tilde{c}_i^{j+1}$, and (iii) $c_i^{j+2}, \bar{c}_i^{j+2}, \tilde{c}_i^{j+2}$ satisfying $S_i^{\{1,\ldots,6\}}$, where the upper indices are taken modulo 3. The clause strings have the same properties as the clause string in the proof of Theorem 1: each literal object appears only once in the clause substrings and if a literal object has the value *false* in τ then it appears as the first or last element in one of the clause substrings. For each variable x_ℓ, we create the same substrings satisfying all $P_\ell^{1,2,3,4}$ as in the proof of Theorem 1.

It can be easily checked that the requirements imposed by all given nested common intervals are fulfilled. It follows by the same argument as in the proof of Theo-

rem 1 that the created substrings can be merged and concatenated into a valid string s. Thus, if ϕ has a satisfying assignment τ, then $\mathcal{C}_\phi$ has a valid string s.

Since the number of objects used in this construction is at most $5n + 6m$, the number of nested common intervals is at most $5n + 9m$, and each nested common interval is of size at most three, i.e., the construction is linear in the size of ϕ, it can be built in linear time, and hence the consistency problem on common intervals is NP-hard. □

Further Variations and Restrictions

Our NP-completeness results also hold for further variations of the above models.

If we preclude a nested common interval to contain any gene multiple times, e.g., $\{\{a, b\}, a\}$, or if we preclude any gene to be left and right extremity of a framed common interval, e.g., $[a\, I\, a]$, our proof techniques still apply and thus NP-completeness still holds. Also, if we restrict any occurrence of a common or nested common interval within a genome to contain each gene only once, NP-completeness still holds.

Instead of restricting the multiplicity for each gene individually, one could define a maximum total number of genes, i.e., a maximum genome length. But going back to the NP-hardness proofs, we find that they also hold in this model. In the proofs for basic and nested common intervals, each gene g with multiplicity $m(g)$ occurs in at least $2m(g)-1$ intersecting gene clusters such that all "allowed" copies of that gene are actually required. Thus, there is no flexibility left to use one gene only $m(g_1) - 1$ times and another gene $m(g_2) + 1$ times. Also, the auxiliary genes used in the proof for framed common intervals have all to be used exactly as often as specified. Hence, in all models, the proofs also hold if only $\sum_g m(g)$ is given as an overall objective. This, in turn, directly implies NP-hardness of the minimization version of the consistency problem on these models: Given a set of gene clusters, find a minimum-length sequence of genes that contains all given clusters.

Since a typical prokaryotic genome consists of one circular chromosome, we are also interested in modeling gene order as circular sequences. In fact, based on the above NP-completeness results, we can deduce NP-completeness of the consistency problem for all the considered gene cluster models on circular sequences in a straight forward fashion.

To redefine the cluster models, we allow any gene cluster to appear at the end of a sequence $(\ldots, g_{|g|-1}, g_{|g|})$ such that its occurrence can be continued at the beginning of the sequence $(g_1, g_2, \ldots)$, i.e., we assume circular gene orders.

Corollary 1 *The consistency problem on circular sequences is NP-complete for the gene cluster models of basic, framed and nested common intervals for the maximum multiplicities and cluster sizes as stated in Theorems 1–3.*

Proof. One can easily formulate an algorithm that verifies a given solution for correctness in polynomial time, which shows that the problem belongs to the complexity class NP.

The consistency problem for the considered gene cluster models on linear sequences is NP-complete according to Theorems 1, 2 and 3.

Let the set of genes $\mathcal{G}_N$, the set of clusters C and the multiplicity function m define an instance of the consistency problem for basic, framed or nested common intervals on linear sequences. Then, we reduce this instance to a problem instance of the corresponding consistency problem on circular sequences in polynomial time as follows:

- $\mathcal{G}_{N+2} := \mathcal{G}_N \cup \{N+1, N+2\}$,
- $C' := C \cup \{c\}$, where c is a gene cluster containing exactly the genes $N+1$ and $N+2$ (the basic or nested common interval $\{N+1, N+2\}$, or the framed common interval $[N+1\ \{\}\ N+2]$, respectively),
- $m' := m \cup \{N+1 \longmapsto 1, N+2 \longmapsto 1\}$.

If C is consistent with respect to m, then there is a linear sequence $g = (a_1, \ldots, a_l)$ satisfying the requirements of m and containing all clusters in C. Obviously, the circular sequence $g' := (a_1, \ldots, a_l, N+1, N+2)$ also contains all clusters in C and cluster c and satisfies the requirements of m'.

Now, we show the opposite implication: If C' is consistent with respect to m', then there is a circular sequence $g' = (a_1, \ldots, a_l, N+1, N+2)$ satisfying the requirements of m' and containing all clusters in C'. Since no cluster in C spans the genes $N+1$ and $N+2$, the linear sequence $g := (a_1, \ldots, a_l)$ contains all clusters in C' except for c, i.e., all clusters in C, and satisfies the requirements of m. □

So far, we only considered genomes composed of one chromosome. As already mentioned in Section 3, the algorithm presented for adjacencies can easily be modified to handle several chromosomes—linear or circular. Also, the NP-completeness results hold for several linear chromosomes. In fact, the proofs are based on the construction of several substrings which could be kept as separate chromosomes instead of concatenating them to one string. In contrast, the proof of the latter lemma builds on exactly one circular chromosome. It is open whether NP-completeness holds for the case of several circular chromosomes for these models.

5 Conclusion

In this paper, we have discussed the consistency problem, i.e., the problem of deciding whether there exists a valid gene order comprising a given set of gene clusters. We have discussed this question for different gene cluster models on sequences

with restricted gene multiplicities. In summary, we identified a strict border between gene cluster models for which we can verify consistency efficiently and those for which we cannot. The complexity rises drastically from linear time for adjacencies to NP-hard for more general cluster models, even if they are strongly restricted.

This raises the question for a sequence-based gene cluster model that, on the one hand, allows some degree of flexibility and, on the other hand, offers a polynomial-time algorithm to verify consistency. The integration of such a model into any of the existing reconstruction methods could increase sensitivity. Actually, first results on both simulated and real data indicate that within segments of conserved gene content, the order of the genes is conserved almost exactly (Wittler, 2010). Thus, a model covering only single missing or additional genes, or the reversal of two neighboring genes could already enhance reconstruction results strongly.

On permutations, the consistency problem for common intervals is the consecutive ones property (C1P) problem (Booth and Lueker, 1976; Habib et al., 2000; Hsu, 2002; Hsu and McConnell, 2004). Thus, one approach that includes additional or missing genes is to relax the condition of the consecutivity of the ones of each row, by allowing gaps, with some restriction on the nature of these gaps. The question is then to decide if there is an ordering of the columns that satisfies these relaxed C1P conditions. In Goldberg et al. (1995), the authors introduced the k-consecutive-ones property (k-C1P): Decide if the columns of a binary matrix can be permuted such that each row contains at most k blocks. This problem is NP-complete for every $k \geq 2$ (Goldberg et al., 1995), and also minimizing the number of gaps in the entire matrix is NP-complete even if each row of this matrix has at most two ones (Haddadi, 2002).

In the spirit of this approach, Chauve et al. (2009) define the gapped C1P: given two integers k and δ, a binary matrix M has the (k, δ)-C1P if its columns can be permuted such that each row contains at most k blocks and no gap larger than δ. While they show that such a property can be decided in polynomial-time when the number of ones in each row of M is bounded, they show that the general case is hard. In particular, they show for all $k \geq 2, \delta \geq 1, (k, \delta) \neq (2, 1)$ that the (k, δ)-C1P is NP-complete. So indeed, in this case, aside from the single open case of the complexity of the (2,1)-C1P, there is also a strict complexity border between the classical C1P ((1,0)-C1P in the gapped C1P context) and this relaxed model.

In Maňuch and Patterson (2010), the authors show also for binary matrices of bounded degree d that the k-C1P is NP-complete even when $d = 3$, which is quite suprising, as this is the weakest form of consecutivity requirement: in each row, only two of the ones must be adjacent. This is shown with an NP-completeness construction based on finding a collection of walks on a hypergraph H that *covers* each hyperedge in H, a technique that inspired some of the NP-completeness constructions in this work.

We implemented the gene order graph to model adjacencies on sequences and integrated this gene cluster model into our unified reconstruction framework, pre-

sented in Stoye and Wittler (2009), available from the web site `bibiserv.techfak.uni-bielefeld.de/rococo/`. An elaborate description of the method and the results can be found in Wittler (2010). We refrain from reporting detailed results here because these are concerned more with the reconstruction method than with the general concept of consistency discussed in this paper. Nevertheless, we would like to mention the following overall findings. Simulations showed that estimating the gene multiplicities using the simple maximum approach does not significantly decrease the accuracy of the reconstruction compared to using the "real" simulated copy numbers. Furthermore, we applied our method to genomic data of Corynebacteria using different gene cluster models: Common intervals on permutations and adjacencies on sequences. A comparison of the results revealed a large overlap. Nevertheless, many conserved segments could only be identified by either of the approaches. This highlights the importance of studying gene cluster reconstruction with respect to different, especially flexible, models for gene clusters and the relaxed model of sequences for gene order.

Acknowledgments

The authors wish to thank Cedric Chauve for valuable discussions. The work of Roland Wittler was supported by the DFG Graduiertenkolleg Bioinformatik (GK 635). The work of Ján Maňuch was supported by NSERC DG. The work of Murray Patterson was supported by NSERC PGS-D3.

References

Z. Adam, M. Turmel, C. Lemieux, and D. Sankoff 2007. Common intervals and symmetric difference in a model-free phylogenomics, with an application to streptophyte evolution. *J. Comp. Biol.*, 14(4):436–445.

A. Bergeron and J. Stoye 2006. On the similarity of sets of permutations and its applications to genome comparison. *J. Comp. Biol.*, 13(7):1340–1354.

A. Bergeron, M. Blanchette, A. Chateau, and C. Chauve 2004. Reconstructing ancestral gene order using conserved intervals, 14–25. In *Proceedings of WABI 2004*, volume 3240 of *LNBI*, Springer Verlag.

A. Bergeron, C. Chauve, and Y. Gingras 2008. Formal models of gene clusters. In A. Z. Ion Măndoiu, editor, *Bioinformatics Algorithms: Techniques and Applications*, Wiley Book Series on Bioinformatics, Wiley.

A. Bhutkar, W. M. Gelbart, and T. F. Smith 2007. Inferring genome-scale rearrangement phylogeny and ancestral gene order: A drosophila case study. *Genome Biol.*, 8(11):R236.

G. Blin, D. Faye, and J. Stoye 2010. Finding nested common intervals efficiently. *J. Comp. Biol.*, 17(9):1183–1194.

S. Böcker, K. Jahn, J. Mixtacki, and J. Stoye 2009. Computation of median gene clusters. *J. Comp. Biol.*, 16(8):1085–1099.

K. S. Booth and G. S. Lueker 1976. Testing for the consecutive ones property, interval graphs and graph planarity using PQ-tree algorithms. *J. Comput. Syst. Sci.*, 13(3):335–379.

C. Chauve and E. Tannier 2008. A methodological framework for the reconstruction of contiguous regions of ancestral genomes and its application to mammalian genomes. *PLoS Comput. Biol.*, 4(11):e1000234.

C. Chauve, J. Maňuch, and M. Patterson 2009. On the gapped consecutive-ones property, 121–125. In *Proceedings of EUROCOMB*, volume 34 of *ENDM*.

M. Csűrös and I. Miklós 2009. Mathematical framework for phylogenetic birth-and-death models. *Arxiv preprint arXiv:0902.0970*.

D. Durand and D. Sankoff 2002. Tests for gene clustering, 144–154. In *Proceedings of RECOMB 2002*, ACM Press.

R. Friedman and A. L. Hughes 2001. Gene duplication and the structure of eukaryotic genomes. *Genome Res.*, 11(3):373–381.

P. Goldberg, M. Golumbic, H. Kaplan, and R. Shamir 1995. Four strikes against physical mapping of DNA. *J. Comp. Biol.*, 2(1):139–152.

M. Habib, R. McConnell, C. Paul, and L. Viennot 2000. Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing. *Theor. Comput. Sci.*, 234(1–2):59–84.

S. Haddadi 2002. A note on the NP–hardness of the consecutive block minimization problem. *International Transactions in Operational Research*, 9(6):775–777.

X. He and M. H. Goldwasser 2005. Identifying conserved gene clusters in the presence of homology families. *J. Comp. Biol.*, 12(6):638–656.

R. Hoberman and D. Durand 2005. The incompatible desiderata of gene cluster properties, 73–87. In *Proceedings of RECOMB Comparative Genomics 2005*, volume 3678 of *LNBI*, Springer Verlag.

W.-L. Hsu 2002. A simple test for the consequtive ones property. *J. Algorithms*, 43(1):1–16.

W.-L. Hsu and R. McConnell 2004. PQ-trees, PC-trees, and planar graphs. In D. P. Mehta and S. Sahni, editors, *Handbook of Data Structures and Applications*.

M. A. Huynen, B. Snel, and P. Bork 2001. Inversions and the dynamics of eukaryotic gene order. *Trends Genet.*, 17(6):304–306.

O. Jaillon, J.-M. Aury, F. Brunet, J.-L. Petit, N. Stange-Thomann, E. Mauceli, L. Bouneau, C. Fischer, C. Ozouf-Costaz, A. Bernot, S. Nicaud, D. Jaffe, S. Fisher, G. Lutfalla, C. Dossat, B. Segurens, C. Dasilva, M. Salanoubat, M. Levy, N. Boudet, S. Castellano, V. Anthouard, C. Jubin, V. Castelli, M. Katinka, B. Vacherie, C. Biemont, Z. Skalli, L. Cattolico, J. Poulain, V. de Berardinis, C. Cruaud, S. Duprat, P. Brottier, J.-P. Coutanceau, J. Gouzy, G. Parra, G. Lardier, C. Chapple, K. J. McKernan, P. McEwan, S. Bosak, M. Kellis, J.-N. Volff, R. Guigo, M. C. Zody, J. Mesirov, K. Lindblad-Toh, B. Birren, C. Nusbaum, D. Kahn, M. Robinson-Rechavi, V. Laudet, V. Schachter, F. Quetier, W. Saurin, C. Scarpelli, P. Wincker, E. S. Lander, J. Weissenbach, and H. Roest Crollius 2004. Genome duplication in the teleost fish tetraodon nigroviridis reveals the early vertebrate proto-karyotype. *Nature*, 431(7011): 946–957.

J. G. Lawrence and J. R. Roth 1996. Selfish operons: Horizontal transfer may drive the evolution of gene clusters. *Genetics*, 143(4):1843–1860.

J. Ma, L. Zhang, B. B. Suh, B. J. Raney, R. C. Burhans, J. W. Kent, M. Blanchette, D. Haussler, and W. Z. Miller 2006. Reconstructing contiguous regions of an ancestral genome. *Genome Res.*, 16(12):1557–1565.

J. Maňuch and M. Patterson 2010. The complexity of the gapped consecutive-ones property problem for matrices of bounded maximum degree, 278–289. In *Proceedings of RECOMB Comparative Genomics 2010*, volume 6398 of *LNBI*, Springer Verlag.

R. Overbeek, M. Fonstein, M. D'Souza, G. D. Pusch, and N. Maltsev 1999. The use of gene clusters to infer functional coupling. *Proc. Natl. Acad. Sci. USA*, 96 (6):2896–2901.

R. D. M. Page and M. A. Charleston 1997. From gene to organismal phylogeny: Reconciled trees and the gene tree/species tree problem. *Molecular Phylogenetics and Evolution*, 7(2):231 – 240.

C. H. Papadimitriou 1994. *Computational complexity*. Addisson-Wesley.

S. Pasek, A. Bergeron, J. L. Risler, A. Louis, E. Ollivier, and M. Raffinot 2005. Identification of genomic features using microsyntenies of domains: domain teams. *Genome Res.*, 15(6):867–874.

S. Rahmann and G. W. Klau 2006. Integer linear programs for discovering approximate gene clusters, 298–306. In *Proceedings of WABI 2006*, volume 4175 of *LNBI*, Springer Verlag.

J. Stoye and R. Wittler 2009. A unified approach for reconstructing ancient gene clusters. *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 6(3):387–400.

T. Uno and M. Yagiura 2000. Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26(2):290–309.

R. Wittler 2010. *Phylogeny-based Analysis of Gene Clusters. Ph.D. Thesis*, Faculty of Technology, Bielefeld University. Online available from `http://bieson.ub.uni-bielefeld.de/volltexte/2010/1627/`.

R. Wittler and J. Stoye 2010. Consistency of sequence-based gene clusters, 252–263. In *Proceedings of RECOMB Comparative Genomics 2010*, volume 6398 of *LNBI*, Springer Verlag.

Figures

List of Figures

1 Artifact arising for reconstruction on sequences 26
2 Illustration of Lemma 1 27
3 Illustration of improper and proper Eulerian cycles 28
4 Clause gadgeds for basic common intervals 29
5 Variable gadget for basic common intervals 30
6 Clause gadgets for nested common intervals 31
7 Variable gadget for nested common intervals 32

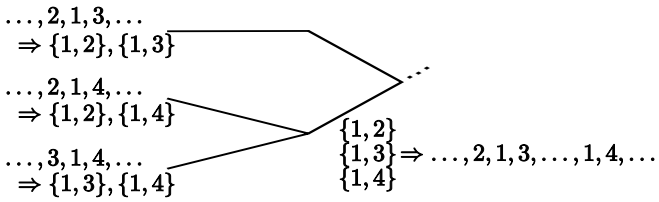


Figure 1: Example for an artifact that arises when gene clusters are reconstructed on the basis of gene orders, where any gene can occur arbitrarily often. A small subtree over three genomes is shown exemplarily. The gene orders on the leaves imply, beside others, the listed adjacencies. Any most parsimonious labeling would assign all three adjacencies to the lowermost internal node, implying at least two copies of gene 1. Since genes are allowed to appear multiple times in a genome, valid gene orders exist, all of which contain gene 1 at least two times — for instance the given gene order.

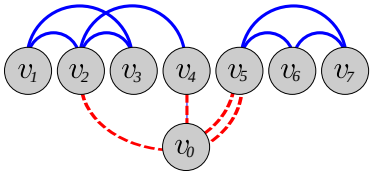
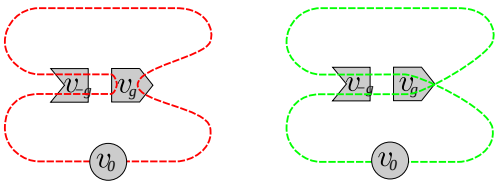


Figure 2: An example to illustrate the proof of Lemma 1. Consider the set of genes $\mathcal{G}_7$ with the multiplicities $m(g) = 2$ for $g \in \{2, 5\}$ and otherwise $m(g) = 1$, and the set $C = \big\{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 4\}, \{5, 6\}, \{5, 7\}, \{6, 7\}\big\}$ of unsigned adjacencies. The gene order graph $G_7(C)$ is depicted including the extensions described in the proof. The solid edges correspond to the original edges as defined by the given adjacencies, and the dashed lines represent the auxiliary edges. The obtained extended graph contains, for instance, the Eulerian cycle $(v_0, v_2, v_1, v_3, v_2, v_4, v_0, v_5, v_6, v_7, v_5, v_0)$, which corresponds to the valid gene order $(2, 1, 3, 2, 4, 5, 6, 7, 5)$.



(a) (b)

Figure 3: Illustration of the relation of improper and proper Eulerian cycles in an extended signed gene order graph as described in the proof of Lemma 2.

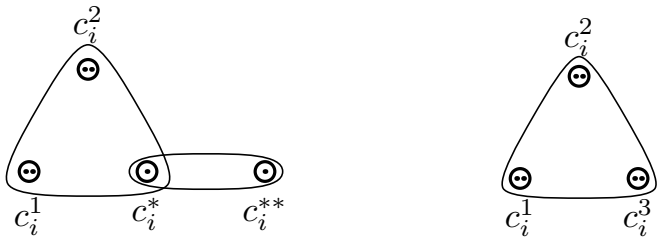


Figure 4: Graphical representations of the (a) 2-clause gadget and (b) 3-clause gadget for clause c_i. The multiplicity of the objects is indicated by the number of dots. Common intervals are depicted by ellipses surrounding two or triangles surrounding three objects, respectively.

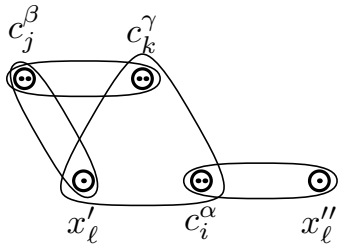


Figure 5: Graphical representation of the variable gadget for variable x_ℓ with positive occurrences c_i^α and c_j^β and negated occurrence c_k^γ in the clauses.

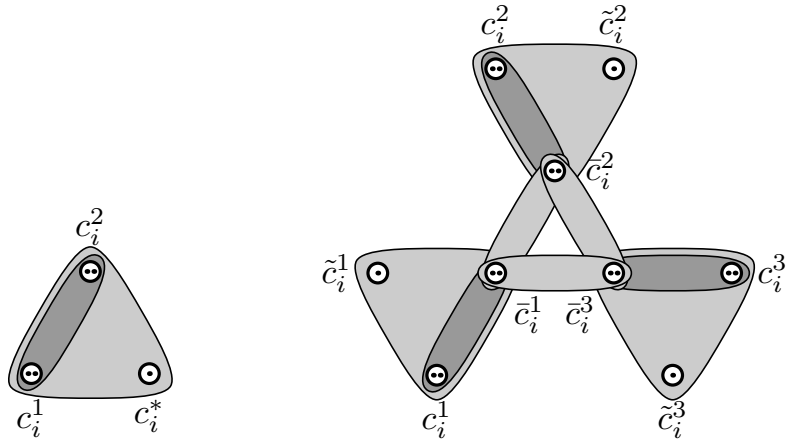


Figure 6: Graphical representations of the (a) 2-clause gadget and (b) 3-clause gadget for clause c_i in the nested common intervals case. The dark shaded ovals depict the nested part of nested intervals of size 3.

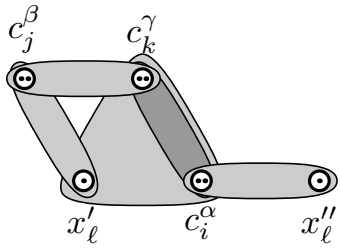


Figure 7: Graphical representation of the variable gadget for variable x_ℓ with positive occurrences c_i^α and c_j^β and negated occurrence c_k^γ in the clauses in the nested common intervals case.