Max-Tolerance Graphs as Intersection Graphs: Cliques, Cycles, and Recognition [*]

Michael Kaufmann[†] Jan Kratochvíl[‡] Katharina A. Lehmann [*]
Amarendran R. Subramanian [*]

Abstract

Max-tolerance graphs can be regarded as generalized interval graphs, where two intervals I_i and I_j only induce an edge in the corresponding graph iff they overlap for an amount of at least $max\{t_i, t_j\}$ where t_i is an individual tolerance parameter associated to each interval I_i. A new geometric characterization of max-tolerance graphs as intersection graphs of isosceles right triangles, shortly called semi-squares, leverages the solution of various graph-theoretic problems in connection with max-tolerance graphs. First, we solve the maximal and maximum cliques problem. It arises naturally in DNA sequence analysis where the maximal cliques might be interpreted as functional domains carrying biologically meaningful information. We prove an upper bound of $O(n^3)$ for the number of maximal cliques in max-tolerance graphs and give an efficient $O(n^3)$ algorithm for their computation. In the same vein, the semi-square representation yields a simple proof for the fact that this bound is asymptotically tight, i.e., a class of max-tolerance graphs is presented where the instances have $\Omega(n^3)$ maximal cliques. Additionally, we answer an open question posed in [8] by showing that max-tolerance graphs do not contain complements of cycles C_n for $n > 9$. By exploiting the new representation more deeply, we can go even further and prove that the recognition problem for max-tolerance graphs is NP-hard.

[*]This work has been supported by the DFG Grant Ka812-11/1 'Evolutionstheorien für natürliche und technische Netzwerke' and by Czech Grant MSM0021620838

[†]Universität Tübingen, WSI für Informatik, Sand 13, 72076 Tübingen, Germany `mk/lehmannk/subraman@informatik.uni-tuebingen.de`

[‡]Department of Applied Mathematics, Charles University, Prague, Czech Republic, `honza@kam.mff.cuni.cz`

1 Introduction

Interval graphs have received substantial consideration in discrete mathematics and computer science [5, 12, 4]. Many combinatorial problems which are, in general, NP-hard have been shown to be efficiently solvable in this case. Prominent examples are the maximum clique problem [5] and the recognition problem [2]. Because of its theoretical as well as practical importance, many variants of interval graphs have been explored [7, 12].

The following variant arises naturally in a bioinformatics application: For comparison of DNA sequences from different organisms or individuals, the initial step in such investigations is often to query a software tool like BLAST [1] by giving a special genomic sequence Q as a parameter. BLAST will return all sequences from its comprehensive database that share a very strong similarity with at least parts of the input query sequence Q. BLAST will *align* the found sequences to Q, i.e., every letter in the query sequence corresponds to one letter (or an inserted gap) of each of the other sequences. In the answer file from BLAST, the query sequence will be displayed in its full length, enumerated from 1 to n, where n denotes the number of letters in Q. For all other sequences only that specific part that shares the similarity with Q is contained in the result (Fig. 1). Let R be the set of all those resulting sequence parts. Since gene sequences often consist of discernible parts, called domains, one subset of R may be aligned to one part of Q and another subset to another part of Q. Roughly speaking, this imposes some kind of natural clustering of the resulting sequences in R. These clusters are assumed to correspond to domains that are often carriers of a certain function in the protein encoded by the sequence. However, to achieve the correspondence between the clusters and the biological structure, a careful formal definition of these clusters has to be established. We interpret the query sequence as a long interval and all the answer sequences as sub-intervals of it. Now it is required that every pair of sequences of a cluster share with each other a certain part of the main interval defined by the query sequence Q. To define a cluster, we demand that for every two elements s_1 and s_2 at least a

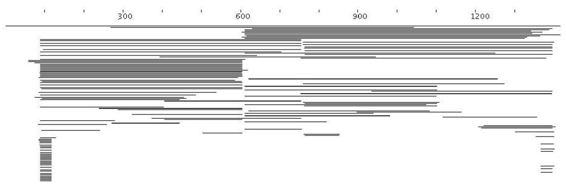


Figure 1: Visualization of a BLAST output.

part of ratio $0 < c < 1$ of s_1 is overlapped by s_2 and vice versa. Note that in two cases ($c = 0$ and $c = 1$) trivial graphs result and these cases are thus omitted here. Additionally we demand that each cluster be maximal with respect to that property, i.e., no other sequence can be added without violating this relation.

This viewpoint of the problem immediately leads to the concept of min- and max-tolerance graphs [8]. In min-tolerance graphs each interval I_i is associated with a tolerance t_i, such that an overlap between two intervals I_i and I_j induces an edge between the corresponding vertices v_i and v_j iff $|I_i \cap I_j| \geq min(t_i, t_j)$. Replacing the operator `min` by `max` provides the definition for max-tolerance graphs. In the application sketched above, t_i is then given by $c \cdot |I_i|$ and the question of maximal clusters of DNA sequences now becomes the problem of finding all maximal cliques in the corresponding max-tolerance graph.

Interval graphs and many variants have been shown to be perfect graphs, for which there are efficient algorithms for many classical NP-complete problems. In the case of max-tolerance graphs, easy examples can be given that contain a C_5, which means that max-tolerance graphs are not perfect in general, so the algorithmic machinery for perfect graphs is not applicable in their case [7]. Despite the apparent practical meaning of max-tolerance graphs, little is known of them from the literature, e.g, that all chordless cycles, trees and cactuses are max-tolerance graphs [8, 10], whereas the 'min'-variant of the tolerance graphs has received more attention.

For min-tolerance graphs for which $t_i \leq |I_i|$ holds for $1 \leq i \leq n$ holds (bounded min-tolerance graphs), a geometric representation of the problem is known where 2-layer parallelograms intersect if and only if the corresponding intervals tolerate each other [3]. In this article, we give a new set of results for max-tolerance graphs also based on a geometric representation. Our main tool is a new characterization of max-tolerance graphs as intersection graphs of semi-squares, which are squares cut in half along the diagonal.

Our article is organized as follows: In Section 2 the necessary definitions are given. Then we prove, in Section 3, the equivalence of max-tolerance graphs and intersection graphs of a corresponding set of similar semi-squares. With the new characterization, we can prove, in Section 4, the maximum/maximal clique problem to be efficiently solvable for max-tolerance graphs by achieving an upper bound. Additionally, we present in Section 5 an efficient algorithm which allows the computation of all maximal cliques in time $O(n^3)$, when the output consists of only the differences between two successive maximal cliques. In Section 6, a worst-case example shows that the given upper bound is asymptotically tight, i.e., there are graphs with $\Theta(n^3)$ maximal cliques. In Section 7, we show that the complement of a cycle C_n is not a max-tolerance graph for $n > 9$, answering an open question from the text book of Golumbic and Trenk, and in Section 8 we derive an NP-hardness proof for the recognition problem of max-tolerance graphs.

2 Definitions

Let S denote a set of n intervals $I_i = (x_i, y_i), 1 \leq i \leq n$, $x_i < y_i$, where x_i denotes the coordinate of the left endpoint and y_i denotes the coordinate of the right endpoint. The length $|I_i|$ of an interval is defined as $y_i - x_i$. Associated with each interval I_i is a tolerance parameter $t_i > 0$ which describes the tolerance of the interval. Note that for max-tolerance graphs we may assume without loss of generality that $t_i \leq |I_i|$, since otherwise the vertex representing I_i is isolated. Let $I_i = (x_i, y_i)$ and $I_j = (x_j, y_j)$ be two intervals. Their *overlap* $|I_i \cap I_j|$ is given by:

$$|I_i \cap I_j| = \min\{y_i, y_j\} - \max\{x_i, x_j\}$$

if the intervals have a common point. A *max-tolerance graph* $G = (V, E)$ is given by $V = \{v_1, ..., v_n\}$ representing the set of intervals where $E = \{\{i, j\} \mid |I_i \cap I_j| \geq \max\{t_i, t_j\},\ 1 \leq i, j \leq n\}$.

Given a graph $G = (V, E)$, a clique is a set of pairwise-connected vertices $C \subseteq V$. A clique is a *maximal clique* if there is no other clique C' of which C is a subset. A clique is a *maximum clique* if its cardinality is maximal. The complement graph $\bar{G}$ of G is defined by $\bar{G} = (V, \bar{E})$ where $\bar{E}$ contains exactly those edges that are not in E.

The *recognition problem of max-tolerance graphs* is defined as follows: For a given graph G, determine whether there is a set of intervals associated with tolerances such that G is the corresponding max-tolerance graph. The following is a folklore observation which will be used lateron more than once.

Lemma 2.1. *Any two disjoint convex polygons in the plane can be separated by a line containing a side of one of them.*

3 Semi-square representation of the problem

In this section we introduce a concept equivalent to the concept of max-tolerance graphs. Let $\mathcal{T}$ be a set of n similar, axis parallel, isosceles triangles $\Delta_i, 1 \leq i \leq n$ in the plane. Each triangle Δ_i is uniquely described by three parameters x_i, y_i, t_i leading to the following set of three points in the plane $\Delta_i(x_i, y_i, t_i) = (a_i, b_i, c_i)$:

$$a_i = (x_i, t_i) \tag{3.1}$$
$$b_i = (x_i, y_i - x_i) \tag{3.2}$$
$$c_i = (y_i - t_i, t_i) \tag{3.3}$$

We call such triangles *semi-squares* for short, since the triangles can be viewed as the lower left halves of squares cut along the diagonal.

We will now show the equivalence of any max-tolerance graph G_t to a corresponding intersection graph G_I of semi-squares: For each interval $I_i \in S$ construct a semi-square $\Delta_i(x_i, y_i, t_i)$ according to equation 3.1-3. The horizontal side $\overline{a_i c_i}$ will be called the *basis*, the vertical side $\overline{a_i b_i}$ is the *cathetus*, and the side $\overline{b_i c_i}$ is the *diagonal hypotenuse* and its slope is -1. The equation of the diagonal hypotenuse of Δ_i is $h_i(x) = y_i - x$ for $x \in [x_i, y_i - t_i]$.

Definition 3.1. *For each max-tolerance graph G_t we call $G_I(G_t)$ the corresponding intersection graph of semi-squares. If there is no ambiguity we omit the parameter and just write G_I.*

The following lemma states the crucial relation between the intersection of two semi-squares and the overlap of the corresponding two intervals:

Lemma 3.1. *Let Δ_i and Δ_j be the semi-squares constructed from intervals I_i and I_j, respectively. Then, Δ_i and Δ_j intersect if and only if the two corresponding intervals have an overlap $|I_i \cap I_j| \geq \max\{t_i, t_j\}$.*

Proof. W.l.o.g. we assume that $x_i \geq x_j$.

1. $t_i \geq t_j$:

$$\begin{aligned} & \Delta_i \cap \Delta_j \neq \emptyset \\ \Leftrightarrow\ & a_i \in \Delta_j \\ \Leftrightarrow\ & t_i \leq y_j - x_i \\ \Leftrightarrow\ & max\{t_i, t_j\} = t_i \leq min\{y_i, y_j\} - x_i = |I_i \cap I_j| \end{aligned}$$

2. $t_i < t_j$:

$$\begin{aligned} & \Delta_i \cap \Delta_j \neq \emptyset \\ \Leftrightarrow\ & (x_i, t_j) \in \Delta_i \cap \Delta_j \\ \Leftrightarrow\ & y_j - t_j \geq x_i \wedge x_i \geq t_j \\ \Leftrightarrow\ & |I_i \cap I_j| = min\{y_i, y_j\} - x_i \geq t_j = max\{t_i, t_j\} \end{aligned}$$

Furthermore, it is clear that every finite set of semi-squares $S = ((x_a, y_a), (x_b, y_b), (x_c, y_c))$, $x_a, y_a, x_b, y_b, x_c, y_c \in \mathbb{R}^+$ can be transformed into a set of intervals I_i with $x_i = x_b$ and $y_i = x_a + y_b$ with tolerance $t_i = y_a$ according to equations 3.1-3. From Lemma 3.1 it follows that for this transformation the intersection graph of the semi-squares corresponds to the max-tolerance graph induced by the intervals. Hence we state the following equivalence:

Theorem 3.1. *Max-tolerance graphs are exactly intersection graphs of semi-squares.*

4 An $O(n^3)$ bound on the number of maximal cliques in max-tolerance graphs

We have just shown the correspondence between an interval I_k and a certain semi-square (a_k, b_k, c_k). We will use this correspondence to give a polynomial-sized bound on the number of maximal cliques. The idea is to characterize each maximal clique by a set of pairwise intersecting semi-squares. Note that unlike the case of rectangles, three semi-squares might pairwise intersect without having any common intersection point.

Proposition 4.1. *Let M be a maximal clique and let Δ_s (Δ_h, Δ_v) be one of its semi-squares with a lowest hypotenuse (highest basis, rightmost cathetus, respectively). Denote the relevant sides of these triangles by s (h, v, respectively). Let $V(s, h, v)$ be the set of all semi-squares Δ such that $\Delta \cap s \neq \emptyset$, $\Delta \cap h \neq \emptyset$, $\Delta \cap v \neq \emptyset$, and such that the hypotenuse of Δ is not strictly below s, the basis of Δ is not strictly above h and the cathetus of Δ is not strictly to the right of v. Then $V(s, h, v) = M$.*

Proof. $M \subseteq V(s, h, v)$: If $\Delta \in M$, the hypotenuse of Δ is by definition not strictly below s. But then $\Delta \cap \Delta_s \neq \emptyset$ implies that Δ intersects s. Similarly for h and v, and hence $\Delta \in V(s, h, v)$.

$V(s, h, v) \subseteq M$: We will show that $V(s, h, v)$ is a clique and then $M = V(s, h, v)$ will follow from the previously proven inclusion and the assumed maximality of M. Suppose for the contrary that $\Delta', \Delta'' \in V(s, h, v)$ and $\Delta' \cap \Delta'' = \emptyset$. By Lemma 2.1, Δ' and Δ'' are separated by a line containing a side of one of them, say the line passing through the hypotenuse of Δ'. Since this line is not lower than s, it also separates Δ'' from Δ_s and $\Delta'' \cap s = \emptyset$, contradicting the assumption $\Delta'' \in V(s, h, v)$.

Corollary 4.1. *Each maximal clique is uniquely determined by one semi-square with the lowest hypotenuse, one with the highest basis and one with the rightmost cathetus. The number of maximal cliques does not exceed n^3 in any max-tolerance graph G.*

Note that the construction of the sets $V(s, h, v)$ for the different parameters s, h, and v just gives an upper bound on the number of maximal cliques and the set of cliques described in this way is just a set of candidates. In the next section, we describe how to efficiently compute the set of candidates and how to determine whether a candidate actually is maximal.

5 Algorithmic aspects

5.1 How to determine the set of candidates

A simple way to determine the set of candidates $V(s, h, v)$ for the different parameters s, h and v is to choose a triple $(\Delta_s, \Delta_h, \Delta_v)$ of extreme semi-squares in the sense of proposition 4.1. Then we check all semi-squares to see whether they fulfill the conditions of the set $V(s, h, v)$. This can be done in $O(1)$ for each semi-square giving this simple approach a total complexity of $O(n^4)$. Note that for $\Theta(n^3)$ maximal cliques of size $\Theta(n)$ we have an output of size $\Theta(n^4)$.

Next, we present an improved algorithm in $O(n^3)$ where the output consists of either a complete clique or the differences from the previous one.

At first, we introduce some additional notation:

For any candidate clique, let S denote the semi-square with the lowest hypotenuse s. All other semi-square belonging to this set can be partitioned into three subsets $P(s), Q(s)$ and $R(s)$ (Fig. 2), where the semi-squares $p \in P(s)$ contain the upper corner b_S of S, the semi-squares $q \in Q(s)$ contain the right corner c_S but not b_S of s and semi-squares $r \in R(s)$ do not contain any corner of s but intersect the hypotenuse of S. Note that because of the lowest hypotenuse of S in the cliques of the candidate set, S contains the left lower corners of the semi-squares from $R(s)$. Extending this notations, for $B \subseteq \{P(s), Q(s), R(s)\}$ and $\alpha \subseteq \{h, v\}$, define $B(\alpha) = \{\Delta \in \bigcup B : \forall_{\beta \in \alpha} \Delta \cap \beta \neq \emptyset\}$. For the sake of brevity, we simplify the notation by omitting several parentheses in a selfexplanatory way, e.g., $PR(s, h, v) = \{P(s), R(s)\}(\{h, v\})$.

Now, we are ready for the description of the algorithm. We start by sorting the set of all semi-squares: once according to the x-coordinate of the left endpoint of their basis and once according to the x-coordinate of the right endpoint of their basis. This is followed by choosing a semi-square S to be the one with the lowest hypotenuse s. Then we extract $P(s)$, $Q(s)$, and $R(s)$ from the sorted lists of semi-squares, where, for the generation of $P(s)$ we choose the list of semi-squares sorted by the x-coordinate of their right endpoints; for $Q(s)$ we choose the list sorted by x-coordinate of left endpoints; and for $R(s)$ we create two lists, $Rl(s)$ and $Rr(s)$, generated from the lists sorted by left and right endpoint, respectively. Then we choose a semi-square from $R(s)$ or $P(s)$ which determines the highest basis. Let h again denote this basis. From $P(s), Rl(s)$, $Rr(s)$, and $Q(s)$, we keep only those elements that intersect h. This defines the sets $P(s, h), Q(s, h)$, as well as $Rl(s, h)$ and $Rr(s, h)$.

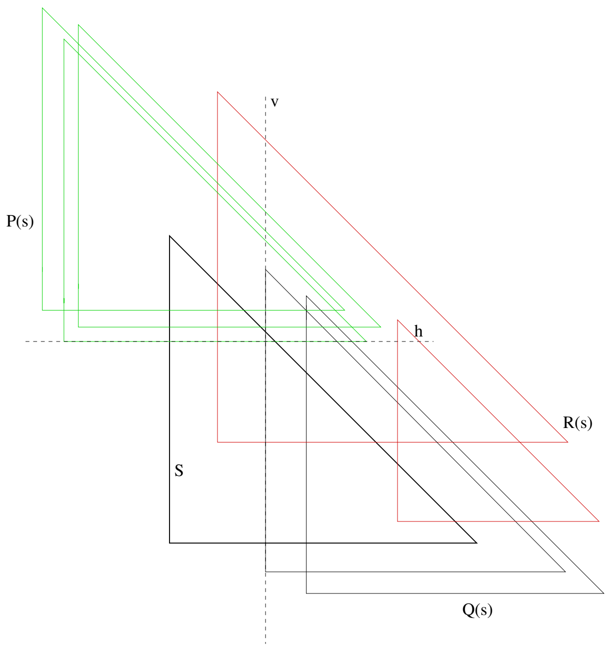


Figure 2: An illustration of $P(s), R(s), Q(s)$ for some semi-square S with hypotenuse s and some arbitrary chosen parameters h and v. Note that $|P(s)| = 3$ whereas $|P(s, h)| = 1$. Analogously, the sizes of $Q(s, h)$ and $Q(s)$ respectively $R(s, h)$ and $R(s)$ differ in this example.

Let $x_1 < x_2 < ... < x_k$ be the different x-coordinates of the left corners of the elements $q_1, q_2, ..., q_k \in Q(s, h) \cup R(s, h) = QR(s, h)$ in this order. Let v_i denote the vertical side of the semi-square q_i. We start with the construction of the clique $V(s, h, v_1)$ in linear time by checking the elements in the list of $P(T, h)$ to determine whether their right corner is right of the vertical v_1 defined by the x-coordinate x_1 of q_1. Note that since these semi-squares are in $P(s)$, their lower left corner is never to the right of the lower left corner of s, and thus also never to the right of x_1. Thus, they intersect a vertical line on position x_1. This gives the set $P(s, h, v_1)$ in time $O(|P(s, h, v_1)|)$. Similarly, for $Q(s, h)$ we check the list to determine whether the elements have their left corner to the left of x_1. Since these semi-squares have their right corner to the right (or on) the right corner of S, an intersection with a vertical line on position x_1 is thus guaranteed. This is done in time $O(|Q(s, h, v_1)|)$. For $R(s, h)$, things are slightly more complicated. Processing the list $Rl(s, h)$ gives the elements with left corners to the left of x_1, and we have to remove the elements where the right corners are also to the left of x_1. The latter elements are determined by processing the list $Rr(s,h)$ up to the point x_1. In a formula, we can state

$$V(s,h,v_1) = P(s,h,v_1) \cup Q(s,h,v_1) \cup R(s,h,v_1)$$

Next assume that we have already computed $V(s,h,v_i)$ for $i \geq 1$. Processing the entries of the four lists between v_i and v_{i+1}, we can determine

$$P(s,h,v_{i+1}) = P(s,h,v_i) \backslash P'(s,h,v_{i+1},v_i),$$

where the last term denotes the set of semi-squares from $P(s,h,v_i)$ with right corners between the vertical sides v_i and v_{i+1} of the semi-squares q_i and q_{i+1}. Similarly,

$$Q(s,h,v_{i+1}) = Q(s,h,v_i) \cup Q'(s,h,v_{i+1},v_i),$$

where the last term denotes the set of semi-squares from $Q(s,h)$ with left corners between v_i and v_{i+1}. $R(s,h,v_{i+1})$ is determined as follows:

$$R(s,h,v_{i+1}) = \\ (R(s,h,v_i) \cup Rl'(s,h,v_{i+1},v_i)) \backslash Rr'(s,h,v_{i+1},v_i)$$

where the last two terms denote the set of semi-squares from the lists $Rl(s,h)$ and $Rr(s,h)$ with respective left and right corners between v_i and v_{i+1}. Now

$$V(s,h,v_{i+1}) = \\ P(s,h,v_{i+1}) \cup Q(s,h,v_{i+1}) \cup R(s,h,v_{i+1})$$

The algorithm is summarized in Algorithm 1.

Global sorting can be accomplished in $O(n\log(n))$ since each extraction for a sublist costs $O(n)$ because we can determine in $O(1)$ if an element belongs to such a sublist by using simple linear algebra in the 2-dimensional plane. The initialising set $V(s,h,v_1)$ for fixed s and h can be computed in linear time by scanning the various sorted lists. Finally, the iteration over the v_i's takes $O(n)$ time since we can use a pointer in each of the sorted lists that only moves right with each iteration. The time complexity involved is then proportional to the width of the stride. In total, the time complexity of our algorithm is $O(n^3)$.

Theorem 5.1. *The set of candidates $V(s,h,v)$ for all choices of s,h and v for maximal cliques can be determined in time $O(n^3)$ if we allow to output solely the difference from the previous candidate clique.*

5.2 How to determine the maximum and maximal cliques

Clearly, the candidates with the maximum cardinality of elements represent the maximum cliques.

Algorithm 1: Algorithm for computing the candidates for maximal cliques

```
Build a list $L_l$ containing all semi-squares sorted by the left $x$-coordinate of their basis.
Build a list $L_r$ containing all semi-squares sorted by the right $x$-coordinate of their basis.
Build a list $L_b$ containing all semi-squares sorted by the $y$-coordinate of their basis.
foreach choice of semi-squares $S$ with hypotenuse $s$ extract $P(s)$ from $L_b$ do
    foreach semi-square in $P(s)$ in descending order with respect to the sorting and $h$ its basis: do
        Extract $Q(s,h)$ and $Rl(s,h)$ from $L_l$.
        Extract $P(s,h)$ and $Rr(s,h)$ from $L_r$.
        For $q_1 \in Q(s,h) \cup Rl(s,h)$ with leftmost vertical side $v_1$ output $V(s,h,v_1)$ as defined above.
        For every $x$-coordinate of $v_i$ $i > 1$ output the difference of $V(s,h,v_i)$ from the previous candidate $V(s,h,v_{i-1})$ as follows:
        ADD $Q'(s,h,v_i,v_{i-1})$
        ADD $Rl'(s,h,v_i,v_{i-1})$
        REMOVE $P'(s,h,v_i,v_{i-1})$
        REMOVE $Rr'(s,h,v_i,v_{i-1})$
    end
end
```

Theorem 5.2. *The maximum cliques can be found in time $O(n^3)$.*

The decision whether a given candidate really represents a maximal clique is more difficult. A straightforward way is to compare all candidates pairwise, which would lead to an $O(n^7)$ algorithm. We can considerably improve this by observing that each candidate $V(s,h,v)$ which is not a maximal clique can be extended in at least one of the three parameters s,h, or v.

By the incremental construction of the candidates, $V(s,h,v_i)$ is extendable to $V(s,h,v_{i+1})$ with respect to parameter v if $P'(s,h,v_{i+1},v_i)$ is empty and $Rr'(s,h,v_{i+1},v_i) \subset Rl'(s,h,v_{i+1},v_i)$ is true. In this case, no semi-square is removed and $V(s,h,v_i)$ can be marked as extendable.

Performing analogous sweeps as before in the diagonal direction with fixed parameters h and v as well as in the vertical direction with fixed parameters s and v, we determine in $O(n^3)$ the cliques which are extendable with respect to parameters s and h. Finally, all candidates that are not marked as extendable are the maximal candidates.

Theorem 5.3. *All maximal cliques can be identified in time $O(n^3)$.*

Next we prove that our bound for the number of possible maximal cliques is not overestimated.

6 A lower bound example

In this section, we present a class of instances leading to $\Omega(n^3)$ maximal cliques. Figure 3 illustrates the idea and in the following we refer to it implicitly. Let m be a fixed parameter, e.g., $m = 4$ as in Figure 3.

Let $1 \leq s \leq m$ be fixed for the moment. We only consider maximal cliques that contain t_s as well as r_s. Now for each pair (h_a, v_b) with $s \leq b \leq a$ we obtain a maximal clique which is made unique by adding k_b to the clique. More formally this maximal clique is comprised of

$$\{t_s, r_s, r_{s+1}, \ldots, r_m, h_a, h_{a+1}, \ldots, h_m, \\ v_a, v_{a-1}, \ldots, v_b, k_b, k_{b-1}, \ldots, k_s\}$$

For our fixed s this gives rise to

$$\left(\sum_{j=1}^{m-s+1} j \right) = \Omega\left((m-s+1)^2\right)$$

maximal cliques. Note that for two different choices for s, say $s_1 \neq s_2$, all these maximal cliques are different since they are made unique by the choice of the t_{s_1} and t_{s_2} respectively. Altogether this adds up to

$$\Omega\left(\sum_{s=1}^{m}(m-s+1)^2\right) = \Omega\left(\sum_{s=1}^{m} s^2\right) = \Omega(m^3)$$

and since we only need exactly $n = 5m$ semi-squares, we conclude that the lower bound is $\Omega(n^3)$ maximal cliques.

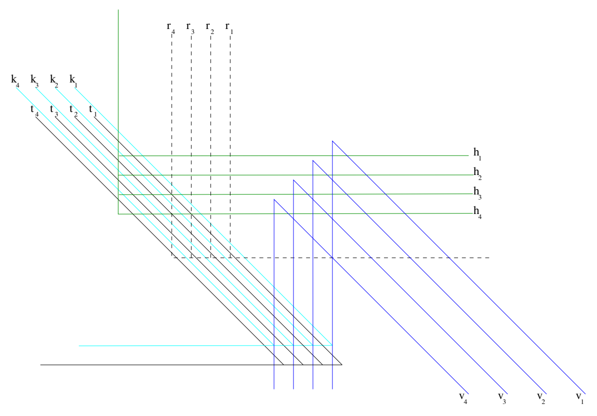


Figure 3: Example for $\Omega(n^3)$ maximal cliques in a max-tolerance graph.

7 Complements of cycles

Golumbic and Trenk [8] note that max-tolerance graphs are not perfect because every (odd) cycle is a max-tolerance graph. In this connection they ask if complements of cycles are also max-tolerance graphs (Question 12.33 in [8]). We show how handy the geometric interpretation is by giving a very simple solution to this problem.

We use Lemma 2.1, which implies that two disjoint semi-squares can be separated by a line containing a side of one of them.

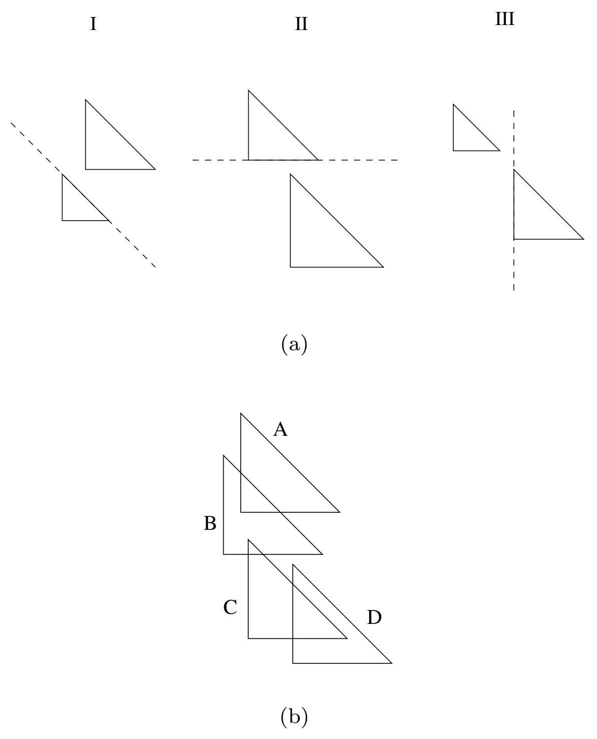


Figure 4: **a)** shows the three different types of positions of disjoint semi-squares. **b)** shows two pairs of disjoint semi-squares (A,C) and (B,D) in the same position.

Two nonadjacent semi-squares must lie in one of the three positions illustrated in Figure 4 (but of course it may happen that more than one case applies, as, e.g., in the right picture whose semi-squares lie also in position II).

Lemma 7.1. *Let A, B, C, D be semi-squares such that the pairs A, C and B, D are disjoint, while any other two pairs intersect. Then the pairs A, C and B, D cannot lie in the same position.*

Proof. Suppose, without loss of generality, that both pairs lie in position II, and e.g., the whole semi-square

C lies below the base line of A, and the whole semi-square D lies below B. Since B intersects C, the base line of B lies below the base line of A. Then D, lying below the base line of B, cannot intersect A (Figure 4).

PROPOSITION 7.1. *The complement of a cycle of length greater than 9 is not a max-tolerance graph.*

Proof. Suppose the complement of C_n is a max-tolerance graph, and consider a semi-square-intersection representation. Color every edge of C_n by one of the colors I, II, or III, depending on the relative position of the (disjoint) semi-squares representing its end-vertices. By the preceding lemma, no two edges of the same color form an induced matching in C_n, and hence each color is used on at most three edges. Thus $n \leq 9$.

8 Recognition

In this section we present another result deduced from the geometric representation, namely the NP-hardness of the recognition problem:

THEOREM 8.1. *Recognition of max-tolerance graphs is NP-hard.*

Proof. We sketch the NP-hardness proof for recognition of intersection graphs of congruent semi-squares.

This proof closely follows along the lines of [9, 11]. In [9, 11] a more general concept called the *noncrossing arc-connected set* is introduced. Two arc-connected sets are called *noncrossing* if their boundaries share at most two connected sections, or in other words, if the boundary of their union can be partitioned into two connected parts, each belonging to the boundary of one of the sets. Any two semi-squares are noncrossing. Thus we can utilize the reduction from [9, 11] which reduces NAE-SAT to the recognition problem. (A CNF-formula is called NAE-satisfiable if it allows a truth assignment of the variables such that every clause contains at least one true and at least one false literal.) For a CNF-formula Φ with exactly 3 variables per clause and at most 4 occurrences per variable (it may be assumed without negations), a graph G is constructed such that if Φ is not NAE-satisfiable, G cannot be represented as an intersection graph of noncrossing arc-conected sets, whereas if Φ is satisfiable, G is an intersection graph of disks. We show that the same construction yields a graph that is an intersection graph of congruent semi-squares (if the formula is NAE-satisfiable). This will be proven by showing how the building blocks of the reduction can be represented. We will also briefly describe the idea of the reduction, but the reader is referred to [9, 11] for details.

Given Φ, first draw the incidence graph of Φ as a rectilinear drawing (i.e., vertices in this graph are variables and clauses of Φ, and edges connect each clause to its variables). Note that the drawing cannot be considered noncrossing since NAE-SAT is polynomial for planar formulas. Add dummy vertices subdividing the edges and representing faces of the drawing to ensure a topologically unique drawing. Now variables will be replaced by variable gadgets which are cycles of length 8. These cycles can be represented either clockwise or counter-clockwise, thus encoding whether the variable is true or false. Edges are replaced by so-called *ladders* (pairs of paths of equal length with corresponding adjacent vertices). In any representation by noncrossing sets, the relative position of the sets representing the last two vertices is the same as that of the first two [11], and this fact is used to transfer the true/false information from the variable to the clause. A crucial role is played by so-called crossover components that allow two ladders to either cross or touch without crossing. These crossover components are placed on pairs of edges leaving each variable, enabling the ladders to lead in the direction of appropriate clauses regardless of the orientation of the representation of the variable cycle (the truth valuation of the variable). The crossover components are also placed in the edge crossings (where they end up being used only in the crossing representation). The most complicated is the clause gadget, formed by a cycle and extensions of the three ladders incoming from the variables (Fig. 8). These extensions must be represented inside the bounding clause cycle and are further connected by three paths (referred to as chains) in such a way that representation by arc-connected sets is impossible if all variables have the same value. A case analysis shows that the clause gadget has a representation by semi-squares when the variables do not have the same value. The figures appear in the Appendix.

Acknowledgements: We thank Kay Nieselt and Stefan Steigele for leading us to this interesting connection between DNA sequence analysis and max-tolerance graphs.

References

[1] S.F. Altschul, W. Gish, W. Miller, E.W. Myers and D.J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215, pp. 403-410, 1990. `http://www.ncbi.nlm.nih.gov/BLAST/`

[2] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and planarity using PQ-tree algorithms. *J. Comput. Sys. Sci.*, 13, pp. 335-379, 1976.

[3] K. Bogart, G. Isaak, J. Laison, and A. Trenk. Comparability invariance results for tolerance orders. *Order*, **18**, pp. 281-294, 2001.

[4] P.C. Fishburn, Interval Orders and Interval Graphs: A Study of Partially Ordered Sets, John Wiley & Sons, New York, 1985.

[5] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, **15**, pp. 835-855, 1965.

[6] M. Garey and D. Johnson. Computers and Intractability - A Guide to the Theory of NP-Completeness, Freeman, 1979

[7] M.C. Golumbic. Algorithmic graph theory and perfect graphs. *Annals of Discrete Mathematics*, **57**, 2004.

[8] M. C. Golumbic, A. Trenk. Tolerance Graphs. Cambridge Studies in Advanced Mathematics 89, Cambridge University Press, 2004, 265 pp.

[9] P. Hliněný, J. Kratochvíl. Representing graphs by disks and balls (a survey of recognition-complexity results) *Discrete Math*, 229 (2001), pp. 101-124.

[10] M.S. Jacobson, F.R. Morris, E.R. Scheinermann. General results on tolerance intersection graphs. *J. Graph Theory*, **15**, pp. 573-577, 1991.

[11] J. Kratochvíl. Intersection graphs of noncrossing arc-connected sets in the plane. In: *Graph Drawing (S. North ed.), Proceedings Graph Drawing '96*, Berkeley, September 1996, Lecture Notes in Computer Science 1190, Springer Verlag, Berlin Heidelberg, 1997, pp. 257-270.

[12] S. Skiena. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, Addison-Wesley, pp. 163-164, 1990.

9 Appendix

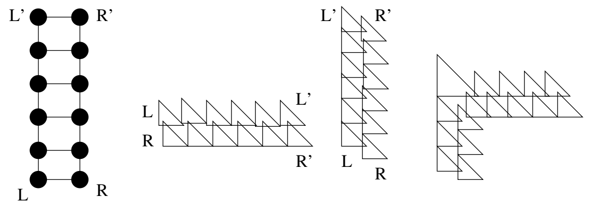


Figure 5: The ladder and vertical, horizontal, and bent representation.

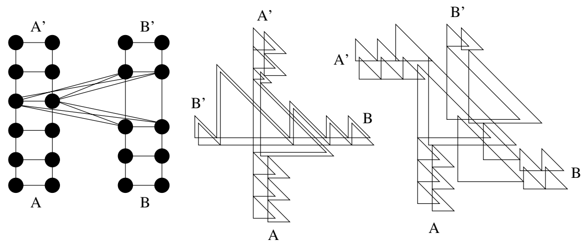


Figure 6: Crossover component of two ladders and crossing and touching representations.

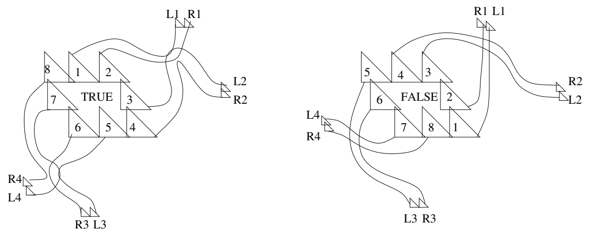


Figure 7: Variable gadget and schematic illustration of TRUE and FALSE representations.

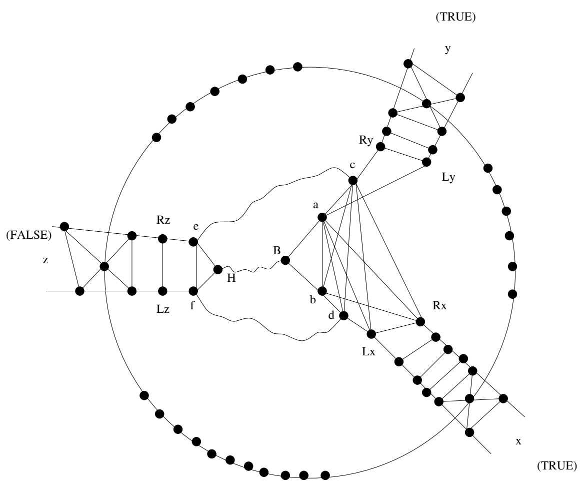


Figure 8: Schematic illustration of the clause gadget.

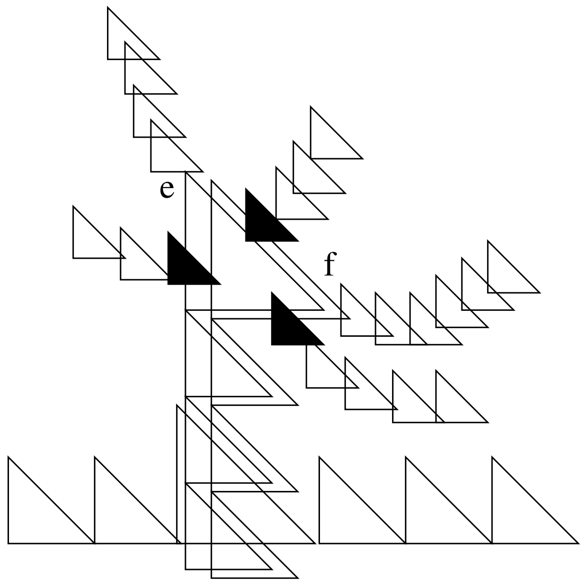


Figure 10: Representations of the efH part of the clause gadget show that H (the bold semi-square) can be placed in all three possible positions, so that the chain starting in H can reach B without crossing the other two chains.

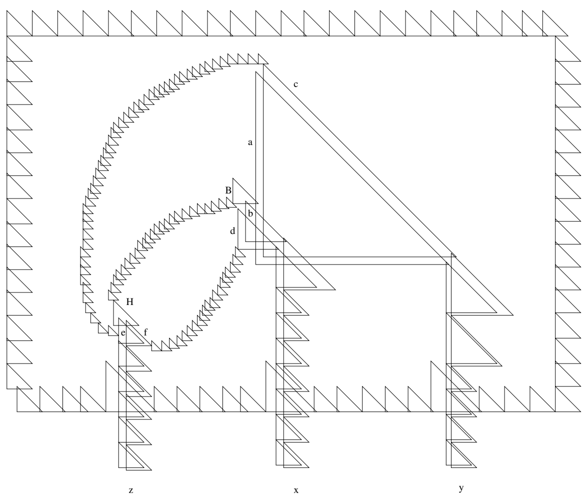
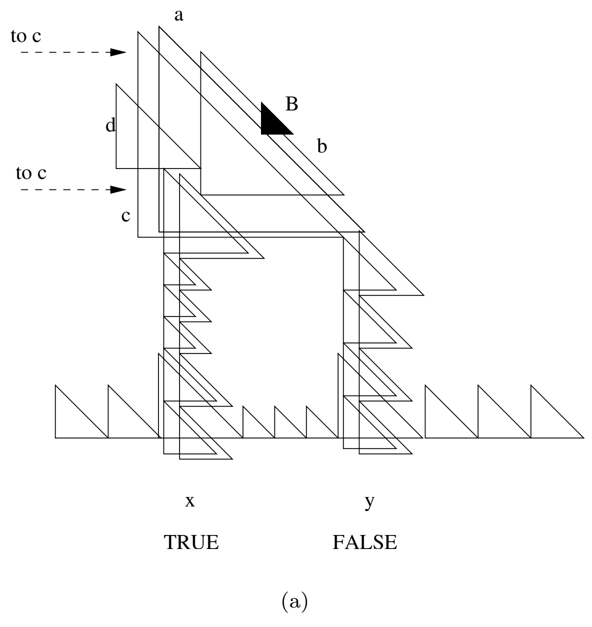


Figure 9: Illustration of the clause gadget representation for $x = y =$TRUE and $z =$FALSE.



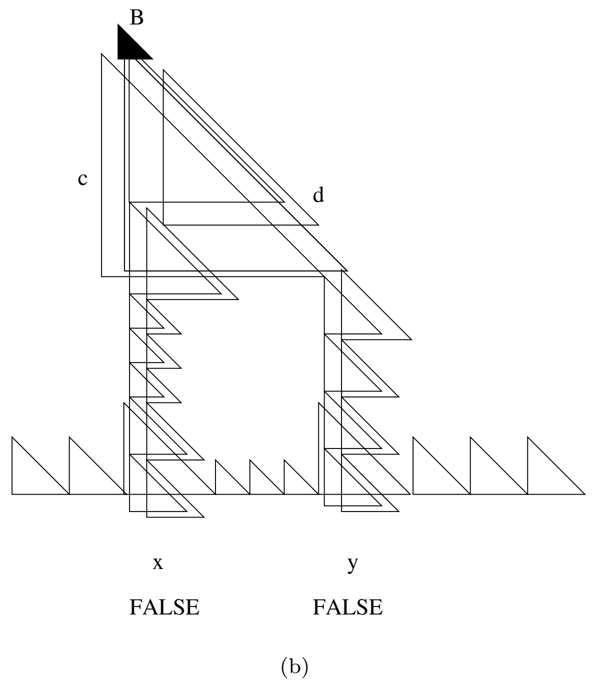


Figure 11: **a)** shows a representation of the *bdBac* part of the clause gadget in the case x =TRUE, y =FALSE showing that c can be reached by its chain either from left or right of d. **b)** shows the representation of the *bdBac* part of the clause gadget in the case x =FALSE, y =FALSE.

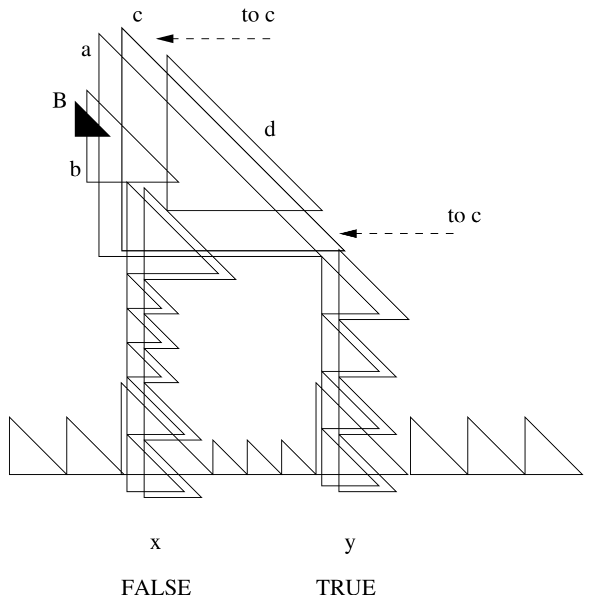


Figure 12: Representation of the *bdBac* part of the clause gadget in the case x =FALSE, y =TRUE showing that c can be reached by its chain either from left or right of d.