The Maximum Interval Graphs on Distance Hereditary Graphs

Ton Kloks Chuan-Min Lee* Sheng-Lung Peng†

Abstract

In this paper we propose a linear-time algorithm to solve the problem of finding an induced interval graph with a maximum number of vertices in a distance hereditary graph.

Keywords: Interval graphs, Distance hereditary graphs, Physical mapping

1 Introduction

A graph $G = (V, E)$ is an *interval graph* if there exists a one-to-one correspondence between V and a family F of intervals of the real line such that two vertices in V are adjacent if and only if their corresponding intervals in F overlap. F is called the *interval model* of G. The *maximum interval graph problem* (MIGP for short) on G is to find a maximum induced interval subgraph of G by deleting the minimum number of vertices of G. A more generalized problem is called the *node-deletion problem* [4, 5, 8]. The node-deletion problem for a graph property π on a graph G is to find a vertex set of minimum cardinality whose deletion (along with all the incident edges) from G leaves a subgraph satisfying the property π. Thus, MIGP is the node-deletion problem with π equaling to "interval graph."

MIGP has an application to physical mapping of a target DNA molecule [1, 2, 3]. Suppose that we are given a set of fragments, where each fragment corresponds to a substring of the target DNA. Then the aim of the physical mapping problem is to reconstruct the relative position of these fragments along the target DNA based on the information of their pairwise overlaps. From the graph viewpoint, we can built a graph $G = (V, E)$ according to the fragments and their pairwise overlaps as follows. Each fragment is represented by a vertex and two vertices are adjacent if and only if their corresponding fragments overlap. If the fragments are accurate and cover the whole target DNA, then the physical mapping problem becomes the one of recognizing whether G is an interval graph and constructing its interval model if so. In laboratories, however, some biological techniques may occasionally produce some new and bad fragments such that G is not interval any more. Therefore, we want to find and remove such bad fragments from G such that G becomes an interval graph.

In this paper, we solve MIGP in linear time for distance hereditary graphs.

2 Preliminaries

We denote the number of vertices of a graph $G = (V, E)$ by n and the number of edges by m. An independent set in a graph is a set of pairwise nonadjacent vertices. For a graph $G = (V, E)$ and $W \subseteq V$, $G[W]$ denotes the subgraph of G induced by the vertex set W. If any two distinct vertices in an induced subgraph G' of G are adjacent, then G' is called a *clique*. For a vertex x of $G = (V, E)$, $N(x) = \{y \in V | (x, y) \in E\}$ is the *neighborhood* of x and $N[x] = N(x) \cup \{x\}$ is the *closed neighborhood* of x. Let $G = (V, E)$ and $H = (W, F)$ be two graphs. We denote the *union* of G and H by $G \cup H = (V \cup W, E \cup F)$ and the *intersection* of G and H by $G \cap H = (V \cap W, E \cap F)$. We let $G - H = (V - W, E - F)$.

Three vertices in a graph G form an *asteroidal triple* (AT for short) if every pair of them is connected by a path which does not pass through the neighborhood of the third. A graph is *AT-free* if no three vertices of G form an AT. Notice that an AT is independent. In a cycle, a *chord* is an edge joining two non-consecutive vertices. A graph G is *chordal* if every cycle in G of length greater than three has a chord.

Theorem 1. [7] A graph G is an interval graph if and only if G is chordal and contains no AT.

Given a graph G, the *distance* between vertices u and v of G, denoted by $d_G(u, v)$, is the number of edges of a shortest path from u to v. A graph is *distance hereditary* if any two distinct vertices have the same distance in every connected induced subgraph containing them. Chang *et al.* showed that every distance hereditary (DH for short) graph has a graceful characteristic [6] that comes from the concept of *twin sets* (which are

*Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, R.O.C., cmlee@cs.ccu.edu.tw

†Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan, R.O.C., lung@csie.ndhu.edu.tw

vertex subsets). We use $TS(G)$ to denote a twin set of the DH graph G.

Definition 1. [6] DH graphs can be defined recursively as follows:

1. A graph consisting of only one vertex is DH, and its twin set is the vertex itself.
2. If G_1 and G_2 are disjoint DH graphs, then the union G of them is also DH, and $TS(G) = TS(G_1) \cup TS(G_2)$. G is said to be obtained from G_1 and G_2 by a *false twin* operation and denoted by $G = G_1 \odot G_2$.
3. If G_1 and G_2 are disjoint DH graphs, then the graph G obtained by connecting every vertex of $TS(G_1)$ to all vertices of $TS(G_2)$ is also DH, and $TS(G) = TS(G_1) \cup TS(G_2)$. G is said to be obtained from G_1 and G_2 by a *true twin* operation and denoted by $G = G_1 \otimes G_2$.
4. If G_1 and G_2 are disjoint DH graphs, then the graph G obtained by connecting every vertex of $TS(G_1)$ to all vertices of $TS(G_2)$ is also DH, and $TS(G) = TS(G_1)$. G is said to be formed from G_1 and G_2 by a *pendant vertex* operation and denoted by $G = G_1 \oplus G_2$.

By the definition above, for any DH graph, a binary ordered decomposition tree can be obtained in $O(n + m)$ time [6]. In this decomposition tree, each leaf node denotes a vertex and each internal node represents one of the three operations: pendant vertex operation ($\oplus$), true twin operation ($\otimes$), and false twin operation ($\odot$).

Throughout the paper, we assume that a decomposition tree of a DH graph is given. In the following, we shall show that the maximum induced interval graph problem on DH graphs can be solved in $O(n + m)$ time using the dynamic programming technique.

3 A linear-time algorithm

Let $G = (V, E)$ be a DH graph. Let $MI(G)$ denote a maximum induced interval graph of G. Let $MI_t(G)$ denote an induced interval graph of G of maximum vertices such that it contains a vertex in $TS(G)$ and let $MI_{t'}(G)$ denote an induced interval graph of G of maximum vertices such that it contains no vertex in $TS(G)$. Assume that S is a family of induced subgraphs of G. Let $\max S$ denote an induced subgraph of S of maximum vertices. Clearly, $MI(G) = \max\{MI_t(G), MI_{t'}(G)\}$.

For simplicity, we assume that a DH graph G is formed from two disjoint DH graphs G_1 and G_2 if G consists of more than one vertex.

Definition 2. Let I_1 and I_2 be induced interval graphs of DH graphs H_1 and H_2, respectively. Let $I_1 \bowtie I_2$ denote the graph obtained by connecting every vertex of $I_1 \cap H_1[TS(H_1)]$ to all vertices of $I_2 \cap H_2[TS(H_2)]$.

Definition 3. We use $\emptyset$ to denote an empty graph and use Φ to denote a graph with vertices $-\infty$. We let $H \cup \Phi = \Phi$ and $H \bowtie \Phi = \Phi$ for any graph H.

The following three lemmas can be easily verified.

Lemma 1. *If G consists of only one vertex, then $MI_t(G) = G$ and $MI_{t'}(G) = \emptyset$.*

Lemma 2. *Suppose that $G = G_1 \odot G_2$. Then*

(1) $MI_{t'}(G) = MI_{t'}(G_1) \cup MI_{t'}(G_2)$.

(2) $MI_t(G) = \max \left\{ \begin{array}{c} MI_t(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI_t(G_2), \\ MI_t(G_1) \cup MI_t(G_2) \end{array} \right\}$.

Lemma 3. *The following statements are true.*

(1) *Suppose that $G = G_1 \otimes G_2$. Then $MI_{t'}(G) = MI_{t'}(G_1) \cup MI_{t'}(G_2)$.*

(2) *Suppose that $G = G_1 \oplus G_2$. Then $MI_{t'}(G) = MI_{t'}(G_1) \cup MI(G_2)$.*

Lemmas 4–8 are concerned with the computation of $MI_t(G)$ for $G = G_1 \otimes G_2$ or $G = G_1 \oplus G_2$.

Lemma 4. *Suppose that $G = G_1 \otimes G_2$ or $G = G_1 \oplus G_2$. Let $I_1 = MI_t(G) \cap G_1$ and $I_2 = MI_t(G) \cap G_2$. If I_1 contains a vertex in $TS(G_1)$ and I_2 contains a vertex in $TS(G_2)$, then $I_1 \cap G_1[TS(G_1)]$ is a clique or $I_2 \cap G_2[TS(G_2)]$ is a clique.*

Definition 4. Let G be a DH graph and I be an induced interval subgraph of G. An AT-free set $\mathcal{A} = \{x, y, z\}$ in I is a *weak* AT-free set if the following conditions are satisfied.

(1) Vertex x is in $TS(G)$ and vertices y and z are not in $TS(G)$,

(2) there is an x, y-path avoiding $N_G(z)$,

(3) there is an x, z-path avoiding $N_G(y)$, and

(4) all y, z-paths pass through a vertex of $N_G(x)$.

If $\mathcal{A} = \{x, y, z\}$ is a weak AT-free set and there is a y, z-path avoiding all vertices in $N_G[x] \cap TS(G)$, then $\mathcal{A}$ is called a *t-weak* AT-free set.

Definition 5. Let G be a DH graph. Let I be an induced interval graph of G and $I_t(G, I) = I \cap G[TS(G)]$. We use $F_t(G, I)$ to the denote the union of all connected components of I in which every component has a vertex in $TS(G)$. We define the following terms.

1. $I_q(G) = \{I|\ I_t(G,I)$ is a clique$\}$ and $I_{q'}(G) = \{I|I_t(G,I) \neq \emptyset,\ I_t(G,I)$ is not a clique, and any two nonadjacent vertices of $I_t(G,I)$ have no common neighbor in $I - I_t(G,I)\}$.
2. $I_{q,0}(G) = \{I|I \in I_q(G)$ and $I_t(G,I) = F_t(G,I)\}$.
3. $I^0_{q,1}(G) = \{I|I \in I_q(G),\ I_t(G,I) \subset F_t(G,I)$, and I has no t-weak AT-free set$\}$.
4. $I^1_{q,1}(G) = \{I|I \in I_q(G),\ I_t(G,I) \subset F_t(,I)$, and I has no weak AT-free set$\}$.
5. $I_{q',0}(G) = \{I|I \in I_{q'}(G)$ and $I_t(G,I) = F_t(G,I)\}$.
6. $I^0_{q',1}(G) = \{I|I \in I_{q'}(G),\ I_t(G,I) \subset F_t(G,I)$, and I has no t-weak AT-free set$\}$.
7. $I^1_{q',1}(G) = \{I|I \in I_{q'}(G),\ I_t(G,I) \subset F_t(G,I)$, and I has no weak AT-free set$\}$.

Notice that $I_t(G,I) \subseteq F_t(G,I)$, $I^1_{q,1}(G) \subseteq I^0_{q,1}(G)$, and $I^1_{q',1}(G) \subseteq I^0_{q',1}(G)$. Let $MI_{q,0}(G)$, $MI^0_{q,1}(G)$, $MI^1_{q,1}(G)$, $MI_{q',0}(G)$, $MI^0_{q',1}(G)$, and $MI^1_{q',1}(G)$ denote the maximum induced interval graphs of $I_{q,0}(G)$, $I^0_{q,1}(G)$, $I^1_{q,1}(G)$, $I_{q',0}(G)$, $I^0_{q',1}(G)$, and $I^1_{q',1}(G)$, respectively.

Lemma 5. *Suppose that $G = G_1 \otimes G_2$ or $G = G_1 \oplus G_2$. Assume that $MI_t(G)$ contains a vertex in $TS(G_1)$ and a vertex in $TS(G_2)$. Let $I_1 = MI_t(G) \cap G_1$ and $I_2 = MI_t(G) \cap G_2$. Let $T_1 = I_1 \cap G_1[TS(G_1)]$ and $F_1 = F_t(G_1, I_1)$. Let $T_2 = I_2 \cap G_2[TS(G_2)]$ and $F_2 = F_t(G_2, I_2)$. Let $i \in \{1,2\}$. The following statements are true.*

(1) *Any two nonadjacent vertices of T_i have no common neighbor in $I_i - T_i$.*

(2) *If $F_i = T_i$, then I_{3-i} has no t-weak AT-free set.*

(3) *If $T_1 \subset F_1$ and $T_2 \subset F_2$, then I_1 and I_2 have no weak AT-free set.*

Lemma 6. *Assume that $G = G_1 \otimes G_2$ or $G = G_1 \oplus G_2$. For $i \in \{1,2\}$, let $I_x \in I_q(G_i)$, $I_y \in I_q(G_{3-i})$, and $I_z \in I_{q'}(G_{3-i})$. Then $I_x \bowtie I_y$ and $I_x \bowtie I_z$ are chordal graphs.*

Lemma 7. *Suppose that $G = G_1 \otimes G_2$. Let $i \in \{1,2\}$. $MI_t(G)$ is one of the following candidates, whichever has maximum number of vertices:*

(1) $MI_t(G_i) \cup MI_{t'}(G_{3-i})$.

(2) $MI_{q,0}(G_i) \bowtie \max \left\{ \begin{array}{l} MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}), \\ MI^0_{q,1}(G_{3-i}), \\ MI^0_{q',1}(G_{3-i}) \end{array} \right\}$.

(3) $MI^0_{q,1}(G_i) \bowtie \max \left\{ \begin{array}{l} MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}) \end{array} \right\}$.

(4) $MI^1_{q,1}(G_i) \bowtie \max \left\{ \begin{array}{l} MI^1_{q,1}(G_{3-i}) \\ MI^1_{q',1}(G_{3-i}) \end{array} \right\}$.

Lemma 8. *Assume that $G = G_1 \oplus G_2$. Let $i \in \{1,2\}$. $MI_t(G)$ is one of the following candidates, whichever has maximum number of vertices.*

(1) $MI_t(G_1) \cup MI_{t'}(G_2)$.

(2) $MI_{q,0}(G_i) \bowtie \max \left\{ \begin{array}{l} MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}), \\ MI^0_{q,1}(G_{3-i}), \\ MI^0_{q',1}(G_{3-i}) \end{array} \right\}$.

(3) $MI^0_{q,1}(G_i) \bowtie \max \left\{ \begin{array}{l} MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}) \end{array} \right\}$.

(4) $MI^1_{q,1}(G_i) \bowtie \max \left\{ \begin{array}{l} MI^1_{q,1}(G_{3-i}) \\ MI^1_{q',1}(G_{3-i}) \end{array} \right\}$.

Lemmas 9–12 are concerned with the computations of $MI_{q,0}(G)$, $MI^0_{q,1}(G)$, $MI^1_{q,1}(G)$, $MI_{q',0}(G)$, $MI^0_{q',1}(G)$, and $MI^1_{q',1}(G)$.

Lemma 9. *Suppose that G consists only one vertex. $MI_{q,0}(G) = G$. $MI^0_{q,1}(G)$, $MI^1_{q,1}(G)$, $MI_{q',0}(G)$, $MI^0_{q',1}(G)$, and $MI^1_{q',1}(G)$ do not exist and are denoted by Φ.*

Lemma 10. *Suppose that $G = G_1 \odot G_2$. Then*

(1) $MI_{q,0}(G) = \max \left\{ \begin{array}{l} MI_{q,0}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI_{q,0}(G_2) \end{array} \right\}$.

(2) $MI^0_{q,1}(G) = \max \left\{ \begin{array}{l} MI^0_{q,1}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI^0_{q,1}(G_2) \end{array} \right\}$.

(3) $MI^1_{q,1}(G) = \max \left\{ \begin{array}{l} MI^1_{q,1}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI^1_{q,1}(G_2) \end{array} \right\}$.

(4) $MI_{q',0}(G) = \max_{i\in\{1,2\}} A$, *where* $A=$
$$\left\{ \begin{array}{l} MI_{q',0}(G_i) \cup \max \left\{ \begin{array}{l} MI_{t'}(G_{3-i}), \\ MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}) \end{array} \right\}, \\ MI_{q,0}(G_i) \cup MI_{q,0}(G_{3-i}) \end{array} \right\}.$$

(5) $MI^0_{q',1}(G) = \max_{i\in\{1,2\}} A$, *where* $A =$
$$\left\{ \begin{array}{l} MI^0_{q',1}(G_i) \cup \max \left\{ \begin{array}{l} MI_{t'}(G_{3-i}), \\ MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}), \\ MI^0_{q,1}(G_{3-i}), \\ MI^0_{q',1}(G_{3-i}) \end{array} \right\}, \\ MI^0_{q,1}(G_i) \cup \max \left\{ \begin{array}{l} MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}), \\ MI^0_{q,1}(G_{3-i}) \end{array} \right\}, \end{array} \right\}.$$

(6) $MI^1_{q',1}(G) = \max_{i\in\{1,2\}} A$, *where* $A =$

$$\left\{\begin{array}{l} MI^1_{q',1}(G_i) \cup \max\left\{\begin{array}{l} MI_{t'}(G_{3-i}), \\ MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}), \\ MI^1_{q,1}(G_{3-i}), \\ MI^1_{q',1}(G_{3-i}) \end{array}\right\}, \\ MI^1_{q,1}(G_i) \cup \max\left\{\begin{array}{l} MI_{q,0}(G_{3-i}), \\ MI_{q',0}(G_{3-i}), \\ MI^1_{q,1}(G_{3-i}) \end{array}\right\} \end{array}\right\}.$$

Lemma 11. *Assume that $G_1 \otimes G_2$. Then*

(1) $MI_{q,0}(G) =$
$$\max\left\{\begin{array}{l} MI_{q,0}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI_{q,0}(G_2), \\ MI_{q,0}(G_1) \bowtie MI_{q,0}(G_2) \end{array}\right\}.$$

(2) $MI^0_{q,1}(G) =$
$$\max\left\{\begin{array}{l} MI^0_{q,1}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI^0_{q,1}(G_2), \\ MI^0_{q,1}(G_1) \bowtie MI_{q,0}(G_2), \\ MI_{q,0}(G_1) \bowtie MI^0_{q,1}(G_2), \\ MI^1_{q,1}(G_1) \bowtie MI^1_{q,1}(G_2) \end{array}\right\}.$$

(3) $MI^1_{q,1}(G) =$
$$\max\left\{\begin{array}{l} MI^1_{q,1}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI^1_{q,1}(G_2), \\ MI^1_{q,1}(G_1) \bowtie MI_{q,0}(G_2), \\ MI_{q,0}(G_1) \bowtie MI^1_{q,1}(G_2), \\ MI^1_{q,1}(G_1) \bowtie MI^1_{q,1}(G_2) \end{array}\right\}.$$

(4) $MI_{q',0}(G) =$
$$\max\left\{\begin{array}{l} MI_{q',0}(G_1) \cup MI_{t'}(G_2), \\ MI_{t'}(G_1) \cup MI_{q',0}(G_2), \\ MI_{q',0}(G_1) \cup MI_{q,0}(G_2), \\ MI_{q,0}(G_1) \cup MI_{q',0}(G_2) \end{array}\right\}.$$

(5) $MI^0_{q',1}(G) =$
$$\max_{i\in\{1,2\}}\left\{\begin{array}{l} MI^0_{q',1}(G_i) \cup MI_{t'}(G_{3-i}), \\ MI^0_{q',1}(G_i) \bowtie MI_{q,0}(G_{3-i}), \\ MI^1_{q',1}(G_i) \bowtie MI^1_{q,1}(G_{3-i}), \\ MI^0_{q,1}(G_i) \bowtie MI_{q',0}(G_{3-i}) \end{array}\right\}.$$

(6) $MI^1_{q',1}(G) =$
$$\max_{i\in\{1,2\}}\left\{\begin{array}{l} MI^1_{q',1}(G_i) \cup MI_{t'}(G_{3-i}), \\ MI^1_{q',1}(G_i) \bowtie MI_{q,0}(G_{3-i}), \\ MI^1_{q',1}(G_i) \bowtie MI^1_{q,1}(G_{3-i}), \\ MI^1_{q,1}(G_i) \bowtie MI_{q'0}(G_{3-i}), \end{array}\right\}.$$

Lemma 12. *Assume that $G_1 \oplus G_2$. Then*

(1) $MI_{q,0}(G) = MI_{q,0}(G_1) \cup MI_{t'}(G_2)$.

(2) *Let* $A = \max\{MI_{q,0}(G_2), MI^1_{q,1}(G_2), MI_{q',0}(G_2), MI^1_{q',1}(G_2)\}$. *Then* $MI^0_{q,1}(G) = \max B$, *where* $B =$
$$\left\{\begin{array}{l} MI^0_{q,1}(G_1) \cup MI_{t'}(G_2), \\ MI^0_{q,1}(G_1) \cup \max\left\{\begin{array}{l} MI_{q,0}(G_2), \\ MI_{q',0}(G_2) \end{array}\right\}, \\ \max\left\{\begin{array}{l} MI_{q,0}(G_1), \\ MI^1_{q,1}(G_1) \end{array}\right\} \cup A \end{array}\right\}.$$

(3) *Let* $A = \max\{MI_{q,0}(G_2), MI^1_{q,1}(G_2), MI_{q',0}(G_2), MI^1_{q',1}(G_2)\}$. *Then,* $MI^1_{q,1}(G) = \max B$, *where* $B =$
$$\left\{\begin{array}{l} MI^1_{q,1}(G_1) \cup MI_{t'}(G_2), \\ MI^1_{q,1}(G_1) \cup \max\left\{\begin{array}{l} MI_{q,0}(G_2), \\ MI_{q',0}(G_2) \end{array}\right\}, \\ \max\left\{\begin{array}{l} MI_{q,0}(G_1), \\ MI^1_{q,1}(G_1) \end{array}\right\} \cup A \end{array}\right\}.$$

(4) $MI_{q',0}(G) = MI_{q',0}(G_1) \cup MI_{t'}(G_2)$

(5) $MI^0_{q',1}(G) = MI^0_{q',1}(G_1) \cup MI_{t'}(G_2)$.

(6) $MI^1_{q',1}(G) = MI^1_{q',1}(G_1) \cup MI_{t'}(G_2)$.

Recall that every DH graph G can be defined by a rooted decomposition tree $\mathcal{T}$. According to Lemmas 1–12, we can compute the maximum interval graph of G by using $\mathcal{T}$ from leaves to root. It is not hard to check that the total time of the computation is linear. Therefore, we have the following theorem.

Theorem 2. *The maximum interval graph problem can be solved in linear time for DH graphs.*

References

[1] F. Alizadeh, K.M. Karp, L.E. Newberg, and D.K. Weisser, Physical mapping of chromosome: a combinatorial problem in molecular biology, *Proceedings of the Fourth ACM-SIAM Symposium on Discrete Algorithms* (SODA'93), ACM Press, 1993, pp. 371–381.

[2] M.C. Golumbic, H. Kaplan, and R. Shamir, On the complexity of DNA physical mapping, *Advances in Applied Mathematics* **15** (1994) 251–261.

[3] T. Jiang and R.M. Karp, Mapping clones with a given ordering or interleaving, *Algorithmica* **21** (1998) 262–284

[4] M.S. Krishnamoorthy and N. Deo, Node-deletion NP-complete problem, *SIAM Journal on Computing* **8** (1979) 619–625.

[5] J.M. Lewis and M. Yannakakis, The node-deletion problem for hereditary properties is NP-complete, *Journal of Computer and System Sciences* **20** (1980) 219–230.

[6] M.S. Chang, S.Y. Hsieh, and G.H. Chen, Dynamic programming on distance hereditary graphs, Lecture Notes in Computer Science **1350** (1997) 344–353.

[7] C. Lekkerkerker and D. Boland, Representation of finite graphs by a set of intervals on the real line, *Fundamenta Mathematicae*, 51 (1962) 45–64.

[8] M. Yannakakis, Node-deletion problems on bipartite graphs, *SIAM Journal of Computing* **10** (1981) 310–327.