The Longest Path Problem is Polynomial on Cocomparability Graphs

Kyriaki Ioannidou and Stavros D. Nikolopoulos

Department of Computer Science, University of Ioannina
GR-45110 Ioannina, Greece
{kioannid,stavros}@cs.uoi.gr

Abstract. The longest path problem is the problem of finding a path of maximum length in a graph. As a generalization of the Hamiltonian path problem, it is NP-complete on general graphs and, in fact, on every class of graphs that the Hamiltonian path problem is NP-complete. Polynomial solutions for the longest path problem have recently been proposed for weighted trees, ptolemaic graphs, bipartite permutation graphs, interval graphs, and some small classes of graphs. Although the Hamiltonian path problem on cocomparability graphs was proved to be polynomial almost two decades ago [9], the complexity status of the longest path problem on cocomparability graphs has remained open until now; actually, the complexity status of the problem has remained open even on the smaller class of permutation graphs. In this paper, we present a polynomial-time algorithm for solving the longest path problem on the class of cocomparability graphs. Our result resolves the open question for the complexity of the problem on such graphs, and since cocomparability graphs form a superclass of both interval and permutation graphs, extends the polynomial solution of the longest path problem on interval graphs [18] and provides polynomial solution to the class of permutation graphs.

Keywords: Longest path problem, cocomparability graphs, permutation graphs, polynomial algorithm, complexity.

1 Introduction

The problem of finding a path of maximum length in a graph (Longest Path Problem) generalizes the Hamiltonian path problem and thus it is NP-complete on general graphs; in fact, it is NP-complete on every class of graphs that the Hamiltonian path problem is NP-complete. It is thus interesting to study the longest path problem on classes of graphs $\mathcal{C}$ where the Hamiltonian path problem is polynomial, since if a graph $G \in \mathcal{C}$ is not Hamiltonian, it makes sense in several applications to search for a longest path of G. Although the Hamiltonian path problem has been extensively studied in the past two decades, only recently did the longest path problem start receiving attention [11,12,13,23,25,26,27].

The Hamiltonian path problem is known to be NP-complete in general graphs [14,15], and remains NP-complete even when restricted to some small

D.M. Thilikos (Ed.): WG 2010, LNCS 6410, pp. 27–38, 2010.

classes of graphs such as split graphs [16], chordal bipartite graphs, split strongly chordal graphs [21], directed path graphs [22], circle graphs [7], planar graphs [15], and grid graphs [19,24]. On the other hand, it admits polynomial time solutions on some known classes of graphs; such classes include interval graphs [1,8], circular-arc graphs [8], biconvex graphs [2], and cocomparability graphs [9]. Note that the problem of finding a longest path on proper interval graphs is easy, since all connected proper interval graphs have a Hamiltonian path which can be computed in linear time [3].

Polynomial time solutions for the longest path problem are known only for small classes of graphs. Specifically, a linear-time algorithm for finding a longest path in a tree was proposed by Dijkstra early in 1960, a formal proof of which can be found in [5]. Recently, through a generalization of Dijkstra's algorithm for trees, Uehara and Uno [25] solved the longest path problem for weighted trees and block graphs in linear time and space, and for cacti in $O(n^2)$ time and space, where n is the number of vertices of the input graph. Polynomial algorithms for the longest path problem have been also proposed on bipartite permutation and ptolemaic graphs having $O(n)$ and $O(n^5)$ time complexity, respectively [23,26]. Furthermore, Uehara and Uno in [25] solved the longest path problem on a subclass of interval graphs in $O(n^3(m + n \log n))$ time, and as a corollary they showed that a longest path on threshold graphs can be found in $O(n + m)$ time and space. Recently, Ioannidou *et al.* [18] showed that the longest path problem has a polynomial solution on interval graphs by proposing an algorithm that runs in $O(n^4)$ time, answering thus the question left open in [25] concerning the complexity of the problem on interval graphs.

In this paper we present a polynomial-time algorithm for solving the longest path problem on the class of cocomparability graphs, an important and well-known class of perfect graphs [16]. The Hamiltonian path problem on cocomparability graphs has been proved to be polynomial [9], while the status of the longest path problem on such graphs was unknown; actually, the status of the longest path problem was unknown even on the smaller class of permutation graphs. Thus, our result resolves the open question for the complexity of the problem on cocomparability graphs, and since cocomparability graphs form a superclass of both interval and permutation graphs, extends the polynomial solution of the longest path problem on interval graphs [18], and also provides polynomial solution to the class of permutation graphs.

2 Theoretical Framework

For basic definitions in graph theory refer to [4,16,20]. A *simple path* (resp. *antipath*) of a graph G is a sequence of distinct vertices $v_1, v_2, \ldots, v_k$ such that $v_i v_{i+1} \in E(G)$ (resp. $v_i v_{i+1} \notin E(G)$), for each i, $1 \leq i \leq k-1$, and is denoted by $(v_1, v_2, \ldots, v_k)$; throughout the paper all paths and antipaths are considered to be simple. We denote by $V(P)$ the set of vertices in the path (antipath) P, and define the *length* of the path (antipath) P to be the number of vertices in P, i.e., $|P| = |V(P)|$. We call *right endpoint* of a path (antipath) $P = (v_1, v_2, \ldots, v_k)$ the last vertex v_k of P. Moreover, if $P = (v_1, v_2, \ldots, v_{i-1}, v_i, v_{i+1}, \ldots, v_j, v_{j+1}, v_{j+2}, \ldots, v_k)$ is a path (antipath) of a graph and $P_0 = (v_i, v_{i+1}, \ldots, v_j)$ is a subpath (subantipath) of P, we shall denote the path (antipath) P by $P = (v_1, v_2, \ldots, v_{i-1}, P_0, v_{j+1}, v_{j+2}, \ldots, v_k)$.

2.1 Partial Orders and Cocomparability Graphs

A *partial order* will be denoted by $\mathcal{P} = (V, <_{\mathcal{P}})$, where V is the finite ground set of elements or vertices and $<_{\mathcal{P}}$ is an irreflexive, antisymmetric, and transitive binary relation on V. Two elements $a, b \in V$ are comparable in $\mathcal{P}$ (denoted by $a \sim_{\mathcal{P}} b$) if $a <_{\mathcal{P}} b$ or $b <_{\mathcal{P}} a$; otherwise, they are said to be incomparable (denoted by $a \parallel b$). An *extension* of a partial order $\mathcal{P} = (V, <_{\mathcal{P}})$ is a partial order $L = (V, <_L)$ on the same ground set that *extends* $\mathcal{P}$, i.e., $a <_{\mathcal{P}} b \Rightarrow a <_L b$, for all $a, b \in V$. The *dual partial order* $\mathcal{P}^d$ of $\mathcal{P} = (V, <_{\mathcal{P}})$ is a partial order $\mathcal{P}^d = (V, <_{\mathcal{P}^d})$ such that for any two elements $a, b \in V$, $a <_{\mathcal{P}^d} b$ if and only if $b <_{\mathcal{P}} a$.

The graph G, edges of which are exactly the comparable pairs of a partial order $\mathcal{P}$ on $V(G)$, is called the *comparability graph* of $\mathcal{P}$, and is denoted by $G(\mathcal{P})$. The complement graph $\overline{G}$, whose edges are the incomparable pairs of $\mathcal{P}$, is called the *cocomparability graph* of $\mathcal{P}$, and is denoted by $\overline{G}(\mathcal{P})$. Alternatively, a graph G is a cocomparability graph if its complement graph $\overline{G}$ has a transitive orientation, corresponding to the comparability relations of a partial order $\mathcal{P}_{\overline{G}}$. Note that a partial order $\mathcal{P}$ uniquely determines its comparability graph $G(\mathcal{P})$ and its cocomparability graph $\overline{G}(\mathcal{P})$, but the reverse is not true, i.e., a cocomparability graph G has as many partial orders $\mathcal{P}_{\overline{G}}$ as the number of the transitive orientations of $\overline{G}$. Also, the class of cocomparability graphs is hereditary.

Let G be a comparability graph, and let $\mathcal{P}_G$ be a partial order which corresponds to G. The graph G can be represented by a directed covering graph with layers $H_1, H_2, \ldots, H_h$, in which each vertex is on the highest possible layer. That is, the maximal vertices of the partial order $\mathcal{P}_G$ are on the highest layer H_h, and for every vertex v on layer H_{i-1} there exists a vertex u on layer H_i such that $v <_{\mathcal{P}_G} u$; such a layered representation of G (respectively $\mathcal{P}_G$) is a called the *Hasse diagram* of G (respectively $\mathcal{P}_G$) [9].

Let $\sigma = (V(G), <_\sigma)$ be a partial order on the vertices of a comparability graph G, such that for any two vertices $v, u \in V(G)$, $v <_\sigma u$ if and only if $v \in H_i$, $u \in H_j$, and $i < j$; hereafter, we equivalently denote $v <_\sigma u$ by $u >_\sigma v$. For simplicity sometimes we shall write $v =_\sigma u$, for vertices $v, u \in V(G)$ which belong to the same layer H_i; we write $v \neq_\sigma u$ to denote that vertices $v, u \in V(G)$ belong to different layers. Also, $v \leq_\sigma u$ implies that either $v <_\sigma u$ or $v =_\sigma u$; again, we equivalently denote $v \leq_\sigma u$ by $u \geq_\sigma v$. Throughout the paper, such an ordering σ is called a *layered ordering* of G. Note that, the partial order σ is an extension of the partial order $\mathcal{P}_G$; in particular, it holds $v <_{\mathcal{P}_G} u$ if and only if $v <_\sigma u$ and $vu \in E(G)$, for any two vertices $u, v \in V(G)$.

Since a comparability graph G does not uniquely determine a partial order, hereafter we will represent a comparability graph G by its Hasse diagram and we will denote the partial order $(V(G), <_{\mathcal{P}_G})$ to which the Hasse diagram of G

corresponds by $\mathcal{P}_G$; that is, the vertices which are on the highest layer H_h of the Hasse diagram are the maximal vertices of the partial order $\mathcal{P}_G$, and for two vertices $u, v \in V(G)$, $v <_{\mathcal{P}_G} u$ if $v \in H_{i-1}$, $u \in H_i$ and $uv \in E(G)$. Thus, we will say that $\mathcal{P}_G$ is the partial order which *corresponds* to the comparability graph G. Note that vertices in the Hasse diagram satisfy the following property: for any three vertices $v, u, w \in V(G)$ such that $v \in H_i$, $u \in H_j$, $w \in H_k$, and $i < j < k$ (or, equivalently, $v <_\sigma u <_\sigma w$), if $vu \in E(G)$ and $uw \in E(G)$, then $vw \in E(G)$.

The following definition and results where given by Damaschke *et al.* in [9], based on which they prove the correctness of their algorithm for finding a Hamiltonian path of a cocomparability graph; note that their algorithm uses the bump number algorithm which is presented in [17].

Definition 1. *(Damaschke et al. [9]): Let G be a comparability graph, and let $\mathcal{P}_G$ be the partial order which corresponds to G. A path $P = (v_1, v_2, \ldots, v_k)$ of the cocomparability graph $\overline{G}$ is monotone if $v_i <_{\mathcal{P}_G} v_j$ implies $i < j$.*

Lemma 1. *(Damaschke et al. [9]): Let G be a comparability graph, and let $\mathcal{P}_G$ be the partial order which corresponds to G. Let $P = (v_1, v_2, \ldots, v_k)$ be a Hamiltonian path of the cocomparability graph $\overline{G}$ such that v_1 is a minimal element of $\mathcal{P}_G$. Then there exists a monotone Hamiltonian path P' of $\overline{G}$ starting with v_1.*

Theorem 1. *(Damaschke et al. [9]): Let G be a cocomparability graph. Then, G has a Hamiltonian path if and only if G has a monotone Hamiltonian path.*

It appears that the above two results hold not only for Hamiltonian paths of a cocomparability graph $\overline{G}$, but also for any path of $\overline{G}$. Indeed, let P be a path of $\overline{G}$ and let $\overline{G'} = \overline{G}[V(P)]$ be the subgraph of $\overline{G}$ induced by the vertices of P. Also, let $\mathcal{P}_{G'}$ be the partial order which corresponds to G' such that $\mathcal{P}_G$ is an extension of $\mathcal{P}_{G'}$, i.e., for any two vertices $u, v \in V(\overline{G})$, if $u <_{\mathcal{P}_G} v$ and $u, v \in V(\overline{G'})$, then $u <_{\mathcal{P}_{G'}} v$. Then, since P is a Hamiltonian path of $\overline{G'}$, from Theorem 1 there exists a monotone path P' of $\overline{G'}$ (with respect to $\mathcal{P}_{G'}$) such that $V(P') = V(P)$. From Definition 1 it is easy to see that P' is also a monotone path of $\overline{G}$ (with respect to $\mathcal{P}_G$), since $\mathcal{P}_G$ is an extension of $\mathcal{P}_{G'}$.

Additionally, since a path P of a cocomparability graph $\overline{G}$ is an antipath of the comparability graph G, and since our algorithm for computing a longest path of a cocomparability graph $\overline{G}$ computes in fact a longest antipath of the comparability graph G, we restate the above definition and results and whenever P denotes a path of a cocomparability graph $\overline{G}$, we refer to P as an antipath of the comparability graph G.

We first restate Definition 1 as follows: an antipath $P = (v_1, v_2, \ldots, v_k)$ of a comparability graph G is monotone if $v_i <_{\mathcal{P}_G} v_j$ implies $i < j$, where $\mathcal{P}_G$ is the partial order which corresponds to G. We next restate Lemma 1 and Theorem 1 in a form stronger than the one stated in [9].

Lemma 2. *Let G be a comparability graph, and let $\mathcal{P}_G$ be the partial order which corresponds to G. Let $P = (v_1, v_2, \ldots, v_k)$ be an antipath of G such that v_1 is a minimal element of $V(P)$ in $\mathcal{P}_G$. Then there exists a monotone antipath P' of G starting with vertex v_1 such that $V(P') = V(P)$.*

Theorem 2. *Let G be a comparability graph. If P is an antipath of G, then there exists a monotone antipath P' of G such that $V(P') = V(P)$.*

The following lemma holds.

Lemma 3. *Let G be a comparability graph, and let σ be the layered ordering of G. Let $P = (v_1, v_2, \ldots, v_k)$ be an antipath of G, and let $v_\ell \notin V(P)$ be a vertex of G such that $v_1 \leq_\sigma v_\ell <_\sigma v_k$ and $v_\ell v_k \in E(G)$. Then there exist two consecutive vertices v_{i-1} and v_i in P, $2 \leq i \leq k$, such that $v_{i-1}v_\ell \notin E(G)$ and $v_\ell <_\sigma v_i$.*

2.2 Normal Antipaths on Comparability Graphs

Our algorithm for computing a longest antipath P of a comparability graph G uses a specific type of antipaths, which we call *normal* antipaths.

Definition 2. *Let G be a comparability graph, and let σ be a layered ordering of G. The antipath $P = (v_1, v_2, \ldots, v_k)$ of G is called normal, if v_1 is a leftmost (i.e., minimal) vertex of $V(P)$ in σ, and for every i, $2 \leq i \leq k$, the vertex v_i is a leftmost vertex of $N_{\overline{G}}(v_{i-1}) \cap \{v_i, v_{i+1}, \ldots, v_k\}$ in σ.*

Based on Lemma 3 and Definition 2, we prove the following result.

Lemma 4. *Let G be a comparability graph, and let σ be the layered ordering of G. Let $P = (v_1, v_2, \ldots, v_k)$ be a normal antipath of G, and let v_ℓ, and v_j be two vertices of P such that $v_\ell <_\sigma v_j$ and $v_\ell v_j \in E(G)$. Then $\ell < j$, i.e., v_ℓ appears before v_j in P.*

Recall that, if $\mathcal{P}_G$ is the partial order corresponding to a comparability graph G, and σ is the layered ordering of G, then $v_\ell <_{\mathcal{P}_G} v_j$ if and only if $v_\ell <_\sigma v_j$ and $v_\ell v_j \in E(G)$, for any two vertices $v_\ell, v_j \in V(G)$. Therefore, the definition of a monotone antipath can be paraphrased as follows: an antipath $P = (v_1, v_2, \ldots, v_k)$ of a comparability graph G is monotone if $v_\ell <_\sigma v_j$ and $v_\ell v_j \in E(G)$ implies that v_ℓ appears before v_j in P. Then, from Lemma 4 we obtain the following result.

Corollary 1. *Let G be a comparability graph. If P is a normal antipath of G, then P is a monotone antipath of G.*

Note that the inverse of Corollary 1 is not always true; for example, see the antipath P in Figure 1. In [9], for proving that for any Hamiltonian path P of a cocomparability graph $\overline{G}$ there exists a monotone Hamiltonian path of $\overline{G}$, Damaschke *et al.* first show that there exists a path $P' = (v_1, v_2, \ldots, v_{|V(\overline{G})|})$ of $\overline{G}$ such that v_1 is a minimal vertex of either $\mathcal{P}_G$ or $\mathcal{P}_G^d$. Using the same arguments, we obtain the following lemma.

Lemma 5. *Let G be a comparability graph, and let $\mathcal{P}_G$ be the partial order which corresponds to G. If P is an antipath of G, then there exists an antipath P' of G such that $V(P') = V(P)$ which starts with a minimal vertex of $V(P)$ in $\mathcal{P}_G$.*

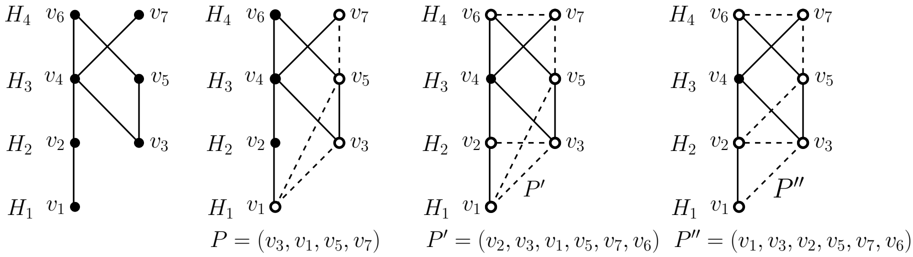

Fig. 1. Illustrating a Hasse diagram of a comparability graph G, an antipath P of G which is neither normal nor longest, an antipath P' of G such that $|P'| > |P|$ which is not normal, and a normal antipath P'' of G such that $V(P'') = V(P')$

The following result is central for the correctness of our algorithm.

Lemma 6. *Let P be a longest antipath of a comparability graph G. Then, there exists a normal antipath P' of G such that $V(P') = V(P)$.*

Figure 1 illustrates a Hasse diagram of a comparability graph G. The antipath $P = (v_3, v_1, v_5, v_7)$ of G is not normal, and there exists no normal antipath $\widehat{P}$ of G such that $V(\widehat{P}) = V(P)$; however, note that P is monotone. Also, P is not a longest antipath of G, since there exists an antipath $P' = (v_2, v_3, v_1, v_5, v_7, v_6)$ of G such that $|P'| > |P|$. Also, P' is not a normal antipath of G and there exists a normal antipath $P'' = (v_1, v_3, v_2, v_5, v_7, v_6)$ of G such that $V(P'') = V(P')$; note that it is easy to see that P'' is a longest antipath of G.

3 The Algorithm

Our algorithm, which we call Algorithm LP_Cocomparability, computes a longest path P of a cocomparability graph G by computing a longest antipath P of the comparability graph $\overline{G}$.

Let G be a comparability graph and let $H_1, H_2, \ldots, H_k$ be the layers of its Hasse diagram. For simplifying our description, we add a dummy vertex u_0 to G such that u_0 belongs to a layer H_0 and $u_0u_i \in E(G)$ for every i, $1 \leq i \leq n$; let G' be the resulting graph. Note that, G' is a comparability graph having a Hasse diagram with layers $H_0, H_1, H_2, \ldots, H_k$, and let σ be a layered ordering of G', where $V(G') = \{u_0, u_1, u_2, \ldots, u_n\}$. It is easy to see that u_0 does not participate in any longest antipath P of G' such that $|P| \geq 2$. In general, a longest antipath P of G' which does not contain the vertex u_0 is also a longest antipath of G. Algorithm LP_Cocomparability computes a longest antipath of G' which is a longest antipath of the original graph G as well. Hereafter, we consider comparability graphs G having assumed that we have already added the dummy vertex u_0. Thus, the antipaths we compute in G are also antipaths of the graph $G \setminus \{u_0\}$.

We next give some definitions and notations necessary for the description of the algorithm. Let $L_j = (v_1, v_2, \ldots, v_k)$ be an arbitrary ordering of the vertices $v_1, v_2, \ldots, v_k$. We denote by $V(L_j)$ the set $\{v_1, v_2, \ldots, v_k\}$ and by $|L_j|$ the cardinality of the set $V(L_j)$. For every vertex $v_z \in L_j$, we denote by $L_j(v_z)$ the ordering $(v_1, v_2, \ldots, v_{z-1}, v_{z+1}, v_{z+2}, \ldots, v_{|L_j|}, v_z)$, and for every index r, $0 \leq r \leq |L_j|$, we denote by $L_j^r(v_z)$ the ordering containing the first r vertices of $L_j(v_z)$; thus:

- $L_j = (v_1, v_2, \ldots, v_k)$,
- $L_j(v_z) = (v_1, v_2, \ldots, v_{z-1}, v_{z+1}, v_{z+2}, \ldots, v_{|L_j|}, v_z)$,
- $L_j^r(v_z) = (v_1, v_2, \ldots, v_r)$ if $1 \leq r \leq z-1$,
- $L_j^r(v_z) = (v_1, v_2, \ldots, v_{z-1}, v_{z+1}, v_{z+2}, \ldots, v_{r+1})$ if $z \leq r \leq |L_j| - 1$,
- $L_j^0(v_z) = \emptyset$, and $L_j^{|L_j|}(v_z) = L_j(v_z)$.

Definition 3. *Let G be a comparability graph, let $H_0, H_1, H_2, \ldots, H_k$ be the layers of its Hasse diagram, let $V(G) = \{u_0, u_1, u_2, \ldots, u_n\}$, and let σ be the layered ordering of G. For every triple p, i, and j, where $1 \leq i \leq j \leq k$ and $u_p \in H_{i-1}$, we define the graph $G(u_p, i, j)$ to be the subgraph $G[S]$, where $S = \{u_x : u_x \in H_\ell, i \leq \ell \leq j\} \setminus \{u_x : u_p u_x \notin E(G)\}$.*

Definition 4. *Let L_j be an ordering of the set $H_j \cap V(G(u_p, i, j))$. We define the graph $G_{u_z}^r(u_p, i, j)$, where $u_z \in L_j$ and $0 \leq r \leq |L_j|$, to be the subgraph $G[S]$, where $S = V(G(u_p, i, j-1)) \cup L_j^r(u_z)$ if $i < j$, and $S = L_j^r(u_z)$ if $i = j$.*

Note that, since the dummy vertex u_0 is adjacent to every other vertex of G, the graph $G(u_p, 1, j)$, $1 \leq j \leq k$, is the subgraph $G[S]$ of G induced by the set $S = \{u_x : u_x \in H_\ell, 1 \leq \ell \leq j\}$. Additionally, $G_{u_z}^{|L_j|}(u_p, i, j) = G(u_p, i, j)$, and if $i < j$, then $G_{u_z}^0(u_p, i, j) = G(u_p, i, j-1)$.

Figure 2 illustrates examples of the graphs defined in Definitions 3 and 4. In particular, the figure to the left illustrates a Hasse diagram of a comparability graph G with layers $H_0, H_1, \ldots, H_5$. The figure in the middle illustrates the subgraph $G(v_1, 2, 4)$ of G induced by the vertices $\{v_3, v_6, v_7, v_8, v_9, v_{10}\}$. The figure to the right illustrates the subgraph $G_{v_9}^2(v_1, 2, 4)$ of G, if we consider the ordering $L_4 = (v_8, v_9, v_{10})$ for the vertices of $H_4 \cap V(G(v_1, 2, 4))$. The subgraph $G_{v_9}^2(v_1, 2, 4)$ of G is induced by the vertices $\{v_3, v_6, v_7, v_8, v_{10}\}$, and it is actually an induced subgraph of $G(v_1, 2, 4)$.

Notation 1. *For every vertex $u_t \in V(G_{u_z}^r(u_p, i, j))$, if $u_t \in H_j$, then we denote by $f(u_t)$ the smallest index such that $f(u_t) < j$, for which there exists a vertex u_x of $G_{u_z}^r(u_p, i, j)$ such that $u_x \in H_{f(u_t)}$ and $u_x u_t \notin E(G)$; in the case where no such index $f(u_t)$ exists, we set $f(u_t) = j$.*

Notation 2. *For every vertex $u_y \in V(G_{u_z}^r(u_p, i, j))$ we denote by $P(u_y; G_{u_z}^r(u_p, i, j))$ a longest normal antipath of $G_{u_z}^r(u_p, i, j)$ with right endpoint the vertex u_y, and by $\ell(u_y; G_{u_z}^r(u_p, i, j))$ the length of $P(u_y; G_{u_z}^r(u_p, i, j))$.*

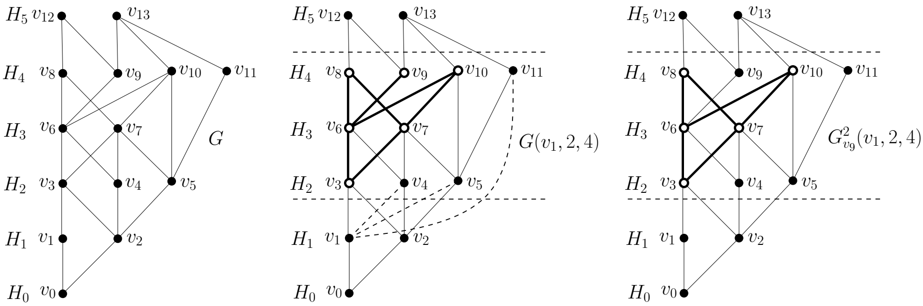

Fig. 2. Illustrating a Hasse diagram of a comparability graph G and the induced subgraphs $G(v_1, 2, 4)$ and $G^2_{v_9}(v_1, 2, 4)$ of G

Note that, if P is a longest normal antipath of $G(u_p, i, j)$ with right endpoint the vertex u_y, i.e., $P = P(u_y; G(u_p, i, j))$, then P is not necessarily a longest antipath of $G(u_p, i, j)$. However, if P is a longest antipath of $G(u_p, i, j)$, then from Lemma 6 there exists in $G(u_p, i, j)$ a normal antipath P' such that $V(P') = V(P)$; let u_y be the right endpoint of the normal antipath P'. Thus, there exists a longest normal antipath $P' = P(u_y; G(u_p, i, j))$ which is also a longest antipath in $G(u_p, i, j)$ for some vertex $u_y \in V(G(u_p, i, j))$.

Given a comparability graph G, Algorithm LP_Cocomparability (presented in Figures 3 and 4) computes for every induced subgraph $G(u_p, i, j)$ and for every vertex u_y of $G(u_p, i, j)$, the length $\ell(u_y; G(u_p, i, j))$ and the corresponding antipath $P(u_y; G(u_p, i, j))$, and outputs the maximum among the values $\{\ell(u_y; G(u_0, 1, k)) : u_y \in V(G(u_0, 1, k))\}$, and the corresponding normal antipath $P(u_y; G(u_0, 1, k))$. We prove that $P(u_y; G(u_0, 1, k))$ is a longest antipath of G.

4 Correctness and Time Complexity

Let G be a comparability graph, let $H_0, H_1, H_2, \dots, H_k$ be the layers of its Hasse diagram, and let σ be the layered ordering of G. We prove the following results.

Lemma 7. *Let L_j be an ordering of the set $H_j \cap V(G(u_p, i, j))$, let $P = (P_1, v_\ell, P_2)$ be a normal antipath of $G^r_{u_z}(u_p, i, j)$, and let v_ℓ be the last vertex of $L^r_j(u_z)$. Then, P_1 and P_2 are normal antipaths of $G^r_{u_z}(u_p, i, j)$.*

Lemma 8. *Let L_j be an ordering of the set $H_j \cap V(G(u_p, i, j))$, and let u_t be the last vertex of $L^r_j(u_z)$. Let P_1 be a normal antipath of $G^{r-1}_{u_z}(u_p, i, j)$ with right endpoint a vertex u_x such that $u_x \in H_\ell$, $f(u_t) \le \ell \le j - 1$, and $u_t u_x \notin E(G)$. Let P_2 be a normal antipath of $G^{r-1}_{u_z}(u_x, \ell + 1, j)$ with right endpoint a vertex u_y such that $u_y \in H_h$, $\ell + 1 \le h \le j$, and $V(P_1) \cap V(P_2) = \emptyset$. Then, $P = (P_1, u_t, P_2)$ is a normal antipath of $G^r_{u_z}(u_p, i, j)$ with right endpoint the vertex u_y.*

Algorithm LP_Cocomparability

Input: a comparability graph G where $V(G) = \{u_0, u_1, u_2, \ldots, u_n\}$, the layers $H_0, H_1, H_2, \ldots, H_k$ of its Hasse diagram, and a layered ordering σ of G.

Output: a longest normal antipath of G.

for $j = 1$ to k
  for $i = j$ downto 1
    for every vertex $u_p \in H_{i-1}$
      let L_j be an ordering of $H_j \cap V(G(u_p, i, j))$
      for every vertex $u_z \in L_j$
        for $r = 1$ to $|L_j|$
          let u_t be the last vertex of $L_j^r(u_z)$
          for every vertex $u_y \in V(G^r_{u_z}(u_p, i, j))$ and $y \neq t$ $\{$*initialization*$\}$
            if $r = 1$ **then**
              $\ell(u_y; G^0_{u_z}(u_p, i, j)) \leftarrow \ell(u_y; G(u_p, i, j-1))$;
              $P(u_y; G^0_{u_z}(u_p, i, j)) \leftarrow P(u_y; G(u_p, i, j-1))$;
            $\ell(u_y; G^r_{u_z}(u_p, i, j)) \leftarrow \ell(u_y; G^{r-1}_{u_z}(u_p, i, j))$;
            $P(u_y; G^r_{u_z}(u_p, i, j)) \leftarrow P(u_y; G^{r-1}_{u_z}(u_p, i, j))$;
          end_for
          if $i = j$ **then** $\{$*case* $i = j\}$
            $\ell(u_t; G^r_{u_z}(u_p, j, j)) \leftarrow |L_j^r(u_z)|$;
            $P(u_t; G^r_{u_z}(u_p, j, j)) \leftarrow L_j^r(u_z)$;
          if $i \neq j$ **then**
            $\ell(u_t; G^r_{u_z}(u_p, i, j)) \leftarrow 1$; $\{$*initialization for* $u_y = u_t\}$
            $P(u_t; G^r_{u_z}(u_p, i, j)) \leftarrow (u_t)$;
            execute process$(G^r_{u_z}(u_p, i, j))$;
        end_for
        $\ell(u_z; G(u_p, i, j)) \leftarrow \ell(u_z; G^{|L_j|}_{u_z}(u_p, i, j))$; $\{$*for the vertex* $u_z \in L_j\}$
        $P(u_z; G(u_p, i, j)) \leftarrow P(u_z; G^{|L_j|}_{u_z}(u_p, i, j))$;
      end_for
      for every vertex $u_y \in V(G(u_p, i, j))$ and $u_y \notin L_j$ $\{$*for* $u_y \notin L_j\}$
        $\ell(u_y; G(u_p, i, j)) \leftarrow \ell(u_y; G^{|L_j|}_{u_z}(u_p, i, j))$;
        $P(u_y; G(u_p, i, j)) \leftarrow P(u_y; G^{|L_j|}_{u_z}(u_p, i, j))$;
      end_for
    end_for
  end_for
end_for

compute the $max\{\ell(u_y; G(u_0, 1, k)) : u_y \in G(u_0, 1, k)\}$ and the corresponding antipath $P(u_y; G(u_0, 1, k))$;

Fig. 3. The algorithm for finding a longest antipath of G

Process($G^r_{u_z}(u_p, i, j)$)

procedure bridge($G^r_{u_z}(u_p, i, j)$)
if $f(u_t) < j$ **then** $\{u_t$ *is the last vertex of* $L^r_j(u_z)\}$
  for $h = f(u_t) + 1$ to j
    for $\ell = f(u_t)$ to $h - 1$
      for every vertex $u_x \in H_\ell \cap V(G^{r-1}_{u_z}(u_p, i, j))$ and $u_x u_t \notin E(G)$
        for every vertex $u_y \in H_h \cap V(G^{r-1}_{u_z}(u_x, \ell + 1, j))$
          $w_1 \leftarrow \ell(u_x; G^{r-1}_{u_z}(u_p, i, j))$; $P'_1 \leftarrow P(u_x; G^{r-1}_{u_z}(u_p, i, j))$;
          $w_2 \leftarrow \ell(u_y; G^{r-1}_{u_z}(u_x, \ell + 1, j))$; $P'_2 \leftarrow P(u_y; G^{r-1}_{u_z}(u_x, \ell + 1, j))$;
          if $w_1 + w_2 + 1 > \ell(u_y; G^r_{u_z}(u_p, i, j))$ **then**
            $\ell(u_y; G^r_{u_z}(u_p, i, j)) \leftarrow w_1 + w_2 + 1$;
            $P(u_y; G^r_{u_z}(u_p, i, j)) \leftarrow (P'_1, u_t, P'_2)$;

procedure append($G^r_{u_z}(u_p, i, j)$)
for $\ell = f(u_t)$ to j $\{u_t$ *is the last vertex of* $L^r_j(u_z)\}$
  for every vertex $u_x \in H_\ell \cap (V(G^{r-1}_{u_z}(u_p, i, j))$ and $u_x u_t \notin E(G)$
    $w_1 \leftarrow \ell(u_x; G^{r-1}_{u_z}(u_p, i, j))$; $P'_1 \leftarrow P(u_x; G^{r-1}_{u_z}(u_p, i, j))$;
    if $w_1 + 1 > \ell(u_t; G^r_{u_z}(u_p, i, j))$ **then**
      $\ell(u_t; G^r_{u_z}(u_p, i, j)) \leftarrow w_1 + 1$;
      $P(u_t; G^r_{u_z}(u_p, i, j)) \leftarrow (P'_1, u_t)$;

return (the value $\ell(u_y; G^r_{u_z}(u_p, i, j))$ and the antipath $P(u_y; G^r_{u_z}(u_p, i, j))$, for every vertex $u_y \in V(G^r_{u_z}(u_p, f(u_t) + 1, j))$ if $f(u_t) < j$, and for $u_y = u_t$ if $f(u_t) = j$);

Fig. 4. The procedure process()

Lemma 9. *For every induced subgraph $G(u_p, i, j)$ of G, and for every vertex $u_y \in V(G(u_p, i, j))$, the value $\ell(u_y; G(u_p, i, j))$ computed by Algorithm LP_Cocomparability is equal to the length of a longest normal antipath of $G(u_p, i, j)$ with right endpoint the vertex u_y and, also, the corresponding computed antipath $P(u_y; G(u_p, i, j))$ is a longest normal antipath of $G(u_p, i, j)$ with right endpoint the vertex u_y.*

Let P be a longest antipath of G such that $|P| \geq 2$. From Lemma 6 we may assume that P is a longest normal antipath of G and let u_y be its right endpoint. Also, P belongs to the graph $G \setminus \{u_0\}$. Since $G(u_0, 1, k) = G \setminus \{u_0\}$ and since Algorithm LP_Cocomparability computes the maximum among the lengths $\{\ell(u_y; G(u_0, 1, k)) : u_y \in V(G(u_0, 1, k))\}$ and the corresponding antipath P', from Lemma 9 we obtain that $|P'| = |P|$. Therefore, we obtain the following.

Theorem 3. *Algorithm LP_Cocomparability computes a longest path of a cocomparability graph in polynomial time.*

References

1. Arikati, S.R., Pandu Rangan, C.: Linear algorithm for optimal path cover problem on interval graphs. Inform. Proc. Lett. 35, 149–153 (1990)
2. Asdre, K., Nikolopoulos, S.D.: The 1-fixed-endpoint path cover problem is polynomial on interval graphs. Algorithmica, doi:10.1007/s00453-009-9292-5
3. Bertossi, A.A.: Finding Hamiltonian circuits in proper interval graphs. Inform. Proc. Lett. 17, 97–101 (1983)
4. Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph Classes: A Survey. SIAM, Philadelphia (1999)
5. Bulterman, R., van der Sommen, F., Zwaan, G., Verhoeff, T., van Gasteren, A., Feijen, W.: On computing a longest path in a tree. Inform. Proc. Lett. 81, 93–96 (2002)
6. Chang, M.S., Peng, S.L., Liaw, J.L.: Deferred-query: An efficient approach for some problems on interval graphs. Networks 34, 1–10 (1999)
7. Damaschke, P.: The Hamiltonian circuit problem for circle graphs is NP-complete. Inform. Proc. Lett. 32, 1–2 (1989)
8. Damaschke, P.: Paths in interval graphs and circular arc graphs. Discrete Math. 112, 49–64 (1993)
9. Damaschke, P., Deogun, J.S., Kratsch, D., Steiner, G.: Finding Hamiltonian paths in cocomparability graphs using the bump number algorithm. Order 8, 383–391 (1992)
10. Deogun, J.S., Steiner, G.: Polynomial algorithms for hamiltonian cycle in cocomparability graphs. SIAM J. Computing 23, 520–552 (1994)
11. Feder, T., Motwani, R.: Finding large cycles in Hamiltonian graphs. In: Proc. of the 16th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA), pp. 166–175. ACM, New York (2005)
12. Gabow, H.N.: Finding paths and cycles of superpolylogarithmic length. In: Proc. of the 36th Annual ACM Symp. on Theory of Computing (STOC), pp. 407–416. ACM, New York (2004)
13. Gabow, H.N., Nie, S.: Finding long paths, cycles and circuits. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 752–763. Springer, Heidelberg (2008)
14. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman, New York (1979)
15. Garey, M.R., Johnson, D.S., Tarjan, R.E.: The planar Hamiltonian circuit problem is NP-complete. SIAM J. Computing 5, 704–714 (1976)
16. Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs. Annals of Discrete Mathematics, vol. 57. North-Holland Publishing Co., Amsterdam (2004)
17. Habib, M., Mörhing, R.H., Steiner, G.: Computing the bump number is easy. Order 5, 107–129 (1988)
18. Ioannidou, K., Mertzios, G.B., Nikolopoulos, S.D.: The longest path problem has a polynomial solution on interval graphs. In: Královič, R., Niwiński, D. (eds.) MFCS 2009. LNCS, vol. 5734, pp. 403–414. Springer, Heidelberg (2009)
19. Itai, A., Papadimitriou, C.H., Szwarcfiter, J.L.: Hamiltonian paths in grid graphs. SIAM J. Computing 11, 676–686 (1982)
20. McKee, T.A., McMorris, F.R.: Topics in Intersection Graph Theory. Society for Industrial and Applied Mathematics, Philadelphia (1999)
21. Müller, H.: Hamiltonian circuits in chordal bipartite graphs. Discrete Math. 156, 291–298 (1996)

22. Narasimhan, G.: A note on the Hamiltonian circuit problem on directed path graphs. Inform. Proc. Lett. 32, 167–170 (1989)
23. Takahara, Y., Teramoto, S., Uehara, R.: Longest path problems on ptolemaic graphs. IEICE Trans. Inf. and Syst. 91-D, 170–177 (2008)
24. Uehara, R.: Simple geometrical intersection graphs. In: Nakano, S.-i., Rahman, M. S. (eds.) WALCOM 2008. LNCS, vol. 4921, pp. 25–33. Springer, Heidelberg (2008)
25. Uehara, R., Uno, Y.: Efficient algorithms for the longest path problem. In: Fleischer, R., Trippen, G. (eds.) ISAAC 2004. LNCS, vol. 3341, pp. 871–883. Springer, Heidelberg (2004)
26. Uehara, R., Valiente, G.: Linear structure of bipartite permutation graphs and the longest path problem. Inform. Proc. Lett. 103, 71–77 (2007)
27. Zhang, Z., Li, H.: Algorithms for long paths in graphs. Theoret. Comput. Sci. 377, 25–34 (2007)