The Longest Path Problem has a Polynomial Solution on Interval Graphs

Article *in* Algorithmica · October 2011

DOI: 10.1007/s00453-010-9411-3 · Source: DBLP

3 authors, including:

George B. Mertzios
Durham University
98 PUBLICATIONS **1,010** CITATIONS
SEE PROFILE

Stavros Nikolopoulos
University of Ioannina
122 PUBLICATIONS **723** CITATIONS
SEE PROFILE

The Longest Path Problem has a Polynomial Solution on Interval Graphs

Kyriaki Ioannidou [1] * George B. Mertzios [2] Stavros D. Nikolopoulos [1]

[1] *Department of Computer Science, University of Ioannina, P.O.Box 1186, GR-45110 Ioannina, Greece.*
{kioannid, stavros}@cs.uoi.gr

[2] *Department of Computer Science, RWTH Aachen University, Ahornstr. 55, D-52074 Aachen, Germany.*
mertzios@cs.rwth-aachen.de

Abstract

The longest path problem is the problem of finding a path of maximum length in a graph. Polynomial solutions for this problem are known only for small classes of graphs, while it is NP-hard on general graphs, as it is a generalization of the Hamiltonian path problem. Motivated by the work of Uehara and Uno in [23], where they left the longest path problem open for the class of interval graphs, in this paper we show that the problem can be solved in polynomial time on interval graphs. The proposed algorithm uses a dynamic programming approach and runs in $O(n^4)$ time, where n is the number of vertices of the input graph.

Keywords: Longest path problem, interval graphs, polynomial algorithm, complexity, dynamic programming.

1 Introduction

A well-known and studied problem in graph theory with numerous applications is the Hamiltonian path problem, i.e., the problem of determining whether a graph contains a simple path in which every vertex of the graph appears exactly once; such a graph is called Hamiltonian. In the case where a graph does not contain a Hamiltonian path, it makes sense in several applications to search for a path of maximum length in the graph; finding such a path is knows as the *longest path problem*. Although the two problems are similar, finding a longest path in a graph seems to be more difficult than deciding whether or not the graph admits a Hamiltonian path. Indeed, it has been proved that even if a graph has a Hamiltonian path, the problem of finding a path of length $n - n^{\varepsilon}$ for any $\varepsilon < 1$ is NP-hard, where n is the number of vertices of the graph [17]. Moreover, there is no polynomial-time constant-factor approximation algorithm for the longest path problem unless P=NP [17]. For related results see also [9–11, 25, 26].

It is clear that the longest path problem is NP-hard on every class of graphs on which the Hamiltonian path problem is NP-complete; note that, the Hamiltonian path problem is known to be NP-complete on general graphs [12, 13], and remains NP-complete even when restricted to some small classes of graphs such as split graphs [15], chordal bipartite graphs, split strongly chordal

*This research is co-financed by E.U.-European Social Fund (75%) and the Greek Ministry of Development-GSRT (25%).

graphs [19], directed path graphs [20], circle graphs [7], planar graphs [13], and grid graphs [16]. On the other hand, there are several classes of graphs on which the Hamiltonian path problem admits polynomial time solutions; these classes include proper interval graphs [3], interval graphs [1, 5, 8], circular-arc graphs [8], biconvex graphs [2], and cocomparability graphs [6]. Thus, if someone is interested in investigating the tractability of the longest path problem, it makes sense to focus on the classes of graphs for which the Hamiltonian path problem is polynomial.

In contrast to the Hamiltonian path problem, there are few known polynomial time solutions for the longest path problem, and these restrict to trees and some small graph classes. 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 [4]. Later, through a generalization of Dijkstra's algorithm for trees, Uehara and Uno [23] 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 and m denote the number of vertices and edges of the input graph, respectively. More recently, polynomial algorithms have been proposed that solve the longest path problem on bipartite permutation graphs in $O(n)$ time and space [24], and on ptolemaic graphs in $O(n^5)$ time and $O(n^2)$ space [22].

In 2004, Uehara and Uno [23] introduced a subclass of interval graphs, namely interval biconvex graphs, which is a superclass of proper interval and threshold graphs, and solved the longest path problem on this class in $O(n^3(m + n \log n))$ time. As a corollary, they showed that a longest path of a threshold graph can be found in $O(n + m)$ time and space. They left open the complexity of the longest path problem on the well-known class of interval graphs.

In this paper, we resolve the open problem posed in [23] by showing that the longest path problem admits a polynomial time solution on the class of interval graphs. In particular, we propose an algorithm for solving the longest path problem on interval graphs which runs in $O(n^4)$ time using a dynamic programming approach. Thus, not only we answer the question left open by Uehara and Uno in [23], but also improve the known time complexity of the problem on interval biconvex graphs, a subclass of interval graphs [23].

The rest of this paper is organized as follows. In Section 2, we review some properties of interval graphs and give the notion of a type of paths, which we call normal paths and is central for our algorithm. In Section 3, we present the three phases of our algorithm for solving the longest path problem on interval graphs, while in Section 4 we prove the correctness and analyze the time complexity of our algorithm. Finally, some concluding remarks are given in Section 5.

2 Theoretical Framework

We consider finite undirected graphs with no loops or multiple edges. For a graph G, we denote its vertex and edge set by $V(G)$ and $E(G)$, respectively. An undirected edge is a pair of distinct vertices $u, v \in V(G)$, and is denoted by uv. We say that the vertex u is adjacent to the vertex v or, equivalently, the vertex u sees the vertex v, if there is an edge uv in G. Let S be a set of vertices of a graph G. Then, the cardinality of the set S is denoted by $|S|$ and the subgraph of G induced by S is denoted by $G[S]$. Furthermore, the induced subgraph $G[S]$ is a *clique* if each two of its vertices are adjacent. The set $N(v) = \{u \in V(G) : uv \in E(G)\}$ is called the *neighborhood* of the vertex $v \in V(G)$ in G, sometimes denoted by $N_G(v)$ for clarity reasons. The set $N[v] = N(v) \cup \{v\}$ is called the *closed neighborhood* of the vertex $v \in V(G)$. A vertex $v \in V(G)$ is called *simplicial* if its neighborhood $N(v)$ induces a clique in G; in this case its closed neighborhood $N[v]$ induces also a clique in G.

A *simple path* of a graph G is a sequence of distinct vertices $v_1, v_2, \ldots, v_k$ such that $v_i v_{i+1} \in E(G)$, 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 considered are simple. We denote by $V(P)$ the set of vertices in the path P, and define the *length* of the path P to be the number of vertices in P, i.e.,

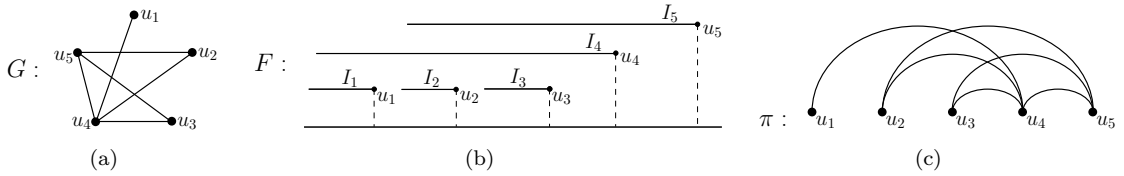

Figure 1: (a) An interval graph G, (b) an intersection model F of G, and (c) the corresponding right-end ordering $\pi = (u_1, u_2, u_3, u_4, u_5)$ of G.

$|P| = |V(P)|$. We call *right endpoint* of a path $P = (v_1, v_2, \ldots, v_k)$ the last vertex v_k of P. Additionally, 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 of a graph and $P_0 = (v_i, v_{i+1}, \ldots, v_j)$ is a subpath of P, we sometimes equivalently use the notation $P = (v_1, v_2, \ldots, v_{i-1}, P_0, v_{j+1}, v_{j+2}, \ldots, v_k)$.

2.1 Structural Properties of Interval Graphs

Interval graphs form a well-known and extensively studied class of perfect graphs [15]. They have important properties, and admit polynomial time solutions for several problems that are NP-complete on general graphs (see e.g. [1, 5, 15, 18]). Moreover, interval graphs have received a lot of attention due to their applicability to DNA physical mapping problems [14], and find many applications in several fields and disciplines such as genetics, molecular biology, scheduling, VLSI circuit design, archaeology and psychology [15].

A graph G is called *interval graph* if its vertices can be put in a one-to-one correspondence with a family F of intervals on the real line such that two vertices are adjacent in G if and only if the corresponding intervals intersect; F is called an *intersection model* for G [1]. The class of interval graphs is *hereditary*, that is, every induced subgraph of an interval graph G is also an interval graph. Ramalingam and Rangan [21] proposed a numbering of the vertices of an interval graph; they stated the following lemma.

Lemma 2.1 *(Ramalingam and Rangan [21]): The vertices of any interval graph G can be numbered with integers $1, 2, \ldots, |V(G)|$ such that if $i < j < k$ and $ik \in E(G)$, then $jk \in E(G)$.*

This numbering, which also results after sorting the intervals of the intersection model of an interval graph G on their right ends [1], can be obtained in $O(|V(G)| + |E(G)|)$ time [21]. An ordering of the vertices according to this numbering is found to be quite useful in solving some graph-theoretic problems on interval graphs [1, 21]. Throughout the paper, such an ordering is called a *right-end ordering* of G. Let u and v be two vertices of G, and let π be a right-end ordering of G; by $u <_\pi v$ we denote that u appears before v in π. In particular, if $\pi = (u_1, u_2, \ldots, u_{|V(G)|})$ is a right-end ordering of G, then $u_i <_\pi u_j$ if and only if $i < j$. In Figure 1 we illustrate the right-end ordering π of an interval graph G. In Figure 1(b) the right endpoints of the intervals in the intersection model F are drawn bold for better visibility.

2.2 Normal Paths

Our algorithm for constructing a longest path of an interval graph G uses a specific type of paths, namely normal paths. We next define the notion of a normal path of an interval graph G.

Definition 2.1 *Let G be an interval graph, and let π be a right-end ordering of G. The path $P = (v_1, v_2, \ldots, v_k)$ of G is called* normal, *if v_1 is the leftmost vertex of $V(P)$ in π, and for every i, $2 \leq i \leq k$, the vertex v_i is the leftmost vertex of $N(v_{i-1}) \cap \{v_i, v_{i+1}, \ldots, v_k\}$ in π.*

For example, in the interval graph G of Figure 1, the path $P = (u_1, u_4, u_2, u_5, u_3)$ is normal. Note that the notion of normal paths in interval graphs is exactly what is called *straight paths* in [8]. Damaschke [8] presents an algorithm for finding a straight Hamiltonian path in an interval graph (Algorithm 3), proving thus that if an interval graph has a Hamiltonian path, then it also has a straight Hamiltonian path. Also, in [8] a path is called *straight* if it is a straight Hamiltonian path in the subgraph induced by its vertex set.

Now, since straight (resp. normal) paths are defined with respect to a given intersection model (resp. right-end ordering) of the graph, the following observation suffices to obtain the correctness of Lemma 2.2: let G be an interval graph, let F be an intersection model of G, let P be a path of G, and let G' be the subgraph of G induced by $V(P)$. Then we can obtain an intersection model F' of G' by simply deleting from F the intervals which correspond to the vertices of $V(G) \setminus V(G')$. Since P is a Hamiltonian path of G', then from [8] there exists a straight Hamiltonian path P' of G' (with respect to F'). By the construction of F', it follows that P' is a straight path in G (with respect to F) as well. Therefore, the following result holds. Note that, hereafter we use the term normal path instead of straight path.

Lemma 2.2 *Let P be a path of an interval graph G. Then, there exists a normal path P' of G, such that $V(P') = V(P)$.*

3 Interval Graphs and the Longest Path Problem

In this section we present our algorithm, which we call Algorithm LP_Interval, for solving the longest path problem on interval graphs; it consists of three phases and works as follows:

- Phase 1: it takes an interval graph G and constructs the auxiliary interval graph H;
- Phase 2: it computes a longest binormal path $\widehat{P}$ on H using Algorithm LP_on_H;
- Phase 3: it computes a longest path P on G from the path $\widehat{P}$;

The proposed algorithm computes a longest binormal path $\widehat{P}$ of the graph H using dynamic programming techniques and then computes a longest path P of G from the path $\widehat{P}$; note that binormal paths are a special type of paths which we define in Section 3.2. We next describe in detail the three phases of our algorithm and prove properties of the constructed graph H which will be used for proving the correctness of the algorithm.

3.1 The interval graph H

In this section we present Phase 1 of the algorithm: given an interval graph G and a right-end ordering π of G, we construct the interval graph H and a right-end ordering σ of H. To this end, we use the following notations.

Notation 3.1 *Let F be the intersection model of an interval graph G, and let $\pi = (v_1, v_2, \ldots, v_{|V(G)|})$ be the right-end ordering of G which we obtain from F. By I_i we denote the interval which corresponds to the vertex v_i in F, and by $l(I_i)$ and $r(I_i)$ we denote the left and the right endpoint of the interval I_i, respectively.*

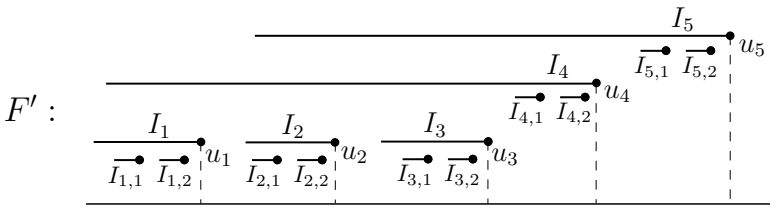

Figure 2: The intersection model F' of the stable-connection graph H, which is obtained from the interval graph G of Figure 1.

▶ **Construction of H and σ:** Let G be an interval graph and let F be the intersection model of G, from which we obtain the right-end ordering $\pi = (v_1, v_2, \ldots, v_{|V(G)|})$ of G. To construct the graph H, for every interval I_i of F we add two disjoint "short" intervals immediately before the right endpoint of I_i.

Formal description: without loss of generality, we may assume that all values $l(I_i)$ and $r(I_i)$ are distinct. Let ε be the smallest distance between two interval endpoints in F. For every interval I_i of F which corresponds to a vertex $v_i \in V(G)$, we add two non-intersecting intervals $I_{i,1} = [r(I_i) - \frac{4\varepsilon}{5}, r(I_i) - \frac{3\varepsilon}{5}]$ and $I_{i,2} = [r(I_i) - \frac{2\varepsilon}{5}, r(I_i) - \frac{\varepsilon}{5}]$. Let $a_{i,1}$ and $a_{i,2}$ be the vertices which correspond to the two new intervals $I_{i,1}$ and $I_{i,2}$, respectively. After processing all intervals I_i, $1 \leq i \leq |V(G)|$, of the intersection model F of G, we obtain an intersection model F' of graph H. Now, set $C = V(G)$ and $A = V(H) \setminus V(G)$.

Thus, H is an interval graph, and the ordering which results from numbering the intervals of F' after sorting them on their right ends is a right-end ordering σ of H. We call the constructed interval graph H the *stable-connection graph* of interval graph G.

In Figure 2, we illustrate the intersection model of the stable-connection graph H of the interval graph G of Figure 1.

Observation 3.1 *For every interval I_i of F, the two new intervals $I_{i,1}$ and $I_{i,2}$ do not intersect with any interval I_k such that $r(I_k) < r(I_i)$. Additionally, the two new intervals intersect with the interval I_i, and with every interval I_ℓ such that $r(I_\ell) > r(I_i)$ and I_ℓ intersects with I_i.*

Hereafter, we will denote by n the number $|V(H)|$ of vertices of the stable-connection graph H and by $\sigma = (u_1, u_2, \ldots, u_n)$ the constructed right-end ordering ordering of H. By construction, the vertex set of H consists of the vertices of $C = V(G)$ and the vertices of A. We will refer to C as the set of *connector vertices* of graph H and to A as the set of *stable vertices* of H; we denote these sets by $C(H)$ and $A(H)$, respectively. Note that $|A(H)| = 2|V(G)|$.

By the construction of the stable-connection graph H, all neighbors of a stable vertex $a \in A(H)$ are connector vertices $c \in C(H)$, such that $a <_\sigma c$. Moreover, observe that all neighbors of a stable vertex form a clique in G and, thus, also in H. Note here that, by the construction of the stable-connection graph H, each stable vertex $a \in A(H)$ is the unique simplicial vertex in the maximal clique induced by the closed neighborhood $N[a]$ in H.

Definition 3.1 *For every connector vertex $u_i \in C(H)$ in $\sigma = \{u_1, u_2, \ldots, u_n\}$, we define by $f(u_i)$ (resp. $h(u_i)$) the smallest (resp. largest) index, such that $f(u_i) < i$ and $u_{f(u_i)}u_i \in E(G)$ (resp. $h(u_i) < i$ and $u_{h(u_i)}u_i \in E(G)$).*

Note that $u_{f(u_i)}$ and $u_{h(u_i)}$ are distinct stable vertices, for every connector vertex u_i.

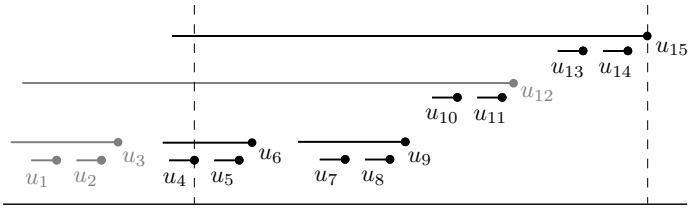

Figure 3: The subgraph $H(4, 15)$ of the stable-connection graph H of Figure 2.

Definition 3.2 *Let H be the stable-connection graph of an interval graph G, and let $\sigma = (u_1, u_2, \ldots, u_n)$ be the right-end ordering of H. For every pair of indices i, j, $1 \leq i \leq j \leq n$, we define the graph $H(i, j)$ to be the subgraph $H[S]$ induced by the set $S = \{u_i, u_{i+1}, \ldots, u_j\} \setminus \{u_k \in C(H) : u_{f(u_k)} <_\sigma u_i\}$.*

The stable-connection graph H of Figure 2 is illustrated in Figure 3, where its vertices (both stable and connector vertices) are numbered according to the right-end ordering $\sigma = (u_1, u_2, \ldots, u_{15})$ of H. The subgraph $H(4, 15)$ is illustrated in Figure 3, where the vertices $V(H(4, 15)) = \{u_4, u_5, u_6, u_7, u_8, u_9, u_{10}, u_{11}, u_{13}, u_{14}, u_{15}\}$ are drawn darker than the others for better visibility.

The following properties hold for every induced subgraph $H(i, j)$, $1 \leq i \leq j \leq n$, and they are used for proving the correctness of Algorithm LP_on_H.

Observation 3.2 *Let u_k be a connector vertex of $H(i, j)$, i.e., $u_k \in C(H(i, j))$. Then, for every vertex $u_\ell \in V(H(i, j))$ such that $u_k <_\sigma u_\ell$ and $u_k u_\ell \in E(H(i, j))$, u_ℓ is also a connector vertex of $H(i, j)$.*

Observation 3.3 *No two stable vertices of $H(i, j)$ are adjacent.*

Lemma 3.1 *Let $P = (v_1, v_2, \ldots, v_k)$ be a normal path of $H(i, j)$. Then:*

(a) For any two stable vertices v_r and v_ℓ in P, v_r appears before v_ℓ in P if and only if $v_r <_\sigma v_\ell$.

(b) For any two connector vertices v_r and v_ℓ in P, if v_ℓ appears before v_r in P and $v_r <_\sigma v_\ell$, then v_r does not see the predecessor $v_{\ell-1}$ of v_ℓ in P.

Proof.

(a) Damaschke in [8] proved that for a normal path $P = (v_1, v_2, \ldots, v_k)$ of an interval graph the following three statements cannot be true simultaneously: vertex v_x appears before v_y in P, $l(I_x) \geq l(I_y)$, and $r(I_x) > r(I_y)$. Since for any two stable vertices v_r and v_ℓ in $H(i, j)$ we have $v_r <_\sigma v_\ell$ if and only if $l(I_r) < r(I_r) < l(I_\ell) < r(I_\ell)$, it follows that $v_r <_\sigma v_\ell$ if and only if v_r appears before v_ℓ in P.

(b) Since $v_r <_\sigma v_\ell$, it follows that $v_\ell \neq v_1$ and, thus, there exists a vertex $v_{\ell-1}$ which appears before v_ℓ in P. Assume that $v_r v_{\ell-1} \in E(H(i, j))$. Since $v_r <_\sigma v_\ell$, and since P is a normal path, v_r should be the next vertex of $v_{\ell-1}$ in P instead of v_ℓ, which is a contradiction. Therefore, $v_r v_{\ell-1} \notin E(H(i, j))$. ∎

ALGORITHM LP_ON_H

Input: a stable-connection graph H, a right-end ordering $\sigma = (u_1, u_2, \ldots, u_n)$ of H.
Output: a longest binormal path of H.

1. `for` $j = 1$ to n
2. `for` $i = j$ downto 1
3. `if` $i = j$ and $u_i \in A(H)$ `then`
4. $\ell(u_i; i, i) \leftarrow 1$; $P(u_i; i, i) \leftarrow (u_i)$;
5. `if` $i \neq j$ `then`
6. `for` every stable vertex $u_r \in A(H)$, $i \leq r \leq j-1$
7. $\ell(u_r; i, j) \leftarrow \ell(u_r; i, j-1)$; $P(u_r; i, j) \leftarrow P(u_r; i, j-1)$; $\{initialization\}$
8. `if` u_j is a stable vertex of $H(i, j)$, i.e., $u_j \in A(H)$, `then`
9. $\ell(u_j; i, j) \leftarrow 1$; $P(u_j; i, j) \leftarrow (u_j)$;
10. `if` u_j is a connector vertex of $H(i, j)$, i.e., $u_j \in C(H)$ and $i \leq f(u_j)$, `then`
11. execute process($H(i, j)$);
12. `compute` $max\{\ell(u_k; 1, n) : u_k \in A(H)\}$ and the corresponding path $P(u_k; 1, n)$;

where the procedure `process()` is as follows:

`process(`$H(i, j)$`)`

13. `for` $y = f(u_j) + 1$ to $j - 1$
14. `for` $x = f(u_j)$ to $y - 1$ $\{u_x$ *and* u_y *are adjacent to* $u_j\}$
15. `if` $u_x, u_y \in A(H)$ `then`
16. $w_1 \leftarrow \ell(u_x; i, j-1)$; $P_1 \leftarrow P(u_x; i, j-1)$;
17. $w_2 \leftarrow \ell(u_y; x+1, j-1)$; $P_2 \leftarrow P(u_y; x+1, j-1)$;
18. `if` $w_1 + w_2 + 1 > \ell(u_y; i, j)$ `then`
19. $\ell(u_y; i, j) \leftarrow w_1 + w_2 + 1$; $P(u_y; i, j) \leftarrow (P_1, u_j, P_2)$;
20. `return` the value $\ell(u_k; i, j)$ and the path $P(u_k; i, j)$, $\forall\ u_k \in A(H(f(u_j) + 1, j - 1))$;

Figure 4: The algorithm for finding a longest binormal path of H.

3.2 Finding a longest binormal path on H

In this section we present Phase 2 of Algorithm LP_Interval. Let G be an interval graph and let H be the stable-connection graph of G constructed in Phase 1. We next present Algorithm LP_on_H, which computes a longest binormal path of the graph H; let us first define binormal paths and give some notations necessary for the description of the algorithm.

Definition 3.3 *Let H be a stable-connection graph, and let P be a path of $H(i, j)$, $1 \leq i \leq j \leq n$. The path P is called* binormal *if P is a normal path of $H(i, j)$, both endpoints of P are stable vertices, and no two connector vertices are consecutive in P.*

Notation 3.2 *Let H be a stable-connection graph, and let $\sigma = (u_1, u_2, \ldots, u_n)$ be the right-end ordering of H. For every stable vertex $u_k \in A(H(i, j))$, we denote by $P(u_k; i, j)$ a longest binormal path of $H(i, j)$ with u_k as its right endpoint, and by $\ell(u_k; i, j)$ the length of $P(u_k; i, j)$.*

Since any binormal path is a normal path, Lemma 3.1 also holds for binormal paths. Moreover, since $P(u_k; i, j)$ is a binormal path, from Lemma 3.1(a) we obtain that its right endpoint u_k is also the rightmost stable vertex of P in σ.

Algorithm LP_on_H, which is presented in Figure 4, computes for every induced subgraph $H(i, j)$ and for every stable vertex $u_k \in A(H(i, j))$, the length $\ell(u_k; i, j)$ and the corresponding

ALGORITHM LP_INTERVAL

Input: an interval graph G and a right-end ordering π of G.
Output: a longest path P of G.

1. Construct the stable-connection graph H of G and the right-end ordering σ of H;
   let $V(H) = C \cup A$, where $C = V(G)$ and A are the sets of connector and stable vertices of H, respectively;
2. Compute a longest binormal path $\widehat{P}$ of H, using Algorithm LP_on_H;
   let $\widehat{P} = (v_1, v_2, \ldots, v_{2k}, v_{2k+1})$, where $v_{2i} \in C$, $1 \leq i \leq k$, and $v_{2i+1} \in A$, $0 \leq i \leq k$;
3. Compute a longest path $P = (v_2, v_4, \ldots, v_{2k})$ of G, by deleting all stable vertices $\{v_1, v_3, \ldots, v_{2k+1}\}$ from the longest binormal path $\widehat{P}$ of H;

Figure 5: The algorithm for solving the longest path problem on an interval graph G.

path $P(u_k; i, j)$. Since $H(1, n) = H$, it follows that the maximum among the values $\ell(u_k; 1, n)$, where $u_k \in A(H)$, is the length of a longest binormal path $P(u_k; 1, n)$ of H.

3.3 Finding a longest path on G

During Phase 3 of our Algorithm LP_Interval, we compute a path P from the longest binormal path $\widehat{P}$ of H, computed by Algorithm LP_on_H, by simply deleting all stable vertices of $\widehat{P}$. In Section 4.2 we prove that the resulting path P is a longest path of the interval graph G.

In Figure 5, we present our Algorithm LP_Interval for solving the longest path problem on an interval graph G; note that Steps 1, 2, and 3 of the algorithm correspond to the presented Phases 1, 2, and 3, respectively.

4 Correctness and Time Complexity

In this section we prove the correctness of our algorithm and analyze its time complexity. More specifically, in Section 4.1 we show that Algorithm LP_on_H computes a longest binormal path $\widehat{P}$ of the graph H, while in Section 4.2 we show that the length of a longest binormal path $\widehat{P}$ of H is equal to $2k + 1$, where k is the length of a longest path of G. Finally, we show that the path P constructed at Step 3 of Algorithm LP_Interval is a longest path of G.

4.1 Correctness of Algorithm LP_on_H

We next prove that Algorithm LP_on_H correctly computes a longest binormal path of the graph H. The following lemmas appear useful in the proof of the algorithm's correctness.

Lemma 4.1 *Let H be a stable-connection graph, and let $\sigma = (u_1, u_2, \ldots, u_n)$ be the right-end ordering of H. Let P be a longest binormal path of $H(i, j)$ with u_y as its right endpoint, let u_k be the rightmost connector vertex of $H(i, j)$ in σ, and let $u_{f(u_k)+1} \leq_\sigma u_y \leq_\sigma u_{h(u_k)}$. Then, there exists a longest binormal path P' of $H(i, j)$ with u_y as its right endpoint, which contains the connector vertex u_k.*

Proof. Let P be a longest binormal path of $H(i, j)$ with u_y as its right endpoint, which does not contain the connector vertex u_k. Assume that $P = (u_y)$. Since u_k is a connector vertex of $H(i, j)$

and $u_{f(u_k)}$ is a stable vertex of $H(i,j)$, we have that $u_i \leq_\sigma u_{f(u_k)} <_\sigma u_y <_\sigma u_k$. Thus, there exists a binormal path $P_1 = (u_{f(u_k)}, u_k, u_y)$ such that $|P_1| > |P|$. However, this is a contradiction to the assumption that P is a longest binormal path of $H(i,j)$.

Therefore, assume now that $P = (u_p, \ldots, u_q, u_\ell, u_y)$. By assumption, P is a longest binormal path of $H(i,j)$ with u_y as its right endpoint that does not contain the connector vertex u_k. Since the connector vertex u_ℓ sees the stable vertex u_y and, also, since u_k is the rightmost connector vertex of $H(i,j)$ in σ, it follows by Observation 3.2 that $u_{f(u_k)} <_\sigma u_y <_\sigma u_\ell <_\sigma u_k$. Thus, u_k sees the connector vertex u_ℓ. Consider first the case where u_k does not see the stable vertex u_q, i.e., $u_q <_\sigma u_{f(u_k)} <_\sigma u_y <_\sigma u_\ell <_\sigma u_k$. Then, it is easy to see that the connector vertex u_ℓ sees $u_{f(u_k)}$, where $u_{f(u_k)}$ is always a stable vertex, and also, from Lemma 3.1(a) it follows that the vertex $u_{f(u_k)}$ does not belong to the path P. Therefore, there exists a binormal path $P_2 = (u_p, \ldots, u_q, u_\ell, u_{f(u_k)}, u_k, u_y)$ in $H(i,j)$, such that $|P_2| > |P|$. This is a contradiction to our assumption that P is a longest binormal path.

Consider now the case where u_k sees the stable vertex u_q. Then, there exists a path $P' = (u_p, \ldots, u_q, u_k, u_y)$ of $H(i,j)$ with u_y as its right endpoint that contains the connector vertex u_k, such that $|P| = |P'|$; since P is a binormal path, it is easy to see that P' is also a binormal path. Thus, the path P' is a longest binormal path of $H(i,j)$ with u_y as its right endpoint, which contains the connector vertex u_k. ∎

Lemma 4.2 *Let H be a stable-connection graph, and let σ be the right-end ordering of H. Let $P = (P_1, v_\ell, P_2)$ be a binormal path of $H(i,j)$, and let v_ℓ be a connector vertex of $H(i,j)$. Then, P_1 and P_2 are binormal paths of $H(i,j)$.*

Proof. Let $P = (v_1, v_2, \ldots, v_{\ell-1}, v_\ell, v_{\ell+1}, \ldots, v_k)$ be a binormal path of $H(i,j)$. Then, from Definition 2.1, v_1 is the leftmost vertex of $V(P)$ in σ, and for every index r, $2 \leq r \leq k$, the vertex v_r is the leftmost vertex of $N(v_{r-1}) \cap \{v_r, v_{r+1}, \ldots, v_k\}$ in σ. It is easy to see that $P_1 = (v_1, v_2, \ldots, v_{\ell-1})$ is a normal path of $H(i,j)$. Indeed, since $V(P_1) \subset V(P)$, then v_1 is also the leftmost vertex of $V(P_1)$ in σ, and additionally, v_r is the leftmost vertex of $N(v_{r-1}) \cap \{v_r, v_{r+1}, \ldots, v_{\ell-1}\}$ in σ, for every index r, $2 \leq r \leq \ell - 1$. Furthermore, since P is binormal and v_ℓ is a connector vertex, it follows that $v_{\ell-1}$ is a stable vertex and, thus, P_1 is a binormal path of $H(i,j)$ as well.

Consider now the path $P_2 = (v_{\ell+1}, v_{\ell+2}, \ldots, v_k)$ of $H(i,j)$. Since P is a binormal path and v_ℓ is a connector vertex, it follows that $v_{\ell+1}$ is a stable vertex and, thus, $v_{\ell+1} <_\sigma v_\ell$ due to Observation 3.2. We first prove that $v_{\ell+1}$ is the leftmost vertex of $V(P_2)$ in σ. Since P is a binormal path, we obtain from Lemma 3.1(a) that $v_{\ell+1}$ is the leftmost stable vertex of $V(P_2)$ in σ. Moreover, consider a connector vertex v_t of P_2. Then, its predecessor v_{t-1} in P_2 is a stable vertex and, thus, $v_{t-1} <_\sigma v_t$ due to Observation 3.2. Since $v_{\ell+1}$ is the leftmost stable vertex of $V(P_2)$ in σ, we have that $v_{\ell+1} \leq_\sigma v_{t-1}$ and, thus, $v_{\ell+1} <_\sigma v_t$. Therefore, $v_{\ell+1}$ is the leftmost vertex of $V(P_2)$ in σ. Additionally, since P is a binormal path, it is straightforward that for every index r, $\ell + 2 \leq r \leq k$, the vertex v_r is the leftmost vertex of $N(v_{r-1}) \cap \{v_r, v_{r+1}, \ldots, v_k\}$ in σ. Thus, P_2 is a normal path. Finally, since P is binormal and $v_{\ell+1}$ is a stable vertex, P_2 is a binormal path as well. ∎

Lemma 4.3 *Let H be a stable-connection graph, and let $\sigma = (u_1, u_2, \ldots, u_n)$ be the right-end ordering of H. Let P_1 be a binormal path of $H(i, j-1)$ with u_x as its right endpoint, and let P_2 be a binormal path of $H(x+1, j-1)$ with u_y as its right endpoint, such that $V(P_1) \cap V(P_2) = \emptyset$. Suppose that u_j is a connector vertex of H and that $u_i \leq_\sigma u_{f(u_j)} \leq_\sigma u_x$. Then, $P = (P_1, u_j, P_2)$ is a binormal path of $H(i,j)$ with u_y as its right endpoint.*

Proof. Let u_z be the first vertex of P_2. Note that u_j is the rightmost vertex of $H(i,j)$ in σ. Since u_j is a connector vertex of H such that $u_i \leq_\sigma u_{f(u_j)} \leq_\sigma u_x <_\sigma u_j$, and since

$u_{f(u_j)}u_j \in E(G)$, from Lemma 2.1 it follows that u_j sees the right endpoint u_x of P_1. Additionally, since $u_z \in V(H(x+1, j-1))$, we have $u_{f(u_j)} \leq_\sigma u_x <_\sigma u_{x+1} \leq_\sigma u_z <_\sigma u_j$ and, thus, u_j sees u_z. Therefore, since $V(P_1) \cap V(P_2) = \emptyset$, it follows that $P = (P_1, u_j, P_2)$ is a path of H. Additionally, since $H(i, j-1)$ and $H(x+1, j-1)$ are induced subgraphs of $H(i, j)$, it follows that P is a path of $H(i, j)$. Hereafter, in the rest of this proof $P_1 = (v_1, v_2, \ldots, v_{p-1})$, $P_2 = (v_{p+1}, v_{p+2}, \ldots, v_\ell)$, $u_x = v_{p-1}$, $u_y = v_\ell$, and $u_j = v_p$.

We first show that $P = (v_1, v_2, \ldots, v_p, \ldots, v_\ell)$ is a normal path. Since v_1 is the leftmost vertex of $V(P_1)$ in σ, it follows that $v_1 \leq_\sigma u_x$. Furthermore, since $u_j = v_p$ is the rightmost vertex of $H(i, j)$ in σ, and since for every vertex $v_k \in V(P_2)$ it holds $u_x <_\sigma u_{x+1} \leq_\sigma v_k <_\sigma v_p$, it follows that v_1 is the leftmost vertex of $V(P)$ in σ. We next show that for every k, $2 \leq k \leq \ell$, the vertex v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_\ell\}$ in σ.

Consider first the case where $2 \leq k \leq p-1$, i.e., $v_k \in V(P_1)$. Since P_1 is a normal path, v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_{p-1}\}$ in σ. Assume that v_{k-1} is a stable vertex. Then, Lemma 3.1(a) implies that $v_{k-1} <_\sigma u_x$ and, due to Observation 3.3, it follows that $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_\ell\}$ is a set of connector vertices. Since every connector vertex $v_r \in V(P_2)$ is a vertex of $H(x+1, j-1)$, it follows that $v_{k-1} <_\sigma u_{x+1} \leq_\sigma u_{f(v_r)}$ and, thus, $v_r \notin N(v_{k-1})$. Additionally, since v_p is the rightmost vertex of $H(i, j)$ in σ, it follows that $v_k <_\sigma v_p$. Therefore, since v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_{p-1}\}$ in σ, it follows that v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_\ell\}$ in σ. Assume now that v_{k-1} is a connector vertex. Since P_1 is a binormal path, v_k is a stable vertex such that $v_k \leq_\sigma u_x$ and v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_{p-1}\}$ in σ. Since for every r, $p+1 \leq r \leq \ell$, the vertex v_r is in $V(H(x+1, j-1))$, it follows that $v_k \leq_\sigma u_x <_\sigma v_r$. Additionally, $v_k <_\sigma u_{x+1} <_\sigma v_p$. Therefore, v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_\ell\}$ in σ.

Consider now the case where $k = p$. Since P_1 is a normal path and v_{p-1} is a stable vertex, $N(v_{p-1}) \cap \{v_p, v_{p+1}, \ldots, v_\ell\}$ is a set of connector vertices, due to Observation 3.3. Additionally, since every connector vertex $v_r \in V(P_2)$ is a vertex of $H(x+1, j-1)$, it follows that $v_{p-1} <_\sigma u_{x+1} \leq_\sigma u_{f(v_r)}$ and, thus, $v_r \notin N(v_{p-1})$. Therefore, $N(v_{p-1}) \cap \{v_p, v_{p+1}, \ldots, v_\ell\} = \{v_p\}$ and, thus, v_p is the leftmost vertex of $N(v_{p-1}) \cap \{v_p, v_{p+1}, \ldots, v_\ell\}$ in σ. Now, in the case where $p+1 \leq k \leq \ell$, since P_2 is a normal path we have that v_{p+1} is the leftmost vertex of $V(P_2) = \{v_{p+1}, v_{p+2}, \ldots, v_\ell\}$ in σ and, thus, v_{p+1} is the leftmost vertex of $N(v_p) \cap \{v_{p+1}, v_{p+2}, \ldots, v_\ell\}$ in σ; also, it directly follows that for every k, $p+2 \leq k \leq \ell$, v_k is the leftmost vertex of $N(v_{k-1}) \cap \{v_k, v_{k+1}, \ldots, v_\ell\}$ in σ.

Concluding, we have shown that P is a normal path of $H(i, j)$. Additionally, since P_1 and P_2 are binormal paths of $H(i, j)$, the path P has stable vertices as endpoints and no two connector vertices are consecutive in P. Therefore, P is a binormal path of $H(i, j)$ with u_y as its right endpoint. ∎

Next, we prove the correctness of Algorithm LP_on_H. For the purposes of the proof we distinguish the notation we use for the values computed by Algorithm LP_on_H, from the notation we use for the optimum values. In particular, by $\ell(u_y; i, j)$ we denote the value computed by Algorithm LP_on_H for the length of a longest binormal path of $H(i, j)$ which has u_y as its right endpoint and by $P(u_y; i, j)$ the corresponding computed path. On the other hand, by $\mathcal{L}(u_y; i, j)$ we denote the optimum value of the length of a longest binormal path of $H(i, j)$ which has u_y as its right endpoint and by $\mathcal{P}(u_y; i, j)$ the corresponding path.

Lemma 4.4 *Let H be a stable-connection graph, and let σ be the right-end ordering of H. For every induced subgraph $H(i, j)$ of H, $1 \leq i \leq j \leq n$, and for every stable vertex $u_y \in A(H(i, j))$, the value $\ell(u_y; i, j)$ computed by Algorithm LP_on_H is equal to the length $\mathcal{L}(u_y; i, j)$ of a longest binormal path of $H(i, j)$ which has u_y as its right endpoint and, also, the corresponding computed path $P(u_y; i, j)$ is a longest binormal path of $H(i, j)$ which has u_y as its right endpoint.*

Proof. Let P be a longest binormal path of the stable-connection graph $H(i,j)$, which has vertex $u_y \in A(H(i,j))$ as its right endpoint. We distinguish two cases (I and II) concerning the set of connector vertices of $H(i,j)$.

Case I. Consider first the case where $C(H(i,j)) = \emptyset$; the graph $H(i,j)$ consists of a set of stable vertices $A(H(i,j))$, which is an independent set, due to Observation 3.3. Therefore, in this case Algorithm LP_on_H sets $\ell(u_y; i, j) = 1$ for every vertex $u_y \in A(H(i,j))$, which is equal indeed to the length of the longest binormal path $\mathcal{P}(u_y; i, j) = (u_y)$ of $H(i,j)$ which has u_y as its right endpoint. Therefore, the lemma holds for every induced subgraph $H(i,j)$, for which $C(H(i,j)) = \emptyset$.

Case II. Consider now the case where $C(H(i,j)) \neq \emptyset$. Let $C(H) = \{c_1, c_2, \ldots, c_k, \ldots, c_t\}$ be the set of connector vertices of H, where $c_1 <_\sigma c_2 <_\sigma \ldots <_\sigma c_k <_\sigma \ldots <_\sigma c_t$. Let $\sigma = (u_1, u_2, \ldots, u_n)$ be the vertex ordering of H constructed in Phase 1. Recall that, by the construction of H, $n = 3t$, and $A(H) = V(H) \setminus C(H)$ is the set of stable vertices of H.

Let $H(i,j)$ be an induced subgraph of H, and let c_k be the rightmost connector vertex of $H(i,j)$ in σ. The proof of the lemma is done by induction on the index k of the rightmost connector vertex c_k of $H(i,j)$. To this end, in both the induction basis and the induction step, we distinguish three cases concerning the position of the stable vertex u_y in the ordering σ: $u_i \leq_\sigma u_y \leq_\sigma u_{f(c_k)}$, $u_{h(c_k)} <_\sigma u_y \leq_\sigma u_j$, and $u_{f(c_k)+1} \leq_\sigma u_y \leq_\sigma u_{h(c_k)}$. In each of these three cases, we examine first the length $\mathcal{L}(u_y; i, j)$ of a longest binormal path of $H(i,j)$ with u_y as its right endpoint and then we compare this value to the length $\ell(u_y; i, j)$ of the path computed by Algorithm LP_on_H. Moreover, we prove that the path $P(u_y; i, j)$ with length $\ell(u_y; i, j)$ computed by Algorithm LP_on_H is indeed a binormal path with u_y as its right endpoint.

Induction basis. We first show that the lemma holds for $k = 1$; i.e., c_1 is the only connector vertex of $H(i,j)$. We distinguish two cases (A1 and A2) concerning the position of the stable vertex u_y in σ.

Case A1: $u_i \leq_\sigma u_y \leq_\sigma u_{f(c_1)}$ or $u_{h(c_1)} <_\sigma u_y \leq_\sigma u_j$. In this case, it is easy to see that the length $\mathcal{L}(u_y; i, j)$ of a longest binormal path P of $H(i,j)$ with u_y as its right endpoint is equal to 1. Indeed, if $u_y \neq u_{f(c_1)}$, then u_y does not see the unique connector vertex c_1 of $H(i,j)$ and, thus, the longest binormal path with u_y as its right endpoint consists of vertex u_y. Now, in the case where $u_y = u_{f(c_1)}$, the unique connector vertex c_1 sees u_y, however, u_y is the leftmost neighbor of c_1 in σ; thus, from Lemma 3.1(a) and the definition of binormal paths, it follows that c_1 does not belong to any binormal path with u_y as its right endpoint. Therefore, in Case A1, we have proved that $\mathcal{L}(u_y; i, j) = 1$ and $\mathcal{P}(u_y; i, j) = (u_y)$. It is easy to see that, in this case, Algorithm LP_on_H (see lines lines 6-7 for $i \leq y \leq j-1$, and 8-9 for $y = j$) correctly computes $\ell(u_y; i, j) = 1$ and $P(u_y; i, j) = (u_y)$.

Case A2: $u_{f(c_1)+1} \leq_\sigma u_y \leq_\sigma u_{h(c_1)}$. In this case, we have $\mathcal{L}(u_y; i, j) = 3$; recall that, in the induction basis, c_1 is the only connector vertex of $H(i,j)$. Algorithm LP_on_H computes (in the subroutine `process()`, with $u_j = c_1$) for every stable vertex u_x of $H(i,j)$ such that $u_{f(c_1)} \leq_\sigma u_x \leq_\sigma u_{y-1}$, the value $\ell(u_x; i, j-1) + \ell(u_y; x+1, j-1) + 1 = 1 + 1 + 1 = 3$ and sets $\ell(u_y; i, j) = 3$. Additionally, it is easy to see that the path $P(u_y; i, j) = (u_x, c_1, u_y)$, computed by Algorithm LP_on_H in this case, is indeed a longest binormal path of $H(i,j)$ with u_y as its right endpoint.

Induction hypothesis. Let now c_k be a connector vertex of H, such that $k \leq t$. Assume that the lemma holds for every induced subgraph $H(i,j)$ of H, which has c_ℓ as its rightmost connector vertex in σ, where $1 \leq \ell \leq k-1$. That is, we assume that for every such graph $H(i,j)$, the value $\ell(u_y; i, j)$ computed by Algorithm LP_on_H is equal to the length $\mathcal{L}(u_y; i, j)$ of a longest binormal path of $H(i,j)$ with u_y as its right endpoint and, also, that the corresponding computed path $P(u_y; i, j)$ is a longest binormal path of $H(i,j)$ which has u_y as its right endpoint.

Induction step. We will show that for every induced subgraph $H(i,j)$ of H, which has c_k as its rightmost connector vertex in σ, the value $\ell(u_y;i,j)$ computed by Algorithm LP_on_H is equal to the length $\mathcal{L}(u_y;i,j)$ of a longest binormal path $\mathcal{P}(u_y;i,j)$ of $H(i,j)$ with u_y as its right endpoint. We distinguish three cases (B1, B2, and B3) concerning the position of the stable vertex u_y in σ.

Case B1: $u_i \leq_\sigma u_y \leq_\sigma u_{f(c_k)}$. We first show that $\mathcal{L}(u_y;i,j) = \mathcal{L}(u_y;i,h(c_k))$ (note that $u_{h(c_k)}$ is the predecessor of c_k in σ). In particular, we show that no vertex to the right of $u_{h(c_k)}$ in σ belongs to a longest binormal path of $H(i,j)$ with u_y as its right endpoint and, thus, such a longest binormal path $\mathcal{P}(u_y;i,j)$ of $H(i,j)$ is actually a path of $H(i,h(c_k))$. On the one hand, we prove that the connector vertex c_k does not belong to any binormal path of $H(i,j)$ with u_y as its right endpoint. In fact, vertex c_k does not see any stable vertex to the left of u_y in σ; therefore, from Lemma 3.1(a) and the definition of binormal paths, it follows that c_k does not belong to any binormal path of $H(i,j)$ with u_y as its right endpoint. On the other hand, we prove that every vertex u_ℓ of $H(i,j)$, where $c_k <_\sigma u_\ell \leq_\sigma u_j$, does not belong to any binormal path of $H(i,j)$ with u_y as its right endpoint. Indeed, since c_k is the rightmost connector vertex of $H(i,j)$, it follows that every vertex u_ℓ of $H(i,j)$, where $c_k <_\sigma u_\ell \leq_\sigma u_j$, is a stable vertex and, thus, again from Lemma 3.1(a) and the definition of binormal paths, it follows that u_ℓ does not belong to any binormal path of $H(i,j)$ with u_y as its right endpoint. Therefore, we have proved that a longest binormal path $\mathcal{P}(u_y;i,j)$ of $H(i,j)$ with u_y as its right endpoint is actually a path of $H(i,h(c_k))$. Furthermore, since $H(i,h(c_k))$ is an induced subgraph of $H(i,j)$, it follows that the path $\mathcal{P}(u_y;i,j)$ is also a longest binormal path of $H(i,h(c_k))$ with u_y as its right endpoint. Thus, it follows that $\mathcal{L}(u_y;i,j) = \mathcal{L}(u_y;i,h(c_k))$.

We next show that, in this case, Algorithm LP_on_H computes $\ell(u_y;i,j) = \mathcal{L}(u_y;i,h(c_k))$. In fact, we show that $\ell(u_y;i,h(c_k)) = \mathcal{L}(u_y;i,h(c_k))$ holds and, also, that Algorithm LP_on_H computes $\ell(u_y;i,j) = \ell(u_y;i,h(c_k))$. Note first that, since $h(c_k) < j$, Algorithm LP_on_H has already computed the value $\ell(u_y;i,h(c_k))$ at a previous iteration, where j was equal to $h(c_k)$.

We first show that $\ell(u_y;i,h(c_k)) = \mathcal{L}(u_y;i,h(c_k))$, i.e., the computed value $\ell(u_y;i,h(c_k))$ is equal to the length $\mathcal{L}(u_y;i,h(c_k))$ of a longest binormal path of $H(i,h(c_k))$ with u_y as its right endpoint. Indeed, consider first the case where $H(i,h(c_k))$ is a graph for which $C(H(i,h(c_k))) = \emptyset$, i.e., $H(i,h(c_k))$ has only stable vertices. Then, as we have shown in the beginning of this proof (cf. Case I), the computed value $\ell(u_y;i,h(c_k)) = 1$ is equal to the length $\mathcal{L}(u_y;i,h(c_k))$ of a longest binormal path of $H(i,h(c_k))$ with u_y as its right endpoint. Consider now the case where $H(i,h(c_k))$ is a graph for which $C(H(i,h(c_k))) \neq \emptyset$, i.e., $H(i,h(c_k))$ has at least one connector vertex (cf. Case II); let c_ℓ be its rightmost connector vertex in σ. Then, $c_\ell <_\sigma c_k$, since $u_{h(c_k)} <_\sigma c_k$. Therefore, by the induction hypothesis, the value $\ell(u_y;i,h(c_k))$ computed by Algorithm LP_on_H is equal indeed to the length $\mathcal{L}(u_y;i,h(c_k))$ of a longest binormal path of $H(i,h(c_k))$ with u_y as its right endpoint.

We now show that, in Case B1, Algorithm LP_on_H computes $\ell(u_y;i,j) = \ell(u_y;i,h(c_k))$. Consider first the case where u_j is a connector vertex of $H(i,j)$, i.e., $u_j = c_k$. Then, Algorithm LP_on_H computes (in lines 6-7) $\ell(u_y;i,j) = \ell(u_y;i,j-1)$, which is equal to $\ell(u_y;i,h(c_k))$, since in this case $j-1 = h(c_k)$; also, note that in Case B1, $y \leq f(c_k)$ (i.e., $y \leq f(u_j)$) and, thus, the value $\ell(u_y;i,j)$ does not change during the execution of the subroutine `process()`.

Consider finally the case where u_j is a stable vertex; then $j-1 > h(c_k)$. If $j-1 = h(c_k)+1$, then Algorithm LP_on_H computes (in lines 6-7) $\ell(u_y;i,j) = \ell(u_y;i,j-1)$, which is equal to $\ell(u_y;i,h(c_k)+1)$. Moreover, the connector vertex $u_{h(c_k)+1} = c_k$ does not see in Case B1 any stable vertex to the left of u_y in σ; therefore, from Lemma 3.1(a) and the definition of binormal paths, it follows that $u_{h(c_k)+1}$ does not belong to any binormal path of $H(i,j)$ with u_y as its right endpoint. Therefore, the computed value $\ell(u_y;i,j) = \ell(u_y;i,h(c_k)+1)$ is equal to the value $\ell(u_y;i,h(c_k))$, which has been computed at a previous iteration, where $j = h(c_k)$.

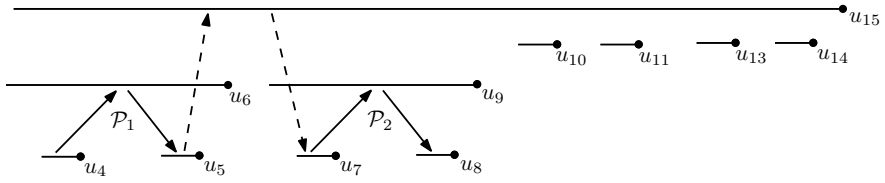

Figure 6: The path $\mathcal{P} = (\mathcal{P}_1, u_{15}, \mathcal{P}_2)$, where $\mathcal{P}_1 = (u_4, u_6, u_5)$ and $\mathcal{P}_2 = (u_7, u_9, u_8)$, is a longest binormal path of the graph $H(i,j) = H(4,15)$ of Figure 3 with u_8 as its right endpoint.

That is, Algorithm LP_on_H computes $\ell(u_y; i, j) = \ell(u_y; i, h(c_k))$. Similarly (to the case where $j - 1 = h(c_k) + 1$), if $j - 1 > h(c_k) + 1$, then Algorithm LP_on_H computes (in lines 6-7) $\ell(u_y; i, j) = \ell(u_y; i, j-1)$, which is again equal to $\ell(u_y; i, h(c_k))$. Therefore, in Case B1, Algorithm LP_on_H computes $\ell(u_y; i, j) = \ell(u_y; i, h(c_k))$ and, also, computes $P(u_y; i, j) = P(u_y; i, h(c_k))$. Then, by the induction hypothesis, this path is also a longest binormal path of $H(i, h(c_k))$ with u_y as its right endpoint. Thus, in Case B1 the lemma holds.

Case B2: $u_{h(c_k)} <_\sigma u_y \leq_\sigma u_j$. Since c_k is the rightmost connector vertex of $H(i,j)$, and since u_y is a stable vertex, it follows that u_y does not see any vertex of $H(i,j)$; furthermore, u_j is a stable vertex. Thus, the longest binormal path of $H(i,j)$ with u_y as its right endpoint consists of vertex u_y, i.e., $\mathcal{L}(u_y; i, j) = 1$. In the case where $u_y = u_j$, one can easily see that Algorithm LP_on_H computes (in lines 8-9) the length $\ell(u_y; i, j) = 1$, and the path $P(u_y; i, j) = (u_y)$, which is clearly a binormal path. Additionally, in the case where $u_{h(c_k)} <_\sigma u_y <_\sigma u_j$, on the one hand Algorithm LP_on_H computes (in lines 6-7) $\ell(u_y; i, j) = \ell(u_y; i, j-1)$ and, on the other hand, the subroutine `process()` is not executed, since u_j is a stable vertex. Thus, in Case B2 the lemma holds.

Case B3: $u_{f(c_k)+1} \leq_\sigma u_y \leq_\sigma u_{h(c_k)}$. In this case, the connector vertex c_k sees u_y. Let $\mathcal{P} = (u_{x'}, \ldots, u_x, c_k, u_{y'}, \ldots, u_y)$ be a longest binormal path of $H(i,j)$ with u_y as its right endpoint, which contains the connector vertex c_k; due to Lemma 4.1, such a path always exists. Let u_x be the predecessor of c_k in the path $\mathcal{P}$; then, $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_y$. Since $\mathcal{P}$ is a binormal path, the vertices $u_{x'}$, u_x, $u_{y'}$, and u_y are all stable vertices. Also, since c_k sees u_y, which is the rightmost stable vertex of $\mathcal{P}$ in σ, all stable vertices of $\mathcal{P}$ belong to the graph $H(i, h(c_k))$. Additionally, since c_k is the rightmost connector vertex of $H(i,j)$ in σ, all connector vertices of $\mathcal{P}$ belong to the graph $H(i, h(c_k) + 1)$. Therefore, all vertices of $\mathcal{P}$ belong to the graph $H(i, h(c_k) + 1)$. Thus, the path $\mathcal{P}$ is a longest binormal path of $H(i, h(c_k) + 1)$ with u_y as its right endpoint, which contains the connector vertex c_k. Therefore, for every graph $H(i,j)$, for which c_k is its rightmost connector vertex in σ and $h(c_k) + 1 \leq j$, we have that $\mathcal{L}(u_y; i, j) = \mathcal{L}(u_y; i, h(c_k) + 1)$. Thus, we will examine only the case where $h(c_k) + 1 = j$, that is, c_k is the rightmost vertex u_j of $H(i,j)$ in σ. An example of this case is illustrated in Figure 6, where $H(i,j) = H(4,15)$ is the stable-connection graph of Figure 3; in this example $u_y = u_8$, $c_k = u_{h(c_k)+1} = u_{15}$, and $u_{f(c_k)} = u_4$.

Next, we examine the length $\mathcal{L}(u_y; i, j)$ of a longest binormal path of $H(i,j)$ with u_y as its right endpoint, in the case where $h(c_k) + 1 = j$. Consider removing the connector vertex c_k from the path $\mathcal{P}$. Then, we obtain the paths $\mathcal{P}_1 = (u_{x'}, \ldots, u_x)$ and $\mathcal{P}_2 = (u_{y'}, \ldots, u_y)$.

Claim 4.1 *Let $\mathcal{P}$, $\mathcal{P}_1$, and $\mathcal{P}_2$ be the paths of Case B3. Then, $\mathcal{P}_1$ is a binormal path of $H(i, j-1)$ which has u_x as its right endpoint, and $\mathcal{P}_2$ is a binormal path of $H(x+1, j-1)$ which has u_y as its right endpoint.*

Proof of Claim 4.1. Since $\mathcal{P}$ is a binormal path of $H(i,j)$, from Lemma 4.2 we obtain that $\mathcal{P}_1$ and $\mathcal{P}_2$ are binormal paths of $H(i,j)$. Since, as we have shown, all vertices of $\mathcal{P}$ belong to $H(i,h(c_k)+1)$, and since $c_k = u_j$ is the rightmost vertex of $H(i,j)$ in σ, it follows that all vertices of $\mathcal{P}_1$ and $\mathcal{P}_2$ belong to the graph $H(i,h(c_k)) = H(i,j-1)$. Therefore, it is straightforward that $\mathcal{P}_1$ is a binormal path of $H(i,j-1)$ which has u_x as its right endpoint.

Next, we show that $\mathcal{P}_2$ is a binormal path of $H(x+1,j-1)$ which has u_y as its right endpoint. Since $\mathcal{P}$ is a binormal path, from Lemma 3.1(a) it follows that for every stable vertex $u_{\ell_2} \in V(\mathcal{P}_2)$, we have $u_x <_\sigma u_{\ell_2} \leq_\sigma u_y \leq_\sigma u_{j-1}$, where $u_{j-1} = u_{h(c_k)}$ is the rightmost vertex of $H(i,j-1)$ in σ, since $u_j = c_k$. Therefore, for every stable vertex $u_{\ell_2} \in V(\mathcal{P}_2)$ it holds $u_{\ell_2} \in A(H(x+1,j-1))$. Additionally, from Lemma 3.1(b) we have that every connector vertex $c_{\ell_2} \in V(\mathcal{P}_2)$ does not see vertex u_x, i.e., $u_x <_\sigma u_{f(c_{\ell_2})} <_\sigma c_{\ell_2} \leq_\sigma u_{j-1}$; thus, $c_{\ell_2} \in C(H(x+1,j-1))$. Summarizing, let H_2 be the induced subgraph of $H(i,j-1)$, with vertex set $V(H_2) = A(H(x+1,j-1)) \cup C(H(x+1,j-1))$; note that the graph H_2 is defined with respect to a stable vertex u_x, where $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_{j-1}$, and also that $H_2 = H(x+1,j-1)$ (for example, in Figure 6, $u_x = u_5$ and $u_j = u_{15}$; thus $H_2 = H(6,14)$, where $V(H(6,14)) = \{u_7, u_8, u_9, u_{10}, u_{11}, u_{13}, u_{14}\}$). Therefore, $\mathcal{P}_2$ is a binormal path of $H(x+1,j-1)$ which has u_y as its right endpoint. ∎

Claim 4.2 *If $\mathcal{P}_1$ is a binormal path of $H(i,j-1)$ which has u_x as its right endpoint, and $\mathcal{P}_2$ is a binormal path of $H(x+1,j-1)$ which has u_y as its right endpoint, then $V(\mathcal{P}_1) \cap V(\mathcal{P}_2) = \emptyset$.*

Proof of Claim 4.2. From the proof of Claim 4.1, recall that H_2 is the induced subgraph of $H(i,j-1)$, with vertex set $V(H_2) = A(H(x+1,j-1)) \cup C(H(x+1,j-1))$; note that the graph H_2 is defined with respect to a stable vertex u_x, where $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_{j-1}$, and also that $H_2 = H(x+1,j-1)$. Therefore, $\mathcal{P}_2$ is a binormal path of H_2 which has u_y as its right endpoint.

Since $\mathcal{P}$ is a binormal path, from Lemma 3.1(a) it follows that for every stable vertex $u_{\ell_1} \in V(\mathcal{P}_1)$, we have $u_i \leq_\sigma u_{x'} \leq_\sigma u_{\ell_1} \leq_\sigma u_x$. Therefore, for every stable vertex $u_{\ell_1} \in V(\mathcal{P}_1)$ it holds $u_{\ell_1} \in A(H(i,x))$. Similarly, since $\mathcal{P}_1$ is a binormal path, u_x is the rightmost stable vertex of $V(\mathcal{P}_1)$ in σ, due to Lemma 3.1(a). Moreover, since $\mathcal{P}_1$ is binormal, every connector vertex $c_{\ell_1} \in V(\mathcal{P}_1)$ sees at least two stable vertices of $\mathcal{P}_1$ and, thus, $u_{f(c_{\ell_1})} <_\sigma u_x$. Actually, since c_{ℓ_1} is a vertex of $\mathcal{P}_1$, and $\mathcal{P}_1$ is a path of $H(i,j)$, it follows that $u_i \leq_\sigma u_{f(c_{\ell_1})} <_\sigma u_x$. Therefore, for every connector vertex $c_{\ell_1} \in V(\mathcal{P}_1)$, we have that $c_{\ell_1} \in C(H(i,j-1)) \setminus \{c_\ell \in C(H(i,j-1)) : u_x \leq_\sigma u_{f(c_\ell)}\} \subseteq C(H(i,j-1)) \setminus C(H(x+1,j-1))$. Summarizing, let H_1 be the induced subgraph of $H(i,j-1)$, with vertex set $V(H_1) = A(H(i,x)) \cup C(H(i,j-1)) \setminus C(H(x+1,j-1))$; note that the graph H_1 is defined with respect to a stable vertex u_x, where $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_{j-1}$ (for example, in Figure 6, $H(i,x) = H(4,5)$, $H(i,j-1) = H(4,14)$, and $H(x+1,j-1) = H(6,14)$; then $A(H(4,5)) = \{u_4, u_5\}$, $C(H(4,14)) = \{u_6, u_9\}$, and $C(H(6,14)) = \{u_9\}$, and thus $V(H_1) = A(H(4,5)) \cup C(H(4,14)) \setminus C(H(6,14)) = \{u_4, u_5, u_6\}$).

Now, it is easy to see that for any stable vertex u_x, where $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_{j-1}$, we have $V(H_1) \cap V(H_2) = \emptyset$. Moreover, $\mathcal{P}_1$ and $\mathcal{P}_2$ belong to the graphs H_1 and H_2, respectively; thus, $V(\mathcal{P}_1) \cap V(\mathcal{P}_2) = \emptyset$. ∎

Since $\mathcal{P} = (\mathcal{P}_1, c_k, \mathcal{P}_2)$ is a longest binormal path of $H(i,j)$ with u_y as its right endpoint, and since the paths $\mathcal{P}_1$ and $\mathcal{P}_2$ belong by Claim 4.2 to two disjoint induced subgraphs (H_1 and H_2, respectively) of $H(i,j)$, it follows that $\mathcal{P}_1$ is a longest binormal path of H_1 with u_x as its right endpoint, and that $\mathcal{P}_2$ is a longest binormal path of H_2 with u_y as its right endpoint (note that $c_k = u_j$ sees every vertex u_z of H_2 and, thus, also of $\mathcal{P}_2$; indeed, since $u_x <_\sigma u_z <_\sigma c_k$ and $u_x c_k \in E(G)$ for every vertex u_z of $\mathcal{P}_2$, from Lemma 2.1 we obtain that $u_z c_k \in E(G)$ for every vertex u_z of $\mathcal{P}_2$). Thus, since $H_2 = H(x+1,j-1)$, we obtain that $|\mathcal{P}_2| = \mathcal{L}(u_y; x+1, j-1)$. We will now show that $|\mathcal{P}_1| = \mathcal{L}(u_x; i, j-1)$. To this end, consider a longest binormal path $\mathcal{P}_0$ of

$H(i, j-1)$ with u_x as its right endpoint. Due to Lemma 3.1(a), u_x is the rightmost stable vertex of $\mathcal{P}_0$ in σ and, thus, all stable vertices of $\mathcal{P}_0$ belong to $A(H_1) = A(H(i, x))$. Furthermore, since $\mathcal{P}_0$ is binormal, every connector vertex c_ℓ of $\mathcal{P}_0$ sees at least two stable vertices of $\mathcal{P}_0$ and, thus, $u_{f(c_\ell)} <_\sigma u_x$, i.e., $c_\ell \in C(H_1) = C(H(i, j-1)) \setminus C(H(x+1, j-1))$. It follows that $V(\mathcal{P}_0) \subseteq V(H_1)$ and, thus, $|\mathcal{P}_0| \leq |\mathcal{P}_1|$. On the other hand, $|\mathcal{P}_1| \leq |\mathcal{P}_0|$, since H_1 is an induced subgraph of $H(i, j-1)$. Thus, $|\mathcal{P}_1| = |\mathcal{P}_0| = \mathcal{L}(u_x; i, j-1)$. Therefore, for the length $|\mathcal{P}| = \mathcal{L}(u_y; i, j)$ of a longest binormal path $\mathcal{P}$ of $H(i, j)$ with u_y as its right endpoint, it follows that $\mathcal{L}(u_y; i, j) = \mathcal{L}(u_x; i, j-1) + \mathcal{L}(u_y; x+1, j-1) + 1$. Also, for the corresponding path $\mathcal{P}(u_y; i, j)$ we have $\mathcal{P}(u_y; i, j) = (\mathcal{P}(u_x; i, j-1), c_k, \mathcal{P}(u_y; x+1, j-1))$.

Hereafter, we examine the results computed by Algorithm LP_on_H in Case B3. Let P be the path of the graph $H(i, j)$ with u_y as its right endpoint, which is computed by Algorithm LP_on_H in Case B3.

Consider first the case where u_j is a connector vertex of $H(i, j)$, i.e., $u_j = c_k$. It is easy to see that the path P constructed by Algorithm LP_on_H (in the subroutine `process()`, line 19) contains the connector vertex c_k. Algorithm LP_on_H computes the length of the path $P = (P_1, c_k, P_2)$, for two paths P_1 and P_2 as follows. The path $P_1 = P(u_x; i, j-1)$ is a path of $H(i, j-1)$ which has u_x as its right endpoint, where u_x is a neighbor of c_k such that $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_y$. The path $P_2 = P(u_y; x+1, j-1)$ is a path of $H(x+1, j-1)$ which has u_y as its right endpoint, where $u_{f(c_k)+1} \leq_\sigma u_y \leq_\sigma u_{h(c_k)}$. Actually, in this case, Algorithm LP_on_H computes (in the subroutine `process()`) the value $w_1 + w_2 + 1 = |P_1| + |P_2| + 1$, for every stable vertex u_x, where $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_y$, and sets $|P|$ to be equal to the maximum among these values. Additionally, Algorithm LP_on_H computes the corresponding path $P = (P_1, c_k, P_2)$. Summarizing, Algorithm LP_on_H computes $\ell(u_y; i, j) = \ell(u_x; i, j-1) + \ell(u_y; x+1, j-1) + 1$ and $P(u_y; i, j) = (P(u_x; i, j-1), c_k, P(u_y; x+1, j-1))$.

Note that the path $P_1 = P(u_x; i, j-1)$ (resp. $P_2 = P(u_y; x+1, j-1)$) has already been computed by Algorithm LP_on_H at a previous iteration. We now show that the computed path $P_1 = P(u_x; i, j-1)$ (resp. $P_2 = P(u_y; x+1, j-1)$) is a longest binormal path $\mathcal{P}(u_x; i, j-1)$ (resp. $\mathcal{P}(u_y; x+1, j-1)$) of $H(i, j-1)$ (resp. of $H(x+1, j-1)$) with u_x (resp. with u_y) as its right endpoint. Indeed, consider first the case where $H(i, j-1)$ (resp. $H(x+1, j-1)$) is a graph for which $C(H(i, j-1)) = \emptyset$ (resp. $C(H(x+1, j-1)) = \emptyset$), i.e., $H(i, j-1)$ (resp. $H(x+1, j-1)$) has only stable vertices. Then, as we have shown in the beginning of this proof (cf. Case I), the computed path $P(u_x; i, j-1)$ (resp. $P(u_y; x+1, j-1)$) is a longest binormal path of $H(i, j-1)$ (resp. of $H(x+1, j-1)$) with u_x (resp. with u_y) as its right endpoint. Consider now the case where $H(i, j-1)$ (resp. $H(x+1, j-1)$) is a graph for which $C(H(i, j-1)) \neq \emptyset$ (resp. $C(H(x+1, j-1)) \neq \emptyset$), i.e., $H(i, j-1)$ (resp. $H(x+1, j-1)$) has at least one connector vertex (cf. Case II); let c_ℓ be its rightmost connector vertex in σ. Then, $c_\ell <_\sigma c_k$, since $u_{j-1} <_\sigma u_j = c_k$. Therefore, by the induction hypothesis, the path $P(u_x; i, j-1)$ (resp. $P(u_y; x+1, j-1)$) computed by Algorithm LP_on_H is indeed a longest binormal path of $H(i, j-1)$ (resp. of $H(x+1, j-1)$) with u_x (resp. with u_y) as its right endpoint. Summarizing, we have proved that $P_1 = P(u_x; i, j-1) = \mathcal{P}(u_x; i, j-1)$ and $P_2 = P(u_y; x+1, j-1) = \mathcal{P}(u_y; x+1, j-1)$ and, thus, $|P_1| = \ell(u_x; i, j-1) = \mathcal{L}(u_x; i, j-1)$ and $|P_2| = \ell(u_y; x+1, j-1) = \mathcal{L}(u_y; x+1, j-1)$.

We now show that the computed path $P = (P_1, u_j, P_2)$ is a longest binormal path $\mathcal{P}(u_y; i, j)$ of $H(i, j)$ with u_y as its right endpoint. Since, as we have proved, P_1 is a binormal path of $H(i, j-1)$ with u_x as its right endpoint, and P_2 is a binormal path of $H(x+1, j-1)$ with u_y as its right endpoint, it follows from Claim 4.2 that $V(P_1) \cap V(P_2) = \emptyset$. Therefore, from Lemma 4.3 we obtain that the computed path $P = (P_1, u_j, P_2)$ is a binormal path as well. Moreover, Algorithm LP_on_H computes (in the subroutine `process()`) for every stable vertex u_x, where $u_{f(c_k)} \leq_\sigma u_x <_\sigma u_y$, the value $\ell(u_x; i, j-1) + \ell(u_y; x+1, j-1) + 1$, and sets $|P|$ to be equal to the

maximum among these values. Thus, the computed path P is a longest binormal path of $H(i,j)$ with u_y as its right endpoint. Summarizing, we have proved that $\ell(u_y;i,j) = \mathcal{L}(u_y;i,j)$ and that $P(u_y;i,j) = \mathcal{P}(u_y;i,j)$, where $u_j = c_k$.

Consider now the case where u_j is a stable vertex of $H(i,j)$. Let c_k be the rightmost connector vertex of $H(i,j)$ in σ; then $h(c_k)+1 \leq j-1$. Since u_j is a stable vertex and also the rightmost vertex of $H(i,j)$, we obtain that u_j does not see any vertex of $H(i,h(c_k)+1)$. In this case, Algorithm LP_on_H correctly computes in lines 6-7 the path $P = P(u_y;i,j) = P(u_y;i,j-1)$. This path is in fact equal to $P(u_y;i,h(c_k)+1)$, since every vertex u_z, $h(c_k)+1 < z \leq j$, to the right of c_k in σ is a stable vertex and, thus, the subroutine `process()` is not executed for any of the graphs $H(i,z)$ (see lines 10-11). Therefore, we have proved that Algorithm LP_on_H computes the path $P = P(u_y;i,j) = P(u_y;i,h(c_k)+1)$, with length $|P| = \ell(u_y;i,j) = \ell(u_y;i,h(c_k)+1)$. Algorithm LP_on_H has already computed the value $\ell(u_y;i,h(c_k)+1)$ at a previous iteration where j was equal to $h(c_k)+1$ (i.e., $u_j = c_k$); moreover, as we have proved in the previous paragraph, in the iteration where $j = h(c_k)+1$ (i.e., in the case where $u_j = c_k$), the computed path $P(u_y;i,h(c_k)+1)$ is optimal, i.e., $P(u_y;i,h(c_k)+1) = \mathcal{P}(u_y;i,h(c_k)+1)$. Therefore, in this case $P = P(u_y;i,j) = \mathcal{P}(u_y;i,h(c_k)+1)$ and $|P| = \ell(u_y;i,j) = \mathcal{L}(u_y;i,h(c_k)+1)$.

Concluding, in both cases where u_j is a connector or a stable vertex of $H(i,j)$, the path $P = P(u_y;i,j)$ of $H(i,j)$ which has u_y as its right endpoint computed by Algorithm LP_on_H is a longest binormal path $\mathcal{P}(u_y;i,j)$ of $H(i,j)$ which has u_y as its right endpoint, and $|P| = \ell(u_y;i,j) = \mathcal{L}(u_y;i,j)$. Thus, the lemma holds in Case B3 as well. ∎

Due to Lemma 4.4, and since the output of Algorithm LP_on_H is the maximum among the lengths $\ell(u_y;1,n)$, $u_y \in A(H(1,n))$, along with the corresponding path, it follows that Algorithm LP_on_H computes a longest binormal path of $H(1,n)$ with right endpoint a vertex $u_y \in A(H(1,n))$. Thus, since $H(1,n) = H$, we obtain the following result.

Lemma 4.5 *Let G be an interval graph. Algorithm LP_on_H computes a longest binormal path of the stable-connection graph H of the graph G.*

4.2 Correctness of Algorithm LP_Interval

We next show that Algorithm LP_Interval correctly computes a longest path of an interval graph G. We first prove the following result.

Lemma 4.6 *Let H be the stable-connection graph of an interval graph G. Then, for any longest path P of G there exists a longest binormal path P' of H, such that $|P'| = 2|P|+1$ and vice versa.*

Proof. Let σ be the right-end ordering of H, constructed in Phase 1.

($\Longrightarrow$) Let $P = (v_1, v_2, \ldots, v_k)$ be a longest path of G, i.e., $|P| = k$. We will show that there exists a binormal path P' of H such that $|P'| = 2k+1$. Since G is an induced subgraph of H, the path P of G is a path of H as well. We construct a path $\widehat{P}$ of H from P, by adding to P the appropriate stable vertices, using the following procedure. Initially, set $\widehat{P} = P$ and for every subpath (v_i, v_{i+1}) of the path $\widehat{P}$, $1 \leq i \leq k-1$, do the following: consider first the case where $v_i <_\sigma v_{i+1}$; then, by the construction of H, v_{i+1} is adjacent to both stable vertices $a_{i,1}$ and $a_{i,2}$ associated with the connector vertex v_i. If $a_{i,1}$ has not already been added to $\widehat{P}$, then replace the subpath (v_i, v_{i+1}) by the path $(v_i, a_{i,1}, v_{i+1})$; otherwise, replace the subpath (v_i, v_{i+1}) by the path $(v_i, a_{i,2}, v_{i+1})$. Similarly, in the case where $v_{i+1} <_\sigma v_i$, replace the subpath (v_i, v_{i+1}) by the path $(v_i, a_{i+1,1}, v_{i+1})$ or $(v_i, a_{i+1,2}, v_{i+1})$, respectively. Finally, consider the endpoint v_1 (resp. v_k) of $\widehat{P}$. If $a_{1,1}$ (resp. $a_{k,1}$) has not already been added to $\widehat{P}$, then add $a_{1,1}$ (resp. $a_{k,1}$) as the first (resp. last) vertex of $\widehat{P}$; otherwise, add $a_{1,2}$ (resp. $a_{k,2}$) as the first (resp. last) vertex of $\widehat{P}$.

By the construction of $\widehat{P}$ it is easy to see that for every connector vertex v of P we add two stable vertices as neighbors of v in $\widehat{P}$, and since in H there are exactly two stable vertices associated with every connector vertex v, it follows that every stable vertex of H appears at most once in $\widehat{P}$. Furthermore, since we add in total $k+1$ stable vertices to P, where $|P| = k$, it follows that $|\widehat{P}| = 2k+1$. Denote now by P' a normal path of H such that $V(P') = V(\widehat{P})$. Such a path exists, due to Lemma 2.2. Due to the above construction, the path $\widehat{P}$ consists of $k+1$ stable vertices and k connector vertices. Thus, since no two stable vertices are adjacent in H due to Observation 3.3, and since P' is a normal path of H, it follows that P' is a binormal path of H. Thus, for any longest path P of G there exists a binormal path P' of H, such that $|P'| = 2|P| + 1$.

($\Longleftarrow$) Consider now a longest binormal path $P' = (v_1, v_2, \ldots, v_\ell)$ of H. Since P' is binormal, it follows that $\ell = 2k+1$, and that P' has k connector vertices and $k+1$ stable vertices, for some $k \geq 1$. We construct a path P by deleting all stable vertices from the path P' of H. By the construction of H, all neighbors of a stable vertex a are connector vertices and form a clique in G; thus, for every subpath (v, a, v') of P', v is adjacent to v' in G. It follows that P is a path of G. Since we removed all the $k+1$ stable vertices of P', it follows that $|P| = k$, i.e., $|P'| = 2|P| + 1$. ∎

Let $\widehat{P}$ be the longest binormal path of H computed in Step 2 of Algorithm LP_Interval, using Algorithm LP_on_H. Then, in Step 3 Algorithm LP_Interval computes the path P by deleting all stable vertices from $\widehat{P}$. By the construction of H, all neighbors of a stable vertex a are connector vertices and form a clique in G; thus, for every subpath (v, a, v') of $\widehat{P}$, v is adjacent to v' in G. It follows that P is a path of G. Moreover, since $\widehat{P}$ is binormal, it has k connector vertices and $k+1$ stable vertices, i.e., $|\widehat{P}| = 2k+1$, where $k \geq 1$. Thus, since we have removed all $k+1$ stable vertices of $\widehat{P}$, it follows that $|P| = k$ and, thus, P is a longest path of G due to Lemma 4.6. Therefore, we have proved the following result.

Theorem 4.1 *Algorithm LP_Interval computes a longest path of an interval graph G.*

4.3 Time Complexity

Let G be an interval graph on $|V(G)| = n$ vertices and $|E(G)| = m$ edges. It has been shown that we can obtain the right-end ordering π of G, which results from numbering the intervals after sorting them on their right ends, in $O(n+m)$ time [1, 21].

First, we show that Step 1 of Algorithm LP_Interval, which constructs the stable-connection graph H of the graph G, takes $O(n^2)$ time. Indeed, for every connector vertex u_i, $1 \leq i \leq n$, we can add two stable vertices in $V(H)$ in $O(1)$ time and we can compute the specific neighborhood of u_i in $O(n)$ time.

Step 2 of Algorithm LP_Interval includes the execution of Algorithm LP_on_H. The subroutine `process()` takes $O(n^2)$ time, due to the $O(n^2)$ pairs of the neighbors u_x and u_y of the connector vertex u_j in the graph $H(i,j)$. Additionally, the subroutine `process()` is executed at most once for each subgraph $H(i,j)$ of H, $1 \leq i \leq j \leq n$, i.e., it is executed $O(n^2)$ times. Thus, Algorithm LP_on_H takes $O(n^4)$ time.

Step 3 of Algorithm LP_Interval can be executed in $O(n)$ time since we simply traverse the vertices of the path $\widehat{P}$, constructed by Algorithm LP_on_H, and delete every stable vertex.

Therefore, we obtain the following result concerning the time complexity of the algorithm.

Theorem 4.2 *A longest path of an interval graph can be computed in $O(n^4)$ time.*

In order to compute the length of a longest path, we need to store one value $\ell(u_y; i, j)$ for every induced subgraph $H(i,j)$ and for every stable vertex u_y of $H(i,j)$. Thus, since there are in total $O(n^2)$ such subgraphs $H(i,j)$, $1 \leq i \leq j \leq n$, and since each one has at most $O(n)$ stable

vertices, we can compute the length of a longest path in $O(n^3)$ space. Furthermore, in order to compute and report a longest path, instead of its length only, for every one of the $O(n^3)$ computed values $\ell(u_y; i, j)$ we have to store a triple of values for the corresponding path $P(u_y; i, j)$, i.e., $(P(u_x; i, j-1), u_j, P(u_y; x+1, j-1))$ (see line 19 of Algorithm LP_on_H); thus, we can compute a longest path in $O(n^3)$ space. Therefore, the space complexity of Algorithm LP_Interval is $O(n^3)$.

5 Concluding Remarks

In this paper we presented a polynomial-time algorithm for solving the longest path problem on interval graphs, which runs in $O(n^4)$ time and, thus, provided a solution to the open problem stated by Uehara and Uno in [23] asking for the complexity status of the longest path problem on interval graphs. It would be interesting to see whether the ideas presented in this paper can be applied to find a polynomial solution to the longest path problem on convex and biconvex graphs, the complexities of which still remain open [23].

References

[1] S.R. Arikati and C. Pandu Rangan, Linear algorithm for optimal path cover problem on interval graphs, *Inform. Proc. Lett.* **35** (1990) 149–153.

[2] K. Asdre and S.D. Nikolopoulos, The 1-fixed-endpoint path cover problem is polynomial on interval graphs, *Algorithmica* (to appear).

[3] A.A. Bertossi, Finding Hamiltonian circuits in proper interval graphs, *Inform. Proc. Lett.* **17** (1983) 97–101.

[4] R. Bulterman, F. van der Sommen, G. Zwaan, T. Verhoeff, A. van Gasteren, and W. Feijen, On computing a longest path in a tree, *Inform. Proc. Lett.* **81** (2002) 93–96.

[5] M.S. Chang, S.L. Peng, and J.L. Liaw, Deferred-query: An efficient approach for some problems on interval graphs, *Networks* **34** (1999) 1–10.

[6] P. Damaschke, J.S. Deogun, D. Kratsch, and G. Steiner, Finding Hamiltonian paths in cocomparability graphs using the bump number algorithm, *Order* **8** (1992) 383–391.

[7] P. Damaschke, The Hamiltonian circuit problem for circle graphs is NP-complete, *Inform. Proc. Lett.* **32** (1989) 1–2.

[8] P. Damaschke, Paths in interval graphs and circular arc graphs. *Discrete Math.* **112** (1993) 49–64.

[9] T. Feder and R. Motwani, Finding large cycles in Hamiltonian graphs, *Proc. of the 16th annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, ACM (2005) 166–175.

[10] H.N. Gabow, Finding paths and cycles of superpolylogarithmic length, *Proc. of the 36th annual ACM Symp. on Theory of Computing (STOC)*, ACM (2004) 407–416.

[11] H.N. Gabow and S. Nie, Finding long paths, cycles and circuits, *Proc. of the 19th annual International Symp. on Algorithms and Computation (ISAAC)*, LNCS **5369** (2008) 752–763.

[12] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, W.H. Freeman, New York, 1979.

[13] M.R. Garey, D.S. Johnson, and R.E. Tarjan, The planar Hamiltonian circuit problem is NP-complete, *SIAM J. Computing* **5** (1976) 704–714.

[14] P.W. Goldberg, M.C. Golumbic, H. Kaplan, and R. Shamir, Four strikes against physical mapping of DNA, *Journal of Comp. Biology* **2** (1995) 139–152.

[15] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs* (Annals of Discrete Mathematics, Vol. 57), North-Holland Publishing Co., Amsterdam, The Netherlands, 2004.

[16] A. Itai, C.H. Papadimitriou, and J.L. Szwarcfiter, Hamiltonian paths in grid graphs, *SIAM J. Computing* **11** (1982) 676–686.

[17] D. Karger, R. Motwani, and G.D.S. Ramkumar, On approximating the longest path in a graph, *Algorithmica* **18** (1997) 82–98.

[18] J.M. Keil, Finding Hamiltonian circuits in interval graphs, *Inform. Proc. Lett.* **20** (1985) 201–206.

[19] H. Müller, Hamiltonian circuits in chordal bipartite graphs, *Discrete Math.* **156** (1996) 291–298.

[20] G. Narasimhan, A note on the Hamiltonian circuit problem on directed path graphs, *Inform. Proc. Lett.* **32** (1989) 167–170.

[21] G. Ramalingam and C. Pandu Rangan, A unified approach to domination problems on interval graphs, *Inform. Proc. Lett.* **27** (1988) 271–274.

[22] Y. Takahara, S. Teramoto, and R. Uehara, Longest path problems on ptolemaic graphs, *IEICE Trans. Inf. and Syst.* **91-D** (2008) 170–177.

[23] R. Uehara and Y. Uno, Efficient algorithms for the longest path problem, *Proc. of the 15th annual International Symp. on Algorithms and Computation (ISAAC)*, LNCS **3341** (2004) 871–883.

[24] R. Uehara and G. Valiente, Linear structure of bipartite permutation graphs and the longest path problem, *Inform. Proc. Lett.* **103** (2007) 71–77.

[25] S. Vishwanathan, An approximation algorithm for finding a long path in Hamiltonian graphs, *Proc. of the 11th annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, ACM (2000) 680–685.

[26] Z. Zhang and H. Li, Algorithms for long paths in graphs, *Theoret. Comput. Sci.* **377** (2007) 25–34.