Computing K-Terminal Reliability of Circular-Arc Graphs

Chien-Min CHEN†(a), Member and Min-Sheng LIN†, Nonmember

SUMMARY Let G be a graph and K be a set of target vertices of G. Assume that all vertices of G, except the vertices in K, may fail with given probabilities. The K-terminal reliability of G is the probability that all vertices in K are mutually connected. This reliability problem is known to be #P-complete for general graphs. This work develops the first polynomial-time algorithm for computing the K-terminal reliability of circular-arc graphs.

key words: algorithm, reliability, circular-arc graphs, interval graphs

1. Introduction

The network model that is developed in this work is a graph with reliable edges and unreliable vertices. The failure probabilities of the vertices are assumed to be mutually independent. A subset of vertices K is specified and the vertices in K are called the target vertices of the network. The K-terminal reliability (KTR) is the probability that all vertices in K are connected to each other by a set of working vertices [1], [10], [12]. This network model can be applied to wireless networks in which communication links are perfectly reliable but in which sites may fail with known probabilities.

The problem of computing KTR is #P-complete [14] for general graphs. The same holds true for chordal graphs [1], circle graphs [11], and polygon-circle graphs [11]. However, polynomial-time algorithms exist for interval graphs [1] and trapezoid graphs [12] (also called bounded multi-tolerance graphs [6]). Additionally, linear-time algorithms exist for proper interval graphs [10] and proper circular-arc graphs [11]. Figure 1 presents the inclusion relations between some graph classes and results of the analysis of the problem of computing KTR. A → B means that class A contains class B. The * symbol indicates a main contribution of this paper.

Fig. 1 Inclusion relations between some graph classes and results of the analysis of the problem of computing KTR. A → B means that class A contains class B. The * symbol indicates a main contribution of this paper.

The network model that is developed in this work is a graph with reliable edges and unreliable vertices. The failure probabilities of the vertices are assumed to be mutually independent. A subset of vertices K is specified and the vertices in K are called the target vertices of the network. The K-terminal reliability (KTR) is the probability that all vertices in K are connected to each other by a set of working vertices [1], [10], [12]. This network model can be applied to wireless networks in which communication links are perfectly reliable but in which sites may fail with known probabilities.

The problem of computing KTR is #P-complete [14] for general graphs. The same holds true for chordal graphs [1], circle graphs [11], and polygon-circle graphs [11]. However, polynomial-time algorithms exist for interval graphs [1] and trapezoid graphs [12] (also called bounded multi-tolerance graphs [6]). Additionally, linear-time algorithms exist for proper interval graphs [10] and proper circular-arc graphs [11]. Figure 1 presents the inclusion relations between the aforementioned graphs and summarizes the above results.

This paper extends the problem of computing the KTR of circular-arc graphs, which are a natural generalization of interval graphs and proper circular-arc graphs.

A circular-arc family A is a set of arcs on a circle. A graph G is a circular-arc graph if a circular-arc family A exists such that two vertices in G are adjacent if and only if their corresponding arcs in A intersect. For a circular-arc family A, G(A) denotes the circular-arc graph constructed from A. For example, Fig. 2(a) shows a family of circular arcs A = [r1, r2, ..., rk] that represents the circular-arc graph in Fig. 2(b). Without loss of generality, the studies presented herein are based on the following assumptions; (1) no two arcs of A have a common endpoint; (2) no single arc of A covers the entire circle; (3) the union of the arcs of A covers the entire circle. Clearly, if the circle contains a point that is not contained in any arc of A, then G(A) is an interval graph and the KTR problem can be solved in polynomial-time [1, 12]. Unlike interval graphs, circular-arc graphs are not even perfect graphs because any chordless cycle is a circular-arc graph, including those of odd-length.

2. Preliminaries

This section presents the preliminaries on which the desired algorithm depends. Consider a circular-arc family A. For convenience, this work will consider arcs in A rather than vertices in its corresponding graph G(A). An arc in A is called a reliable arc if its corresponding vertex in G(A) is a target vertex; otherwise, it is called an unreliable arc. For simplicity, if no ambiguity arises, let K denote both the set of reliable arcs in A and the set of target vertices in G(A). Two arcs in A are connected if a path of arcs (a sequence of adjacent arcs) links them on the circle. A connected arcs component in A is a subset of arcs of A of which any pair are connected on the circle.

Copyright © 2016 The Institute of Electronics, Information and Communication Engineers
Definition 1. A subset \( S \subseteq A \setminus \{x, y\} \) is an \( x, y \)-separator in \( A \) for two non-overlapping arcs \( x, y \in A \) if \( x \) and \( y \) are contained in distinct connected arc components in \( A \setminus S \). If no proper subset of \( S \) is an \( x, y \)-separator in \( A \) then \( S \) is called a minimal \( x, y \)-separator in \( A \).

The following well-known lemma provides the necessary and sufficient condition for a minimal \( x, y \)-separator.

**Lemma 1.** [8] Let \( S \) be an \( x, y \)-separator in \( A \). \( S \) is a minimal \( x, y \)-separator in \( A \) if and only if for any arc \( r \) of \( S \), there is a path from \( x \) to \( y \) that intersects \( S \) only in \( r \).

Definition 2. A subset \( S \subseteq A \setminus K \) is a K-separator in \( A \) if \( S \) is an \( x, y \)-separator in \( A \) for some reliable arcs \( x, y \in K \). If no proper subset of \( S \) is a K-separator in \( A \) then \( S \) is called a minimal K-separator in \( A \).

Notably, all arcs of a K-separator in \( A \) are unreliable.

Definition 3. Add a new point between every pair consecutive endpoints of arcs on the circle. These new points are called scanpoints. A scanpoint \( p \) is called reliable if some reliable arc contains \( p \); otherwise it is called unreliable. For instance, in Fig. 2 (a), the black and white points on the circle represent the reliable and unreliable scanpoints, respectively.

Definition 4. The sequence of scanpoints on the circle is split into runs of consecutive reliable scanpoints or unreliable scanpoints, which are called reliable runs or unreliable runs, respectively. For example, Fig. 2 (a) presents three unreliable runs - (1, 2), (3, 4, 5) and (6, 7, 8).

Notably, if only one reliable run (and hence only one unreliable run) exists on the circle, then all reliable arcs in \( K \) must be mutually connected, independently of the failure probabilities of the unreliable arcs. Therefore, without loss of generality, assume that at least two reliable runs (and therefore at least two unreliable runs) exist on the circle.

The following definition of an unreliable scanline is similar to that of a scanline that is used in tree-width and minimum fill-in algorithms [4, 5, 9].

Definition 5. An unreliable scanline \( s \) of \( A \) is a line segment in the circle that connects two unreliable scanpoints \( p_1 \) and \( p_2 \) that are in two distinct unreliable runs. For each unreliable scanline \( s \), define \( A(s) \) as the set of arcs of \( A \) that contain at least one endpoint of \( s \). For example, in Fig. 2 (a), unreliable scanline \( s_{15} \) connects unreliable scanpoints 3 and 6 and \( A(s_{15}) = \{r_6, r_7, r_8\} \).

The following lemma specifies the relationship between minimal K-separators and unreliable scanlines in \( A \).

**Lemma 2.** Given a circular-arc family \( A \), for every minimal K-separator \( S \) in \( A \), there exists an unreliable scanline \( s \) in \( A \) such that \( S = A(s) \).

**Proof.** The proof is similar to that in an earlier work [5]. Let \( S \) be a minimal K-separator in \( A \). According to Definition 2, there exist two reliable arcs \( x \) and \( y \) such that \( S \) is a minimal \( x, y \)-separator in \( A \) and \( S \) contains only unreliable arcs. Let \( c_x \) and \( c_y \) be the connected arc components of \( A \setminus S \) that contain \( x \) and \( y \), respectively. Then, “generalized arcs” \( A(c_x) \) and \( A(c_y) \) arise as the union of all arcs in \( c_x \) and \( c_y \), respectively. Clearly, \( A(c_x) \) and \( A(c_y) \) have an empty intersection. Define the head (tail) of an arc as the starting (ending) point of the arc when the circle is traversed clockwise. Let \( A(c_x) \rightarrow A(c_y) \) be the portion of the circle from the tail of \( A(c_x) \) clockwise to the head of \( A(c_y) \).

Suppose all scanpoints in \( A(c_x) \rightarrow A(c_y) \) but not in any arcs of \( A \setminus S \) are reliable. Because \( S \) contains only unreliable arcs, it contains no arc that intersects \( A(c_x) \rightarrow A(c_y) \). Accordingly, \( A(c_x) \) and \( A(c_y) \) are connected in \( A \setminus S \). This finding contradicts the fact that \( A(c_x) \) and \( A(c_y) \) have an empty intersection in \( A \setminus S \). Thus, at least one unreliable scanpoint, \( p_1 \), is contained in \( A(c_x) \rightarrow A(c_y) \) but not in any arcs of \( A \setminus S \). Similarly, at least one unreliable scanpoint, \( p_2 \), is contained in \( A(c_x) \rightarrow A(c_y) \) but not in any arcs of \( A \setminus S \). Clearly, the reliable arcs \( x \) and \( y \) must lie on the portion of the circle from \( p_2 \) to \( p_1 \) in the clockwise and counterclockwise directions, respectively. Consequently, \( p_1 \) and \( p_2 \) are in distinct unreliable runs. Let \( s \) be the unreliable scanline that connects \( p_1 \) and \( p_2 \). Since no arc of \( A \setminus S \) contains \( p_1 \) or \( p_2 \), \( A(s) \subseteq S \).
However, since every unreliable arc $r$ of $S$ intersects $A(c_i)$ and $A(c_j)$ by Lemma 1, the unreliable arc $r$ contains at least one of the scanpoints $p_1$ and $p_2$. Therefore, $S \subseteq A(s)$. Accordingly, Lemma 2 holds.

Clearly, the number of unreliable scanlines in $\mathcal{A}$ is at most $O(|\mathcal{A}|^2)$. Thus, the following corollary can easily be obtained.

**Corollary 1.** A circular-arc family $\mathcal{A}$ has at most $O(|\mathcal{A}|^2)$ minimal $K$-separators in $\mathcal{A}$.

### 3. Algorithm

To compute the KTR of circular-arc graphs, this section presents the use of a dynamic programming algorithm that has been used similarly to compute the 2-terminal reliability for multi-tolerance graphs proposed in our previous work [6]. Let $\Delta$ and $\Omega$ represent the set of all minimal $K$-separators and the set of all unreliable scanlines in $\mathcal{A}$, respectively. For $s \in \Omega$, let $F(s)$ be the event in which all arcs in $A(s)$ fail. Because all elements of $[A(s) | s \in \Omega]$ are $K$-separators and $[A(s) | s \in \Omega]$ contains all minimal $K$-separators of $\mathcal{A}$ by Lemma 2, the KTR of $G(\mathcal{A})$ for a given circular-arc family $\mathcal{A}$, $R_K(\mathcal{A})$, is given by

$$R_K(\mathcal{A}) = 1 - Pr\left(\bigcup_{s \in \Omega} F(s)\right).$$  \hspace{1cm} (1)

The problem of computing Eq. (1) is called the union of products problem (UPP). Ball and Provan [2] proved that the UPP is $\#P$-complete. All exact algorithms for solving $\#P$-complete problems are widely known to have exponential time complexity. However, Ball et al. [3] developed lattice-theoretic techniques to simplify the UPP. The following lemma and propositions will demonstrate that $\Omega$ has a specific partial order $\preceq$ that forms a semi-lattice and exhibits closure and convexity.

Beginning at the first unreliable scanpoint of any unreliable run and moving clockwise on the circle, label all unreliable scanpoints. Let $p_1(s)$ and $p_2(s)$ denote the two unreliable scanpoints that are connected by an unreliable scanline $s$, and without loss of generality, assume that $p_1(s) < p_2(s)$. Note that $p_1(s)$ and $p_2(s)$ are in distinct unreliable runs, according to Definition 5.

**Definition 6.** An unreliable scanline $s_i$ is said to be less than or equal to an unreliable scanline $s_j$, denoted as $s_i \preceq s_j$, if and only if $p_1(s_i) \leq p_1(s_j)$ and $p_2(s_i) \geq p_2(s_j)$. If $s_i \preceq s_j$ but $s_i \neq s_j$, then $s_i$ is said to be less than $s_j$, and is written $s_i < s_j$. For example, in Fig. 2(a), $s_1 < s_9 < s_{21}$ and $s_1 < s_{14} < s_{21}$ hold, but neither $s_9 < s_{14}$ nor $s_{14} < s_9$ holds.

According to the above definition, clearly, $(\Omega, \preceq)$ is a poset. Lemma 3 specifies that $(\Omega, \preceq)$ is also a meet-semilattice, as can be easily confirmed using Definition 6.

**Lemma 3.** $(\Omega, \preceq)$ forms a meet-semilattice, and the meet ($\wedge$) of two unreliable scanlines $s_i$ and $s_j$ such that $p_1(s_i) = s_i \wedge s_j$ such that $p_1(s_i) = \min\{p_1(s_i), p_1(s_j)\}$ and $p_2(s_k) = \max\{p_2(s_i), p_2(s_j)\}$.

Notably, the least element of $\Omega$ is the unreliable scanline $s$ such that $p_1(s) = 1$ and $p_2(s) = z$. With respect to the example in Fig. 2(a), the least element of $\Omega$ is $s_1$ with $p_1(s_1) = 1$ and $p_2(s_1) = 8$ and the meet of $s_{10}$ and $s_{15}$ is $s_{10} \wedge s_{15} = s_9$.

**Proposition 1.** (Convexity) For $s_i, s_j, s_k \in \Omega$, if $s_i \preceq s_j \preceq s_k$, then $A(s_i) \cap A(s_k) \subseteq A(s_j)$.

**Proof.** Because $p_1(s_i) < p_2(s_i)$ for all $s \in \Omega$ and $p_1(s_i) \leq p_1(s_j) \leq p_1(s_k)$ and $p_2(s_i) \leq p_2(s_j) \leq p_2(s_k)$, the unreliable scanpoints $p_1(s_i), p_1(s_j), p_1(s_k), p_2(s_i), p_2(s_j), p_2(s_k)$, and $p_2(s_k)$ are ordered clockwise around the circle. Let $r = A(s_i) \cap A(s_k)$; thus, arc $r$ contains both $p_1(s_i)$ and $p_1(s_k)$ or both $p_2(s_i)$ and $p_2(s_k)$ according to Definition 5. Therefore, arc $r$ also contains $p_1(s_i)$ or $p_2(s_i)$ and $r \in A(s_j)$. Thus, $A(s_i) \cap A(s_k) \subseteq A(s_j)$.

**Proposition 2.** (Closure) For $s_i, s_j \in \Omega, A(s_i \wedge s_j) \subseteq A(s_i) \cup A(s_j)$.

**Proof.** Three cases must be considered.

Case 1: $s_i \preceq s_j$. Because $s_i \wedge s_j = s_i$, the proof is trivial.

Case 2: $s_j \preceq s_i$. The argument is similar to that made in Case 1.

Case 3: Neither $s_i \preceq s_j$ nor $s_j \preceq s_i$. This case splits into two subcases.

Case 3.1: $p_1(s_i) \leq p_1(s_j)$ and $p_2(s_i) \leq p_2(s_j)$. Hence, $p_1(s_i \wedge s_j) = p_1(s_i)$ and $p_2(s_i \wedge s_j) = p_2(s_i)$. Therefore, for each unreliable arc $r \in A(s_i \wedge s_j)$, $r$ contains $p_1(s_i)$ or $p_2(s_i)$, so $r \in A(s_j)$ or $r \in A(s_j)$. Accordingly, $A(s_i \wedge s_j) \subseteq A(s_i) \cup A(s_j)$.

Case 3.2: $p_1(s_i) \geq p_1(s_j)$ and $p_2(s_i) \geq p_2(s_j)$. The argument is similar to that made in Case 3.1. Therefore, this proposition holds in all cases.

**Proposition 3.** $\bigcup_{s_i \in \Omega} F(s_i) = \bigcup_{s_i \in \Omega} F^c(s_i)$, where $F^c(s_i)$ is the complement of event $F(s_i)$.

**Proof.** $(\Rightarrow)$ Assume that $F = \bigcup_{s_i \in \Omega} F(s_i)$. Let $O \subseteq \Omega$ be the set of unreliable scanlines $s_i$ such that event $F(s_i)$ occurs in $F$ and let $s^* = \bigwedge_{s_i \in O} s_i$. Thus, event $F(s^*)$ does not occur for all $s_i < s^*$. Also, $F(s^*)$ occurs for $s_i \in O$, so all arcs in $A(s_i)$ fail. By Proposition 2, $A(s^*) \subseteq A(s_i)$. Therefore, all arcs in $A(s^*)$ fail, so $F(s^*)$ occurs and $s^* \in O$. Consequently,
event $FS(s')$ occurs, implying that $\bigcup_{s_i \in \Omega} FS(s_k)$ occurs.

$(\Leftarrow)$ Assume that $\bigcup_{s_i \in \Omega} FS(s_k)$ occurs. Then, at least one event $FS(s)$ occurs, implying that $F(s)$ occurs. Therefore, $\bigcup_{s_i \in \Omega} F(s)$ occurs. $\square$

**Proposition 4.** For $i \neq j$, $FS(s_i) \cap FS(s_j) = \emptyset$.

**Proof.** The proof is by contradiction. Suppose that both events $FS(s_i)$ and $FS(s_j)$ occur. By the definition of $FS(s)$, both events $F(s_i)$ and $F(s_j)$ occur, but events $F(s_f)’$ and $F(s_j)$ do not occur for all $s_f < s_i$ and for all $s_f < s_j$, respectively. Three cases must be considered.

Case 1: $s_i < s_j$. Since $F(s_f)'$ does not occur for all $s_f < s_i$, $F(s_i)$ does not occur. This result contradicts the occurrence of $F(s_i)$.

Case 2: $s_j < s_i$. The argument is similar to that in Case 1.

Case 3: Neither $s_i < s_j$ nor $s_j < s_i$. Let $s^* = s_i \land s_j$. Since $s^* < s_i$ and $s^* < s_j$, $F(s')$ does not occur when events $FS(s)$ and $FS(s')$ occur. Furthermore, since both $F(s_i)$ and $F(s_j)$ occur, all arcs in $A(s_i) \cup A(s_j)$ fail. By Proposition 2, $A(s^*) \subseteq A(s_i) \cup A(s_j)$, so all arcs in $A(s^*)$ fail. Accordingly, $F(s^*)$ occurs and a contradiction is obtained. $\square$

By Propositions 3 and 4, Eq. (1) can be rewritten as

$$R_K(\mathcal{A}) = 1 - \Pr\left[\bigcup_{s_i \in \Omega} FS(s_k)\right] \quad \text{(by Proposition 3)}$$

$$= 1 - \sum_{s_i \in \Omega} \Pr[FS(s_k)]. \quad \text{(by Proposition 4)}$$

Define the conditional event $E(s_k)$, for all $s_k \in \Omega$, as

$$E(s_k) = \{ \text{for all } s_i < s_k, \text{ event } F(s_i) \text{ does not occur given that event } F(s_k) \text{ occurs} \}.$$

Thus, $Pr[FS(s_k)] = Pr[E(s_k)] \cdot Pr[F(s_k)]$ and the above equation can be expressed as

$$R_K(\mathcal{A}) = 1 - \sum_{s_i \in \Omega} \Pr[E(s_k)] \cdot Pr[F(s_k)].$$

Let $q_r$ be the failure probability of unreliable arc $r$. The computation of $Pr[F(s_k)] = \prod_{r \in A(s_k)} q_r$ in Eq. (2) is trivial.

From Lemma 3 and Propositions 1 and 2, $\langle \Omega, \leq \rangle$ forms a meet-semilattice and exhibits convexity and closure. Accordingly, a dynamic programming method [6], [13] for computing $Pr[E(s_k)]$ in Eq. (2) is given by

$$Pr[E(s_k)] = 1 - \sum_{s_i < s_k} \Pr[E(s_i)] \cdot \prod_{r \in A(s_k) \setminus A(s_i)} q_r.$$  \hspace{1cm} (3)

The KTR of a circular-arc graph can be computed using Eqs. (2) and (3) in $O(n \cdot m^2)$ time, where $m$ is the number of unreliable scanlines in $\mathcal{A}$ and $n$ is the number of arcs in $\mathcal{A}$. Since the number of unreliable scanlines in $\mathcal{A}$ does not exceed $O(n^2)$, the KTR of an $n$-arc circular-arc graph can be computed in $O(n^5)$ time. However, the complexity can be reduced to $O(n^4)$ as follows.

Let $f(s_i, s_j)$ denote the probability that all vertices in $A(s_i) \setminus A(s_j)$ fail. Equation (3) can be rewritten as

$$Pr[E(s_k)] = 1 - \sum_{s_i < s_k} (Pr[E(s_i)] \cdot f(s_i, s_k)). \quad (4)$$

The following proposition presents another dynamic programming method to reduce the complexity of the computation of $f(s_i, s_k)$.

**Proposition 5.** [6] For $s_i, s_j, s_k \in \Omega$ and $s_i < s_j < s_k$,

$$f(s_i, s_k) = f(s_j, s_k) \cdot \prod_{r \in A(s_j) \setminus A(s_i)} q_r.$$

**Proof.** All arcs of $A(s_i) \setminus A(s_k)$ can be grouped into two disjoint sets according to whether they belong to $A(s_j)$. Hence, $A(s_i) \setminus A(s_j) = (A(s_i) \setminus A(s_k)) \setminus (A(s_j) \setminus A(s_k))$. First, by Proposition 1, $A(s_i) \cap A(s_k) \subseteq A(s_j)$, and thus $(A(s_i) \setminus A(s_k)) \setminus (A(s_j) \setminus A(s_k))$ can be simplified to $A(s_j) \setminus A(s_k)$. Next, $(A(s_i) \setminus A(s_k)) \setminus (A(s_j) \setminus A(s_k))$ can be rewritten as $(A(s_i) \setminus A(s_k)) \cap A(s_j)$. Therefore, $A(s_i) \setminus A(s_j) = A(s_j) \setminus A(s_k) \cup (A(s_i) \setminus A(s_k)) \cap A(s_j)$ and the proof follows. $\square$

The bottleneck in the computation of $Pr[E(s_k)]$ using Eq. (4) is the computation of $f(s_i, s_k)$ for all $s_i < s_k$. However, the computation time of $f(s_i, s_k)$ can be reduced by using the “immediate predecessor”.

**Definition 7.** Let $s, s'$ and $s''$ be unreliable scanlines. The $s'$ is said to be an immediate predecessor of $s$ if and only if $s' < s$ and no $s''$ exists such that $s'' < s' < s$. For example, in Fig. 2 (a), the immediate predecessors of $s_{15}$ are $s_9$ and $s_{14}$.

Let $P(s)$ represent the set of all immediate predecessors of unreliable scanline $s$. The following property can be easily confirmed.

**Proposition 6.** $P(s) = \{ s' \in \Omega | p_1(s') = p_1(s) - 1 \text{ and } p_2(s') = p_2(s) \}$

The set $\{s_i | s_i < s_k\}$ in Eq. (4) can be divided into two disjoint parts. The first part consists of the unreliable scanlines that are the immediate predecessors of $s_k$. The second part consists of the unreliable scanlines that are less than some immediate predecessor of $s_k$. Accordingly,

$$\{s_i | s_i < s_k\} = \mathcal{P}(s_k) \cup \bigcup_{s_i \in \mathcal{P}(s_k)} \{s_i | s_i < s_{i'}\}. \hspace{1cm} \text{Hence, } f(s_i, s_k) \text{ in Eq. (4) is computed as follows, according to whether } s_i \text{ belongs to } \mathcal{P}(s_k).$$
The upper part of Eq. (5) is trivially obtained from the definition of $f(s_i, s_k)$ and the lower part of Eq. (5) is obtained from Proposition 5. Based on the above formulations, the formal algorithm for computing the KTR of a circular-arc graph is presented as follows.

The following proposition states that the set $A(s_k) \setminus A(s_k')$, for $s_k, s_k' \in \Omega$ and $s_k' \in P(s_k)$, in line 23 of Algorithm 1 has at most one element.

**Proposition 7.** For unreliable scanline $s', s \in \Omega$ and $s' \in P(s)$, $|A(s') \setminus A(s)| \leq 1$.

**Proof.** Because $s' \in P(s)$, according to Proposition 6, either $p_1(s') = p_1(s) - 1$ and $p_2(s') = p_2(s)$ or $p_1(s') = p_1(s)$ and $p_2(s') = p_2(s) + 1$.

Case 1: $p_1(s') = p_1(s) - 1$ and $p_2(s') = p_2(s)$. If there exists an unreliable arc $r$ such that $p_1(r) - 1$, the tail of $r$, and $p_1(r)$ are ordered clockwise around the circle, then $A(s') \setminus A(s) = \emptyset$.

**Algorithm 1.** Compute the KTR of a circular-arc graph

Input: A circular-arc family $A$ of $n$ sorted arcs and a set of reliable arcs $K$

Output: $R_k(A)$—the KTR of $G(A)$

1. Step 1: Add a scanpoint between every two consecutive endpoints of arcs in $A$.
2. Step 2: Split the sequence of scanpoints into reliable runs or unreliable runs.
3. Step 3: Starting from the first unreliable scanpoint of an arbitrary unreliable run, label all unreliable scanpoints from 1 to $z$ in the clockwise direction, where $z$ is the number of unreliable scanpoints.
4. Step 4: // compute $\Omega$ and $A(s), P(s)$ for each unreliable scanline $s \in \Omega$ //
5. $\Omega \leftarrow \emptyset$;
6. for $p_1 \leftarrow 1$ to $z$-1 step 1 do begin
7. for $p_2 \leftarrow z$ to $p_1+1$ step -1 do begin
8. if (unreliable scanpoints $p_1$ and $p_2$ are in distinct unreliable runs) then begin
9. $s_k$—the unreliable scanline connecting $p_1$ and $p_2$; $\Omega \leftarrow \Omega \cup \{s\}$;
10. $A(s) \leftarrow \emptyset$; $P(s) \leftarrow \emptyset$;
11. for each arc $r \in A \setminus K$ do if (r contains scanpoints $p_1$ or $p_2$) then $A(s) \leftarrow A(s) \cup \{r\}$;
12. if (there exists $s' \in \Omega$ such that $s'$ connecting $p_1-1$ and $p_2$) then $P(s) \leftarrow P(s) \cup \{s\}$;
13. if (there exists $s' \in \Omega$ such that $s'$ connecting $p_1$ and $p_2+1$) then $P(s) \leftarrow P(s) \cup \{s\}$;
14. end-if
15. end-for
16. end-for
17. $\{s_1, s_2, ..., s_m\} \leftarrow$ topologically sort all unreliable scanlines of $\Omega$;
18. Step 5: // pre-compute all $f(s_i, s_k)$ for $1 \leq k \leq m$ and $s_i < s_k$, according to Eq. (5) //
19. for $k \leftarrow 1$ to $m$ step 1 do begin
20. for each $s_k \in P(s_k)$ do begin
21. $f(s_k, s_k) \leftarrow 1$; // the initial value of upper part of Eq. (5) //
22. for each $s_i$ with $s_i < s_k$ do $f(s_i, s_k) \leftarrow f(s_i, s_k)$; // the initial value of lower part of Eq. (5) //
23. for each $r \in A(s_k) \setminus A(s_i)$ do begin
24. $f(s_k, s_i) \leftarrow f(s_k, s_i) \cdot q_r$; // the upper part of Eq. (5) //
25. for each $s_i$ with $s_i < s_k$ do if $(r \in A(s_i))$ then $f(s_i, s_k) \leftarrow f(s_i, s_k) \cdot q_r$;
26. end-for
27. end-for
28. end-for
29. Step 6: // compute $\Pr[E(s_k)], 1 \leq k \leq m$, and $R_k(A)$ //
30. for $k \leftarrow 1$ to $m$ step 1 do $\Pr[E(s_k)] \leftarrow 1 - \sum_{s_i < s_k} \left( \Pr[E(s_i)] \cdot f(s_i, s_k) \right)$; // Eq. (4) //
31. $R_k(A) \leftarrow 1 - \sum_{s \in \Omega} \left( \Pr[E(s)] \cdot \prod_{r \in A(s)} q_r \right)$; // Eq. (2) //
32. return $R_k(A)$;
end-algorithm
Case 2: \( p_1(s') = p_1(s) \) and \( p_2(s') = p_2(s) + 1 \). If there exists an unreliable arc \( r \) such that scanpoint \( p_2(s) \), the head of \( r \), and scanpoint \( p_2(s) + 1 \) are ordered clockwise around the circle, then \( A(s') \cap A(s) = \{ r \} \); else \( A(s') \cap A(s) = \emptyset \).

In either case, \( |A(s') \cap A(s)| \leq 1 \). □

Theorem 1. Given an \( n \)-vertex circular-arc graph \( G \), the KTR problem on \( G \) can be solved in \( O(n^2) \) time.

Proof. Notably, if a circular-arc family \( \mathcal{A} \) is not given, then it can be constructed in \( O(n^2) \) time [7] from a given \( n \)-vertex circular-arc graph \( G = G(\mathcal{A}) \). Consider Algorithm 1. Step 1 takes \( O(n) \) time because the number of endpoints on the circle is exactly \( 2n \). In Step 2, because checking whether a scanpoint is reliable or unreliable requires \( O(n) \) time, splitting the sequence of scanpoints into reliable runs or unreliable runs takes \( O(n^2) \) time. Clearly, Step 3 takes \( O(n) \) time. Notably, the number of unreliable scanpoints, \( z \), is at most \( O(n) \). In Step 4, computing all \( A(s) \) for \( s \in \Omega \) in line 11 takes \( O(n^2) \) time. Then, computing all \( A(s) \) for \( s \in \Omega \) in lines 12 and 13 takes \( O(n^2) \) time. Because the number of unreliable scanlines, \( |\Omega| = m \), does not exceed \( O(n^2) \) and checking whether \( s' < s \), for any two unreliable scanlines \( s' \) and \( s \), takes constant time, all unreliable scanlines can be topologically ordered in \( O(n^2) \) time in line 17. In Step 5, based on Proposition 6, the for-loop in line 20 is executed at most twice for each \( s_k \). Therefore, all initial values of \( f(s_k, s_k) \) in line 21 and \( f(s_i, s_j) \) in line 22 can be computed in \( O(m) = O(n^2) \) and \( O(m^2) = O(n^4) \) time, respectively. From Proposition 7, the size of \( A(s_k) \cap A(s_k) \) in line 23 does not exceed one, so the body of the for-loop in line 23 is executed at most \( O(m) \) times throughout Step 5. Thus, all values of \( f(s_k, s_k) \) in line 24 and of \( f(s_i, s_j) \) in line 25 can be updated in \( O(m) = O(n^2) \) time and \( O(m^2) = O(n^4) \) time, respectively. After all values of \( f(s_i, s_j), 1 \leq k \leq m \) and \( s_i < s_j \) are pre-computed in Step 5, all values of \( \Pr[E(s_j)] \) in line 30 are computed in \( O(m^2) = O(n^4) \) time in Step 6. Then, the value of \( R_k(\mathcal{A}) \) in line 31 is computed in \( O(mn) = O(n^3) \) time. Therefore, the total time complexity of the algorithm is \( O(n^4) \). □

4. Conclusions

This work presents the first polynomial-time algorithm for computing the \( K \)-terminal reliability of circular-arc graphs, which are a natural generalization of the well-studied class of interval graphs. Extending the results to other “geometrical” classes of graphs, such as circular trapezoid graphs, would be particularly interesting.

Acknowledgments

The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan, for financially supporting this research under Contract No. MOST 104-2221-E-027-018-

References