Tesis de Licenciatura

Problemas en grafos intersección y overlap de arcos alrededor de un círculo

Sergio Fernando Mera
smera@dc.uba.ar

Director: Dr. Guillermo A. Durán
Co-Director: Dr. Min Chih Lin

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Diciembre de 2001

A mis viejos, Ricardo y Susana

Agradecimientos

A Willy Durán, mi director de tesis, por su gran predisposición, su guía y su profundo conocimiento del tema.

A Irene y Pablo, por su cuidadoso análisis de este trabajo y sus comentarios, que ayudaron a dar los últimos retoques a la versión final de esta tesis.

A Hernán, que tuvo la infinita paciencia de escuchar cientos de demostraciones que no cerraban. A Agus, que con su increíble capacidad de "edición"combatió mi cuestionable sentido estético y logró que esta tesis fuera un trabajo presentable. A ambos les quiero agradecer todo lo que aportaron, tanto en lo académico como en lo humano.

A los amigos que me dio la facultad: Ale, Rula, Andran, Nacho, Claudia, Tobe, Manu, Beto, Mariano G., Hernán, Agus, Daniel y algunos más que seguramente debo estar olvidando. Ellos me dieron el empuje necesario para poder seguir adelante a lo largo de todos estos años de estudio.

A mis amigos de Algoritmos I: Diego G., Diego P., Norberto, Javier y Martin, por las discusiones compartidas en pos de una cátedra mejor.

A Juan Paz, que me enseñó a ver la realidad con otros ojos.

A Analía, mi gran amor, por haber podido contar siempre con ella y ser en gran parte responsable de esta tesis.

A Mauro, Marcelo, Darío y Ariel, con los que compartí muchísimas horas de ensayos y pizzas.

A los docentes de esta Facultad, que su inagotable vocación los lleva a seguir peleando por la educación y la ciencia en este país.

A mis viejos, Ricardo y Susana, y a mi hermana Gaby. Quiero agradecerles todas las invaluables oportunidades que me dieron, y la forma en que me apoyaron todos estos años.

A mi abuela Negra, por todos los domingos que pasamos juntos.

A Pelu, por sus valiosas acotaciones e inteligentes comentarios.

Resumen

Los grafos arco-circulares son los grafos intersección de arcos alrededor de un círculo, mientras que los grafos circulares son los grafos intersección de cuerdas dentro de un círculo. Los grafos overlap de arco-circulares (CAO) son una superclase de los grafos circulares que poseen una representación overlap en arcos alrededor del círculo. Esta clase de grafos no ha sido muy estudiada en la literatura, siendo introducidos por primera vez a principios de la década del '90. Algunos problemas NP-Completos tienen solución eficiente en grafos pertenecientes a esta clase.

En esta tesis presentamos un caracterización de los grafos CAO utilizando ideas aportadas por un teorema de Jayme Szwarcfiter [35] para una caracterización de grafos circulares. Demostramos que pidiendo condiciones similares a las que cumplen los grafos circulares sobre sucesivas orientaciones de un grafo, éstas son necesarias y suficientes para asegurar que el grafo es CAO.

Presentamos algunos algoritmos eficientes para la resolución de los problemas de encontrar el mínimo transversal de los cliques y el máximo conjunto de cliques disjuntos en grafos arco-circulares Helly. La idea general para estos algoritmos fue presentada en la tesis doctoral de Guillermo Durán [9] y su implementación de manera eficiente fue sugerida por Min Chih Lin [28]. Estos algoritmos resuelven ambos problemas con complejidad $O(n.log(n))$ mientras que los más eficientes conocidos en la literatura lo hacían en $O(n^2)$ [21].

Índice

Capítulo 1

Introducción

Consideremos una familia finita de conjuntos no vacíos. El grafo intersección de esta familia se obtiene representando cada conjunto por un vértice. Dos vértices están conectados por una arista si y sólo si los correspondientes conjuntos se intersecan.

Los grafos arco-circulares, de aquí en adelante llamados grafos CA, son los grafos intersección de arcos alrededor de un círculo, y fueron introducidos a mediados de la década del '60, siendo Alan Tucker quien aportó los primeros resultados teóricos. Tucker [37] propuso un algoritmo de reconocimiento de complejidad $O(n^3)$, mas adelante, Spinrad [33] simplificó el algoritmo de Tucker para el caso que el grafo dado pueda ser cubierto por dos cliques. Varios años después, Hsu [23] encontró un algoritmo más eficiente (complejidad $O(n.m)$). Uno de los principales problemas abiertos de esta clase es determinar si existe algún algoritmo de reconocimiento lineal en m y n. En [15], Gavril propuso algoritmos de complejidad polinomial para reconocer algunas subclases de los grafos arco-circulares.

Ha sido probado que diversos problemas NP-Completos para la clase general de los grafos tienen resolución polinomial para los grafos arco-circulares. Este es el caso de los problemas de conjunto independiente máximo ([19], [20], [25], [29]), clique máximo ([4],[24]), partición en cliques mínima ([20], [25]) y conjunto dominante mínimo ([8], [25]). En cambio, el número cromático permanece NP-Completo para grafos arco-circulares ([14]) y grafos arco-circulares Helly ([16]) (de ahora en más llamados grafos HCA). El problema de isomorfismo en grafos, de complejidad desconocida para el caso general, es polinomial para los arco-circulares ([23]). Los grafos arco-circulares Helly fueron caracterizados por Gavril [16] de manera que conducen a un algoritmo eficiente para el reconocimiento de esta subclase. Los grafos arco-circulares y sus subclases tienen aplicaciones en genética, control del tránsito, diseño de compiladores, estadística y problemas de almacenamiento.

Los grafos circulares, de aquí en adelante llamados grafos C, son los grafos intersección de cuerdas dentro de un círculo. Fueron introducidos en [12], donde se muestra una aplicación para resolver un problema de reordenamiento de vagones de un tren propuesto por Knuth [27], usando pilas y colas. Se pueden encontrar algoritmos de tiempo polinomial para reconocer esta clase de grafos en [7], [13], [31] y [34]. Los problemas de clique máximo y conjunto independiente máximo en grafos circulares pueden ser resueltos en tiempo polinomial ([17], [24], [30], [32]), pero el problema de coloreo se mantiene NP-Hard [14].

El grafo overlap de una familia finita de conjuntos no vacíos se obtiene representando cada conjunto por un vértice. Dos vértices están conectados por una arista si y sólo si los correspondientes conjuntos se intersecan pero ninguno está incluido en el otro. Los grafos circulares son equivalentes a los

grafos overlap de intervalos (conocidos en la literatura como grafos overlap). Un grafo G es overlap de intervalos si existe un modelo overlap para sus intervalos (de aquí en adelante los llamaremos grafos IO). La prueba de la equivalencia entre ambas clases puede encontrarse en [17].

Los grafos overlap de arcos alrededor del círculo, de aquí en adelante llamados grafos CAO, fueron poco estudiados en la literatura. Existe un trabajo de T. Kashiwabara [26] que describe algoritmos para encontrar un conjunto independiente máximo en un grafo CAO de complejidad $O(n^2)$ y para encontrar un clique máximo de complejidad $O(n^5)$. En la tesis doctoral de Guillermo Durán [9], también hay resultados sobre esta familia de grafos, donde se caracteriza a los grafos CAO no circulares y se presentan propiedades de estos grafos en relación a otras familias de grafos.

El objetivo de esta tesis es continuar la línea de investigación desarrollada por Guillermo Durán en su tesis doctoral [9]. Tomamos algunos de los temas estudiados por Durán y presentamos nuevos resultados sobre los grafos pertenecientes a las clases overlap de arco-circulares y arco-circular Helly.

En este primer capítulo enumeramos los tópicos de esta tesis, un conjunto de definiciones básicas y las convenciones de notación usadas a lo largo de todo el trabajo.

En el capítulo 2, presentamos una caracterización de los grafos CAO utilizando ideas propuestas en una caracterización para grafos overlap de intervalos introducidas por Jayme Szwarcfiter [35]. El problema de caracterizar grafos CAO ha sido mencionado como abierto en [9]. En la primera sección de este capítulo se definen elementos básicos para trabajar sobre representaciones circulares, como órdenes circulares y propiedades asociadas. En la segunda sección se enuncia el teorema en sí, el cual fija las condiciones necesarias y suficientes para que un grafo sea CAO.

El capítulo 3 está destinado a analizar algunos algoritmos sobre grafos arco-circulares Helly. Se presentan dos algoritmos que resuelven los problemas de encontrar el mínimo transversal de los cliques y el máximo conjunto de cliques disjuntos sobre esta clase de grafos. Ambos algoritmos tienen una complejidad de $O(n.log(n))$, si tomamos como datos de entrada una representación en arcos circulares que cumple la propiedad de Helly. De esta forma se mejora la complejidad de los algoritmos presentados en [21], que poseen un orden de $O(n^2)$. La idea general de los algoritmos propuestos está planteada en [9] y las implementaciones para alcanzar el orden $O(n.log(n))$ fueron sugeridas por Min Chih Lin [28].

Por último, en el capítulo 4 se presentan las conclusiones de esta tesis y algunas posibilidades de trabajo futuro.

1.1 Definiciones básicas y notación

Denotamos un grafo G por un par $(V(G), E(G))$, donde $V(G)$ representa un conjunto finito de vértices, y $E(G)$, un conjunto de pares no ordenados de

vértices de G, llamados aristas. Sean $n = |V(G)|$ y $m = |E(G)|$.

Un vértice v es adyacente a otro vértice w en G si $(v, w) \in E(G)$. Decimos que v y w son los extremos de la arista.

Un grafo orientado $\vec{G}$ está formado por un par $(V(\vec{G}), E(\vec{G}))$, donde $V(\vec{G})$ representa un conjunto finito de vértices, y $E(\vec{G})$, un conjunto de pares ordenados de vértices de G, llamados aristas. Un vértice v es adyacente a otro vértice w en $\vec{G}$ si $(v \to w) \in E(\vec{G})$.

El complemento de un grafo G, denotado por $\overline{G}$, es el grafo que tiene el mismo conjunto de vértices de G y tal que dos vértices distintos son adyacentes en $\overline{G}$ si y sólo si no son adyacentes en G.

Un grafo H es un subgrafo de un grafo G si $V(H) \subseteq V(G)$ y $E(H) \subseteq E(G) \cap (V(H) \times V(H))$. Si $V(H) = V(G)$, decimos que H es un subgrafo generador de G. Dado un conjunto de vértices $X \subseteq V(G)$, el subgrafo de G inducido por X es el subgrafo H de G tal que $V(H) = X$ y $E(H)$ es el conjunto de aristas de G que tiene ambos extremos en X.

Dos grafos G y H son isomorfos si existe una biyección entre $V(G)$ y $V(H)$ que conserva las adyacencias. En este caso, notamos $G = H$.

Un camino simple en un grafo G es una secuencia de vértices distintos $P = v_1, ..., v_k$, donde $(v_i, v_{i+1}) \in E(G)$, para $1 \leq i \leq k-1$. Una cuerda en P es una arista que une dos vértices no consecutivos de P. Un circuito en un grafo G es una secuencia de vértices $C = v_1, ..., v_k$, no necesariamente distintos, donde $v_1 = v_k$, y $(v_i, v_{i+1}) \in E(G)$, para $1 \leq i \leq k-1$.

Un ciclo en un grafo G es una secuencia de vértices $C = v_1, ..., v_k, v_{k+1}$, donde $v_1, ..., v_k$ es un camino simple, v_1 es adyacente a v_k, $v_1 = v_{k+1}$ y $k \geq 3$. Una cuerda en C es cualquier cuerda del camino $v_1, ..., v_k$. Si los vértices que unen la cuerda en C están a distancia 2, decimos que la cuerda es corta. Un ciclo es un ciclo inducido si no posee cuerdas. Llamamos C_k al ciclo inducido por k vértices (C_3 es también llamado triángulo).

Un grafo G es conexo si para todo par de vértices distintos v y w de G, existe un camino de v a w.

Un grafo G es completo si cualquier par de vértices distintos de G son adyacentes. Llamamos K_n al grafo completo con n vértices.

Un conjunto de vértices M de un grafo G es un subgrafo completo si el subgrafo inducido por M es completo. Un clique es un subgrafo completo maximal de G.

Sean G_1 y G_2 dos grafos que poseen el mismo conjunto de vértices. La *unión* $G_1 \cup G_2$ es el grafo que posee los mismos vértices que G_1 y G_2, y las aristas de $E(G_1) \cup E(G_2)$.

Un concepto muy usado a lo largo de este trabajo es el de la propiedad de Helly. Una familia de subconjuntos S satisface la propiedad de Helly cuando toda subfamilia de S consistente en subconjuntos que se intersecan de a pares tiene intersección no vacía.

Las definiciones que no fueron dadas aquí pueden encontrarse en [6], [9], [18] ó [22].

Capítulo 2

Grafos overlap de arco-circulares

2.1 Generalidades

Recordaremos aquí las definiciones de las clases de grafos arco-circulares y overlap de arco-circulares, utilizadas con frecuencia a lo largo de este capítulo.

2.1.1 Grafos arco-circulares

Un grafo G es arco-circular si existe un conjunto de arcos φ (que llamamos representación) alrededor de un círculo y una correspondencia 1-1 entre vértices de G y arcos de φ, de manera que dos vértices distintos son adyacentes si y sólo si los arcos correspondientes se intersecan. Es decir, un grafo arco-circular es el grafo intersección de arcos alrededor de un círculo. Supondremos en este trabajo que los arcos son abiertos. Sin pérdida de generalidad, podemos asumir que ningún par de arcos tiene un extremo común y que ningún arco cubre el perímetro total de la circunferencia. La Figura 2.1 muestra un grafo arco-circular y una representación posible para él.

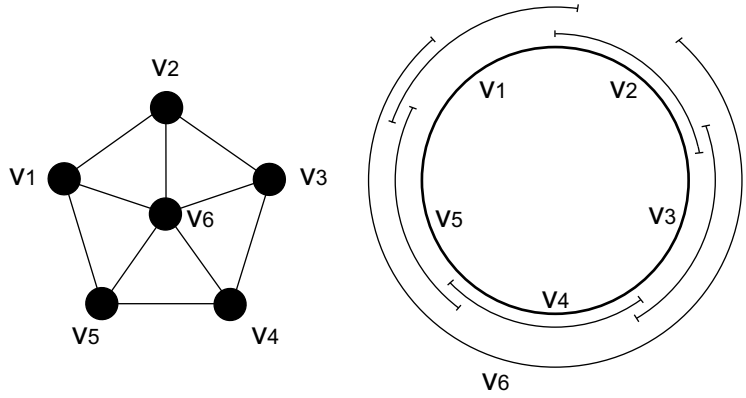


Figura 2.1: Grafo arco-circular y una posible representación.

2.1.2 Grafos overlap de arco-circulares

Como se menciona en la introducción, el grafo overlap de una familia finita de conjuntos no vacíos se obtiene representando cada conjunto por un vértice. Dos vértices estarán conectados por una arista si y sólo si los correspondientes conjuntos se intersecan pero ninguno está incluido en el otro.

Si llevamos este concepto a los grafos arco-circulares, vemos que un grafo G es overlap de arco-circulares si existe un conjunto de arcos φ (que nuevamente llamamos representación) alrededor de un círculo y una correspondencia 1-1 entre vértices de G y arcos de φ, de manera que dos vértices distintos

son adyacentes si y sólo si los arcos correspondientes se solapan. Dos arcos se solapan cuando su intersección es no vacía, pero ningún arco está incluido en el otro. La Figura 2.2 muestra un grafo overlap de arco-circular y una representación posible para él.

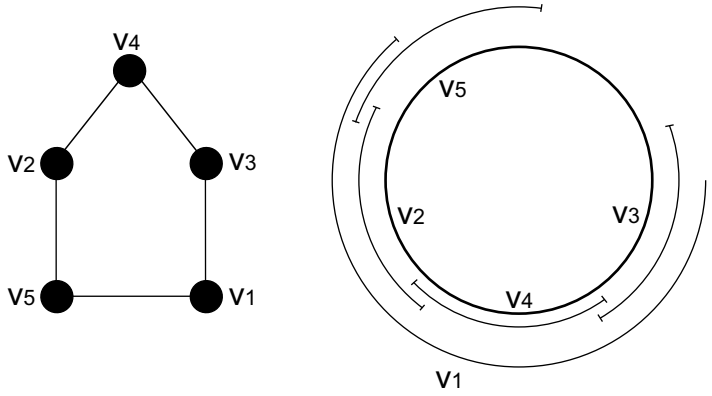


Figura 2.2: Grafo overlap de arco-circulares y una posible representación.

2.2 Órdenes circulares

Las definiciones, lemas y propiedades que presentamos en esta sección son introducidas por primera vez en esta tesis y serán utilizados en la siguientes secciones.

Sea φ una representación arco-circular de un grafo G. Sean i, j, k extremos distintos correspondientes a arcos de φ. Diremos que j está *entre* i y k si partiendo de i, y recorriendo el círculo en sentido horario, j se encuentra antes que k.

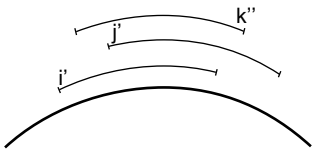


Figura 2.3: En este ejemplo, el extremo $j^{'}$ está entre $i^{'}$ y $k^{''}$.

Sea φ una representación arco-circular de un grafo G. Sean $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ arcos de φ correspondientes a vértices $v_i, v_j \in V(G)$ ($v_i^{''}$ le sigue a $v_i^{'}$ en el sentido horario). Diremos que el arco $[v_i^{'}, v_i^{''}]$ *corta* al arco $[v_j^{'}, v_j^{''}]$ si partiendo de $v_i^{'}$, y recorriendo el círculo en sentido horario, $v_j^{'}$ se encuentra antes que $v_i^{''}$. Notemos que esta propiedad no es simétrica, dado que v_j y v_i se cortan mutuamente sólo cuando ambos cubren todo el círculo.

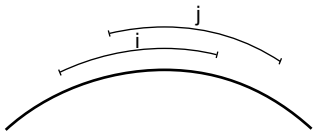


Figura 2.4: El arco $[v_i', v_i'']$ corta al arco $[v_j', v_j'']$.

Sea G un grafo, Q e I una partición disjunta de aristas $\bar{G} = Q \cup I$ de su complemento $\bar{G}$ y R_k un orden lineal $v_{1_k}, ..., v_{n_k}$ de los vértices de G asociado a $v_k \in V(G)$. Sea $v_p = v_{1_k}$. Para simplificar la notación diremos que $v_i < v_j$ bajo un orden R_k cuando R_k asigna a v_i una numeración menor que a v_j. Luego R_k es llamado un *orden circular* asociado a v_k, Q e I cuando:

1. $(v_p, v_k) \notin E(I)$.
2. Si bajo R_k vale $v_j < v_k$ entonces $(v_p, v_j) \notin E(I)$.
3. Bajo R_p vale $v_p < v_k$.
4. Si existen tres vértices tales que bajo R_t vale $v_i < v_t < v_j$, bajo R_i vale $v_j < v_i < v_t$ y $(v_t, v_j) \in E(G)$ entonces $(v_t, v_i) \notin E(Q)$.
5. Cualquier vértice distinto de v_p que cumple todos los items presentados es numerado por el orden R_k con un número menor.
6. Sean R_{p3} y R_{k3} los órdenes que quedan definidos si sólo valen las condiciones 1, 2, 4 y 5. Si estos órdenes así definidos no cumplen la condición 3, luego:
   (a) $v_p < v_{i_{p3}}$ en R_p para todo $v_{i_{p3}}$ tal que bajo R_{p3} quedó numerado entre 1 y el número asignado a v_p.
   (b) $v_k < v_{i_{k3}}$ en R_k para todo $v_{i_{k3}}$ tal que bajo R_{k3} quedó numerado entre 1 y el número asignado a v_k.

Una orientación $\vec{G_{R_k}}$ de G *es inducida* por un orden circular R_k cuando $(v_i \to v_j) \in E(\vec{G_{R_k}}) \Rightarrow i < j$. A fin de simplificar la notación, para representar el orden de los vértices de G bajo el orden circular R_k se notará $[v_1, ..., v_n]_{R_k}$

Sean $R_k = \{v_{1_k}, ..., v_{n_k}\}$, $R_s = \{v_{1_s}, ..., v_{n_s}\}$ dos órdenes circulares asociados a $v_k, v_s \in V(G)$. Diremos que R_k es un *desplazamiento circular* de R_s cuando $\exists j, 1 \leqslant j \leqslant n$, tal que R_k puede definirse en función de R_s de la siguiente manera:

$$v_{i_k} = \begin{cases} v_{(j+i)_s} & \text{si } i + j \leqslant n, \\ v_{(j+i-n)_s} & \text{si no.} \end{cases}$$

Por ejemplo, $[v_2, v_5, v_1, v_7]$ es un desplazamiento circular de $[v_1, v_7, v_2, v_5]$.

Sea φ una representación arco-circular de un grafo G, donde a cada $v_i \in$ V(G) le corresponde un intervalo $[v_i^{'}, v_i^{''}]$ y sea s_{v_j} una función asociada a $v_j \in V(G)$ cuyo dominio es el conjunto de los extremos de los arcos de φ y su imagen es $\mathbb{N}$[1..2n]. Diremos que s_{v_j} es una *linealización canónica* asociada a v_j cuando se comporta de acuerdo a la siguiente construcción:

1. Fijar el cero del círculo en la posición de $v_j^{'}$. El crecimiento de los ángulos se define en sentido horario.
2. Sea el conjunto de todos los extremos $\beta = \{v_1^{'}, v_1^{''}, v_2^{'}, v_2^{''}, ..., v_n^{'}, v_n^{''}\}$
3. Primera etapa:
   (a) Eliminar de β los extremos $v_i^{''}$ tal que su arco asociado $[v_i^{'}, v_i^{''}]$ corta al arco $[v_j^{'}, v_j^{''}]$.
   (b) Partiendo de $v_j^{'}$, recorrer el círculo en sentido horario, numerando en sentido creciente y consecutivo los extremos de los arcos que pertenecen a β a medida que aparecen.
4. Segunda etapa:
   (a) Sea ξ el conjunto de todos los $v_i^{''}$ que se eliminaron de β en la primera etapa.
   (b) Partiendo de $v_j^{'}$, recorrer el círculo en sentido horario y, continuando con la numeración de la primera etapa, numerar en sentido creciente y consecutivo los extremos de los arcos que pertenecen a ξ a medida que aparecen.

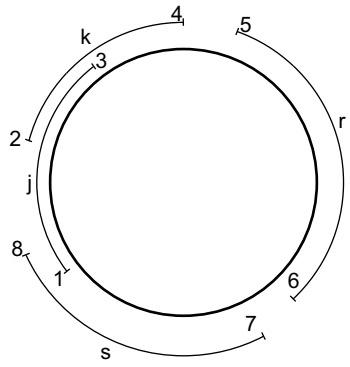


Figura 2.5: Linealización canónica en función del arco $[v_j^{'}, v_j^{''}]$

A fin de simplificar la escritura, cuando valga $s_{v_j}(v_i^{'}) < s_{v_j}(v_k^{'})$ se notará $[s(v_i^{'}) < s(v_k^{'})]_{v_j}$. Con frecuencia se trabajarán con funciones s_{v_i} donde v_i es el primer vértice en función de un orden circular R_j, luego, nuevamente para simplificar la escritura, en lugar de escribir $s_{v_{1_j}}(v_{i_j}^{'})$, se notará $[s(v_i^{'})]_{R_j}$.

Sea φ una representación arco-circular de un grafo. Diremos que R_k es un *orden circular canónico* asociado a $v_k \in V(G)$ cuando define una numeración $v_1, ..., v_n$ sobre los vértices de G tal que existe una linealización canónica $s_{v_{1_k}}$ sobre φ tal que $v_i < v_j \Leftrightarrow [s(v_i^{'}) < s(v_j^{'})]_{R_k}$ y v_{1_k} es elegido de la siguiente forma:

- Sea $\triangle$ el conjunto de todos los arcos que cortan a $[v_k^{'}, v_k^{''}]$.
- Eliminar de $\triangle$ los arcos que junto con $[v_k^{'}, v_k^{''}]$ cubran todo el círculo.
- Si el conjunto $\triangle$ es vacío entonces $v_{1_k} = v_k$.
- Si no, partiendo de $v_k^{'}$ recorrer el círculo en sentido anti-horario hasta encontrar el último $v_i^{'} \in \triangle$. Tomar $v_{1_k} = v_i$.

Por ejemplo, si vemos la figura 2.5, el orden circular canónico R_j asociado a v_j queda definido de la siguiente manera: $R_j = \{v_s, v_j, v_k, v_r\}$.

Lema 2.2.1 *Sea φ una representación arco-circular de un grafo G. Si R_k y R_s son dos órdenes circulares canónicos asociados a $v_k, v_s \in V(G)$, entonces R_k es un desplazamiento circular de R_s.*

Demostración. Sea $R_s = [v_1, ..., v_n]_{R_s}$ y $R_t = [v_1, ..., v_n]_{R_t}$ y supongamos que R_t no es un desplazamiento circular de R_s. Luego existe v_{i_s} tal que se cumple que $v_{i_s} = v_{j_t}$ pero $v_{i-1_s} \neq v_{j-1_t}$. Esto significa que comenzando del extremo $v_{1_s}^{'}$ en φ y recorriendo el círculo en sentido horario en algún punto se alcanzó a $v_{i-1_s}^{'}$ y seguidamente a $v_{i_s}^{'}$. Sin embargo, al construir a R_t, siguiendo el mismo procedimiento y partiendo de $v_{1_t}^{'}$, se alcanzó a $v_{j-1_t}^{'}$ para encontrar seguidamente a $v_{j_t}^{'}$. Esto en un absurdo, dado que partiendo de $v_{i_s}^{'}$ y recorriendo el círculo en sentido anti-horario el primer comienzo de arco que se encuentra es o bien $v_{j-1_t}^{'}$ o bien $v_{i-1_s}^{'}$. Por lo tanto R_t es un desplazamiento circular de R_s.□

Lema 2.2.2 *Si φ es una representación arco-circular de un grafo G y R_k es el orden circular canónico asociado a $v_k \in V(G)$, entonces existe una partición disjunta de aristas $\bar{G} = Q \cup I$ de su complemento $\bar{G}$ tal que R_k es un orden circular asociado a v_k, Q e I.*

Demostración. Partimos las aristas $(v_i, v_j) \in E(G)$ en subgrafos Q e I de la siguiente manera:

$(v_i, v_j) \in E(Q) \Leftrightarrow [v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ están uno incluido en el otro.
$(v_i, v_j) \in E(I) \Leftrightarrow [v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ no se intersecan.

Veamos ahora que el orden circular canónico R_k asociado a v_k, Q e I cumple con las condiciones necesarias para ser un orden circular. Para ello verificaremos que se cumplen las condiciones (1)-(6) impuestas en la definición de orden circular. Sea $v_{1_k} = v_p$. Por construcción vale (1) $(v_p, v_k) \notin E(I)$, dado

que el extremo elegido como primero en la construcción se interseca con v_k. Todos los vértices que quedaron numerados entre v_p y v_k se intersecan con v_p, dado que v_p se interseca con v_k y se numeró a los vértices recorriendo el círculo en sentido horario, partiendo de v_p. Por lo tanto (2) si bajo R_k vale $v_j < v_k$ entonces $(v_p, v_{j_k}) \notin E(I)$. Como v_p junto con v_k no cubren todo el círculo y lo mismo sucede con v_p y v_{1_p} (por construcción), partiendo del extremo v'_{1_p} y recorriendo el círculo en sentido horario, primero se encuentra a v'_p y luego a v'_k. Por lo tanto, (3) bajo R_p vale $v_p < v_k$. Para probar (4) supongamos que valen las hipótesis pero $(v_i, v_t) \in E(Q)$. Como bajo R_t vale $i < t < j$ esto significa que el arco $[v'_t, v''_t]$ está incluido en el arco $[v'_i, v''_i]$ y como $(v_t, v_j) \in E(G)$ y $t < j$ bajo R_t, el arco $[v'_t, v''_t]$ corta al arco $[v'_j, v''_j]$. Por la posición de los arcos, esto significa que el arco $[v'_i, v''_i]$ también corta al arco $[v'_j, v''_j]$ y al numerar los arcos bajo el orden R_i vale $i < j$. Esto es un absurdo dado que por hipótesis bajo R_i vale $j < i$. Por lo tanto $(v_i, v_t) \notin E(Q)$.

Para probar el punto (5) se presentan dos alternativas:

Caso 1: Supongamos que no existe v_c tal que v_k corte a v_c . Luego, por construcción, v_p es el vértice que cumple con todos los items anteriores y numera a v_k con la máxima numeración. Bajo estas condiciones al eliminar la condición (3) ésta sigue valiendo, con lo cual la condición (6) también se cumple.

Caso 2: Supongamos que existe v_c tal que v_k corta a v_c . Por construcción, $v_c \neq v_p$. Veamos entonces que no es posible que v_c cumpla las seis condiciones impuestas. Supongamos que si y que $v_c = v_p$. Si se elimina la condición (3), los órdenes definidos son tales que $v_c < v_k$ bajo R_{k3} y $v_k < v_c$ bajo R_{c3}, por lo que no cumplen la condición (3). Luego se deben cumplir (6a) y (6b). Pero la condición (6b) no se cumple dado que v_c bajo R_{k3} quedó numerado entre 1 y el número asignado a v_k, y por lo tanto la condición (6b) exige que bajo R_k valga $v_k < v_c$ y esto es imposible ya que $v_c = v_p$. Por lo tanto no se puede elegir a un vértice v_c tal que v_k corte a v_c. Notar que esto cubre el caso en el que v_c, junto con v_k, cubren todo el círculo, con lo cual el algoritmo de numeración presentado funciona correctamente y R_k es un orden circular. □

Corolario 2.2.1 *Sea φ una representación arco-circular de un grafo G y sea R_k un orden circular asociado a $v_k \in V(G)$. Si $[v'_k, v''_k]$ es el arco de φ asociado a v_k, entonces v_{1_k} es un vértice tal que su arco asociado $[v'_{1_k}, v''_{1_k}]$ en φ no es cortado por el arco $[v'_k, v''_k]$.*

Demostración. Trivial, dada la demostración del caso 2 del lema 2.2.2. □

Corolario 2.2.2 *Sea φ una representación arco-circular de un grafo G y sea R_k un orden circular asociado a $v_k \in V(G)$. Si $[v'_k, v''_k]$ es el arco de φ asociado a v_k, entonces v_{1_k} es un vértice tal que su arco asociado $[v'_{1_k}, v''_{1_k}]$ en φ corta al arco $[v'_k, v''_k]$.*

Demostración. Trivial, dado el corolario 2.2.1 y sabiendo que, dado que R_k es un orden circular, vale $v_{1_k}, v_k \notin E(I)$. □

Corolario 2.2.3 *Sea φ una representación arco-circular de un grafo G y sea R_k un orden circular asociado a $v_k \in V(G)$. Sea $[v_k^{'}, v_k^{''}]$ el arco de φ asociado a v_k. Si $v_i \in V(G)$ es un vértice tal que su arco asociado $[v_i^{'}, v_i^{''}]$ en φ es cortado por el arco $[v_k^{'}, v_k^{''}]$, entonces vale $k < i$.*

Demostración. Como $[v_k^{'}, v_k^{''}]$ corta a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.1, $v_i \neq v_{1_k}$. Supongamos por el absurdo que $i < k$ bajo R_k. Esto significa que partiendo de $v_{1_k}^{'}$ y recorriendo el círculo en sentido horario primero se encuentra el extremo $v_i^{'}$ y luego $v_k^{'}$. Esto significa que $v_{1_k}^{'}$ se encuentra entre $v_k^{'}$ y $v_i^{'}$. Como $[v_k^{'}, v_k^{''}]$ corta a $[v_i^{'}, v_i^{''}]$ esto implica que $[v_k^{'}, v_k^{''}]$ corta a $[v_{1_k}^{'}, v_{1_k}^{''}]$. Esto es un absurdo dado el corolario 2.2.1. Por lo tanto bajo R_k vale $k < i$. □

Corolario 2.2.4 *Sea φ una representación arco-circular de un grafo G y sea R_k un orden circular asociado a $v_k \in V(G)$. Sea $[v_k^{'}, v_k^{''}]$ el arco de φ asociado a v_k. Si $v_i \in V(G), v_i \neq v_{1_k}$, es un vértice tal que su arco asociado $[v_i^{'}, v_i^{''}]$ en φ corta a $[v_k^{'}, v_k^{''}]$, y ambos arcos no cubren todo el círculo, entonces el arco $[v_{1_k}^{'}, v_{1_k}^{''}]$, asociado a v_{1_k}, corta a $[v_i^{'}, v_i^{''}]$.*

Demostración. Supongamos que $[v_{1_k}^{'}, v_{1_k}^{''}]$ no corta a $[v_i^{'}, v_i^{''}]$. Dadas las hipótesis, y por el corolario 2.2.2, sabemos que tanto $[v_{1_k}^{'}, v_{1_k}^{''}]$ como $[v_i^{'}, v_i^{''}]$ cortan a $[v_k^{'}, v_k^{''}]$. Luego lo que debe suceder es que $[v_i^{'}, v_i^{''}]$ corta a $[v_{1_k}^{'}, v_{1_k}^{''}]$. Por lo tanto v_i cumple con todas las condiciones para ser el primer vértice de R_k, y numera a v_k con una numeración mayor que la asignada por v_{1_k}. Esto es un absurdo dado que $v_{1_k} \neq v_i$, luego la única alternativa es que $[v_{1_k}^{'}, v_{1_k}^{''}]$ corte a $[v_i^{'}, v_i^{''}]$.

2.3 Caracterización de grafos overlap de arco-circulares

Para la formulación y demostración de la caracterización de los grafos overlap de arco-circulares (CAO), utilizamos ideas propuestas en una caracterización para grafos circulares de J. Szwarcfiter. El teorema planteado por Szwarcfiter es el siguiente:

Teorema 2.3.1 ([35]) *G es un grafo circular si y sólo si existe una partición disjunta de aristas $\bar{G} = Q \cup I$ de su complemento $\bar{G}$ y un orden lineal R de sus vértices, tal que las orientaciones $\vec{G}$, $\vec{Q}$, $\vec{I}$ inducidas por R satisfacen*
para $i < j < k$,
(B1): $(v_i \to v_j) \in E(\vec{I}) \Rightarrow (v_i \to v_k) \in E(\vec{I})$.
(B2): $(v_i \to v_j) \in E(\vec{Q}) \wedge (v_j \to v_k) \in E(\vec{Q}) \Rightarrow (v_i \to v_k) \in E(\vec{Q})$.
(B3): $(v_i \to v_j) \in E(\vec{G}) \wedge (v_j \to v_k) \in E(\vec{G}) \Rightarrow (v_i \to v_k) \notin E(\vec{Q})$.
(B4): $(v_i \to v_j) \in E(\vec{G}) \wedge (v_j \to v_k) \in E(\vec{I}) \Rightarrow (v_i \to v_k) \in E(\vec{I})$.
(B5): $(v_i \to v_j) \in E(\vec{Q}) \wedge (v_j \to v_k) \in E(\vec{G}) \Rightarrow (v_i \to v_k) \notin E(\vec{I})$.

El teorema que caracteriza a la clase CAO toma las ideas básicas del teorema 2.3.1 y refuerza las condiciones pedidas sobre el grafo. De esta manera se pueden sortear las dificultades adicionales que se presentan al trabajar con representaciones en overlap de arcos circulares.

Teorema 2.3.2 *G es un grafo overlap de arco-circulares si y sólo si existe una partición disjunta de aristas $\bar{G} = Q \cup I$ de su complemento $\bar{G}$ y para cada vértice $v \in V(G)$ existe un orden circular R_v asociado a v, Q e I tal que las orientaciones $\vec{G_{R_v}}$, $\vec{Q_{R_v}}$, $\vec{I_{R_v}}$ inducidas por R_v satisfacen*
para $i < j < k$,
(B1): $(v_i \to v_j) \in E(\vec{I_{R_i}}) \Rightarrow (v_i \to v_k) \in E(\vec{I_{R_i}})$.
(B2): $(v_i \to v_j) \in E(\vec{Q_{R_v}}) \wedge (v_j \to v_k) \in E(\vec{Q_{R_v}}) \Rightarrow (v_i \to v_k) \in E(\vec{Q_{R_v}})$.
(B3): $(v_i \to v_j) \in E(\vec{G_{R_i}}) \wedge (v_j \to v_k) \in E(\vec{G_{R_i}}) \Rightarrow (v_i \to v_k) \notin E(\vec{Q_{R_i}})$.
(B4): $(v_i \to v_j) \in E(\vec{G_{R_i}}) \wedge (v_j \to v_k) \in E(\vec{I_{R_i}}) \Rightarrow (v_i \to v_k) \in E(\vec{I_{R_i}})$.
(B5): $(v_i \to v_j) \in E(\vec{Q_{R_v}}) \wedge (v_j \to v_k) \in E(\vec{G_{R_v}}) \Rightarrow (v_i \to v_k) \notin E(\vec{I_{R_v}})$.
y R_s es un desplazamiento circular de R_t para todo s, $t \in V(G)$.

Idea intuitiva sobre la demostración.

La idea básica que utilizamos en la demostración del teorema consiste en "linealizar"la representación en arcos circulares de G por medio de un orden circular. Cada orden circular asociado a un vértice $v_k \in V(G)$ asigna una numeración a los vértices del grafo, que permite "ver" a la arcos a través de un orden total, definido por la función $s_{v_{1_k}}$.

Si bien bajo este orden puede perderse información sobre algunos solapamientos entre arcos, el orden R_{v_k} mantiene los solapamientos entre el arco

asociado al vértice v_k con el resto de los arcos. De esta forma, la función $s_{v_{1_k}}$ asigna a los extremos de los arcos un orden tal que respeta sus posiciones relativas en función del arco asociado a v_k.

Una vez fijado el orden circular, el grafo debe cumplir las condiciones (B1)-(B5). Si esto sucede para cada vértice del grafo, el grafo es overlap de arco-circulares

Demostración.($\Rightarrow$) Sea G un grafo overlap de arco-circulares, φ una representación en arcos alrededor del círculo de él y sea R_i el orden circular canónico asociado a $v_i \in V(G)$ para cada vértice de G. Partir las aristas $(v_i, v_j) \in E(G)$ en subgrafos Q e I de la siguiente manera:

$$(v_i, v_j) \in E(Q) \Leftrightarrow ([s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i} \vee [s(v_i^{'}) < s(v_j^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_i})$$

$$(v_i, v_j) \in E(I) \Leftrightarrow [s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$$

Como R_i es un orden circular canónico, esto define una partición disjunta de aristas del complemento de G, donde en I están las aristas de $\bar{G}$ que en la representación sus vértices correspondientes no se intersecan, y en Q las aristas de $\bar{G}$ que en la representación sus vértices correspondientes están incluidos uno en otro. De esta forma, por el lema 2.2.2, quedan definidos n órdenes circulares. Sean $\vec{G_{R_i}}, \vec{Q_{R_i}}, \vec{I_{R_i}}$ las correspondientes orientaciones de G, Q e I inducidas por R_i. Veamos ahora que se cumplen (B1)-(B5).

B1: Como $(v_i \to v_j) \in E(\vec{I_{R_i}})$, tenemos $[s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$. Además dado que sabemos que bajo el orden R_i vale $i < j$, esto implica que $[s(v_i^{''}) < s(v_j^{'})]_{R_i}$. También sabemos que bajo R_i tenemos $j < k$ por lo que, por construcción de s, vale $[s(v_j^{'}) < s(v_k^{'})]_{R_i}$. Por lo tanto, $[s(v_i^{''}) < s(v_k^{'})]_{R_i}$ con lo cual $(v_i, v_k) \in E(I)$, y bajo R_i esto significa $(v_i \to v_k) \in E(\vec{I_{R_i}})$.

B2: Dado que $(v_i \to v_j) \in E(\vec{Q_{R_v}})$ y $(v_j \to v_k) \in E(\vec{Q_{R_v}})$ por construcción de Q sabemos que en la representación $[v_i^{'}, v_i^{''}] \subset [v_j^{'}, v_j^{''}] \subset [v_k^{'}, v_k^{''}]$ o bien $[v_k^{'}, v_k^{''}] \subset [v_j^{'}, v_j^{''}] \subset [v_i^{'}, v_i^{''}]$. Pero como bajo el orden R_v vale $i < j < k$, dado que R_v es un orden canónico su construcción implica que $[v_i^{'}, v_i^{''}] \subset [v_j^{'}, v_j^{''}] \subset [v_k^{'}, v_k^{''}]$. Por lo tanto, bajo el orden R_i vale $i < k$ y eso implica que $[s(v_i^{'}) < s(v_k^{'}) < s(v_k^{''}) < s(v_i^{''})]_{R_i}$. Por construcción de Q, esto significa que $(v_i, v_k) \in E(Q)$, y bajo el orden R_v vale $(v_i \to v_k) \in E(\vec{Q_{R_v}})$.

B3: Tenemos que $(v_i \to v_j) \in E(\vec{G_{R_i}})$ y $(v_j \to v_k) \in E(\vec{G_{R_i}})$. Como R_i es un orden canónico y vale $i < j < k$, tenemos que $[s(v_i^{'}) < s(v_j^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i}$ y $[s(v_j^{'}) < s(v_k^{'}) < s(v_j^{''}) < s(v_k^{''})]_{R_i}$. Esto implica que $[s(v_i^{'}) < s(v_k^{'})]_{R_i}$ y $[s(v_k^{''}) > s(v_i^{''})]_{R_i}$, con lo cual, por construcción de Q, tenemos que $(v_i, v_k) \notin E(Q)$. Por lo tanto, $(v_i \to v_k) \notin E(\vec{Q_{R_i}})$

B4: Sabemos que $(v_i \to v_j) \in E(\vec{G_{R_i}})$ y $(v_j \to v_k) \in E(\vec{I_{R_i}})$. Como R_i es un orden canónico y vale $i < j < k$, tenemos que $[s(v_i^{'}) < s(v_j^{'}) < s(v_i^{''}) <$

$s(v_j^{''})]_{R_i}$ y $[s(v_j^{''}) < s(v_k^{'})]_{R_i}$. Esto implica que $[s(v_i^{''}) < s(v_k^{'})]_{R_i}$. Luego, por construcción de I, tenemos que $(v_i, v_k) \in E(I)$ y bajo el orden R_i vale $(v_i \rightarrow v_k) \in E(\vec{I_{R_i}})$.

B5: Como $(v_i \rightarrow v_j) \in E(\vec{Q_{R_v}})$, por construcción de Q sabemos que un arco está incluido en el otro. Como además bajo R_v vale $i < j$ y como R_v es un orden canónico, sabemos que el arco $[v_i^{'}, v_i^{''}]$ está incluido en $[v_j^{'}, v_j^{''}]$, por lo que vale $[s(v_i^{'}) < s(v_j^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_v}$. Por otra parte tenemos que $(v_j \rightarrow v_k) \in E(\vec{G_{R_v}})$, con lo cual $[s(v_j^{'}) < s(v_k^{'}) < s(v_j^{''}) < s(v_k^{''})]_{R_v}$. Por lo tanto, llegamos a que $[s(v_i^{'}) < s(v_k^{'}) < s(v_i^{''})]_{R_v}$, es decir, que ambos arcos se intersecan. Veamos ahora que sucede bajo el orden R_i. Si los arcos $[v_i^{'}, v_i^{''}]$ y $[v_k^{'}, v_k^{''}]$ no cubren todo el círculo, por construcción del orden canónico R_i vale que $i < j < k$. Por otra parte, si $[v_i^{'}, v_i^{''}]$ y $[v_k^{'}, v_k^{''}]$ cubren todo el círculo, significa que $v_k^{'}$ no fue tomado en cuenta como candidato para ser el primer vértice de R_i, por lo que también vale $i < j < k$. Por lo tanto las desigualdades que valían para R_v también valen para R_i. Luego, por construcción de I, vale $(v_i, v_k) \notin E(I)$, y esto significa que $(v_i \rightarrow v_k) \notin E(\vec{I_{R_v}})$.

Por último, por el lema 2.2.1, R_s es un desplazamiento circular de R_t para todo s, t ∈ V(G).

(⇐) Sea G un grafo, $\bar{G} = Q \cup I$ una partición disjunta de aristas de su complemento $\bar{G}$ y para cada vértice v ∈ V(G) existe un orden circular R_v asociado a v tal que las orientaciones $\vec{G_{R_v}}$, $\vec{Q_{R_v}}$, $\vec{I_{R_v}}$ inducidas por R_v satisfacen (B1)-(B5). Además, R_s es un desplazamiento circular de R_t para todo s, t ∈ V(G). Probamos que G es un grafo overlap de arco-circulares. El argumento es por inducción en n. Si $n = 1$ no hay nada que probar. Vamos a probar que podemos construir una representación de arcos $[v_i^{'}, v_i^{''}]$ alrededor del círculo tal que:

(C0) $[s(v_i^{'}) < s(v_j^{'})]_{R_v} \Leftrightarrow i < j$ bajo R_v

(C1) $(v_i, v_j) \in E(G) \Leftrightarrow [s(v_i^{'}) < s(v_j^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i} \vee [s(v_j^{'}) < s(v_i^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_i}$

(C2) $(v_i, v_j) \in E(I) \Leftrightarrow [s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$

Como una consecuencia de (C1) y (C2) vale:

(C3) $(v_i, v_j) \in E(Q) \Leftrightarrow [s(v_i^{'}) < s(v_j^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_i} \vee [s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i}$

Las implicaciones (B1)-(B5) se mantienen válidas cuando nos restringimos a los subgrafos inducidos $G^{'}$ de G (y también se mantienen válidas para $\bar{G'}$). Por lo tanto, podemos aplicar la hipótesis inductiva a G - v_k, donde v_k es un vértice cualquiera de G numerado bajo el orden R_k de forma que $v_{t_k} = v_k$ para algún t entre 1 y n. Esto nos permite construir un conjunto de arcos alrededor del círculo $\varphi - v_{t_k}$, con arcos correspondientes al conjunto de vértices G - $\{v_{t_k}\}$ y tal que (C0), (C1) y (C2) son satisfechas. Luego obtenemos un nuevo

conjunto de arcos alrededor del círculo insertando apropiadamente en $\varphi - v_{t_k}$ un nuevo intervalo $[v_t^{'}, v_t^{''}]$, sin cambiar las posiciones relativas de los otros arcos. Este nuevo intervalo corresponde al vértice v_{t_k}, y se inserta de la siguiente manera:

(D1) *Posición de* $v_{t_k}^{'}$.

- Insertar inicialmente a $v_{t_k}^{'}$ entre $v_{t-1_k}^{'}$ y $v_{t+1_k}^{'}$ de tal forma que no exista ningún extremo de arco entre $v_{t_k}^{'}$ y $v_{t+1_k}^{'}$.
- Desplazar a $v_{t_k}^{'}$ en el sentido anti-horario mientras exista algún extremo de arco entre $v_{t_k}^{'}$ y $v_{t-1_k}^{'}$ y no se sobrepase a ningún extremo $v_i^{''}$, tal que:
  1. $(v_i, v_t) \in E(I)$.
  2. Bajo R_k valga $t < i$.

(D2) *Posición de* $v_{t_k}^{''}$.

- Insertar inicialmente a $v_{t_k}^{''}$ entre $v_{t-1_k}^{'}$ y $v_{t_k}^{'}$ de tal forma que no exista ningún extremo de arco entre $v_{t_k}^{''}$ y $v_{t_k}^{'}$.
- Desplazar a $v_{t_k}^{''}$ en el sentido anti-horario mientras exista algún extremo de arco entre $v_{t_k}^{'}$ y $v_{t_k}^{''}$ y se cumplan las siguientes condiciones:
  1. No sobrepasar a ningún $v_i^{''}$ tal que $(v_i, v_t) \in E(G)$ y al sobrepasarlo $[v_t^{'}, v_t^{''}]$ quede incluido en $[v_i^{'}, v_i^{''}]$.
  2. No sobrepasar a ningún $v_i^{''}$ tal que $(v_i, v_t) \in E(Q)$ y al sobrepasarlo $[v_i^{'}, v_i^{''}]$ deje de estar incluido en $[v_t^{'}, v_t^{''}]$.
  3. No sobrepasar a ningún $v_i^{'}$ tal que $(v_i, v_t) \in E(G)$ y al sobrepasarlo $[v_t^{'}, v_t^{''}]$ deje de intersecar a $[v_t^{'}, v_t^{''}]$.

Probamos que el grafo resultante de la construcción anterior satisface la hipótesis. Las condiciones (C0), (C1) y (C2) valen para los vértices v_i con $i \neq t$ por hipótesis inductiva. Mostramos que la construcción φ satisface estas condiciones en relación al vértice v_t. Estas condiciones son:

(E0) $[s(v_i^{'}) < s(v_j^{'})]_{R_v} \Leftrightarrow i < j$ bajo R_v

(E1) $(v_i, v_t) \in E(G) \Leftrightarrow [s(v_i^{'}) < s(v_t^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i} \vee [s(v_t^{'}) < s(v_i^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$

(E2) $(v_i, v_t) \in E(I) \Leftrightarrow [s(v_i^{''}) < s(v_t^{'})]_{R_i} \vee [s(v_t^{''}) < s(v_i^{'})]_{R_i}$

A lo largo de la demostración a los contrarrecíprocos de (B1)-(B5) los llamaremos ($\Leftarrow$B1)-($\Leftarrow$B5).

Prueba para (E0).

Por hipótesis inductiva tenemos que $[s(v_i^{'}) < s(v_j^{'})]_{R_v} \Leftrightarrow i < j$ bajo R_v, con $i \neq j \neq t$. Veamos que esta propiedad también vale para t bajo R_k. Por (D1), $v_t^{'}$ quedó posicionado entre $v_{t-1_k}^{'}$ y $v_{t+1_k}^{'}$. Luego, por construcción de s, $[s(v_{t-1}^{'}) < s(v_t^{'}) < s(v_{t+1}^{'})]_{R_k}$ y naturalmente bajo R_k vale $t-1 < t < t+1$. Por lo tanto $[s(v_i^{'}) < s(v_j^{'})]_{R_k} \Leftrightarrow i < j$ bajo R_k. Analicemos ahora los que sucede con R_q, $q \neq k$. Como R_q es un desplazamiento circular de R_k, bajo R_q el vértice v_t es el inmediato posterior a v_{t-1} o el inmediato anterior a v_{t+1}.

Caso 1: v_t es el inmediato posterior a v_{t-1} bajo R_q.
Por construcción de s, $[s(v_{t-1}^{'}) < s(v_t^{'})]_{R_q}$ y por hipótesis tenemos $[s(v_{t-1}^{'}) > s(v_i^{'})]_{R_q}$ para todo $i \geqslant t+1$. Además vale $t-1 < t$, luego $[s(v_i^{'}) < s(v_j^{'})]_{R_q} \Leftrightarrow i < j$ bajo R_q.

Caso 2: v_t es el inmediato anterior a v_{t+1} bajo R_q.
Por construcción de s, $[s(v_t^{'}) < s(v_{t+1}^{'})]_{R_q}$ y por hipótesis tenemos $[s(v_i^{'}) < s(v_{t+1}^{'})]_{R_q}$ para todo $i \leqslant t-1$. Luego $[s(v_i^{'}) < s(v_j^{'})]_{R_q} \Leftrightarrow i < j$ bajo R_q .

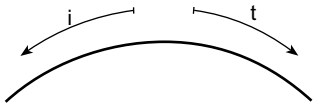
Por lo tanto queda probado (E0).

Prueba para (E1).

($\Rightarrow$): Asumamos que $(v_i, v_t) \in E(G)$. Si $[s(v_i^{'}) < s(v_t^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i} \vee [s(v_t^{'}) < s(v_i^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$ es falso, como la construcción de s asegura $[s(v_t^{'}) < s(v_t^{''})]_{R_i}$ y $[s(v_i^{'}) < s(v_i^{''})]_{R_i}$, lo único que podría pasar es uno de los siguientes cuatro casos:

1. $[s(v_i^{'}) < s(v_i^{''}) < s(v_t^{'}) < s(v_t^{''})]_{R_i}$
2. $[s(v_t^{'}) < s(v_t^{''}) < s(v_i^{'}) < s(v_i^{''})]_{R_i}$
3. $[s(v_i^{'}) < s(v_t^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$
4. $[s(v_t^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i}$

Caso 1: $[s(v_i^{'}) < s(v_i^{''}) < s(v_t^{'}) < s(v_t^{''})]_{R_i}$.

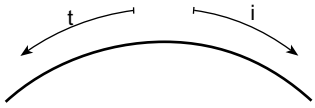


Por construcción de s, sabemos que el arco $[v_i^{'}, v_i^{''}]$ no corta al arco $[v_t^{'}, v_t^{''}]$. Veamos que tampoco pasa lo contrario. Asumamos que $[v_t^{'}, v_t^{''}]$ corta a $[v_i^{'}, v_i^{''}]$. Por construcción de s sabemos que $v_t \neq v_{1_i}$ y que ambos arcos no cubren todo el círculo, luego por el corolario 2.2.4, vale que $[v_{1_i}^{'}, v_{1_i}^{''}]$ corta a $[v_t^{'}, v_t^{''}]$. Como $[v_t^{'}, v_t^{''}]$ corta a $[v_i^{'}, v_i^{''}]$, por construcción de s tenemos que $[s(v_t^{'}) < s(v_i^{'})]_{R_i}$.

Esto es un absurdo, dado que contradice las hipótesis. Por lo tanto, los arcos $[v_i^{'}, v_i^{''}]$ y $[v_t^{'}, v_t^{''}]$ no se intersecan.

Luego, por (D2), al posicionar $v_t^{''}$ se sobrepasó a $v_i^{'}$ y los arcos dejaron de intersecarse, con lo cual por ($D2_3$), tenemos $(v_i, v_t) \notin E(G)$, una contradicción.

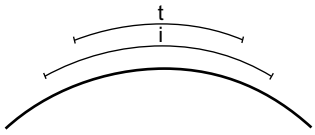
Caso 2: $[s(v_t^{'}) < s(v_t^{''}) < s(v_i^{'}) < s(v_i^{''})]_{R_i}$.



Por construcción de s, sabemos que el arco $[v_t^{'}, v_t^{''}]$ no corta al arco $[v_i^{'}, v_i^{''}]$. Veamos que tampoco pasa lo contrario. Supongamos que $[v_i^{'}, v_i^{''}]$ corta a $[v_t^{'}, v_t^{''}]$. Luego, por el corolario 2.2.3, bajo R_i vale $i < t$. Luego, por (E0), $[s(v_i^{'}) < s(v_t^{'})]_{R_i}$. Esto es una contradicción, con lo cual ambos arcos no se intersecan en la representación.

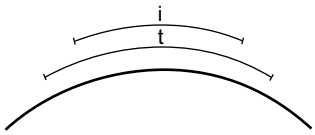
Luego, por (D2), al posicionar $v_t^{''}$ se sobrepasó a $v_i^{'}$ y los arcos dejaron de intersecarse, con lo cual por ($D2_3$), tenemos $(v_i, v_t) \notin E(G)$, una contradicción.

Caso 3: $[s(v_i^{'}) < s(v_t^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$.



Por construcción de s tenemos que en la representación el arco $[v_t^{'}, v_t^{''}]$ está incluido en el arco $[v_i^{'}, v_i^{''}]$. Por (D2), al posicionar $v_t^{''}$ se sobrepasó a $v_i^{''}$, y en ese momento el arco $[v_t^{'}, v_t^{''}]$ pasó a estar incluido en $[v_i^{'}, v_i^{''}]$. Por ($D2_1$), tenemos $(v_i, v_t) \notin E(G)$, una contradicción.

Caso 4: $[s(v_t^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i}$.



Por construcción de s sabemos que el arco $[v_i^{'}, v_i^{''}]$ está incluido en el arco $[v_t^{'}, v_t^{''}]$.

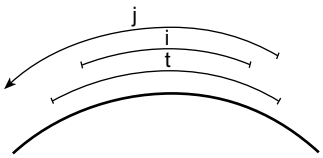
Por (D2), no se siguió desplazando a $v_t^{''}$ en sentido anti-horario porque existe un extremo de arco entre $v_i^{''}$ y $v_t^{''}$ que cumple con una de las siguientes condiciones:

1. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.

2. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.
3. Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.

Si $v_j^{''} = v_i^{''}$, estamos en la condición 2 y $(v_i, v_t) \in E(Q)$. Esto es un absurdo, luego el extremo que impidió el desplazamiento de $v_t^{''}$ no es $v_i^{''}$.

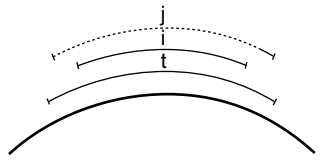
Caso 4.1: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.



Analicemos qué sucede bajo el orden R_j en función de las posiciones relativas de los extremos de los arcos. Sabemos que el arco $[v_j^{'}, v_j^{''}]$ corta a $[v_t^{'}, v_t^{''}]$. Como $v_j^{''}$ está entre $v_i^{''}$ y $v_t^{''}$, el arco $[v_i^{'}, v_i^{''}]$ está incluido en el arco $[v_j^{'}, v_j^{''}]$, con lo cual $[v_j^{'}, v_j^{''}]$ corta a $[v_i^{'}, v_i^{''}]$. Por el corolario 2.2.3, bajo R_j vale $j < t, i$. Como $[v_i^{'}, v_i^{''}]$ está incluido en el arco $[v_t^{'}, v_t^{''}]$, por construcción de s tenemos que $[s(v_j^{'}) < s(v_t^{'}) < s(v_i^{'})]_{R_j}$, y por (E0) bajo R_j vale $j < t < i$.

Por otra parte, vale $[s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_j}$, luego por (C3) tenemos que $(v_j, v_i) \in E(Q)$. Luego, $(v_j \to v_i) \in E(\vec{Q_{R_j}})$ y $(v_j \to v_t) \in E(\vec{G_{R_j}})$. Por ($\Leftarrow$B3), $(v_t \to v_i) \notin E(\vec{G_{R_j}})$ y como $t < i$ vale $(v_i, v_t) \notin E(G)$. Esto es un absurdo, por lo que este caso no puede ocurrir.

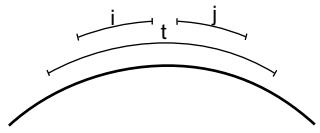
Caso 4.2: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.



Esto significa que el arco $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_i^{'}, v_i^{''}]$ como a $[v_j^{'}, v_j^{''}]$. Por el corolario 2.2.3, bajo R_t vale $t < i, j$

Bajo este caso se presentan tres alternativas en función de la posición del extremo $v_j^{'}$.

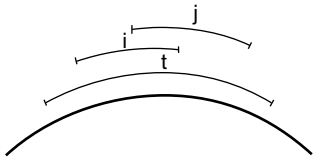
Caso 4.2.1: $v_j^{'}$ se encuentra entre $v_i^{''}$ y $v_j^{''}$.



Como en la representación los arcos $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ no se intersecan. Por construcción de s vale $[s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$. Luego, por (C2), $(v_i, v_j) \in E(I)$.

Bajo R_t tenemos que $t < i, j$. Por la posición relativa de $v_j^{'}$, sabemos que $[s(v_i^{'}) < s(v_j^{'})]_{R_t}$ y por (E0), esto implica que $t < i < j$. Luego, $(v_i \to v_j) \in E(\vec{I_{R_t}})$ y $(v_t \to v_j) \in E(\vec{Q_{R_t}})$. Por ($\Leftarrow$B4), $(v_t \to v_i) \notin E(\vec{G_{R_t}})$ y como $t < i$ vale $(v_i \to v_t) \notin E(G)$. Esto es una contradicción.

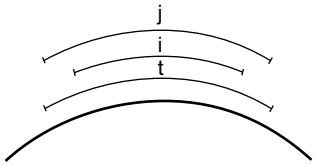
Caso 4.2.2: $v_j^{'}$ se encuentra entre $v_i^{'}$ y $v_i^{''}$.



En la representación los arcos $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ se solapan. Como $[v_i^{'}, v_i^{''}]$ corta a $[v_j^{'}, v_j^{''}]$, por el corolario 2.2.3 bajo R_i vale $i < j$ y esto implica que $[s(v_i^{'}) < s(v_j^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i}$. Por (C1), $(v_i, v_j) \in E(G)$.

Bajo R_t tenemos que $t < i, j$. Por la posición relativa de $v_j^{'}$, sabemos que $[s(v_i^{'}) < s(v_j^{'})]_{R_t}$ y por (E0), esto implica que $t < i < j$. Luego, $(v_i \to v_j) \in E(\vec{G_{R_t}})$ y $(v_t \to v_j) \in E(\vec{Q_{R_t}})$. Por ($\Leftarrow$B3), $(v_t \to v_i) \notin E(\vec{G_{R_t}})$ y como $t < i$ vale $v_i, v_t \notin E(G)$, un absurdo.

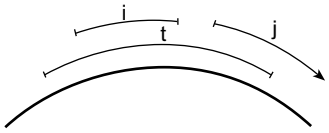
Caso 4.2.3: $v_j^{'}$ se encuentra entre $v_t^{'}$ y $v_i^{'}$.



En la representación el arco $[v_i^{'}, v_i^{''}]$ está incluido en el arco $[v_j^{'}, v_j^{''}]$. Como $[v_j^{'}, v_j^{''}]$ corta a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.3, bajo R_j vale $j < i$ y esto implica que $[s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_j}$. Por (C3), $(v_j, v_i) \in E(Q)$.

Bajo R_t tenemos que $t < i, j$. Por la posición relativa de $v_j^{'}$, sabemos que $[s(v_j^{'}) < s(v_i^{'})]_{R_t}$ y por (E0), esto implica que $t < j < i$. Luego, $(v_t \to v_j) \in E(\vec{Q_{R_t}})$ y $(v_j \to v_i) \in E(\vec{Q_{R_t}})$. Por (B2), $(v_t \to v_i) \in E(\vec{Q_{R_t}})$ y como $t < i$ vale $(v_i, v_t) \in E(Q)$. Esto significa que $(v_i, v_t) \notin E(G)$, una contradicción. Por lo tanto el caso 4.2 no puede ocurrir.

Caso 4.3: Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.



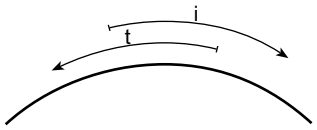
Esto significa que el arco $[v_t^{'}, v_t^{''}]$ junto con el arco $[v_j^{'}, v_j^{''}]$ no cubren todo el círculo. La posición de $v_j^{'}$ nos indica que el arco $[v_i^{'}, v_i^{''}]$ no corta al arco $[v_j^{'}, v_j^{''}]$. El hecho que $[v_t^{'}, v_t^{''}]$ y $[v_j^{'}, v_j^{''}]$ no cubren todo el círculo sumado a que $[v_i^{'}, v_i^{''}]$ está incluido en $[v_t^{'}, v_t^{''}]$, implica que $[v_j^{'}, v_j^{''}]$ no corta a $[v_i^{'}, v_i^{''}]$. Por lo tanto, los arcos $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ no se intersecan en la representación.

Por construcción de s esto implica que $[s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$. Luego, por (C2), $(v_i, v_j) \in E(I)$.

Como $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_i^{'}, v_i^{''}]$ como a $[v_j^{'}, v_j^{''}]$, por el corolario 2.2.3 sabemos que bajo R_t vale $t < i, j$. Esto significa que $[s(v_i^{'}) < s(v_j^{'})]_{R_t}$ y por (E0), esto implica que $t < i < j$. Por lo tanto $(v_i \to v_j) \in E(\vec{I}_{R_t})$ y $(v_t \to v_j) \in E(\vec{G}_{R_t})$. Por ($\Leftarrow$B4), $(v_t \to v_i) \notin E(\vec{G}_{R_t})$ y como $t < i$ vale $(v_i, v_t) \notin E(G)$, un absurdo. Esto significa que el caso 4 no puede ocurrir, de modo que la única alternativa es que valga $[s(v_i^{'}) < s(v_t^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i} \vee [s(v_t^{'}) < s(v_i^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$.

($\Leftarrow$): Asumamos que vale $[s(v_i^{'}) < s(v_t^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i} \vee [s(v_t^{'}) < s(v_i^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$ y supongamos que $(v_i, v_t) \notin E(G)$. Analicemos los dos casos que se presentan por separado:

Caso 1: $[s(v_t^{'}) < s(v_i^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$.



Bajo este caso, notemos que el arco $[v_i^{'}, v_i^{''}]$ no corta al arco $[v_t^{'}, v_t^{''}]$, porque si así fuera, el lema 2.2.3 nos diría que bajo R_i vale $i < t$ y por (E0) llegaríamos a que $[s(v_i^{'}) < s(v_t^{'})]_{R_i}$, un absurdo. Luego, como ambos arcos no cubren todo el círculo, al desplazar $v_t^{''}$ para alcanzar su posición definitiva se sobrepasó a $v_i^{''}$. Al sobrepasarlo, $[v_i^{'}, v_i^{''}]$ dejó de estar incluido en $[v_t^{'}, v_t^{''}]$, por lo tanto por ($D2_2$) $(v_i, v_t) \notin E(Q)$. Por hipótesis $(v_i, v_t) \notin E(G)$, luego la única posibilidad es que $(v_i, v_t) \in E(I)$.

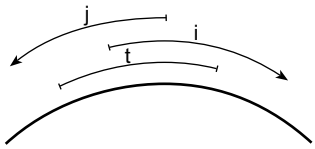
Por (D2), no se siguió desplazando a $v_t^{''}$ en sentido anti-horario porque existe un extremo de arco entre $v_i^{'}$ y $v_t^{''}$ que cumple con una de las siguientes condiciones:

1. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.
2. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.

3. Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.

Si $v_j^{'} = v_i^{'}$, estamos en la condición 3 y $(v_i, v_t) \in E(G)$. Esto es un absurdo, luego el extremo que impidió el desplazamiento de $v_t^{''}$ no es $v_i^{'}$.

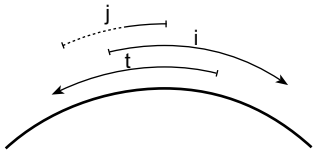
Caso 1.1: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.



Por la posición relativa de los extremos de los arcos sabemos que el arco $[v_j^{'}, v_j^{''}]$ corta tanto a $[v_t^{'}, v_t^{''}]$ como a $[v_i^{'}, v_i^{''}]$. Luego, por el corolario 2.2.3, bajo R_j vale $j < t, i$. En la representación los arcos $[v_j^{'}, v_j^{''}]$ y $[v_i^{'}, v_i^{''}]$ se solapan, luego por construcción de s esto implica que $[s(v_j^{'}) < s(v_i^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_j}$. Por (C1), $(v_j, v_i) \in E(G)$.

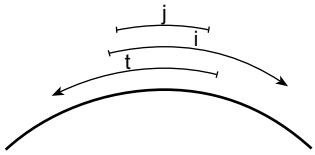
Como vale $j < t, i$, dada la posición relativa de los extremos de los arcos, por construcción de s tenemos que $[s(v_j^{'}) < s(v_t^{'}) < s(v_i^{'})]_{R_j}$. Por lo tanto, por (E0), bajo R_j vale $j < t < i$. Esto significa que $(v_j \to v_t) \in E(\vec{G}_{R_j})$ y $(v_j \to v_i) \in E(\vec{G}_{R_j})$. Por ($\Leftarrow$B4), $(v_t \to v_i) \notin E(\vec{I}_{R_j})$ y como $t < i$ vale $v_i, v_t \notin E(I)$. Esto contradice la hipótesis.

Caso 1.2: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.



Bajo este caso, el arco $[v_j^{'}, v_j^{''}]$ está incluido en el arco $[v_t^{'}, v_t^{''}]$. Sabemos que el arco $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_j^{'}, v_j^{''}]$ como a $[v_i^{'}, v_i^{''}]$. Luego, por el corolario 2.2.3, bajo R_t vale $t < j, i$. Este caso presenta dos alternativas:

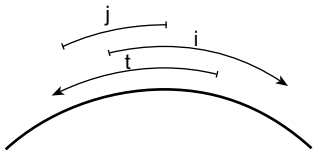
Caso 1.2.1: $v_j^{'}$ se encuentra entre $v_i^{'}$ y $v_j^{''}$.



Bajo R_t vale $t < i, j$, y el arco $[v_j^{'}, v_j^{''}]$ está incluido en el arco $[v_i^{'}, v_i^{''}]$. Por construcción de s vale $[s(v_i^{'}) < s(v_j^{'})]_{R_t}$. Por (E0) tenemos que $t < i < j$ bajo R_t. Luego $(v_t \to v_i) \in E(\vec{I}_{R_t})$, pero por (B1) esto implica que $(v_t \to v_j) \in$

$E(\vec{I_{R_t}})$. Esto contradice la hipótesis que afirma $(v_j, v_t) \in E(Q)$, con lo cual este caso no puede ocurrir.

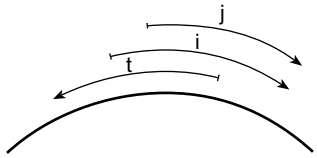
Caso 1.2.2: $v_j^{'}$ se encuentra entre $v_t^{'}$ y $v_i^{'}$.



La posición de $v_j^{'}$ nos indica que en la representación los arcos $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ se solapan. Como $[v_j^{'}, v_j^{''}]$ corta a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.3, bajo R_j vale $j < i$ y esto implica que $[s(v_j^{'}) < s(v_i^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_j}$. Por (C1), $(v_i, v_j) \in E(G)$.

Como bajo R_t vale $t < i, j$, y $v_j^{'}$ se encuentra entre $v_t^{'}$ y $v_i^{'}$, sabemos que $[s(v_j^{'}) < s(v_i^{'})]_{R_t}$. Por (E0), tenemos que bajo R_t vale $t < j < i$ y por construcción de s, $(v_t \rightarrow v_j) \in E(\vec{Q_{R_t}})$ y $(v_j \rightarrow v_i) \in E(\vec{G_{R_t}})$. Por (B5), $(v_t \rightarrow v_i) \notin E(\vec{I_{R_t}})$ y como $t < i$ vale $(v_i, v_t) \notin E(I)$. Esto contradice las hipótesis, por lo cual el caso 1.2 no puede ocurrir.

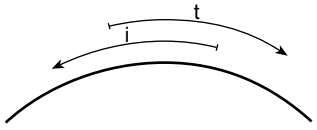
Caso 1.3: Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.



Bajo este caso el arco $[v_t^{'}, v_t^{''}]$ se solapa con el arco $[v_j^{'}, v_j^{''}]$. Sabemos que el arco $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_j^{'}, v_j^{''}]$ como a $[v_i^{'}, v_i^{''}]$. Luego, por el corolario 2.2.3, bajo R_t vale $t < j, i$.

Dada la posición de $v_j^{'}$, vale $[s(v_i^{'}) < s(v_j^{'})]_{R_t}$. Por (E0) tenemos que $t < i < j$ bajo R_t. Luego $(v_t \rightarrow v_i) \in E(\vec{I_{R_t}})$, pero por (B1) esto implica que $(v_t \rightarrow v_j) \in E(\vec{I_{R_t}})$. Esto contradice la hipótesis que afirma $(v_j, v_t) \in E(G)$, con lo cual el caso 1 no puede ocurrir.

Caso 2: $[s(v_i^{'}) < s(v_t^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i}$.



Veamos primero que bajo este caso $(v_i, v_t) \notin E(I)$. Supongamos por el absurdo lo contrario. Por (D1) sabemos que $v_t^{'}$ está entre $v_{t-1}^{'}$ y $v_{t+1}^{'}$. Por otro lado, $v_i^{''}$ está entre $v_t^{'}$ y $v_t^{''}$, por lo tanto $v_i^{''}$ está entre $v_{t-1}^{'}$ y $v_t^{''}$. Esto presenta dos alternativas:

Caso a: $v_i^{''}$ está entre $v_{t-1}^{'}$ y $v_{t+1}^{'}$. Por (D1), al insertar a $v_t^{'}$ se sobrepasó a $v_i^{''}$ y esto significa que $(v_i, v_t) \notin E(I)$. Esto es un absurdo, luego este caso no puede ocurrir.

Caso b: $v_i^{''}$ está entre $v_{t+1}^{'}$ y $v_t^{''}$. Esto implica que el arco $[v_i^{'}, v_i^{''}]$ corta al arco $[v_{t+1}^{'}, v_{t+1}^{''}]$ y por el corolario 2.2.3 bajo R_i vale $i < t+1$. Por construcción de s vale $[s(v_i^{'}) < s(v_{t+1}^{'}) < s(v_i^{''})]_{R_i}$, y por (C2) esto significa que $(v_i, v_{t+1}) \notin E(I)$. Por lo tanto $(v_i \rightarrow v_{t+1}) \notin E(\vec{I}_{R_i})$ y por ($\Leftarrow$B1), $(v_i \rightarrow v_t) \notin E(\vec{I}_{R_i})$. Como el arco $[v_i^{'}, v_i^{''}]$ corta al arco $[v_t^{'}, v_t^{''}]$, por el corolario 2.2.3 sabemos que bajo R_i vale $i < t$, luego $(v_i, v_t) \notin E(I)$, una contradicción.

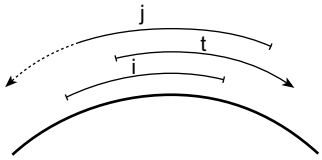
Por lo tanto $(v_i, v_t) \notin E(I)$. Como por hipótesis suponemos que $(v_i, v_t) \notin E(G)$, la única posibilidad es que $(v_i, v_t) \in E(Q)$.

Por (D2), no se siguió desplazando a $v_t^{''}$ en sentido anti-horario porque existe un extremo de arco entre $v_i^{''}$ y $v_t^{''}$ que cumple con una de las siguientes condiciones:

1. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.
2. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.
3. Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.

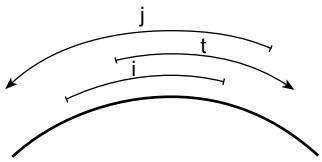
No puede ocurrir que $v_j^{''} = v_i^{''}$, dado que en ese caso estaríamos bajo la condición 1 e implicaría que $(v_i, v_t) \in E(G)$, contradiciendo la hipótesis.

Caso 2.1: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.



Este caso presenta dos alternativas:

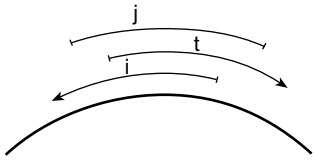
Caso 2.1.1: $v_j^{'}$ se encuentra entre $v_j^{''}$ y $v_i^{'}$.



Luego el arco $[v_i^{'}, v_i^{''}]$ está incluido en el arco $[v_j^{'}, v_j^{''}]$. Como el arco $[v_j^{'}, v_j^{''}]$ corta tanto a $[v_i^{'}, v_i^{''}]$ como a $[v_t^{'}, v_t^{''}]$, por el corolario 2.2.3, sabemos que bajo R_j vale $j < i, t$. Esto implica que $[s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_i^{''})]_{R_j}$. Por (C3), $(v_i, v_j) \in E(Q)$.

Como $t < i, j$, dadas las posiciones relativas de los extremos de los arcos, tenemos que $[s(v_i^{'}) < s(v_t^{'})]_{R_j}$. Por (E0), tenemos que bajo R_j vale $j < i < t$. Luego $(v_j \to v_i) \in E(\vec{Q_{R_j}})$ y $(v_i \to v_t) \in E(\vec{Q_{R_j}})$. Por (B2), $(v_j \to v_t) \in E(\vec{Q_{R_j}})$, lo que significa que $(v_j, v_t) \in E(Q)$. Esto contradice la hipótesis que afirma $(v_j, v_t) \in E(G)$, por lo cual este caso no puede ocurrir.

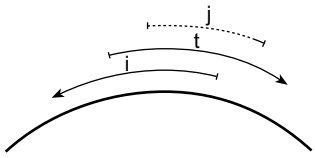
Caso 2.1.2: $v_j^{'}$ se encuentra entre $v_i^{'}$ y $v_t^{'}$.



En este caso el arco $[v_i^{'}, v_i^{''}]$ se solapa con el arco $[v_j^{'}, v_j^{''}]$. Como $[v_i^{'}, v_i^{''}]$ corta a $[v_j^{'}, v_j^{''}]$, por el corolario 2.2.3 sabemos que bajo R_i vale $i < j$. Luego vale $[s(v_i^{'}) < s(v_j^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i}$ y por (C1) tenemos que $(v_i, v_j) \in E(G)$.

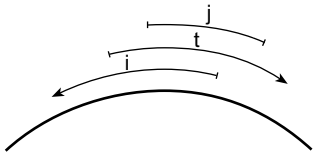
Como el arco $[v_i^{'}, v_i^{''}]$ también corta al arco $[v_t^{'}, v_t^{''}]$, por el corolario 2.2.3 sabemos que bajo el orden R_i vale $i < j, t$. Por las posiciones de los extremos de los arcos tenemos que $[s(v_i^{'}) < s(v_j^{'}) < s(v_t^{'})]_{R_i}$, con lo cual, por (E0) bajo R_i vale $i < j < t$. Esto implica que $(v_i \to v_j) \in E(\vec{G_{R_i}})$ y $(v_j \to v_t) \in E(\vec{G_{R_i}})$. Por (B3), $(v_i, v_t) \notin E(\vec{Q_{R_i}})$, un absurdo dado que por hipótesis $(v_i, v_t) \in E(Q)$.

Caso 2.2: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.



Aquí se presentan dos alternativas

Caso 2.2.1: $v_j^{'}$ se encuentra entre $v_t^{'}$ y $v_i^{''}$.

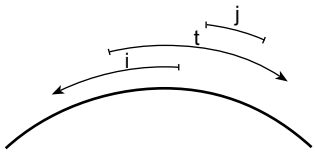


En este caso el arco $[v_i^{'}, v_i^{''}]$ se solapa con el arco $[v_j^{'}, v_j^{''}]$. Sabemos que el arco $[v_i^{'}, v_i^{''}]$ corta tanto a $[v_j^{'}, v_j^{''}]$ como a $[v_t^{'}, v_t^{''}]$. Luego, por el corolario 2.2.3, bajo R_i vale $i < j, t$. Por construcción de s vale $[s(v_i^{'}) < s(v_j^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_i}$, con lo cual por (C1), vale $(v_i, v_j) \in E(G)$.

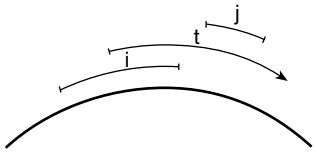
Dada la posición de $v_j^{'}$, y sabiendo que bajo R_i vale $i < j, t$ por construcción de s vale $[s(v_i^{'}) < s(v_t^{'}) < s(v_j^{'})]_{R_i}$, y por (E0), bajo R_i vale $i < t < j$.

Esto significa que $(v_i \to v_t) \in E(\vec{Q_{R_i}})$ y $(v_t \to v_j) \in E(\vec{Q_{R_i}})$. Por (B2), esto implica que $(v_i \to v_j) \in E(\vec{Q_{R_i}})$, contradiciendo el resultado anterior que afirma $(v_i, v_j) \in E(G)$. Por lo tanto este caso no puede ocurrir.

Caso 2.2.2: $v_j^{'}$ se encuentra entre $v_i^{''}$ y $v_j^{''}$.



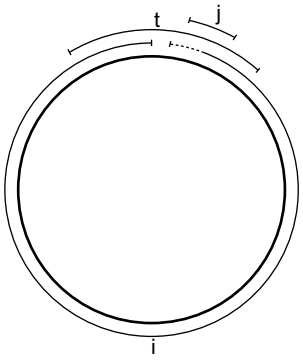
Caso 2.2.2.1: Supongamos que en la representación los arcos $[v_j^{'}, v_j^{''}]$ y $[v_i^{'}, v_i^{''}]$ no se intersecan.



Esto implica que $[s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$ y por (C2) $(v_i, v_j) \in E(I)$.

Como $[v_i^{'}, v_i^{''}]$ corta a $[v_t^{'}, v_t^{''}]$, por el corolario 2.2.3 bajo R_i vale $i < t$. Si bajo R_i tenemos que $j < i$, luego $(v_j \to v_i) \in E(\vec{I_{R_i}})$ y por (B1), $(v_j \to v_t) \in E(\vec{I_{R_i}})$. Esto es un absurdo dado que $(v_j, v_t) \in E(Q)$ por hipótesis. Por otro lado si bajo R_i vale $i < j$, por construcción de s tenemos que $[s(v_i^{'}) < s(v_t^{'}) < s(v_j^{'})]_{R_i}$. Esto por (E0) implica que bajo R_i vale $i < t < j$. Luego $(v_i \to v_t) \in E(\vec{Q_{R_i}})$ y $(v_t \to v_j) \in E(\vec{Q_{R_i}})$. Por (B2), $(v_i \to v_j) \in E(\vec{Q_{R_i}})$ y esto es un absurdo dado que habíamos llegado a que $(v_i, v_j) \in E(I)$. Luego este caso no puede ocurrir.

Caso 2.2.2.2: Supongamos que en la representación los arcos $[v_j^{'}, v_j^{''}]$ y $[v_i^{'}, v_i^{''}]$ se intersecan.



Como el arco $[v_j^{'}, v_j^{''}]$ está incluido en el arco $[v_t^{'}, v_t^{''}]$ y dada la posición de $v_j^{'}$, esto significa que $[v_i^{'}, v_i^{''}]$ junto con $[v_t^{'}, v_t^{''}]$ cubren todo el círculo. Luego,

por el corolario 2.2.3 bajo R_t vale $t < i$. Por otra parte como $(v_t \to v_i) \notin E(\vec{I_{R_t}})$, por ($\Leftarrow$B1), $(v_t \to v_{t+1}) \notin E(\vec{I_{R_t}})$ y como bajo R_t vale $t < t+1$, tenemos que $(v_t, v_{t+1}) \notin E(I)$ (este dato será usado más adelante).

Analicemos nuevamente la posición de $v_i^{''}$. Como habíamos dicho anteriormente, $v_i^{''}$ está entre $v_t^{'}$ y $v_t^{''}$, por lo tanto $v_i^{''}$ está entre $v_{t-1}^{'}$ y $v_t^{''}$. Esto presenta dos alternativas:

Caso a: $v_i^{''}$ está entre $v_{t-1}^{'}$ y $v_{t+1}^{'}$. Por (D1), al insertar a $v_t^{'}$ se sobrepasó a $v_i^{''}$ y como $(v_i, v_t) \notin E(I)$, bajo R_t debe valer $i < t$. Esto es un absurdo, luego este caso no puede ocurrir.

Caso b: $v_i^{''}$ está entre $v_{t+1}^{'}$ y $v_t^{''}$. Esto implica que el arco $[v_i^{'}, v_i^{''}]$ corta al arco $[v_{t+1}^{'}, v_{t+1}^{''}]$ y por el corolario 2.2.3 bajo R_i vale $i < t+1$. Por construcción de s vale $[s(v_i^{'}) < s(v_{t+1}^{'}) < s(v_i^{''})]_{R_i}$, y por (C2) esto significa que $(v_i, v_{t+1}) \notin E(I)$. Analicemos las posibilidades restantes sobre el arco (v_i, v_{t+1}):

Caso b1: $(v_i, v_{t+1}) \in E(Q)$.

Bajo R_i vale $i < t+1$ luego, por (C3) y (E0), esto significa que $[s(v_i^{'}) < s(v_{t+1}^{'}) < s(v_{t+1}^{''}) < s(v_i^{''})]_{R_i}$ y por la posición de los extremos de $[v_t^{'}, v_t^{''}]$, implica que $[s(v_t^{'}) < s(v_{t+1}^{'}) < s(v_{t+1}^{''}) < s(v_i^{''})]_{R_i}$. Esto significa que en la representación los arcos $[v_{t+1}^{'}, v_{t+1}^{''}]$ y $[v_j^{'}, v_j^{''}]$ no se intersecan. Esto implica que $[s(v_{t+1}^{''}) < s(v_j^{'})]_{R_j} \vee [s(v_j^{''}) < s(v_{t+1}^{'})]_{R_j}$ y por (C2) $(v_{t+1}, v_j) \in E(I)$.

Caso b1.1: $(v_i, v_j) \in E(Q)$.

Notemos que el arco $[v_t^{'}, v_t^{''}]$ corta a los arcos $[v_{t+1}^{'}, v_{t+1}^{''}]$, $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$. Luego, por el corolario 2.2.3, sabemos que bajo R_t vale $t < t+1, i, j$. Luego por construcción de s tenemos que $[s(v_t^{'}) < s(v_{t+1}^{'}) < s(v_i^{'}) < s(v_j^{'})]_{R_t}$ y por (E0) vale $t < t+1 < i < j$. Luego $(v_{t+1} \to v_i) \in E(\vec{Q_{R_t}})$, $(v_i \to v_j) \in E(\vec{Q_{R_t}})$ y por (B2) $(v_{t+1} \to v_j) \in E(\vec{Q_{R_t}})$, un absurdo.

Caso b1.2: $(v_i, v_j) \in E(G)$.

Como el arco $[v_t^{'}, v_t^{''}]$ corta a $[v_j^{'}, v_j^{''}]$, bajo R_t vale $t < j$. Además sabemos que $t < t+1$, y como $t+1$ es el inmediato siguiente a t bajo R_t tenemos que $t < t+1 < j$.

Supongamos que $(v_t, v_{t+1}) \in E(G)$. Luego $(v_t \to v_{t+1}) \in E(\vec{G_{R_t}})$, $(v_{t+1} \to v_j) \in E(\vec{I_{R_t}})$, y por (B4), $(v_t \to v_j) \in E(\vec{I_{R_t}})$. Como bajo R_t vale $t < j$, tenemos que $(v_t, v_j) \in E(I)$, un absurdo.

La otra alternativa es que $(v_t, v_{t+1}) \in E(Q)$. Recordemos que bajo R_i vale $i < t < t+1$.

Veamos que bajo R_i vale $j < i$. Si $v_j = v_{1_i}$ no hay nada que probar. Si, en cambio, $v_j \neq v_{1_i}$, como el arco $[v_j^{'}, v_j^{''}]$ corta a $[v_i^{'}, v_i^{''}]$ y ambos arcos no cubren todo el círculo, por el corolario 2.2.4 vale que $[v_{1_i}^{'}, v_{1_i}^{''}]$ corta a $[v_j^{'}, v_j^{''}]$. Por construcción de s esto implica que $[s(v_j^{'}) < s(v_i^{'})]_{R_i}$, y por (E0) esto significa que bajo R_i vale $i < j$.

Luego $(v_j \to v_t) \in E(\vec{Q_{R_i}})$, $(v_t \to v_{t+1}) \in E(\vec{Q_{R_i}})$ y por (B2) $(v_j \to v_{t+1}) \in E(\vec{Q_{R_i}})$. Esto es una contradicción, por lo cual este caso no puede ocurrir.

Caso b2: $(v_i, v_{t+1}) \in E(G)$.

Sabemos que el arco $[v_t^{'}, v_t^{''}]$ corta a $[v_{t+1}^{'}, v_{t+1}^{''}]$ y a $[v_i^{'}, v_i^{''}]$. Luego, por el corolario 2.2.3 bajo R_t vale $t < t+1, i$. Como $v_{t+1}^{'}$ es el extremo inmediato posterior a $v_t^{'}$ tenemos que $t < t+1 < i$ bajo R_t.

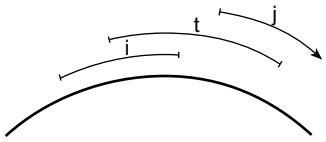
Si $(v_t, v_{t+1}) \in E(G)$ vale que $(v_t \to v_{t+1}) \in E(\vec{G_{R_t}})$, $(v_{t+1} \to v_i) \in E(\vec{G_{R_t}})$ y por (B3) $(v_t \to v_i) \notin E(\vec{Q_{R_t}})$. Esto es una contradicción, por lo tanto la única alternativa es que $(v_t, v_{t+1}) \in E(Q)$.

Si esto sucede, analicemos lo que pasa bajo R_i. El arco $[v_i^{'}, v_i^{''}]$ corta tanto a $[v_t^{'}, v_t^{''}]$ como a $[v_{t+1}^{'}, v_{t+1}^{''}]$. Luego, por el corolario 2.2.3, bajo R_i vale $i < t, t+1$ y esto por construcción de s implica que $[s(v_i^{'}) < s(v_t^{'}) < s(v_{t+1}^{'})]_{R_i}$. Luego, por (E0), en R_i vale $i < t < t+1$.

Entonces tenemos que $(v_i \to v_t) \in E(\vec{Q_{R_i}})$, $(v_t \to v_{t+1}) \in E(\vec{Q_{R_i}})$ y por (B2) $(v_i \to v_{t+1}) \in E(\vec{Q_{R_i}})$. Esto es una contradicción, dado que asumíamos que $(v_i, v_{t+1}) \in E(G)$.

En conclusión, en la representación los arcos $[v_j^{'}, v_j^{''}]$ y $[v_i^{'}, v_i^{''}]$ no se intersecan y el caso 2.2.2.2 no puede ocurrir.

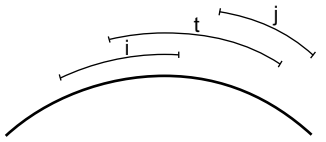
Caso 2.3: Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.



Esto implica que los arcos $[v_t^{'}, v_t^{''}]$ y $[v_j^{'}, v_j^{''}]$ no cubren todo el círculo. Por la posición de $v_i^{''}$ y $v_j^{'}$ en la representación, esto significa que el arco $[v_i^{'}, v_i^{''}]$ no puede estar incluido en el arco $[v_j^{'}, v_j^{''}]$.

Analicemos las diferentes alternativas en función de la posición de los arcos $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ en la representación.

Caso 2.3.1: Los arcos $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ no se intersecan.



Por construcción de s esto significa que $[s(v_i^{''}) < s(v_j^{'})]_{R_i} \vee [s(v_j^{''}) < s(v_i^{'})]_{R_i}$, con lo cual, por (C2), $(v_i, v_j) \in E(I)$.

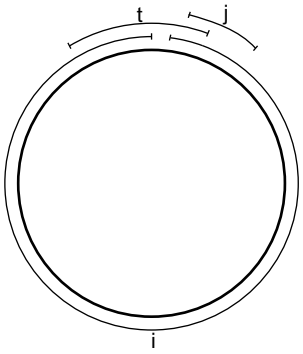
Veamos que ocurre bajo R_t. Como el arco $[v_t^{'}, v_t^{''}]$ corta al arco $[v_j^{'}, v_j^{''}]$, por el corolario 2.2.3 sabemos que bajo R_t vale $t < j$.

Por otro lado, el arco $[v_t^{'}, v_t^{''}]$ es cortado por el arco $[v_i^{'}, v_i^{''}]$. Veamos que bajo R_t vale $i < t$. Si $v_i = v_{1_t}$ no hay nada que probar. Si, en cambio, $v_i \neq v_{1_t}$, como $[v_t^{'}, v_t^{''}]$ y $[v_i^{'}, v_i^{''}]$ no cubren todo el círculo (porque sino $[v_i^{'}, v_i^{''}]$ y $[v_j^{'}, v_j^{''}]$ se intersecarían), por el corolario 2.2.4 tenemos que el arco $[v_{1_t}^{'}, v_{1_t}^{''}]$

corta al arco $[v_i^{'}, v_i^{''}]$. Por construcción de s esto significa que $[s(v_i^{'}) < s(v_t^{'})]_{R_t}$, y por (E0) bajo R_t vale $i < t$.

Luego vale $(v_i \to v_t) \in E(\vec{Q_{R_t}})$, $(v_t \to v_j) \in E(\vec{G_{R_t}})$ y por (B5) $(v_i \to v_j) \notin E(\vec{I_{R_t}})$. Como bajo R_t vale $i < j$, luego $(v_i, v_j) \notin E(I)$, lo cual es un absurdo.

Caso 2.3.2: El arco $[v_j^{'}, v_j^{''}]$ está incluido en el arco $[v_i^{'}, v_i^{''}]$.

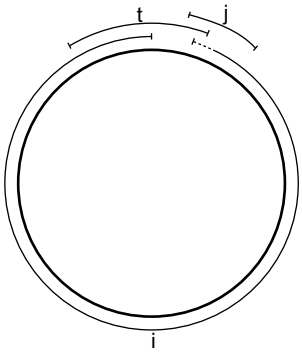


Bajo este caso el arco $[v_j^{'}, v_j^{''}]$ es cortado tanto por el arco $[v_t^{'}, v_t^{''}]$ como por el arco $[v_i^{'}, v_i^{''}]$. Veamos que bajo R_j vale $t < i < j$. Por la posición relativa de los extremos del arco $[v_i^{'}, v_i^{''}]$, sabemos que $v_i \neq v_{1_j}$, porque el arco $[v_t^{'}, v_t^{''}]$ cumple las condiciones para ser el primer vértice de R_j y numera a v_j con un número mayor. Como el arco $[v_i^{'}, v_i^{''}]$ junto con $[v_j^{'}, v_j^{''}]$ no cubren todo el círculo, por el corolario 2.2.4 sabemos que $[v_{1_j}^{'}, v_{1_j}^{''}]$ corta a $[v_i^{'}, v_i^{''}]$. Luego si $v_t = v_{1_j}$, por construcción de s vale $[s(v_t^{'}) < s(v_i^{'}) < s(v_j^{'})]_{R_j}$, y por (E0) bajo R_j vale $t < i < j$. Si $v_t \neq v_{1_j}$, como el arco $[v_t^{'}, v_t^{''}]$ junto con $[v_j^{'}, v_j^{''}]$ no cubren todo el círculo, por el corolario 2.2.4 sabemos que $[v_{1_j}^{'}, v_{1_j}^{''}]$ corta a $[v_t^{'}, v_t^{''}]$ y a $[v_i^{'}, v_i^{''}]$. Luego, por construcción de s vale $[s(v_t^{'}) < s(v_i^{'}) < s(v_j^{'})]_{R_j}$, y por (E0) bajo R_j vale $t < i < j$.

Por construcción de s, esto implica que $[s(v_i^{'}) < s(v_j^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_j}$, con lo cual, por (C3), $(v_i, v_j) \in E(Q)$

Luego vale $(v_t \to v_i) \in E(\vec{Q_{R_j}})$ y que $(v_i \to v_j) \in E(\vec{Q_{R_j}})$. Por lo tanto, por (B2) $(v_t \to v_j) \in E(\vec{Q_{R_j}})$. Como bajo R_j vale $t < j$, luego $(v_j, v_t) \in E(Q)$, lo cual es una contradicción según las hipótesis.

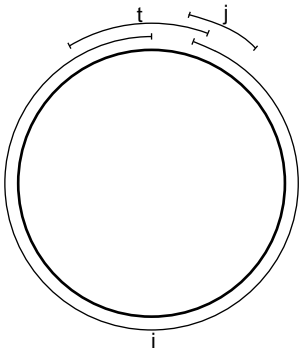
Caso 2.3.3: El arco $[v_j^{'}, v_j^{''}]$ se solapa con el arco $[v_i^{'}, v_i^{''}]$.



Como el arco $[v_j^{'}, v_j^{''}]$ corta a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.3 bajo R_j vale $j < i$. Veamos que además bajo R_j vale $t < j$. Si $v_t = v_{1_j}$ no hay nada que probar. Si, en cambio, $v_t \neq v_{1_j}$, como el arco $[v_t^{'}, v_t^{''}]$ junto con $[v_j^{'}, v_j^{''}]$ no cubren todo el círculo, por el corolario 2.2.4 sabemos que $[v_{1_j}^{'}, v_{1_j}^{''}]$ corta a $[v_t^{'}, v_t^{''}]$. Luego, por construcción de s vale $[s(v_t^{'}) < s(v_j^{'})]_{R_j}$, y por (E0) bajo R_j vale $t < j$.

Esto implica, por construcción de s, que $[s(v_j^{'}) < s(v_i^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_j}$. Por (C1) vale $(v_i, v_j) \in E(G)$.

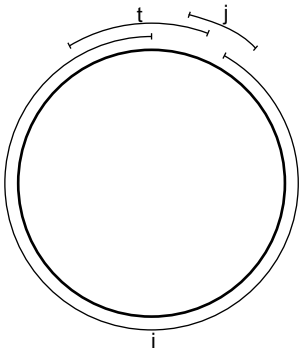
Caso 2.3.3.1: Supongamos que el arco $[v_i^{'}, v_i^{''}]$ junto con $[v_t^{'}, v_t^{''}]$ cubren todo el círculo.



Esto significa que $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_j^{'}, v_j^{''}]$ como a $[v_i^{'}, v_i^{''}]$, y por el corolario 2.2.3, bajo R_t vale $t < j, i$. Luego, por construcción de s tenemos que $[s(v_t^{'}) < s(v_j^{'}) < s(v_i^{'})]_{R_t}$, y por (E0), bajo R_t vale $t < j < i$.

Luego, tenemos que $(v_t \rightarrow v_j) \in E(\vec{G_{R_t}})$ y que $(v_j \rightarrow v_i) \in E(\vec{G_{R_t}})$. Por lo tanto, por (B3) $(v_t \rightarrow v_i) \notin E(\vec{Q_{R_t}})$. Como bajo R_t vale $t < i$, luego $(v_j, v_t) \notin E(Q)$, lo cual es una contradicción según las hipótesis.

Caso 2.3.3.2: Supongamos que el arco $[v_i^{'}, v_i^{''}]$ junto con $[v_t^{'}, v_t^{''}]$ no cubren todo el círculo.



Veamos que bajo R_t vale $i < t$. Si $v_i = v_{1_t}$ no hay nada que probar. Si, en cambio, $v_i \neq v_{1_t}$, como $[v_t^{'}, v_t^{''}]$ y $[v_i^{'}, v_i^{''}]$ no cubren todo el círculo, por el corolario 2.2.4 tenemos que el arco $[v_{1_t}^{'}, v_{1_t}^{''}]$ corta al arco $[v_i^{'}, v_i^{''}]$. Por construcción de s esto significa que $[s(v_i^{'}) < s(v_t^{'})]_{R_t}$, y por (E0) bajo R_t vale $i < t$. Además sabíamos que $t < j$, luego bajo R_t vale $i < t < j$.

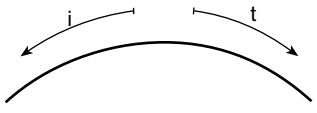
Veamos lo que sucede bajo R_i. Como $[v_i^{'}, v_i^{''}]$ corta a $[v_t^{'}, v_t^{''}]$, por el corolario 2.2.3, bajo R_i vale $i < t$. Si $v_j = v_{1_i}$ tenemos que bajo R_i vale $j < i < t$. Si esto no es así, como $[v_i^{'}, v_i^{''}]$ es cortado por $[v_j^{'}, v_j^{''}]$ y ambos arcos no cubren todo el círculo, por el corolario 2.2.4 tenemos que el arco $[v_{1_i}^{'}, v_{1_i}^{''}]$ corta a $[v_j^{'}, v_j^{''}]$. Por construcción de s esto significa que $[s(v_j^{'}) < s(v_i^{'})]_{R_i}$, y por (E0) bajo R_i vale $j < i < t$.

Entonces tenemos que bajo R_t vale $i < t < j$ y bajo R_i vale $j < i < t$. Como además sabemos que $(v_t, v_j) \in E(G)$, por propiedad de orden circular tenemos que $(v_t, v_i) \notin E(Q)$. Esto es una contradicción, por lo tanto el caso 2 no puede ocurrir y la única posibilidad es que $(v_i, v_t) \in E(G)$. Esto completa la prueba para E1. □

Prueba para (E2). ($\Leftarrow$): Asumamos que vale $[s(v_i^{''}) < s(v_t^{'})]_{R_i} \vee [s(v_t^{''}) < s(v_i^{'})]_{R_i}$ y supongamos que $(v_i, v_t) \notin E(I)$.

Veamos primero que ambos arcos no se intersecan, analizando los dos casos de la disyunción por separado:

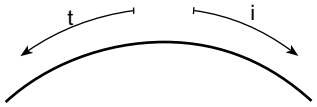
Caso 1: $[s(v_i^{'}) < s(v_i^{''}) < s(v_t^{'}) < s(v_t^{''})]_{R_i}$.



Por construcción de s, sabemos que el arco $[v_i^{'}, v_i^{''}]$ no corta al arco $[v_t^{'}, v_t^{''}]$. Veamos que tampoco pasa lo contrario. Asumamos que $[v_t^{'}, v_t^{''}]$ corta a $[v_i^{'}, v_i^{''}]$.

Por construcción de s sabemos que $v_t \neq v_{1_i}$ y que ambos arcos no cubren todo el círculo, luego por el corolario 2.2.4, vale que $[v'_{1_i}, v''_{1_i}]$ corta a $[v'_t, v''_t]$. Como $[v'_t, v''_t]$ corta a $[v'_i, v''_i]$, por construcción de s tenemos que $[s(v'_t) < s(v'_i)]_{R_i}$. Esto es un absurdo, dado que contradice las hipótesis. Esto significa que $[v'_t, v''_t]$ no corta a $[v'_i, v''_i]$, y que ambos arcos no se intersecan en la representación.

Caso 2: $[s(v'_t) < s(v''_t) < s(v'_i) < s(v''_i)]_{R_i}$.



Por construcción de s, sabemos que el arco $[v'_t, v''_t]$ no corta al arco $[v'_i, v''_i]$. Veamos que tampoco pasa lo contrario. Supongamos que $[v'_i, v''_i]$ corta a $[v'_t, v''_t]$. Luego, por el corolario 2.2.3, bajo R_i vale $i < t$. Luego, por (E0), $[s(v'_i) < s(v'_t)]_{R_i}$. Esto es una contradicción, con lo cual ambos arcos no se intersecan en la representación.

Por lo tanto, en ambos casos los arcos no se intersecan en la representación. Luego, por (D2), al posicionar v''_t se sobrepasó a v''_i y el arco $[v'_i, v''_i]$ dejó de estar incluido en el arco $[v'_t, v''_t]$, con lo cual por ($D2_2$), tenemos $(v_i, v_t) \notin E(Q)$. Como suponíamos que $(v_i, v_t) \notin E(I)$, esto implica que $(v_i, v_t) \in E(G)$.

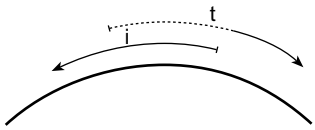
De la misma forma, por (D2), al posicionar v''_t se sobrepasó a v'_i y los arcos dejaron de intersecarse, con lo cual por ($D2_3$), tenemos $(v_i, v_t) \notin E(G)$, una contradicción. Por lo tanto este caso no puede ocurrir y la única posibilidad es $(v_i, v_t) \in E(I)$.

($\Rightarrow$): Asumamos que $(v_i, v_t) \in E(I)$ y supongamos que es falso que $[s(v''_i) < s(v'_t)]_{R_i} \vee [s(v''_t) < s(v'_i)]_{R_i}$.

Esto significa que vale $[s(v''_i) > s(v'_t)]_{R_i} \wedge [s(v''_t) > s(v'_i)]_{R_i}$, lo cual plantea cuatro casos posibles:

1. $[s(v'_i) < s(v'_t) < s(v''_t) < s(v''_i)]_{R_i}$
2. $[s(v'_i) < s(v'_t) < s(v''_i) < s(v''_t)]_{R_i}$
3. $[s(v'_t) < s(v'_i) < s(v''_i) < s(v''_t)]_{R_i}$
4. $[s(v'_t) < s(v'_i) < s(v''_t) < s(v''_i)]_{R_i}$

Casos 1 y 2: $[s(v_i^{'}) < s(v_t^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i} \vee [s(v_i^{'}) < s(v_t^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i}$.

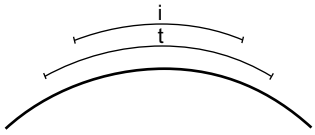


Por (D1) sabemos que $v_t^{'}$ está entre $v_{t-1}^{'}$ y $v_{t+1}^{'}$. Por otro lado, $v_i^{''}$ está entre $v_t^{'}$ y $v_i^{'}$, por lo tanto $v_i^{''}$ está entre $v_{t-1}^{'}$ y $v_i^{'}$. Esto presenta dos alternativas:

Caso a: $v_i^{''}$ está entre $v_{t-1}^{'}$ y $v_{t+1}^{'}$. Por (D1), al insertar a $v_t^{'}$ se sobrepasó a $v_i^{''}$ y esto significa que $(v_i, v_t) \notin E(I)$. Esto es un absurdo.

Caso b: $v_i^{''}$ está entre $v_{t+1}^{'}$ y $v_i^{'}$. Esto implica que el arco $[v_i^{'}, v_i^{''}]$ corta tanto al arco $[v_{t+1}^{'}, v_{t+1}^{''}]$ como a $[v_t^{'}, v_t^{''}]$, y por el corolario 2.2.3 bajo R_i vale $i < t, t+1$. Por construcción de s vale $[s(v_i^{'}) < s(v_t^{'}) < s(v_{t+1}^{'}) < s(v_i^{''})]_{R_i}$, y por (C2) esto significa que $(v_i, v_{t+1}) \notin E(I)$. Por (E0) también implica que bajo R_i vale $i < t < t+1$. Por lo tanto $(v_i \rightarrow v_{t+1}) \notin E(\vec{I}_{R_i})$ y por ($\Leftarrow$B1), $(v_i \rightarrow v_t) \notin E(\vec{I}_{R_i})$. Como bajo R_i vale $i < t$, luego $(v_i, v_t) \notin E(I)$, una contradicción. Por lo tanto no pueden ocurrir ni el caso 1 ni el caso 2.

Caso 3: $[s(v_t^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_t^{''})]_{R_i}$.

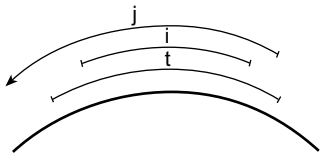


Por (D2), no se siguió desplazando a $v_t^{''}$ en sentido anti-horario porque existe un extremo de arco entre $v_i^{''}$ y $v_t^{''}$ que cumple con una de las siguientes condiciones:

1. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.
2. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.
3. Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.

Si $v_j^{''} = v_i^{''}$, estamos en la condición 2 y $(v_i, v_t) \in E(Q)$. Esto es un absurdo, luego el extremo que impidió el desplazamiento de $v_t^{''}$ no es $v_i^{''}$.

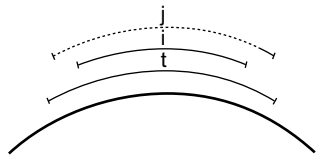
Caso 3.1: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.



Analicemos que sucede bajo el orden R_j en función de las posiciones relativas de los extremos de los arcos. Sabemos que el arco $[v_j^{'}, v_j^{''}]$ corta a $[v_t^{'}, v_t^{''}]$. Como $v_j^{''}$ está entre $v_i^{''}$ y $v_t^{''}$, el arco $[v_j^{'}, v_j^{''}]$ corta al arco $[v_i^{'}, v_i^{''}]$. Por el corolario 2.2.3, bajo R_j vale $j < t, i$. Como $[v_i^{'}, v_i^{''}]$ está incluido en el arco $[v_t^{'}, v_t^{''}]$, por construcción de s tenemos que $[s(v_j^{'}) < s(v_t^{'}) < s(v_i^{'})]_{R_j}$, y por (E0) bajo R_j vale $j < t < i$.

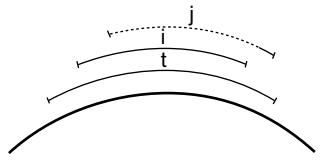
Por otra parte, vale $[s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_j}$, luego por (C3) tenemos que $(v_j, v_i) \in E(Q)$. Luego, $(v_j \to v_t) \in E(\vec{G}_{R_j})$ y $(v_t \to v_i) \in E(\vec{I}_{R_j})$. Por (B4), $(v_j \to v_i) \in E(\vec{I}_{R_j})$. Esto es un absurdo, por lo que este caso no puede ocurrir.

Caso 3.2: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.



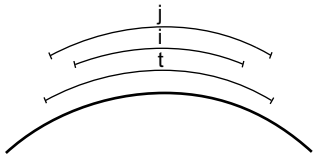
Como el arco $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_j^{'}, v_j^{''}]$ como a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.3, bajo R_t vale $t < i, j$. Veamos lo que sucede en función de la posición de $v_j^{'}$:

Caso 3.2.1: $v_j^{'}$ se encuentra entre $v_i^{'}$ y $v_j^{''}$.



Como $t < i, j$, por construcción de s vale $[s(v_t^{'}) < s(v_i^{'}) < s(v_j^{'})]_{R_t}$, y por (E0) bajo R_t vale $t < i < j$. Luego tenemos que $(v_t \to v_i) \in E(\vec{I}_{R_t})$, y por (B1), esto implica que $(v_t \to v_j) \in E(\vec{I}_{R_t})$. Como bajo R_t vale $t < j$, significa que $(v_j, v_t) \in E(I)$. Esto contradice las hipótesis, luego este caso no puede ocurrir.

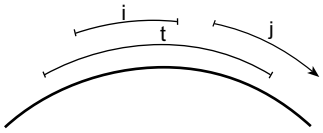
Caso 3.2.2: $v_j^{'}$ se encuentra entre $v_t^{'}$ y $v_i^{'}$.



Como el arco $[v_j^{'}, v_j^{''}]$ corta a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.3 sabemos que bajo R_j vale $j < i$. Esto implica, por construcción de s, que $[s(v_j^{'}) < s(v_i^{'}) < s(v_i^{''}) < s(v_j^{''})]_{R_j}$ y por (C3), que $(v_j, v_i) \in E(Q)$.

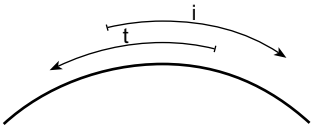
Como $t < i, j$, por construcción de s vale $[s(v_t^{'}) < s(v_j^{'}) < s(v_i^{'})]_{R_t}$, y por (E0) bajo R_t vale $t < j < i$. Luego $(v_t \to v_j) \in E(\vec{Q}_{R_t})$, $(v_j \to v_i) \in E(\vec{Q}_{R_t})$ y por (B2), $(v_t \to v_i) \in E(\vec{Q}_{R_t})$. Esto es un absurdo, por lo que el caso 3.2 no puede ocurrir.

Caso 3.3: Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.



Como el arco $[v_t^{'}, v_t^{''}]$ corta tanto al arco $[v_i^{'}, v_i^{''}]$ como al arco $[v_j^{'}, v_j^{''}]$, por el corolario 2.2.3 bajo R_t vale $t < i, j$. Por construcción de s vale $[s(v_t^{'}) < s(v_i^{'}) < s(v_j^{'})]_{R_t}$, y por (E0) bajo R_t vale $t < i < j$. Luego tenemos que $(v_t \to v_i) \in E(\vec{I}_{R_t})$, y por (B1), esto implica que $(v_t \to v_j) \in E(\vec{I}_{R_t})$. Como bajo R_t vale $t < j$, significa que $(v_j, v_t) \in E(I)$. Esto contradice las hipótesis, por lo tanto el caso 3 no puede ocurrir.

Caso 4: $[s(v_t^{'}) < s(v_i^{'}) < s(v_t^{''}) < s(v_i^{''})]_{R_i}$.



Bajo este caso, notemos que el arco $[v_i^{'}, v_i^{''}]$ no corta al arco $[v_t^{'}, v_t^{''}]$, porque si así fuera, el lema 2.2.3 nos diría que bajo R_i vale $i < t$ y por (E0) llegaríamos a que $[s(v_i^{'}) < s(v_t^{'})]_{R_i}$, un absurdo. Por lo tanto, ambos arcos no cubren todo el círculo.

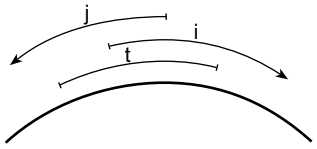
Por (D2), no se siguió desplazando a $v_t^{''}$ en sentido anti-horario porque existe un extremo de arco entre $v_i^{'}$ y $v_t^{''}$ que cumple con una de las siguientes condiciones:

1. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.

2. Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.
3. Existe un $v_j^{'}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ deja de intersecar a $[v_j^{'}, v_j^{''}]$.

Si $v_j^{'} = v_i^{'}$, estamos en la condición 3 y $(v_i, v_t) \in E(G)$. Esto es un absurdo, luego el extremo que impidió el desplazamiento de $v_t^{''}$ no es $v_i^{'}$.

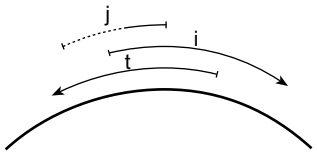
Caso 4.1: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t^{'}, v_t^{''}]$ queda incluido en $[v_j^{'}, v_j^{''}]$.



Analicemos que sucede bajo el orden R_j en función de las posiciones relativas de los extremos de los arcos. Sabemos que el arco $[v_j^{'}, v_j^{''}]$ corta tanto a $[v_t^{'}, v_t^{''}]$ como a $[v_i^{'}, v_i^{''}]$. Por el corolario 2.2.3, bajo R_j vale $j < t, i$. Por construcción de s tenemos que $[s(v_j^{'}) < s(v_t^{'}) < s(v_i^{'})]_{R_j}$, y por (E0) bajo R_j vale $j < t < i$.

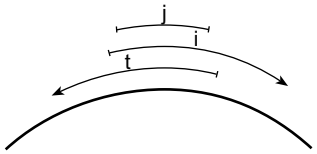
Por otra parte, vale $[s(v_j^{'}) < s(v_i^{'}) < s(v_j^{''}) < s(v_i^{''})]_{R_j}$, luego por (C1) tenemos que $(v_j, v_i) \in E(G)$. Luego, $(v_j \to v_t) \in E(\vec{G}_{R_j})$ y $(v_t \to v_i) \in E(\vec{I}_{R_j})$. Por (B4), $(v_j \to v_i) \in E(\vec{I}_{R_j})$. Esto es un absurdo, por lo que este caso no puede ocurrir.

Caso 4.2: Existe un $v_j^{''}$ tal que $(v_j, v_t) \in E(Q)$ y si se lo sobrepasa, $[v_j^{'}, v_j^{''}]$ deja de estar incluido en $[v_t^{'}, v_t^{''}]$.



Como el arco $[v_t^{'}, v_t^{''}]$ corta tanto a $[v_j^{'}, v_j^{''}]$ como a $[v_i^{'}, v_i^{''}]$, por el corolario 2.2.3, bajo R_t vale $t < i, j$. Veamos lo que sucede en función de la posición de $v_j^{'}$:

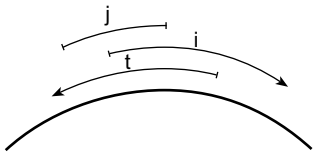
Caso 4.2.1: $v_j^{'}$ se encuentra entre $v_i^{'}$ y $v_j^{''}$.



Como $t < i, j$, por construcción de s vale $[s(v_t^{'}) < s(v_i^{'}) < s(v_j^{'})]_{R_t}$, y por (E0) bajo R_t vale $t < i < j$. Luego tenemos que $(v_t \to v_i) \in E(\vec{I}_{R_t})$, y por

(B1), esto implica que $(v_t \to v_j) \in E(\vec{I_{R_t}})$. Como bajo R_t vale $t < j$, significa que $(v_j, v_t) \in E(I)$. Esto contradice las hipótesis, luego este caso no puede ocurrir.

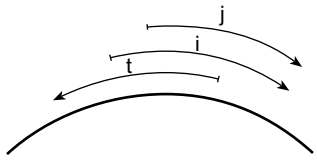
Caso 4.2.2: v_j' se encuentra entre v_t' y v_i'.



Como el arco $[v_j', v_j'']$ corta a $[v_i', v_i'']$, por el corolario 2.2.3 sabemos que bajo R_j vale $j < i$. Esto implica, por construcción de s, que $[s(v_j') < s(v_i') < s(v_j'') < s(v_i'')]_{R_j}$ y por (C1), que $(v_j, v_i) \in E(G)$.

Como $t < i, j$, por construcción de s vale $[s(v_t') < s(v_j') < s(v_i')]_{R_t}$, y por (E0) bajo R_t vale $t < j < i$. Luego $(v_t \to v_j) \in E(\vec{Q_{R_t}})$, $(v_j \to v_i) \in E(\vec{G_{R_t}})$ y por (B5), $(v_t \to v_i) \notin E(\vec{I_{R_t}})$. Esto es un absurdo, por lo que el caso 4.2 no puede ocurrir.

Caso 4.3: Existe un v_j' tal que $(v_j, v_t) \in E(G)$ y si se lo sobrepasa, $[v_t', v_t'']$ deja de intersecar a $[v_j', v_j'']$.



Como el arco $[v_t', v_t'']$ corta tanto a $[v_j', v_j'']$ como a $[v_i', v_i'']$, por el corolario 2.2.3, bajo R_t vale $t < i, j$.

Por construcción de s vale $[s(v_t') < s(v_i') < s(v_j')]_{R_t}$, y por (E0) bajo R_t vale $t < i < j$. Luego tenemos que $(v_t \to v_i) \in E(\vec{I_{R_t}})$, y por (B1), esto implica que $(v_t \to v_j) \in E(\vec{I_{R_t}})$. Como bajo R_t vale $t < j$, significa que $(v_j, v_t) \in E(I)$. Esto contradice las hipótesis, por lo tanto el caso 4 no puede ocurrir y la única posibilidad es que $[s(v_i'') < s(v_t')]_{R_i} \vee [s(v_t'') < s(v_i')]_{R_i}$. Esto completa la prueba para (E2).

Por lo tanto la hipótesis inductiva es cierta para grafos de n vértices. En particular (C1) implica que G es un grafo overlap de arco-circulares, con lo cual queda probado el teorema 2.3.2. □

Capítulo 3

Algoritmos en grafos HCA

En este capítulo presentamos un algoritmo de complejidad O($n.log(n)$) que encuentra el mínimo transversal de los cliques para grafos arco-circulares Helly, suponiendo que la representación en arcos circulares que verifica la propiedad de Helly es dada como parámetro de entrada. Realizando unas pequeñas modificaciones a este algoritmo, construimos un segundo algoritmo para determinar el máximo número de cliques disjuntos para estos grafos, también con una complejidad de O($n.log(n)$).

Los problemas de conjunto transversal de los cliques y conjunto de cliques disjuntos fueron estudiados en el transcurso de los últimos diez años. Pueden encontrarse referencias en la literatura en [10], [1], [2], [3], [5], [11] y [36]. En [21] se presentan dos algoritmos para resolver estos problemas en grafos arco-circulares Helly con una complejidad de O(n^2).

Los dos algoritmos presentados son de tipo goloso. Ambos se diferencian por completo, en cuanto a la manera de resolver el problema, de los propuestos en [21] cuya formulación es mucho más compleja. Tanto los algoritmos presentados en [21] como los propuestos en esta tesis toman como parámetro de entrada una representación arco-circular Helly. Está demostrado en [16] que dado un grafo HCA es posible construir una representación en arcos circulares que verifique la propiedad de Helly con un orden de complejidad de O(n^3).

Recordemos que una familia de subconjuntos S satisface la propiedad de Helly cuando toda subfamilia de S consistente en subconjuntos que se intersecan de a pares tiene intersección no vacía. Un grafo G es arco-circular Helly (HCA) si existe una representación arco-circular de G tal que los arcos satisfacen la propiedad de Helly.

Los algoritmos son presentados en forma de pseudo-código, con el nivel de detalle suficiente para justificar su orden de complejidad y funcionamiento. La implementación de los algoritmos fue realizada en el lenguaje de programación Java.

3.1 Mínimo transversal de los cliques

El problema del mínimo transversal de los cliques consiste en encontrar un cubrimiento mínimo de cliques con vértices. Es decir, dado un grafo G, se trata de encontrar un conjunto de vértices tal que todos los cliques de G contengan al menos algún vértice de este conjunto y el cardinal del conjunto sea mínimo.

3.1.1 Definiciones y propiedades

Los grafos arco-circulares Helly fueron caracterizados y reconocidos en [15]. Allí se prueba que si G es un grafo HCA y φ es una posible representación Helly, los cliques de G se identifican encontrando los puntos de intersección entre los arcos de φ. Esto es fácil de ver, porque claramente un punto de intersección es un subgrafo completo. Si no fuera clique, habría un conjunto de arcos que no verifica la propiedad de Helly.

También se probó en [15] que el número de cliques de un grafo HCA está acotado superiormente por n.

A continuación presentamos algunas definiciones que nos serán útiles a lo largo de este capítulo:

Sea G un grafo HCA, y φ una posible representación Helly de G. Sea C un arco de φ. Si C interseca a todos los otros arcos de φ, diremos que C es un *arco universal* de G. Un arco C es un *arco propio* de φ si no está contenido por ningún otro arco de φ.

Denotamos el conjuntos de los arcos de la representación arco-circular φ como $A = \{A_1, A_2, ..., A_n\}$ donde $A_j = [e_{i_j}, e_{f_j}]$ con $e_{i_j}, e_{f_j} \in [0, 2\Pi)$. Por convención, tomaremos a los ángulos creciendo en sentido horario. Cuando sobre una representación φ hablamos de 'por la izquierda' y 'por la derecha', nos referimos a desplazarse sobre el círculo en sentido anti-horario y horario respectivamente.

3.1.2 Presentación del algoritmo

El algoritmo propuesto para encontrar el mínimo transversal de los cliques de un grafo G opera sobre una representación φ de G que cumple la propiedad de Helly y consta de varias fases. Presentaremos primero una idea general de cada fase, para después detallar cada una justificando su complejidad algorítmica y la estructura de datos que utiliza.

1. **Inicialización**. Precalcula datos sobre la representación de G que luego son usados por el resto de las fases. Posee una complejidad de $O(n.log(n))$.

2. **Extremos distintos**. Modifica la representación φ para generar otra representación equivalente, pero garantizando que los extremos de los arcos son todos distintos. Su complejidad es de $O(n)$.

3. **Arco universal y propios**. Detecta si una representación φ posee un arco universal. También detecta los arcos propios de φ. Esta fase requiere que todos los extremos de los arcos posean valores distintos. Cuando se detecta un arco universal, la solución del problema es dicho arco, dado que el vértice correspondiente a este arco está presente en todos los cliques del grafo intersección. Posee un complejidad de $O(n)$.

4. **Puntos de intersección**. Esta fase identifica los puntos de intersección de φ, que como vimos, corresponden a los cliques del grafo intersección. Eventualmente puede detectar dos arcos que cubren todo el círculo. Si esto ocurre, entonces la solución del problema son estos dos arcos, ya que cualquier clique del grafo intersección debe contener al menos uno de los vértices correspondientes a ellos, y por la ausencia de vértice universal (lo garantiza la fase anterior), es la mejor solución posible. Esta fase posee una complejidad de $O(n)$.

5. **Filtro y precálculos**. Elimina de la representación los arcos no propios. Luego calcula para cada arco propio cuál es el punto de intersección que está fuera del arco y que está lo más cerca posible por la derecha (vale aclarar que a esta altura, no puede existir un arco universal). A continuación, para cada punto de intersección determina cuál es el arco propio que lo contiene y se extiende lo más posible hacia la derecha (para todo punto de intersección siempre hay un arco propio que lo contiene). Estos cálculos se realizan mediante búsqueda binaria. La complejidad de esta fase es de $O(n.log(n))$.

6. **Búsqueda**. Esta fase toma un punto de intersección cualquiera (el algoritmo toma el primero de ellos), y busca cuál es el arco que pasa por encima de él y llega lo más a la derecha posible. Luego calcula cuál es el primer punto de intersección que no cubre este arco por la derecha, y se repite nuevamente el procedimiento hasta que vuelva a un punto de intersección que haya pasado previamente. Este punto de intersección será el resultado de esta fase. Este procedimiento posee una complejidad de $O(n)$.

7. **Principal**. El punto de intersección resultante de la fase de búsqueda, es de alguna manera considerado como un punto 'bueno'. Esto significa que aplicando a partir de este punto el procedimiento goloso que se usó en la búsqueda se puede encontrar la solución óptima al problema. Esta fase posee una complejidad de $O(n)$.

3.1.3 Tipos de datos principales

Veamos primero los tipos de datos principales utilizados por el algoritmo:

Tipo Arco

Un elemento de tipo Arco es una tupla con dos campos de tipo real. Representa a un arco de una representación arco-circular, donde los dos componentes de la tupla representan al ángulo inicial y final del arco.

$$< \text{ánguloInicial: } \mathbb{R}, \text{ánguloFinal: } \mathbb{R} >$$

Tipo Extremo

Un elemento de tipo Extremo representa a un extremo de un arco. Está modelado mediante una tupla, cuyos elementos son: el ángulo del extremo, si es un extremo inicial o final y una referencia al arco que conforma.

$$< \text{ángulo: } \mathbb{R}, \text{ tipo: } \{\text{Inicial} \mid \text{Final}\}, \text{ referencia: Int} >$$

Tipo RepresentaciónAC

Modela una representación arco-circular de un grafo. Lo constituye una tupla con dos elementos, un arreglo de elementos de tipo Arco y otro arreglo de elementos de tipo Extremo. Si bien almacena información redundante, esto nos será útil para mejorar el orden en algunas operaciones.

$$< \text{arcos: [Arco], extremos: [Extremo]} >$$

3.1.4 Fase 1: Inicialización

La fase de inicialización construye un elemento de tipo RepresentaciónAC en función de un arreglo de elementos de tipo Arco, que recibe como parámetro. Adicionalmente ordena el arreglo de extremos de la representación en función de su ángulo. Si el ángulo coincide, se consideran menores los extremos iniciales de un arco que los finales:

Parámetros: $arcos : [Arco]$

- *rep:RepresentaciónAC*
- Para cada arco a_i del arreglo *arcos*
  - Crear e_1*:Extremo*, con los valores correspondientes al ángulo inicial de a_i.
  - Crear e_2*:Extremo*, con los valores correspondientes al ángulo final de a_i.
  - Agregar e_1 y e_2 a *rep.extremos*
- Ordenar *rep.extremos* por ángulo ascendentemente. En caso de coincidir ordenar por tipo de extremo, siendo menor un extremo inicial que uno final.

Complejidad de esta fase del algoritmo

La generación del arreglo *rep.extremos* posee una complejidad de $O(n)$, dado que se recorren todos los arcos de la representación y por cada uno se crean dos elementos de tipo Extremo. La creación de un elemento de tipo Extremo posee orden constante.

Luego se ordena el arreglo *rep.extremos* por su ángulo/tipo. Esta operación se realiza en un orden de $O(n.log(n))$, utilizando un algoritmo de ordenamiento de tipo Heap Sort.

Por lo tanto, esta fase posee un orden de $O(n.log(n))$.

3.1.5 Fase 2: Extremos distintos

Esta fase se encarga de garantizar que los siguientes sub-algoritmos reciban una representación equivalente con todos sus extremos distintos. Recibe como parámetro un elemento de tipo RepresentaciónAC, con sus extremos ordenados mediante el algoritmo de la fase 1.

Este algoritmo recorre los extremos de los arcos de la representación consecutivamente, es decir, de forma ascendente por ángulo. Notemos que los elementos del arreglo de extremos que poseen el mismo ángulo y tipo son consecutivos en el arreglo, con lo cual se pueden identificar todos los grupos de extremos iguales en orden O(n). Veamos también que dado un extremo 'e', su extremo más cercano por la derecha y por la izquierda son sus vecinos inmediatos a izquierda y a derecha en el arreglo (visto circularmente). Esto significa que la operación de encontrar al extremo más cercano por la derecha o por la izquierda posee un orden O(cte).

El algoritmo agrupa los extremos de acuerdo a su valor (ángulo) y el tipo de extremo: Final o Inicial. Para un grupo de k extremos de tipo 'Inicial' que tienen el mismo valor 'a', si el extremo más cercano por la izquierda tiene valor 'b' entonces estos k extremos son redistribuidos con los siguientes valores:

$$a - \frac{k(a-b)}{3k}, a - \frac{(k-1)(a-b)}{3k}, ..., a - \frac{2(a-b)}{3k}, a - \frac{(a-b)}{3k}$$

Para un grupo de k extremos de tipo 'Final' que tienen el mismo valor 'a', y el extremo más cercano por la derecha tiene valor 'b' entonces estos extremos son reubicados de la siguiente forma:

$$a + \frac{(b-a)}{3k}, a + \frac{2(b-a)}{3k}, ..., a + \frac{(k-1)(b-a)}{3k}, a + \frac{k(b-a)}{3k}$$

Notemos que el arreglo de extremos sigue ordenado después de estas redistribuciones.

El algoritmo devuelve un nuevo elemento de tipo RepresentaciónAC, pero con todos sus extremos distintos. Presentamos a continuación el pseudocódigo del algoritmo:

Parámetros: rep : $RepresentacionAC$

- *resultado:RepresentacionAC*
- resultado := rep
- Recorrer *rep.extremos* identificando grupos
  - Por cada grupo $k = \{e_0, ..., e_{k-1}\}$ de extremos con el mismo valor y tipo
    - Si el grupo es de extremos de tipo *Inicial*
      - Encontrar el extremo m menor a e_0 módulo 2Π más cercano por la izquierda

- Para cada extremo e_i del grupo k
  - $resultado.extremos[i] := e_i.angulo - \frac{(k-i)(e_i.angulo - m.angulo)}{3k}$
  - Ajustar $resultado.extremos[i]$ módulo 2Π
- Si el grupo es de extremos de tipo $Final$
- Encontrar el extremo m mayor a e_{k-1} módulo 2Π más cercano por la derecha
- Para cada extremo e_i del grupo k
  - $resultado.extremos[i] := e_i.angulo + \frac{i(m.angulo - e_i.angulo)}{3k}$
  - Ajustar $resultado.extremos[i]$ módulo 2Π
- Devolver $resultado$

Complejidad y corrección de esta fase del algoritmo

Por construcción, esta nueva representación no tiene extremos repetidos. Veamos ahora que la representación es equivalente. Comparando la representación generada con la original, los arcos han extendido sus dos extremos, por lo tanto si dos arcos se intersecan en la representación original, también se intersecan en la nueva. Ahora si dos arcos no se intersecan en la representación original, tampoco se intersecan en la nueva por la forma de redistribuir los extremos. Por lo tanto la representación generada es equivalente a la original, pero todos sus extremos son distintos.

Analicemos ahora su complejidad. El algoritmo recorre los $2n$ extremos de arcos de la representación y a medida que los va recorriendo identifica los grupos de arcos con el mismo valor. Como los extremos están ordenados por ángulo/tipo, los elementos de estos grupos se ubican consecutivamente dentro del arreglo, por lo tanto identificar todos los grupos posee un orden de O($2n$).

En el caso de ser un grupo de extremos iniciales se intenta encontrar el extremo menor (módulo 2Π) más cercano por la izquierda. Por el orden que poseen los elementos del arreglo, si en el arreglo el primer elemento del grupo es $e[i]$, el extremo menor más cercano por la izquierda es $m = e[i-1]$ (módulo 2Π).

Por otro lado, en el caso de ser un grupo de extremos finales se intenta encontrar el extremo mayor (módulo 2Π) más cercano por la derecha. Por el orden que poseen los elementos del arreglo, si en el arreglo el último elemento del grupo es $e[i]$, el extremo mayor más cercano por la derecha es $m = e[i+1]$ (módulo 2Π).

Luego, tanto si se trata de un grupo de extremos iniciales como finales, la búsqueda del extremo m posee un orden O(cte). Una vez encontrado el extremo m, a cada extremo del grupo se lo reubica siguiendo la fórmula presentada, cuyo cálculo también posee orden constante.

Suponiendo, en el peor de los casos, que todo extremo de la representación pertenezca a un grupo de extremos iguales, el algoritmo realiza $2n$

reubicaciones de orden constante. Resumiendo, en total tenemos $2n$ operaciones para identificar todos los grupos de extremos con el mismo valor, más $2n$ operaciones para reubicar los extremos de dichos grupos. En total son $4n$ operaciones, por lo que esta fase posee un orden $O(n)$.

3.1.6 Fase 3: Arco universal y propios

Esta fase identifica los arcos propios de una representación arco-circular y, si existe, identifica un arco universal. Recibe como parámetro un elemento de tipo RepresentaciónAC, con todos sus extremos distintos, y ordenados mediante el algoritmo de la fase 1.

El algoritmo utiliza como tipo de dato adicional un tipo *Cola*, que modela una cola de extremos de arcos. Un objeto de tipo *Cola* posee las operaciones estandar de una cola de elementos.

Utilizamos dos propiedades, fácilmente verificables, para encontrar el arco universal:

1. Si hay un arco universal, siempre hay un arco propio que verifica esa condición.

2. Si un arco $A = [e_i, e_f]$ es universal, entonces dentro de $[e_i, e_f]$ debe encontrarse al menos un extremo de cada uno de los n arcos de la representación.

De la condición 1 se desprende que podemos descartar aquellos arcos que sabemos que no son propios. La condición 2 nos permite concluir que si sabemos la cantidad de extremos que se encuentran en el intervalo $[e_i, e_f]$, y la cantidad de arcos cuyos extremos están ambos incluidos en $[e_i, e_f]$, podemos determinar si A es universal o no.

El algoritmo recorre los extremos de los n arcos en sentido horario, avanzando por el arreglo de extremos de la representación. A partir del primer extremo e_i de tipo $Inicial$, comienza a registrar los sucesivos extremos que va visitando. Se contabiliza cuántos arcos tienen sus dos extremos registrados mediante la variable *contadorArcos*.

En caso que el algoritmo detecte un extremo de tipo $Final$ que se encuentre después de haber registrado su correspondiente extremo $Inicial$ en algún paso anterior (y no sea precisamente e_i), significa que el arco identificado está contenido en el arco cuyo extremo inicial es e_i, y por lo tanto este arco es marcado como "no propio". Cuando el algoritmo encuentra el extremo de tipo $Final$, e_f, cuyo extremo inicial es e_i, el algoritmo está en condiciones de determinar si el arco $[e_i, e_f]$ es un arco universal (por la propiedad 2).

A continuación el algoritmo descarta el primer extremo e_i y los subsiguientes extremos registrados hasta que encuentra un extremo de tipo $Inicial$ de algún arco que no esté marcado como "no propio". Luego vuelve al principio, pero con información acerca de los extremos registrados que no fueron descartados.

Parámetros: $rep : RepresentacionAC$

- $indice : int := 0$
- $contadorArcos : int := 0$
- $c : Cola$
- $arco : Arco$
- Mientras la cantidad de arcos cerrados de rep sea menor a n
  - Si c no posee elementos, posicionar $indice$ en el primer extremo de rep de tipo $Inicial$.
  - Asignar a $arco$ el arco al que pertenece $rep.extremos[indice]$.
  - Incrementar en uno la cantidad de visitas a $arco$.
  - Si ya se visitó dos veces a $arco$
    - Incrementar $contadorArcos$
  - Agregar a c el extremo $rep.extremos[indice]$.
  - Si $rep.extremos[indice]$ es un extremo de tipo $Final$
    - Si visité los dos extremos del arco $arco$.
      - Marcar a $arco$ como Cerrado
      - Si el extremo inicial de $arco$ es el primer extremo de la cola
        - Si $c.cantidadDeElementos() - contadorArcos = n$
          - Marcar a $arco$ como Universal
        - Hacer
          - Decrementar en uno la cantidad de visitas del primer elemento de c.
          - Si la cantidad de visitas del primer elemento de c es igual a uno, decrementar en uno $contadorArcos$
          - Desencolar al primer elemento de c
        - Mientras no pase que $c.cantidadDeElementos() = 0$ o el primer elemento de c sea un extremo de un arco no propio de tipo $Inicial$
      - Si no
        - Marcar a $arco$ como No Propio.
  - Incrementar $indice$ módulo $2n$

Complejidad y corrección de esta fase del algoritmo

Veamos que el algoritmo es correcto. Si un arco es propio, no está contenido en ningún otro arco, y por el funcionamiento del algoritmo nunca va a ser marcado como "no propio". Además, en algún momento su extremo $Inicial$ será tomado como primer primer extremo (e_i) , por lo que será examinado por el test de arco universal. Este comportamiento, más la propiedad 1, nos

asegura la corrección del algoritmo para detectar un arco universal, si éste existe.

Ahora si un arco no es propio, está contenido en algún arco propio, por lo tanto debe ser descubierto como "no propio"durante el análisis de este último. Por lo tanto, el algoritmo funciona correctamente para detectar arcos propios.

Analicemos el orden del algoritmo. La condición de parada del mismo se cumple cuando todos los arcos están "cerrados". El algoritmo marca un arco como "cerrado"cuando visita su extremo $Final$ luego de visitar en algún paso anterior a su extremo $Inicial$ (para el mismo e_i). Esto significa que los extremos de los arcos son visitados hasta dos veces, porque dado un arco a_i cualquiera, a lo sumo en la segunda vuelta a la circunferencia los extremos se visitan en un orden tal que el arco a_i es marcado como "cerrado". En total, se realizan a lo sumo $4n$ visitas a los $2n$ extremos de los arcos, por lo tanto este algoritmo posee una complejidad de $O(n)$.

3.1.7 Fase 4: Puntos de intersección

Esta fase identifica los puntos de intersección de una representación arco-circular Helly o bien devuelve dos arcos que cubren todo el círculo. Como la representación es arco-circular Helly, los puntos de intersección representan los cliques del grafo.

El algoritmo requiere que los extremos de la representación sean todos distintos. También se asume que la representación no posee un arco universal. Esta fase es ejecutada luego de la fase 3 con lo cual estas condiciones son cumplidas. Adicionalmente, los extremos de los arcos quedaron ordenados con el criterio de la fase 1.

Veamos conceptualmente su funcionamiento. Este algoritmo examina en el sentido horario cada par de extremos contiguos, interesándose sólo en aquellos pares donde el primer extremo es de tipo $Inicial$ y el segundo de tipo $Final$.

En el caso que los dos extremos del par sean de un mismo arco, cualquier otro arco que lo interseca lo contiene. Por lo tanto todos los arcos que lo intersecan forman un clique y el punto de intersección correspondiente al clique puede ser cualquier punto que esté entre estos dos extremos (el algoritmo elige el punto medio entre ellos).

Ahora si los dos extremos no son de un mismo arco es que tenemos dos extremos a_i y b_j, con $i \neq j$. Si analizamos el intervalo $[a_i, b_j]$ vemos que $[a_i, b_j] \subseteq (A_i \cap Aj)$. Si la contención es estricta, $[b_i, a_j] \cap (A_i \cap A_j)$ es no vacía. Esto significa que A_i y A_j cubren todo el círculo, con lo cual si se llega a este caso el algoritmo termina devolviendo estos dos arcos. Por otro lado, si $[a_i, b_j] = A_i \cap A_j$, para cualquier otro arco A_k que interseque con A_i y A_j sucede que $[a_i, b_j] \subseteq A_k$. Por lo tanto el clique formado por los arcos que contienen a $[a_i, b_j]$ puede ser representado por cualquier punto entre a_i y b_j (el algoritmo elige el punto medio entre ellos).

Parámetros: rep : $RepresentacionAC$

- Para cada extremo e_i de $rep.extremos$
  - Sea e_j el extremo consecutivo a e_i por la derecha
  - Si e_i es un extremo $Inicial$ y e_j es un extremo $Final$
    - Si e_i y e_j pertenecen al mismo arco
      - Agregar al punto medio entre e_i y e_j al conjunto de puntos de intersección
    - Si no
      - Sean A_i y A_j los arcos correspondientes a e_i y e_j respectivamente
      - Si A_i y A_j cubren todo el círculo
        - Devolver como resultado A_i y A_j. Terminar
      - Si no
        - Agregar al punto medio entre e_i y e_j al conjunto de puntos de intersección

Complejidad y corrección de esta fase del algoritmo

Primero veamos que el algoritmo no omite ningún clique. Como sabemos, un clique del grafo intersección corresponde a un conjunto maximal de arcos con intersección en común. La intersección de un conjunto de arcos está conformada por una o dos secciones continuas sobre el círculo, a las que llamaremos *tramos*. Un tramo es una sección maximal continua dentro del círculo, donde sobre cualquier punto del tramo se intersecan todos los arcos de un clique dado. En la figura 3.1 podemos ver como quedan definidos los tramos de una representación arco-circular. Los tramos 1 y 2 pertenecen a cliques distintos, mientras que el 3 y el 4 definen la intersección de un mismo clique. Notemos que como máximo un clique puede tener dos tramos de intersección, porque de otro modo la representación no cumpliría la propiedad de Helly.

Tanto si la intersección de un conjunto de arcos está formada por uno o dos tramos, cada tramo debe comenzar y terminar sobre algún extremo de los arcos que se intersecan, ya que los otros puntos no imponen ninguna condición al respecto. Además, el extremo donde comienza el tramo debe ser de tipo $Inicial$, porque de ser $Final$ el arco al que pertenece este extremo no podría tener intersección más allá del extremo en cuestión. De manera análoga, el extremo que marca el fin del tramo debe ser de tipo $Final$.

Veamos que dentro de un tramo no puede haber otro extremo. Supongamos por el absurdo que esto ocurre. Si el arco al que pertenece este extremo es uno de los arcos que pertenecen a la intersección del tramo, entonces este tramo debería ser más pequeño ya que debería comenzar antes o terminar después (en el sentido horario) dependiendo si el extremo es $Inicial$ o $Final$. Esto es así porque de otra manera habría puntos del tramo sobre los cuales

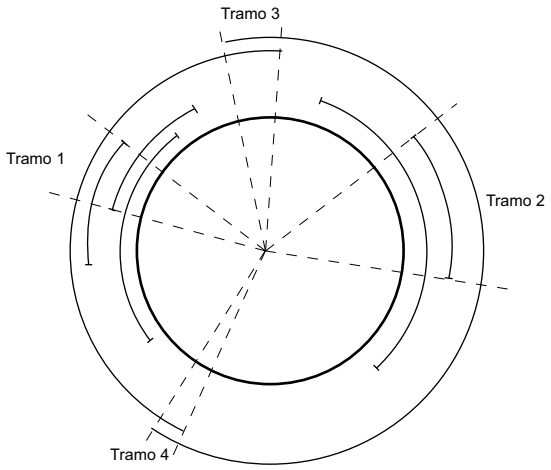


Figura 3.1: Tramos definidos sobre una representación.

no intersecarían todos arcos de la intersección. Si, por otro lado, el extremo es de un arco que no interviene en la intersección, entonces este arco interseca a todos los arcos que intervienen en la intersección, y esto significa que el conjunto original de arcos no es maximal con lo que el clique no es tal, un absurdo.

Hasta ahora vimos que todo clique del grafo intersección puede tener uno o varios tramos sobre el círculo. Cada uno de estos tramos comienza en un extremo de tipo $Inicial$ y termina en un extremo contiguo de tipo $Final$, que son exactamente los casos que examina el algoritmo. Para los cliques que poseen un solo tramo de intersección, el algoritmo determina un punto de intersección en forma exitosa. Para los que tienen dos tramos de intersección, habría que verificar que el algoritmo no determine varios puntos de intersección para un mismo clique.

Supongamos un clique C que tiene dos tramos de intersección, y elijamos un tramo $[a_i, b_j]$ cualquiera. Es obvio que $i \neq j$, porque sino el arco $A_i = \{a_i, b_i\}$ no puede contener a otros tramos de intersección. Por lo tanto, los arcos A_i y A_j contienen otros tramos de intersección, y esto implica que A_i se extiende en el sentido horario y A_j en el anti-horario. Como existe otro tramo distinto a $[a_i, b_j]$ y ambos arcos deben estar presentes, esto implica que los arcos A_i y A_j cubren todo el círculo. Llegado este punto, el algoritmo detecta este caso, deja de buscar puntos de intersección y devuelve a A_i y a A_j.

Por último habría que verificar que el algoritmo sólo determina puntos de intersección correspondientes a los cliques del grafo. Supongamos que no es así, y que el algoritmo identifica un punto de intersección p que no se corresponde con ningún clique del grafo. Por la manera que el algoritmo identifica puntos, p es el punto medio de una intersección $[a_i, b_j]$. Si $i = j$, significa que p identifica al clique representado por un arco aislado, con lo cual llegamos a un absurdo. Si $i \neq j$, el punto de intersección p identifica a dos arcos A_i y A_j

que se intersecan. Como la representación cumple la propiedad de Helly, y no hay ningún extremo de arco entre a_i y b_j, esto significa que todos los arcos que poseen intersección no vacía con $[a_i, b_j]$ forman un subgrafo completo. Si no fuera clique, habría un conjunto de arcos que no verifica la propiedad de Helly, con lo cual llegamos nuevamente a un absurdo. En conclusión, el algoritmo sólo identifica puntos que se corresponden con cliques del grafo.

Analicemos ahora la complejidad. El algoritmo recorre el conjunto de extremos del grafo en orden (recordar que el arreglo de extremos está ordenado según la fase 1). Para cada extremo se obtiene el extremo consecutivo por la derecha y dada la forma en que están ordenados los extremos, esta operación posee orden O(cte). Luego, a partir de cada extremo y su consecutivo, se verifica su tipo y si pertenecen al mismo arco (también con O(cte) dada la estructura de *RepresentacionAC*). Tanto el cálculo del punto medio como la verificación de si dos arcos cubren todo el círculo posee O(cte), por lo tanto el algoritmo posee una complejidad de O(n). Notemos que la cantidad de puntos de intersección está acotada superiormente por n, dado que el grafo es HCA.

3.1.8 Fase 5: Filtro y precálculos

Esta fase es la encargada de calcular cierta información útil que luego es utilizada por la fase de búsqueda. Es importante notar que sólo se llega a esta fase cuando se sabe que el grafo a analizar no posee un arco universal ni dos arcos que cubran todo el círculo.

Para empezar, esta fase descarta los arcos no propios, dado que estos arcos no aportan a la solución del problema. Luego ordena ascendentemente a los arcos propios resultantes por su extremo $Inicial$ y a los puntos de intersección por su valor angular.

En el siguiente paso, calcula para cada arco el punto de intersección mas cercano por la derecha y que no pertenezca al arco. También calcula para cada punto de intersección cual es el arco (propio) que lo contiene y se extiende más a la derecha.

Parámetros: $rep : RepresentacionAC, puntosInterseccion : [Punto]$

- Para cada arco A_i de rep
  - Si A_i es un arco no propio, descartarlo de rep.
- Ordenar $rep.arcos$ por su extremo $Inicial$ ascendentemente.
- Ordenar $puntosInterseccion$ por su valor angular ascendentemente.
- Para cada arco A_i de $rep.arcos$
  - Buscar en $puntosInterseccion$ mediante búsqueda binaria el punto más cercano por la derecha que no pertenezca a A_i
- Para cada punto de intersección p_i de $puntosInterseccion$

- Buscar en *rep.arcos* mediante búsqueda binaria el arco más cercano por la izquierda a p_i

Complejidad y corrección de esta fase del algoritmo

La corrección de la mayoría de las operaciones realizadas puede ser verificada fácilmente. La secciones que merecen una explicación más detallada son las que corresponden a las búsquedas de arcos y puntos de intersección.

La primera búsqueda calcula para cada arco propio cuál es el punto de intersección que está fuera del arco lo más cerca posible por la derecha. Esta búsqueda siempre encuentra un punto de intersección, dado que a esta altura, no puede existir un arco universal. Se utiliza búsqueda binaria para hallar este punto de intersección que tiene que estar lo más cerca por la derecha del extremo de tipo *Final* del arco en cuestión.

La segunda búsqueda calcula, para cada punto de intersección p_i, cuál es el arco propio que lo contiene y se extiende lo más posible hacia la derecha. Notemos que para todo punto de intersección siempre hay un arco propio que lo contiene. Como sólo se consideran los arcos propios, buscar un arco propio que verifica la condición mencionada es equivalente a buscar un arco cuyo extremo *Inicial* esté lo más cerca posible por izquierda de p_i. Como sólo se trabaja con arcos propios, el extremo *Final* de este arco llega lo más lejos posible por la derecha comparado con otros que también contienen al punto. Para realizar la implementación también utilizamos búsqueda binaria.

Veamos ahora la complejidad de esta fase. Descartar los arcos propios posee un orden de O(n), ya que estos arcos fueron identificados por la fase 3. Ordenar los arcos y los puntos de intersección posee un orden de O($n.log(n)$), dado que se utiliza un algoritmo de ordenamiento de tipo Heap Sort. Para ambas búsquedas se utiliza búsqueda binaria, con un orden de O($n.log(n)$) dado que sabemos que la cantidad de puntos de intersección está acotada por n. En total esta fase posee una complejidad de O($n.log(n)$).

3.1.9 Fase 6: Búsqueda

Esta fase se encarga de encontrar el primer punto de intersección a partir del cual será calculado el cubrimiento mínimo.

La idea básica del procedimiento es la siguiente: el algoritmo toma un punto de intersección p_1 cualquiera y busca cuál es el arco A_1 que contiene a p_1 y llega lo más a la derecha posible. A continuación, calcula cual es el punto de intersección p_2 que no está contenido por A_1 y se encuentra lo más cerca posible por la derecha . Este procedimiento se repite hasta que el algoritmo vuelva a un punto de intersección que haya visitado previamente. El punto que vuelve a visitar es el primer punto de intersección a partir del cual será calculado el cubrimiento mínimo.

Parámetros: $rep : RepresentacionAC, puntosInterseccion : [Punto]$

- $p : Punto$
- $p := puntosInterseccion[1]$
- Mientras no se repita p
  - Buscar el arco A_i que contiene a p y se extiende más a la derecha. (precalculado en fase 5)
  - Buscar el punto de intersección p_i más cercano a A_i por la derecha que no pertenece a A_i. (precalculado en fase 5)
  - $p := p_i$

Complejidad y corrección de esta fase del algoritmo

La corrección de esta fase del algoritmo será demostrada en la fase siguiente. En cuanto a su complejidad, notemos que el algoritmo siempre llega a un punto que se repite, ya que la cantidad de puntos de intersección distintos está acotado superiormente por n. En el peor caso, en cada iteración se alcanza un nuevo punto de intersección, siendo n el límite para la cantidad de iteraciones. Como los puntos y arcos buscados en cada iteración están precalculados en la fase anterior, cada iteración posee un orden O(cte). Por lo tanto el orden de complejidad de esta fase es O(n).

3.1.10 Fase 7: Principal

Aquí mostramos como se integran las fases anteriores para resolver el problema de encontrar el mínimo transversal de los cliques sobre grafos HCA. Se recibe como parámetro un arreglo de elementos de tipo *Arco* y se devuelve como resultado los arcos que pertenecen a un cubrimiento mínimo para ese conjunto de arcos.

Una vez que la fase anterior (búsqueda) calcula el punto de intersección "bueno", la fase principal realiza nuevamente el mismo procedimiento, pero partiendo de dicho punto. El arco seleccionado por cada punto de intersección es elegido como integrante del cubrimiento mínimo. El algoritmo lleva un contador de la cantidad de cliques cubiertos y se detiene cuando los ha cubierto todos.

Parámetros: $arcos : [Arco]$

- $rep : RepresentacionAC$
- $puntosInterseccion : [Punto]$
- $punto : Punto$
- $resultado : [Arco]$
- $cliquesCubiertos : int := 0$
- **{Fase 1}** $rep := Inicializacion(arcos)$
- **{Fase 2}** $rep := ExtremosDistintos(rep)$

- **{Fase 3}** $ArcoUniversalYPropios(rep)$
- Si existe un arco universal A_i
  - Devolver A_i y terminar.
- Si no
  - **{Fase 4}** $puntosInterseccion = PuntosDeInterseccion(rep)$
  - Si hay dos arcos A_i y A_j que cubren todo el círculo
    - Devolver A_i, A_j y terminar.
  - Si no
    - **{Fase 5}** $FiltroYPrecalculos(rep, puntosInterseccion)$
    - **{Fase 6}** $punto := Busqueda(rep, puntosInterseccion)$
    - **{Fase Principal}**
    - Mientras $cliquesCubiertos < puntosInterseccion.cantidadDeElementos()$
      - Buscar el arco A_i que contiene a *punto* y se extiende más a la derecha
      - Agregar A_i a *resultado*
      - Buscar al punto de intersección p_i más cercano a A_i por la derecha que no pertenece a A_i
      - Incrementar *cliquesCubiertos* la cantidad de puntos de intersección entre *punto* y p_i
      - $punto := p_i$
    - Devolver *resultado*

Corrección del algoritmo

Hasta ahora hemos probado la corrección de todas las fases, faltando sólo las dos últimas: Búsqueda y Principal.

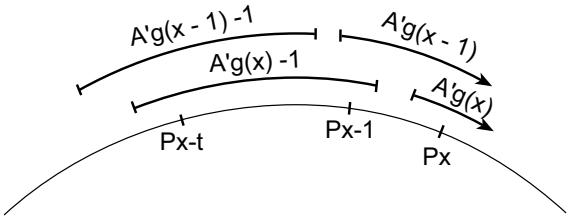
Llamemos $p_1^{'}$ al punto de intersección encontrado por la fase de búsqueda. Partiendo de este punto se encuentran $p_2^{'}$, $p_3^{'}$, ..., $p_{s+1}^{'} = p_1^{'}$ puntos de intersección. Cada uno de estos s puntos de intersección ha elegido un arco propio que los contiene, que notamos como $A_1^{'}, A_2^{'}, ..., A_s^{'}$.

La secuencia $p_1^{'}, p_2^{'}, p_3^{'}, ..., p_s^{'}$ puede no ser una secuencia ordenada de puntos en el sentido horario. Por eso, podemos reordenarlos de acuerdo al sentido horario como $p_1, p_2, ..., p_s$ utilizando una función biyectiva $f : \{1, 2, ..., s\} \rightarrow \{1, 2, ..., s\}$ para mapear este reordenamiento. Esto significa que $p_i = p_{f(i)}^{'}$.

Si hay k puntos de intersección, sabemos que $p_i = p_{i+k}$ y que $A_i^{'} = A_{i+s}^{'}$. Para poder indexar los puntos de intersección circularmente utilizando f, definamos la función g como $g(x) = f(((x - 1)\ mod\ s) + 1)$. A partir de las definiciones anteriores, dado un punto de intersección p_j podemos afirmar que el arco $A_{g(j)-1}^{'}$ termina antes de p_j y después del punto de intersección inmediatamente anterior a p_j (el cual no tiene porque ser uno de los p_i).

Sea $A'_{g(x)-1}$ el arco que contiene mayor cantidad de puntos de intersección entre los puntos p_i. Sea t la cantidad de puntos p_i que contiene. Por lo tanto este arco contiene exactamente en sentido horario a $p_{x-t}, p_{x-t+1}, ..., p_{x-1}$.

Consideremos ahora al arco $A'_{g(x-1)-1}$. Este arco termina justo antes del punto p_{x-1} y como es un arco propio debe comenzar antes que $A'_{g(x)-1}$. Esto implica que contiene a los puntos $p_{x-t}, p_{x-t+1}, ..., p_{x-2}$. Notemos que también debe contener a p_{x-t-1}, porque si no fuera de ese modo, el punto de intersección $p'_{g(x-1)-1}$ no estaría contenido por el arco $A'_{g(x-1)-1}$, un absurdo. Por lo tanto, $A'_{g(x-1)-1}$ también contiene t puntos de intersección entre los puntos p_i.



Este razonamiento se puede aplicar iterativamente para $A'_{g(x-2)-1}$, $A'_{g(x-3)-1}$, ..., $A'_{g(x-s+1)-1}$, llegando a la conclusión que todos los arcos A' contienen exactamente t puntos de intersección p_i. También sabemos que cada arco A'_j siempre termina entre p'_{j+1} y el punto de intersección inmediatamente anterior a p'_{j+1} (que no es necesariamente un punto entre los p_i). Este resultado garantiza que la búsqueda final puede comenzar en cualquier p_i sin variar la cantidad de arcos que cubren los k puntos de intersección.

Por lo tanto, para cubrir los k puntos de intersección se necesitan $\lceil \frac{s}{t} \rceil$ arcos, ya que cada arco A' cubre t puntos de intersección de tipo p_i de un total de s. Por la forma que el algoritmo elige los arcos, si comienza por el punto p_j, el primer arco cubre todos los puntos de intersección que están en $[p_j, p_{j+t})$, el segundo arco los que están en $[p_{j+t}, p_{j+2t})$ y así hasta el arco número $\lceil \frac{s}{t} \rceil$, que cubre los que están en $[p_{j+\lceil \frac{s}{t} \rceil t-t}, p_{j+\lceil \frac{s}{t} \rceil t})$. De esta forma, estos $\lceil \frac{s}{t} \rceil$ arcos cubren los k puntos de intersección.

Faltaría ver que no hay una solución mejor. Supongamos que existe una solución mejor con a lo sumo $\lceil \frac{s}{t} \rceil - 1$ arcos. Por lo tanto debe existir un arco C de esta solución que contiene más de t puntos de intersección de tipo p_i. Sean $p_{h-t}, ..., p_{h-1}, p_h$ puntos de intersección cubiertos por C, siendo p_h el último de ellos. En el momento de elegir el arco que contiene a p_{h-t}, y que termina lo más a la derecha posible, el algoritmo eligió a A'_{h-t} que permite llegar justo antes de p_h. Pero el arco C contiene a p_{h-t} y llega por lo menos hasta p_h, con lo cual el algoritmo debería haber elegido a C en lugar de a A'_{h-t}. Esto es un absurdo, con lo cual no existe una solución mejor.

Complejidad del algoritmo

Hasta ahora hemos calculado la complejidad de todas las fases salvo la Principal. El cálculo que se realiza es el mismo que en la fase de búsqueda, con lo cual cada iteración tiene un orden O(cte). Como cada vez que se agregar un arco al conjunto resultado se cubre por lo menos un clique, en el peor caso se realizan n iteraciones, con lo cual el orden de la fase principal es O(n).

Resumiendo, el orden total del algoritmo está dado por la suma de la complejidad de las fases anteriores más la complejidad de la fase principal.

Fase	**Orden**
Inicialización	n.log(n)
Extremos Distintos	n
Arco Universal y Propios	n
Puntos de Intersección	n
Filtro y Precálculos	n.log(n)
Búsqueda	n
Principal	n
Total	n.log(n)

Por lo tanto, el algoritmo posee una complejidad de O($n.log(n)$).

3.2 Conjunto máximo de cliques disjuntos

El problema del máximo número de cliques disjuntos consiste en encontrar un conjunto de cliques que no tengan ningún vértice en común y cuyo cardinal sea máximo.

3.2.1 Presentación del algoritmo

El algoritmo que proponemos para la resolución de este problema es una leve modificación del algoritmo presentado en la sección 3.1. Los cambios realizados sólo se ven reflejados en modificaciones en la fase **Puntos de Intersección** (fase 4 del algoritmo anterior) y en la fase **Principal** (fase 7 del algoritmo anterior).

Las restantes fases permanecen inalteradas, por lo que no volveremos a explayarnos sobre ellas. Los tipos de datos utilizados también son los mismos que presentamos en la sección 3.1.3.

3.2.2 Fase $4'$: Puntos de intersección modificado

Esta fase identifica algunos de los puntos de intersección de una representación arco-circular Helly. A diferencia de la fase 4 del algoritmo anterior, se descartan aquellos cliques que intersecan con todos los demás y pudieran ser descubiertos en este algoritmo.

El algoritmo requiere que los extremos de la representación sean todos distintos. Adicionalmente, los extremos deben estar ordenados con el criterio de la fase 1.

La primera parte del funcionamiento del algoritmo es idéntico al de la fase 4, salvo cuando se detectan dos arcos que cubren todo el círculo. Este algoritmo modificado descarta esos arcos, ya que en el peor de los casos está dejando de considerar un punto de intersección que interseca con todos los demás cliques (significa que cualquier solución que contenga este clique, no puede tener otro clique).

Al finalizar la búsqueda de puntos de intersección, si encontró al menos un punto de intersección, devuelve los puntos encontrados de la misma forma que lo hacía el algoritmo original.

En el caso de no haber encontrado ningún punto de intersección, esto significa que todos los cliques están formados por pares de arcos que cubren todo el círculo (y que todos los cliques se intersecan de a pares). Bajo estas condiciones, el algoritmo devuelve un punto de intersección que contenga al primer par de arcos detectados que cubren todo el círculo.

Parámetros: $rep : RepresentacionAC$

- $\{Primera\ parte\}$
- Para cada extremo e_i de $rep.extremos$
  - Sea e_j el extremo consecutivo a e_i por la derecha

- Si e_i es un extremo $Inicial$ y e_j es un extremo $Final$
  - Si e_i y e_j pertenecen al mismo arco
    - Agregar al punto medio entre e_i y e_j al conjunto de puntos de intersección
  - Si no
    - Sean A_i y A_j los arcos correspondientes a e_i y e_j respectivamente
    - Si A_i y A_j cubren todo el círculo
      - Si es la primera vez que se llega a esta condición
        - Sea $A_{p_i} = A_i$ y $A_{p_j} = A_j$
    - Si no
      - Agregar al punto medio entre e_i y e_j al conjunto de puntos de intersección
- {*Segunda parte*}
- Si no se encontró ningún punto de intersección
  - Sea R_1 la región definida por el extremo $Inicial$ de A_{p_i} y el extremo $Final$ de A_{p_j}
  - Sea R_2 la región definida por el extremo $Inicial$ de A_{p_j} y el extremo $Final$ de A_{p_i}
  - Para cada extremo e_i de *rep.extremos* y mientras no se haya encontrado a $R_{1'}$
    - Sea e_j el extremo consecutivo a e_i por la derecha
    - Si e_i es un extremo $Inicial$ y e_j es un extremo $Final$
      - Sea R_i la región definida por e_i y e_j.
      - Si la región R_i está contenida en la región R_2
        - Sea $R_{1'} = R_i$
  - Para cada arco A_i de *rep.arcos*
    - Si A_i contiene a la región R_1 y A_i no contiene a la región $R_{1'}$
      - Agregar el punto medio de la región R_1 al conjunto de puntos de intersección. Terminar
    - Si A_i contiene a la región R_1' y A_i no contiene a la región R_1
      - Agregar el punto medio de la región R_1' al conjunto de puntos de intersección. Terminar
  - Agregar el punto medio de la región R_1 al conjunto de puntos de intersección.

Complejidad y corrección de esta fase del algoritmo

La primera parte del algoritmo es idéntica a la fase 4 del algoritmo para encontrar el transversal de los cliques mínimo (salvo que se descartan los

pares de arcos que cubren todo el círculo), con lo cual no entraremos en detalles. Recordemos que la complejidad de la primera parte es O(n).

La segunda parte del algoritmo se ejecuta cuando no se identifica ningún punto de intersección. Esto implica que todos cliques se intersecan de a pares y por lo tanto la maxima cantidad de cliques disjuntos es uno y cualquier conjunto unitario de un clique es solución. Esto significa que hay que hallar un punto de intersección cualquiera que represente un clique. Para esto, el algoritmo almacena el primer par de extremo (a_i,b_j) donde se detectó que sus arcos correspondientes A_i y A_j cubren todo el círculo. El punto de intersección que se devuelve contiene a A_i y a A_j.

Como los arcos A_i y A_j cubren todo el círculo, se intersecan tanto en $[a_i, b_j]$ como en $[a_j, b_i]$. La región $[a_i, b_j]$ está definida por extremos consecutivos, con lo cual no hay ningún extremo de arco incluido en dicha región. El primer paso de algoritmo es encontrar una región incluida en $[a_j, b_i]$ con esta misma propiedad (tener en cuenta que la inclusión no es estricta). Para esto, recorre todos los pares de extremos contiguos de tipo (a_x,b_y) que estén contenidos en $[a_j, b_i]$. Al encontrar un par (siempre existe al menos uno) el algoritmo define la región $[a_j^{'}, b_i^{'}]$, con las mismas características que $[a_i, b_j]$.

Ahora bien, como el clique a devolver incluye a los arcos A_i y A_j y el grafo analizado es HCA, el punto de intersección que representa a dicho clique debe estar en la región $[a_i, b_j]$ o en la región $[a_j^{'}, b_i^{'}]$.

Para saber exactamente cuál es la región por donde se podría ubicar el punto de intersección, el algoritmo recorre los arcos de la representación de a uno, verificando si se intersecan con ambas regiones. En el caso que un arco interseque solamente con una de las dos regiones, esto significa que el punto de intersección debe ser elegido dentro de esta región (el algoritmo elige el punto medio de la misma). Si todos los arcos intersecan con las dos regiones, se puede elegir el punto de intersección de cualquiera de las dos regiones. En este caso el algoritmo elige el punto medio de la región $[a_i, b_j]$.

Nos queda por analizar la complejidad de la segunda parte de esta fase. La búsqueda de la región $[a_j^{'}, b_i^{'}]$ posee un orden O(n). Esto es así porque se itera para cada extremo de la representación y en cada iteración se verifica si una región del círculo está incluida en otra (O(cte)). A continuación se recorren los n arcos verificando para cada uno si pertenece a alguna de las dos regiones definidas. Esta operación posee un orden O(cte), por lo que esta verificación posee una complejidad de O(n). Esto significa que la segunda parte de esta fase posee un orden de O(n).

Resumiendo, cada parte de esta fase del algoritmo posee una complejidad de O(n), por lo tanto, esta fase tiene un orden de O(n).

3.2.3 Fase 7′: Principal

Aquí se integran las fases anteriores para resolver el problema de encontrar un conjunto máximo de clique disjuntos sobre grafos HCA. Se recibe como parámetro un arreglo de elementos de tipo Arco y se devuelve un conjun-

to de puntos de intersección representando cliques disjuntos para el grafo intersección.
Parámetros: $arcos : [Arco]$

- $rep : RepresentacionAC$
- $puntosInterseccion : [Punto]$
- $punto : Punto$
- $resultado : [Punto]$
- $cliquesCubiertos : int := 0$
- **{Fase 1}** $rep := Inicializacion(arcos)$
- **{Fase 2}** $rep := ExtremosDistintos(rep)$
- **{Fase 4'}** $puntosInterseccion = PuntosDeInterseccion(rep)$
- Si $puntosInterseccion.cantidadDeElementos() = 1$
  - Devolver $puntosInterseccion[1]$ y terminar.
- Si no
  - **{Fase 3}** $ArcoUniversalYPropios(rep)$
  - Si hay un arco universal
    - Devolver $puntosInterseccion[1]$ y terminar.
  - Si no
    - **{Fase 5}** $FiltroYPrecalculos(rep, puntosInterseccion)$
    - **{Fase 6}** $punto := Busqueda(rep, puntosInterseccion)$
    - **{Fase Principal}**
    - Mientras $cliquesCubiertos < puntosInterseccion.cantidadDeElementos()$
      - Agregar *punto* a *resultado*
      - Buscar el arco A_i que contiene a *punto* y se extiende más a la derecha
      - Buscar al punto de intersección p_i más cercano a A_i por la derecha que no pertenece a A_i
      - Incrementar *cliquesCubiertos* la cantidad de puntos de intersección entre *punto* y p_i
      - $punto := p_i$
    - Si $cliquesCubiertos > puntosInterseccion.cantidadDeElementos()$
      - Eliminar el último punto de intersección agregar a *resultado*
    - Devolver *resultado*

Corrección del algoritmo

Analicemos esta última fase de algoritmo. Al aplicar la fase $4^{'}$, ésta puede haber descartado puntos de intersección para algunos cliques. Esto no genera ningún inconveniente, ya que los cliques descartados son aquellos que intersecan con todos los demás cliques del grafo. En el caso que identifique sólo un clique, esto significa que todos los cliques se intersecan entre sí de a pares, con lo cual, la mayor cantidad de cliques disjuntos es 1. A continuación se aplica la fase 3, para detectar un arco universal e identificar a los arcos propios. En el caso que el algoritmo encuentre un arco universal, esto también significa que todos los cliques se intersecan entre sí de a pares y la mayor cantidad de cliques disjuntos es 1.

Las fase 6 y la fase principal sólo consideran los arcos propios. Veamos que esto no afecta corrección del algoritmo. Sea $A_{\neg p}$ un arco no propio que impide que dos cliques C_1 y C_2 sean seleccionados como elementos del conjunto solución porque ambos cliques lo comparten. Luego, existe un arco propio A_p que incluye a $A_{\neg p}$ y tanto C_1 como C_2 comparten al arco A_p. Por lo tanto el no considerar a los arcos no propios no es relevante para computar la solución del problema.

Como vimos en el sección 3.1.10, la fase de búsqueda devuelve un punto $p_1^{'}$ a partir del cual se genera una secuencia de puntos de intersección $p_2^{'}, ..., p_s^{'}, p_{s+1}^{'} = p_1^{'}$. Cada punto de intersección $p_{i'}$ ha elegido un arco $A_{i'}$ que lo contiene y más se extiende hacia la derecha. Siguiendo el mismo procedimiento, reordenemos la secuencia $p_1^{'}, p_2^{'}, ..., p_s^{'}$ en el sentido horario mediante la función biyectiva f, asignando cada punto $p_i = p_{f(i)}$. Definamos nuevamente $g(x) = f(((x-1)\ mod\ s)+1)$ y consideremos que $p_i = p_{i+k}$ y que $A_i^{'} = A_{i+s}^{'}$. Con estas mismas definiciones, en la sección 3.1.10 se probó que cada arco $A_{f(i)}^{'}$ contiene a los puntos $p_i, p_{i+1}, ..., p_{i+t-1}$, y que p_i es el primer punto de intersección inmediatamente después del extremo inicial del arco $A_{f(i)}^{'}$. También se demostró que el punto p_{i+t} es $p_{f(i)+1}^{'}$ y que la cantidad de arcos necesarios para cubrir todos los puntos de intersección es $\lceil \frac{s}{t} \rceil$.

Veamos primero que la fase principal elige exactamente $\lfloor \frac{s}{t} \rfloor$ puntos de intersección. Si s es múltiplo de t, entonces $\lceil \frac{s}{t} \rceil = \frac{s}{t} = \lfloor \frac{s}{t} \rfloor$, lo cual significa que cada punto de intersección p_i está exactamente en uno de los $\frac{s}{t}$ arcos. Por lo tanto, el acumulador llega exactamente a k y el algoritmo no se deshace del último punto de intersección elegido, y se eligen $\lfloor \frac{s}{t} \rfloor$ puntos de intersección.

Si s no es múltiplo de t, entonces $\lceil \frac{s}{t} \rceil = \lfloor \frac{s}{t} \rfloor + 1$ y el punto de intersección $p_1^{'}$ se encuentra en el primero y en el último de los $\lceil \frac{s}{t} \rceil$ arcos que cubren todos los puntos de intersección. Por lo tanto, todos los puntos de intersección son contados al menos una vez, y $p_1^{'}$ es contado en el primer arco y en el último. Estos significa que el acumulador supera a k, por lo que el algoritmo descarta el último punto de intersección, quedándose con $\lfloor \frac{s}{t} \rfloor$ puntos.

Veamos ahora que los puntos de intersección seleccionados representan cliques disjuntos. Supongamos por el absurdo que los cliques que representan los puntos de intersección de la solución no son disjuntos. Esto implica

que hay dos puntos de intersección que están incluidos en un mismo arco. Además, siempre podemos suponer que este arco es propio, porque si no lo fuera, seguro existe otro arco propio en el que ambos están incluidos.

Tomemos entonces a un arco propio que los contiene y llamémoslo A. Sabemos también que los puntos de intersección conflictivos pertenecen a la secuencia $p_1, p_2, ..., p_s$, porque el algoritmo selecciona los puntos de intersección de dicha secuencia. Si consideramos que la búsqueda final comenzó en el punto p_j, los puntos de intersección son elegidos por el algoritmo en el siguiente orden: $p_j, p_{j+t}, p_{j+2t}, ..., p_{j+\lfloor \frac{s}{t} \rfloor t-t}$. Por lo tanto, el arco A que contiene a dos de los puntos de esta secuencia también debe contener a todos los p_i que están en medio de ellos dos. A partir de la secuencia elegida por el algoritmo, es fácil ver que por lo menos hay $t+1$ puntos de tipo p_i consecutivos entre ellos dos.

Sea p_{j+xt} el primero de los dos puntos conflictivos. Si analizamos el arco que pasa por él y se extiende lo más a la derecha posible, vemos que dicho arco termina justo antes del punto p_{j+xt+t}, dado que cada arco contiene exactamente a t puntos de intersección entre los p_i. Pero el arco A también pasa por p_{j+xt} y se extiende por lo menos hasta alcanzar a p_{j+xt+t}. Esto es un absurdo, por lo tanto los puntos de intersección seleccionados representan cliques disjuntos.

Finalmente, faltaría ver que no existe otro conjunto de puntos de intersección con mayor cardinalidad que también represente a cliques disjuntos del grafo. Si suponemos que este conjunto existe, debe contener por lo menos $\lfloor \frac{s}{t} \rfloor + 1$ puntos de intersección. Ahora bien, el círculo se puede particionar en r intervalos semi-cerrados de la siguiente manera: Sean $p_1^{''}, p_2^{''}, ..., p_r^{''}$ los puntos de la supuesta mejor solución (con $r \geq \lfloor \frac{s}{t} \rfloor + 1$) ordenados de menor a mayor en función de su ángulo, y sean $(p_1^{''}, p_2^{''}], (p_2^{''}, p_3^{''}], ..., (p_r^{''}, p_1^{''}]$ los r intervalos así definidos.

Si consideramos los s puntos de intersección p_i, vemos que por lo menos uno de estos r intervalos contiene menos de t puntos p_i (porque hay por lo menos $\lfloor \frac{s}{t} \rfloor + 1$ intervalos). Sin pérdida de generalidad, podemos decir que $(p_j^{''}, p_{j+1}^{''}]$ contiene a $p_{x+1}, p_{x+2}, ..., p_{x+y}$, con $y < t$. Analicemos ahora uno de los arcos $A_i^{'}$ que pasa por $p_x, p_{x+1}, p_{x+2}, ..., p_{x+t-1}$ y cuyo primer punto de intersección a su derecha es p_{x+t}. El punto $p_j^{''}$ está incluido en este arco, porque pertenece a $[p_x, p_{x+1})$. Por otra parte, el punto $p_{j+1}^{''}$ está incluido en $[p_{x+y}, p_{x+t})$ y por lo tanto también está incluido en $A_i^{'}$. Entonces, hay un arco que contiene a $p_j^{''}$ y a $p_{j+1}^{''}$, con lo cual los cliques representados por estos puntos no son disjuntos y la supuesta mejor solución no sería válida. Por lo tanto el algoritmo devuelve un conjunto con cardinalidad máxima.

Complejidad del algoritmo

En secciones anteriores ya calculamos la complejidad de todas las fases salvo la principal. La fase principal de este algoritmo es muy similar a la fase

principal del algoritmo anterior, con la diferencia que en ésta, el algoritmo almacena puntos de intersección en lugar de arcos y se aplica la fase $4'$ antes que la fase 3. Estos cambios no tienen efecto en la complejidad de la fase, por lo tanto el orden es de $O(n)$, igual que la fase principal del algoritmo anterior.

Por lo tanto, este algoritmo posee un orden de complejidad de $O(n.log(n))$.

Capítulo 4

Conclusiones y trabajo futuro

En el capítulo 2, caracterizamos a los grafos overlap de arco-circulares pidiendo condiciones necesarias y suficientes sobre sucesivas orientaciones de una partición del complemento de un grafo. Este problema, planteado como abierto en [9], fue demostrado en base a ideas aportadas por un teorema de Jayme Szwarcfiter [35] para una caracterización de grafos circulares (overlap de intervalos).

Si bien el teorema probado en esta tesis permite caracterizar a esta clase de grafos, no lleva naturalmente a un algoritmo de reconocimiento eficiente. Una posible línea de trabajo que se desprende de esta demostración es la de investigar posibles algoritmos de reconocimiento, basados en la idea que sugiere el teorema de analizar a un grafo CAO como "n grafos circulares". Queda pendiente también investigar la complejidad de problemas conocidos (coloreo, conjunto dominante) sobre grafos CAO e intentar proponer algoritmos eficientes para su resolución.

En el capítulo 3, presentamos dos algoritmos eficientes para resolver los problemas de encontrar el mínimo transversal de los cliques y el máximo conjunto de cliques disjuntos en grafos arco-circulares Helly. Ambos algoritmos toman como entrada una representación arco-circular Helly de un grafo y tienen una complejidad de $O(n.log(n))$.

La idea básica para estos algoritmos fue presentada en la tesis doctoral de Guillermo Durán [9] y la implementación de manera eficiente fue sugerida por Min Chih Lin [28]. Los algoritmos propuestos introducen una mejora en el orden de complejidad con respecto a los algoritmos conocidos hasta el momento [21], que resuelven estos problemas en un orden de $O(n^2)$, una vez conocida la representación arco-circular Helly del grafo.

Ambos algoritmos fueron implementados en Java, y constituyen una herramienta de utilidad para futuros desarrollos teóricos y prácticos. Los algoritmos fueron testeados con grafos pequeños (no más de diez vértices), por la dificultad de de generar grafos HCA y construir una representación en arcos circulares que cumpla la propiedad de Helly. Un posible trabajo pendiente es el de implementar los algoritmos propuestos por Gavril [16] para la caracterización de grafos HCA y la construcción de representaciones en arcos circulares adecuadas.

Como trabajo futuro, queda pendiente investigar otros problemas algorítmicos en grafos arco-circulares Helly. Es sabido que el problema de coloreo permanece NP-Completo para grafos HCA [16], mientras que clique máximo puede ser resuelto fácilmente en tiempo polinomial utilizando, por ejemplo, las ideas de los algoritmos presentados en esta tesis.

Bibliografía

[1] T. Andrae, "On the clique-transversal number of chordal graphs", Discrete Mathematics 191 (1998), 3-11.

[2] T. Andrae, C. Flotow, "On covering all cliques of a chordal graph", Discrete Mathematics 149 (1996), 299-302.

[3] T. Andrae, M. Schughart, Z. Tuza, "Clique-transversal sets of line graphs and complements of line graphs ", Discrete Mathematics 88 (1991), 11-20.

[4] A. Apostolico, S. Hambrush, "Finding maximun cliques on circular-arc graphs", Information Processing Letters 26 N. 4 (1987), 209-215.

[5] V. Balachandhran, P. Nagavamsi, C. Pandu Ragan, "Clique transversal and clique independence on comparability graphs", Information Processing Letters 58 (1996), 181-184.

[6] C. Berge, "Graphs", North-Holland, 1985.

[7] A. Bouchet, "Reducing prime graphs and recognizing circle graphs", Combinatorica 7, 3 (1987), 243-254.

[8] M. Bonuccelli, "Dominating sets and domatic number of circular-arc graphs", Report S80-18, Dipartamento di Informatica, Universita di Pisa, Italy (1980).

[9] G. Durán, "Sobre grafos intersección de arcos y cuerdas en un círculo", Tesis Doctoral, Universidad de Buenos Aires, 2000.

[10] G. Durán, M. Lin, J. Szwarcfiter, "On clique-transversal and clique-independent sets", enviado para su publicación.

[11] P. Erdös, T. Gallai, Z. Tuza, "Covering the cliques of a graph with vertices", Discrete Mathematics 108 (1992), 279-289.

[12] E. Even, A. Itai, "Queues, stacks and graphs, in Theory of Machines and Computations", Academic Press, New York (1971), 71-86.

[13] C. Gabor, K. Supowit, W. Hsu, "Recognizing circle graphs in polynomial time", Journal. of the ACM, Vol. 36, 3 (1989), 435-473.

[14] M. Garey, D. Johnson, G. Miller, C. Papadimitriou, "The complexity of coloring circular arcs and chords", SIAM J. Algebraic and Discrete Methods 1 (1980), 216-227.

[15] F. Gavril, "Algorithms on circular-arc graphs", Networks 4 (1974), 357-369.

[16] F. Gavril, "Intersection graphs of Helly families of subtrees", Discrete Applied Mathematics 66 (1996), 45-56.

[17] F. Gavril, "Algorithms for a maximum clique and a maximun independent set of a circle graph", Network 3 (1973), 261-273.

[18] M. Golumbic, "Algorithm Graph Theory and Perfect Graphs", Academic Press, New York, 1980.

[19] M. Golumbic, P. Hammer, "Stability in circular-arc graphs", Journal of Algorithms 9 (1988), 314-320.

[20] U. Gupta, D. Lee, J. Leung, "Efficient algorithms for interval graphs and circular-arc graphs", Networks 12 (1982), 459-467.

[21] V. Guruswami, C. Pandu Rangan, "Algorithmic aspects of clique-transversal and clique-independent sets", Discrete Applied Mathematics 100 (2000), 183-202.

[22] F. Harary, "Graph Theory", Addison-Wesley, 1969.

[23] W. Hsu, "O($n.m$) algorithms for the recognition and isomorphism problems on circular-arc graphs", SIAM J. Comput. 24 (1995), 411-439.

[24] W. Hsu, "Maximun weight clique algorithms for circular-arc graphs and circle graphs", SIAM J. Comput. 14 (1985), 224-231.

[25] W. Hsu, K. Tsai, "Linear time algorithms on circular-arc graphs", Information Processing Letters 40 N. 3 (1991), 123-129.

[26] T. Kashiwabara, S. Masuda, K. Nakajima, T. Fujisawa, "Polynomial Time Algorithms on Circular-Arc Overlap Graphs", Networks 21 (1991), 195-203.

[27] D. Knuth, "The Art of Computing Programming", Vol. 1, Addison-Wesley (1969).

[28] M. Lin, comunicación personal, 2001.

[29] S. Massuda, K. Nakajima, "An optimal algorithm for finding a maximun independent set of a circular-arc graph", SIAM J. Comput. 17 (1988), 41-52.

[30] S. Massuda, K. Nakajima, T. Kashiwabara, T. Fujisawa, "Efficient algorithms for finding maximun cliques of an overlap graph ", Networks 20 (1990), 157-171.

[31] W. Naji, "Reconnaissance des graphes de cordes", Discrete Mathematics 54 (1985), 329-337.

[32] D. Rotem, J. Urrutia, "Finding maximum cliques in circle graphs", Networks 11, (1981), 269-278.

[33] J. Spinrad, "Circular-arc graphs with clique cover number two", Journal of Comb. Theory B 44, (1988), 300-306.

[34] J. Spinrad, "Recognition of circle graphs", Journal of Algorithms, 16 (2), (1994), 264-282.

[35] J. Szwarcfiter, "On edge transivity of directed graphs", Discrete Mathematics 141 (1995), 227-235.

[36] Z. Tuza, "Covering all cliques of a graph", Discrete Mathematics 86 (1990), 117-126.

[37] A. Tucker, "An efficient test for circular-arc graphs", SIAM J. Comput. 9 (1980), 1-24.