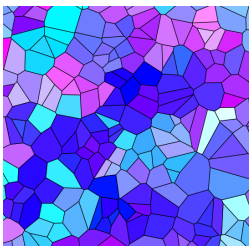

Taller de Geometría Discreta y Computacional
Encuentro Nacional de Computación 2021

Inmersiones de árboles en la malla con un mínimo número de dobleces de trayectorias

Nestaly Marín[1*], Adriana Ramírez-Vigueras[2], Oriol Solé-Pi[3], Fabiano de Souza Oliveira[4], Jayme Luiz Szwarcfiter[5], Vitor Tocci Ferreira de Luca[4], y Jorge Urrutia[2]

[1]Posgrado en Ciencia e Ingeniería de la Computación, Universidad Nacional Autónoma de México.
[2]Instituto de Matemáticas, Universidad Nacional Autónoma de México.
[3]Facultad de Ciencias, Universidad Nacional Autónoma de México.
[4]Instituto de Matemática e Estatística, Universidade do Estado do Rio de Janeiro.
[5]COPPE, NCE, IM, Universidade Federal do Rio de Janeiro, e Instituto de Matemática e Estatística, Universidade do Estado do Rio de Janeiro.

PLÁTICA

Resumen
Sea T un árbol cuyos vértices tengan grado máximo cuatro. Una inmersión de T en la malla cuadrada es una inmersión de T en la que cada uno de sus vértices es representado por un punto distinto con coordenadas enteras y sus aristas son representadas como segmentos verticales u horizontales con interiores ajenos. En esta plática presentamos un algoritmo de tiempo $O(n)$ que encuentra una inmersión de T en la malla cuadrada en la que se minimiza el máximo número de pares de aristas adyacentes no alineadas (dobleces) de cualquier trayectoria de T.

Palabras clave: Mínimo número de dobleces, Inmersión en la malla, Árboles

1 Introducción

Las inmersiones de gráficas en las que los vértices son puntos en el plano y las aristas son segmentos de línea son llamadas *inmersiones rectilíneas*. Fáry [2] demostró que toda gráfica plana tiene una inmersión rectilínea. Garg y Tamassia [3] probaron que decidir si una gráfica dada tiene una inmersión rectilínea es un problema *NP*-difícil . De Luca, Oliveira, y Szwarcfiter [1] propusieron el problema de encontrar inmersiones de árboles en la malla cuadrada en las que se minimice el máximo número de dobleces de cualquier trayectoria y presentaron un algoritmo de tiempo $O(n^2)$ para encontrar dicha inmersión. Nosotros presentamos un algoritmo de tiempo $O(n)$ para resolver este problema. En adelante supondremos que los árboles considerados aquí no tienen vértices de grado dos, ya que en dado caso siempre existe una inmersión de ellos en la que las aristas incidentes en cualquier vértice de grado dos se dibujen alineadas.

2 Definiciones

Sea $T = (V, E)$ un árbol tal que el grado máximo de sus vértices es cuatro.

Definición 1. Una inmersión de T en la malla cuadrada, a la cual llamaremos s-modelo de T siguiendo la notación en [1], es una inmersión de T en el plano tal que sus vértices son representados por puntos con coordenadas enteras, y sus aristas son representadas con segmentos de línea verticales u horizontales cuyos interiores son ajenos.

Definición 2. Dado un s-modelo $\mathcal{S}$ de T y una trayectoria P de T, el número de dobleces de P es el número pares de aristas consecutivas en P que no están alineadas en $\mathcal{S}$.

Definición 3. Un sl-modelo de T consiste en una colección de órdenes locales $\{\ell_v\}_{v \in V}$, donde ℓ_v indica las direcciones en que los vecinos de v deben colocarse con respecto a v, sujetas a rotación.

*Contacto
nestaly@ciencias.unam.mx

Todo s-modelo $\mathcal{S}$ de T está asociado a un sl-modelo $\mathcal{M}$ de T dado por el orden en que los vecinos de cada vértice se dibujan a su alrededor. En tal caso, decimos que $\mathcal{S}$ es una *representación* de $\mathcal{M}$.

Sea $e = (u, v)$ una arista de T.

Definición 4. Definimos a $T[e, v]$ como el subárbol de T compuesto por la unión de todas las trayectorias de T que contienen a e y terminan en u.

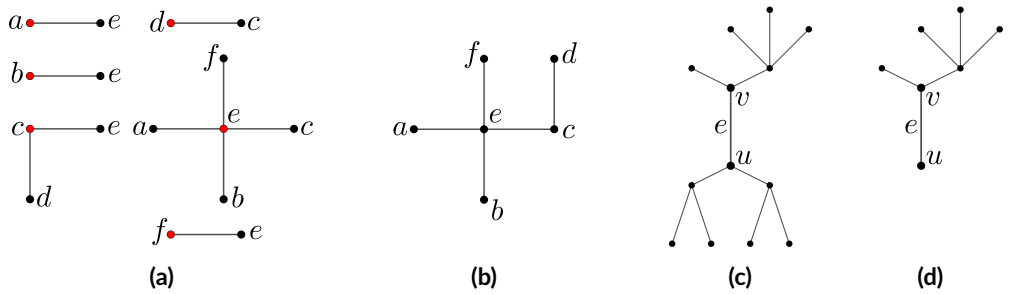

Figura 1. (a) Un sl-modelo $\mathcal{M}$ de un árbol. (b) Un s-modelo que es una representación de $\mathcal{M}$. (c) Un árbol T y una arista $e = (u, v)$. (d) $T[e, v]$.

Definición 5. Definimos a $b_{\mathcal{M}}(e, v)$ como el máximo número de dobleces en el sl-modelo $\mathcal{M}$ de entre todas las trayectorias de T que contienen a e y terminan en u. Definimos a $b(e, v)$ como el mínimo valor de $b_{\mathcal{M}}(e, v)$ entre todos los sl-modelos posibles de T.

Podemos ver al valor $b(e, v)$ como el valor asignado a la arista $e = (u, v)$ cuando se orienta de v hacia u. Por lo tanto, existen $2(n-1)$ valores de la forma $b(e, v)$.

Definición 6. Un sl-modelo $\mathcal{M}$ de T cumple con el valor $b(e, v)$ si $b_{\mathcal{M}}(e, v) = b(e, v)$; $\mathcal{M}$ cumple fuertemente con $b(e, v)$ si cumple con $b(e', v')$ para todo par (e', v') con $T[e', v'] \in T[e, v]$

3 Problemas

Problema 1. *Dado un árbol T tal que el grado máximo de los vértices de T es cuatro, encontrar un s-modelo de T en el que se minimice el máximo número de dobleces de cualquier trayectoria de T.*

4 Resultados

Lema 1. *Para todo sl-modelo $\mathcal{M}$ de T, existe un s-modelo que es la representación de $\mathcal{M}$. Además, dicho s-modelo puede ser encontrado a partir de $\mathcal{M}$ en tiempo $O(n)$.*

Por el lema 1, el problema 1 se reduce a encontrar un sl-modelo de T *óptimo*, i.e. que minimice el máximo número de dobleces de cualquier trayectoria. Dicho sl-modelo se obtiene a partir de los $2(n-1)$ valores de la forma $b(e, v)$, los cuales se pueden obtener en tiempo $O(n)$ como probamos en el teorema 1. Por último, en el teorema 2 probamos que el sl-modelo de T obtenido es óptimo.

Teorema 1. *Los $2(n-1)$ valores de la forma $b(e, v)$ y los sl-modelos de sus subárboles correspondientes que cumplan fuertemente con cada uno de ellos pueden ser computados en tiempo $O(n)$.*

Teorema 2. *El sl-modelo $\mathcal{M}$ computado por nuestro algoritmo minimiza el máximo número de dobleces de cualquier trayectoria de T.*

Referencias

[1] Vitor Tocci Ferreira de Luca, Fabiano de Souza Oliveira, and Jayme Luiz Szwarcfiter. Minimum number of bends of paths of trees in a grid embedding. In *LAGOS 2021*, 2021.

[2] István Fáry. On straight lines representation of planar graphs. *Acta Sci. Math. Szeged*, 11(4):229–233, 1948.

[3] Ashim Garg and Roberto Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM Journal on Computing*, 31(2):601–625, 2001.