Universidad Nacional Autónoma de México

Facultad de Ciencias

Titulo

T E S I S

1

QUE PARA OBTENER EL TÍTULO DE:

Licenciado en Ciencias de la Computación

PRESENTA:

uriel ochoa

TUTOR

adri

Ciudad Universitaria, CDMX, 2021

Hoja de datos del jurado

1. Datos del alumno
   Ochoa
   Gonzalez
   Uriel
   null@ciencias.unam.mx
   Universidad Nacional Autonoma de Mexico
   Facultad de Ciencias
   Ciencias de la Computación
   411

2. Datos del tutor
   Dra.
   Adriana

Kira

— Blumi

Agradecimientos

Índice general

Lista de figuras

44 Lista de tablas

Lista de definiciones

1

Introduccion

2 Conceptos Fundamentales

2.1 Gráficas

Esta sección y la siguiente (2.1.1) fueron basadas en *Discrete Mathematics: An Open Introduction*[3]

Definición 2.1
Una **gráfica** es un par ordenado $G = (V, E)$ que consiste de un conjunto no vacío V (llamado vértices) y un conjunto E (llamado aristas) formado por duplas de elementos de V.2.7a $\dashv$

Definición 2.2
La **adyacencia** se define de forma distinta para vértices que para aristas. Dos vértices son adyacentes si estan conectados por una arista. Dos aristas son adyacentes si comparten un vértice. $\dashv$

Definición 2.3
Un **recorrido** es una secuencia de vértices tal que, vértices consecutivos en dicha secuencia son adyacentes en la gráfica. $\dashv$

Definición 2.4
Un **camino** es un recorrido que no repite vértices (o aristas) excepto, tal vez, el primero y el último. Si un camino empieza y termina en el mismo vértice, se llama **ciclo**. $\dashv$

Cuando dos gráficas son básicamente la misma, pero no necesariamente iguales, son llamadas *isomorfas*. La definición de isomorfismo es como sigue:

Definición 2.5
Un **isomorfismo** entre dos gráficas G_1 y G_2 es una biyección $f : V_1 \to V_2$ entre los

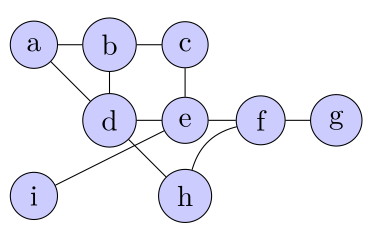
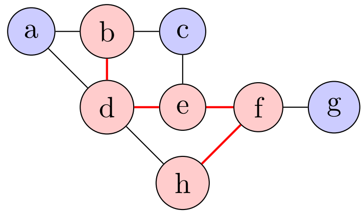
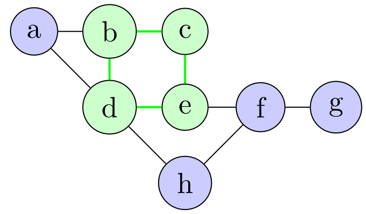


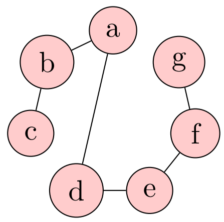
Figura 2.1 – Una gráfica conexa G.



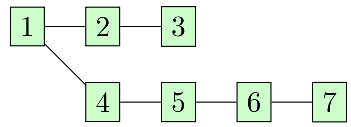
(a) Un camino (en rojo) en G.



(b) Un ciclo (en verde) en G.



(a) Gráfica G'



(b) Gráfica H'

Figura 2.3 – Isomorfismo entre G' y H'.

vértices de las gráficas tal que $\{a, b\}$ es una arista en G_1 si, y solo si, $\{f(a), f(b)\}$ es una arista en G_2. ⊣

Dos gráficas son isomorfas si existe un isomorfismo entre ellas. En ese caso escribimos $G_1 \cong G_2$

Definición 2.6

Decimos que $G' = (V', E')$ es una **subgráfica** de $G = (V, E)$, y escribimos $G' \subseteq G$, si se cumple que $V' \subseteq V$ y que $E' \subseteq E$. ⊣

Definición 2.7

Decimos que $G' = (V', E')$ es una **subgráfica inducida** de $G = (V, E)$ si se cumple que $V' \subseteq V$ y que cada arista en E cuyos vértices siguen estando en V' también es una arista en E'. ⊣

Definición 2.8

Una gráfica es **conexa** cuando podemos llegar de cualquier vértice a cualquier otro vértice siguiendo algún camino de aristas. ⊣

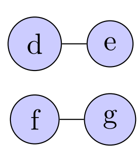


Figura 2.4 – Una subgráfica de G. No inducida, pues la arista (e, f) no aparece.

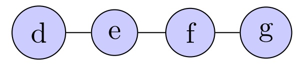


Figura 2.5 – Una subgráfica inducidad de G.

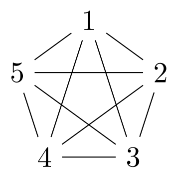


Figura 2.6 – Grafica plana, no representada en su forma plana.

Definición 2.9

Al número de aristas que emanan de un vértice dado le llamamos el **grado** de ese vértice. El máximo (o el mínimo) grado *de una gráfica* hace referencia al máximo (o el mínimo) grado de sus vértices. ⊣

Definición 2.10

Cuando una gráfica *puede* ser dibujada en el plano sin que ninguna de sus aristas se crucen, se dice que es **plana**. ⊣

2.1.1 Árboles

Definición 2.11

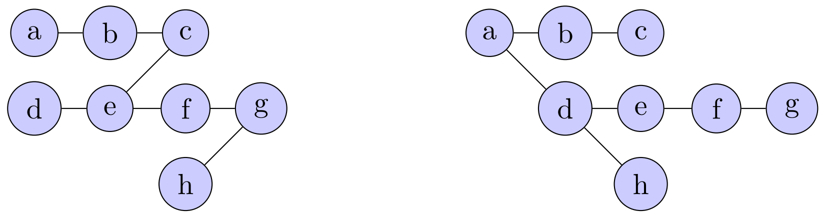
Un **árbol** es una gráfica *conexa* y que no contiene *ciclos*. ⊣

Cuando designamos un nodo como *raíz*, todos los demás vértices en el árbol pueden ser identificados por su posición relativa a la raíz. Esto es debido a que existe un único camino entre cualesquiera dos vértices en un árbol.

Si dos vértices son adyacentes, decimos que uno de ellos es el **padre** del otro, el cual es llamado **hijo**. De los dos, el padre es el que se encuentra más cerca de la ráiz. Por lo tanto la raíz del árbol es un padre pero no es el hijo de ningún vértice.

Así mismo, el hijo de el hijo de un vértice es el **nieto** de ese vértice (convirtiéndose en **abuelo**). Generalizando, decimos que un vértice v es **descendiente** de un vértice u si u es un vértice en el camino de v a la raíz. Entonces, llamaremos a u, un **ancestro** de v.

El propósito de este lenguaje es ayudarnos a recorrer el árbol. Recorrer el árbol, visitando cada nodo en algún orden, es un paso escencial en muchos algoritmos. Cuando visitamos primero todos los vértices de una misma generación antes de visitar la siguiente, decimos que hacemos una **búsqueda a lo ancho** (en inglés BFS o *Breadth First Search*). Por otro lado, cuando primero viajamos a lo más alejado posible de la raíz, y luego retrocedemos hasta una disyunción y poder viajar hacia adelante otra vez, le llamamos **búsqueda en profundidad** (en inglés DFS o *Depth First Search*). Decimos "búsqueda" porque



(a) Un árbol generador de G. **(b)** Otro árbol generador de G.

Figura 2.7 – Árbol generador

la mayoría de las veces recorremos el árbol tratando de encontrar vértices con cierta propiedad.

Definición 2.12
Dada una gráfica conexa G, un **árbol generador** de G es una subgráfica de G que es un árbol y que contiene todos los vértices de G. Todas las gráficas conexas tienen árbol generador. $\dashv$

2.2 Gráficas geométricas

Definición 2.13
Una gráfica es **geométrica** cuando es dibujada en el plano (no necesariamente es plana) utilizando segmentos (parte de una línea, definido por dos puntos), y ningúna tercia de vértices yacen sobre la misma línea recta, es decir que no son *colineales*. [5] $\dashv$

Aunque la definición de gráfica plana es muy similar, vale la pena denotar las diferencias: la planaridad es la cualidad de una gŕafica de *poder* ser dibujada sin que sus aristas se crucen, mientras que una gráfica geométrica necesariamente está representada en el plano.

Las siguientes definiciones fueron basadas en “On embedding a graph in the grid with the minimum number of bends”[6], excepto donde se indique lo contrario.

Definición 2.14
Una **gráfica de plano**, $G = (V, E)$ es una gráfica geométrica como la definida anteriormente salvo dos diferencias; el conjunto de aristas E es un conjunto de curvas simples que conectan dos vértices en V y, las curvas simples de E no se cruzan entre ellas, excepto, posiblemente, en los extremos. Una gráfica de plano divide al conjunto de puntos en el plano que no pertenecen a las aristas en regiones conectadas topológicamente, llamadas *caras*. La región no acotada es llamada *cara externa*. Una gráfica es planar si es isomorfa a alguna gráfica de plano. $\dashv$

Definición 2.15
Le llamaremos **representación planar** al conjunto de listas ordenadas de aristas $P(f)$,

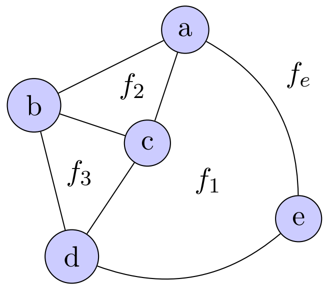


Figura 2.8 – Gráfica de plano

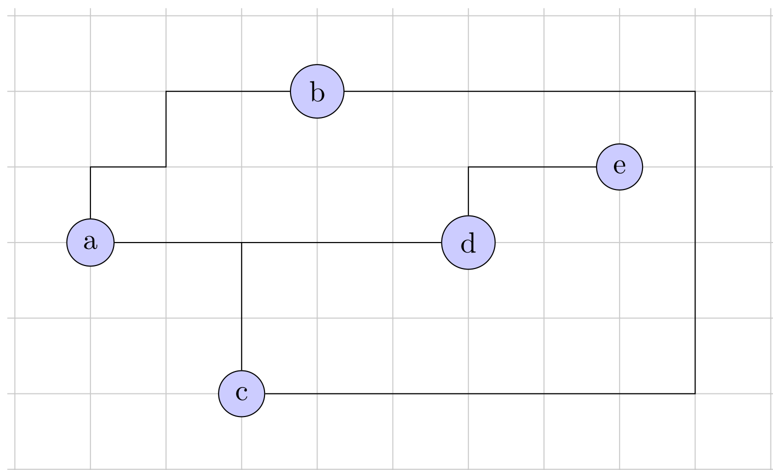


Figura 2.9 – Gráfica ortogonal

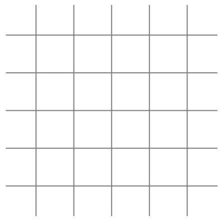
una por cada cara f, que describen la estructura topológica de la gráfica al listar las aristas que aparecen en el contorno de cada cara, y especificando la cara externa. ⊣

Definición 2.16

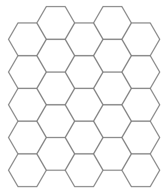
Una **gráfica ortogonal** es una gráfica de plano cuyas aristas son secuencias alternantes de segmentos horizontales y verticales. La *representacion ortogonal* describe la forma de una gráfica ortogonal, sin considerar la longitud de los segmentos. ⊣

Definición 2.17

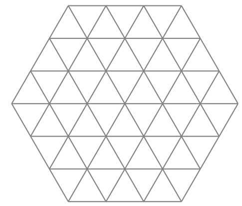
La **malla rectilinea** o **malla ortogonal** es la gráfica planar infinita cuyos vértices (o puntos) tienen coordenadas enteras, y cuyas aristas ligan a cada par de vértices que se encuentran a distancia unitaria. Un **encajamiento planar** en la malla $M = (V_M, E_M)$, de una gráfica $G = (V, M)$ es una correspondencia $Q : G \mapsto M$ que cumple que: (1) Q mapea cada vértice en G a una posición distinta de la malla; (2) Q mapea cada arista e de G a un camino en la malla $Q(e)$ cuyos extremos son mapeos de vértices ligados por e; (3) Los caminos $Q(e')$ y $Q(e'')$ correspondientes a cada par de aristas $\{e', e''\}$ de G, no tienen puntos en común, excepto, tal vez, los extremos. ⊣



(a) Rectilínea

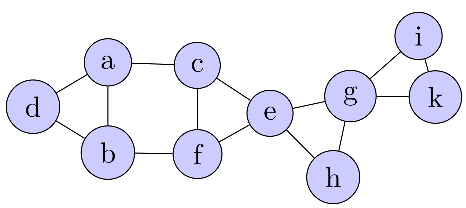


(b) Hexagonal

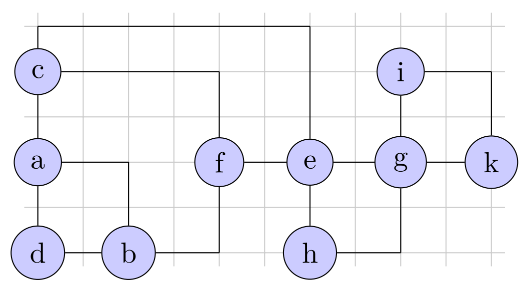


(c) Triangular

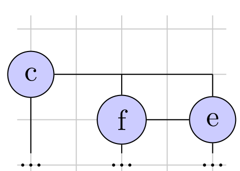
Figura 2.10 – Mallas



(a) G



(b) Q(G)



(c) Este mapeo no cumple la tercera condición

Figura 2.11 – Encajamiento en la malla

(a) No hay doblez entre a y c **(b)** Sí hay doblez entre a y c

Figura 2.12 – Doblez

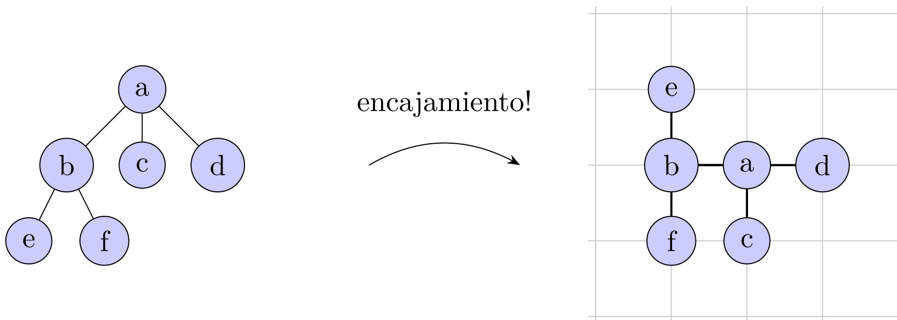


Figura 2.13 – Modelo en línea recta

La gráfica planar $Q(G)$ definida por el encajamiento (o incrustamiento) Q es isomorfa a G. Una gráfica admite ser incrustada en la malla si, y solo si, es planar y sus vértices tienen un grado máximo menor o igual 4. Para acortar, nos referiremos a estas graficas como *4-planares*.

Definición 2.18

Contamos un *doblez*[2] por cada par de aristas consecutivas en una gráfica ortogonal que no yacen sobre la misma línea. ⊣

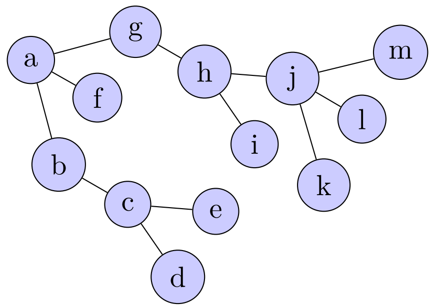
Definición 2.19

Un **modelo en línea recta**[2], o **s-modelo**, [1] se refiere a la gráfica ortogonal que resulta del encajamiento planar de un árbol en la malla. Como podemos ver en las figuras correspondientes, un arbol puede ser encajado sin dobleces en las aristas que no incluyen otro nodo del arbol, es decir que todas las aristas se mapean a caminos rectos. ⊣

Definición 2.20

Un **modelo local en línea recta**[4] de un árbol T consta de T, junto con una colección de *ordenamientos locales* $\{l_v\}_{v\in V}$, donde $\{l_v\}$ indica los ángulos entre las aristas incidentes a v, sin importar rotaciones. Para nuestro caso ortogonal, todos los ángulos descritos por $\{l_v\}$ son multiplos de $\pi/2$. ⊣

[1] Del inglés: *straight model*, cuya traducción propiamente sería modelo-rl



(a) El árbol

$$\{0_a, \pi/2_a, \pi_a, 0_b, \pi_b, 0_c, \pi/2_c,$$
$$\pi_c, 0_d, 0_e, 0_f, 0_g, \pi_g, 0_h, \pi/2_h, \pi_h,$$
$$0_i, 0_j, \pi/2_j, \pi_j, 3\pi/2_j, 0_k, 0_l, 0_m\}$$

(b) Ordenamientos locales.

Figura 2.14 – Un sl-modelo

2.3 Notación asintótica

Definiciones tomadas de *Introduction to Algorithms*[1].

En algoritmos, al analizar tamaños de entrada suficientemente grandes como para hacer relevante únicamente el crecimiento del tiempo de ejecución, decimos que estamos estudiando la eficiencia **asintótica** del algoritmo. Es decir, nos interesa cómo crece el tiempo de ejecución de un algoritmo cuando el tamaño de la entrada está en el límite, creciendo sin restricción.

Definición 2.21
Cuando solamente tenemos el **límite asintótico superior**, utilizamos notación de $\mathcal{O}$. Dada una función $g(n)$, denotamos $\mathcal{O}(g(n))$ (pronunciada “o grande de g de n” o, algunas veces, solo “o de g de n”) al conjunto de funciones ⊣

$$\begin{aligned}\mathcal{O}(g(n)) = \{f(n) :\ & \text{existen dos constantes positivas } c \text{ y } n_0 \\ & \text{tal que } 0 \leq f(n) \leq cg(n) \text{ para toda } n \geq n_0\}\end{aligned}$$

Usamos la notación $\mathcal{O}$ para dar una cota superior de una función, dentro de un factor constante.

Definición 2.22
Así como la notación de $\mathcal{O}$ nos provee de un límite asintótico *por arriba* o *superior*, Ω nos indica el **límite asintótico inferior**. Dada una función $g(n)$, denotamos $\Omega(g(n))$ (pronunciada “omega grande de g de n” o, algunas veces, solo “omega de g de n”) al conjunto de funciones ⊣

$$\begin{aligned}\Omega(g(n)) = \{f(n) :\ & \text{existen dos constantes positivas } c \text{ y } n_0 \\ & \text{tal que } 0 \leq cg(n) \leq f(n) \text{ para toda } n \geq n_0\}\end{aligned}$$

Definición 2.23

Por último, cuando tenemos "ensandwichada" la función, decimos que tenemos un **límite asintótico justo** señalada por Θ. Dada una función $g(n)$, denotamos $\Theta(g(n))$ (pronunciada "theta grande de g de n" o, algunas veces, solo "theta de g de n") al conjunto de funciones

$$\begin{aligned}\Theta(g(n)) = \{f(n) :\ & \text{existen tres constantes positivas } c_1, c_2 \text{ y } n_0 \\ & \text{tal que } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ para toda } n \geq n_0\}\end{aligned}$$

$\dashv$

3 Renderizado

El propósito final de este escrito es mostar el algortimo que, dado un árbol T, encuentra un encajamiento en la malla que minimiza el número máximo de veces que cualquier camino en T, se dobla. Hay bastante que desempacar de ese enunciado pero tener el propósito último resumido en una oración nos será útil para mantenernos enfocados.

El primer concepto que abarcaremos es el *modelo recto local* (2.20). Dentro de la definición de modelo recto local que incluye los ordenamientos locales, este concepto esta relacionado, pero no es igual, al de *sistemas de rotación*, ya que este último codifica solamente el orden en el que las aristas aparecen alrededor de un vértice v. Basado en el hecho de que un vertice puede tener distintos ordenamientos locales para el mismo conjunto de aristas adyacentes, podemos observar que entonces un mismo árbol puede tener distintos ordenamientos locales y, por tanto, múltiples *sl-modelos*, al conjunto de todos los sl-modelos de un árbol T ser7a denotado por $\mathcal{SM}(\mathcal{T})$. Dado que los *s-modelos* son la representaci7on gr7afica de un 7arbol en el plano, cada uno tiene asociado un sl-modelo que es la descripci7on de 7esa representacion. Diremos que el s-modelo es la renderizaci7on del sl-modelo.

Lema 3.1

Para todo sl-modelo $\mathcal{M}$, existe un s-modelo que es una renderizaci7on de $\mathcal{M}$, adem7as, dicho modelo puede ser encontrado en tiempo $\mathcal{O}(n)$. ⊣

Bibliografía

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to Algorithms* (2nd). The MIT Press.
2. de Luca, V. T. F., de Souza Oliveira, F., & Szwarcfiter, J. L. (2021). Minimum Number of Bends of Paths of Trees in a Grid Embedding. *Procedia Computer Science*, *195*, 118-126.
3. Levin, O. (2021). *Discrete Mathematics: An Open Introduction.*
4. Luca, V. T. F., Marín, N., Oliveira, F. S., Ramírez-Vigueras, A., Solé-Pi, O., Szwarcfiter, J. L., & Urrutia, J. (2022). Grid straight-line embeddings of trees with a minimum number of bends per path. *Inf. Process. Lett.*, *174*(106210), 106210.
5. Pach, J. (2018). Geometric graph theory. En *Handbook of Discrete and Computational Geometry, Third Edition* (Tercera edición). Chapman; Hall/CRC;CRC Press.
6. Tamassia, R. (1987). On embedding a graph in the grid with the minimum number of bends. *SIAM j. comput.*, *16*(3), 421-444.