Análisis De Algoritmos Primera Entrega Del Proyecto

Andrea Carolina Bayona y Camilo Andrés Moreno

Resumen—El presente Artículo contiene tres (3) problemas, y la información asociada a cada uno de ellos.

Index Terms—Análisis, Algoritmos, Problemas, Solucion, Grafo, Árbol, Hamming

1. Introducción

EL análisis de algoritmos es una parte importante de la Teoría de complejidad computacional más amplia, que provee estimaciones teóricas para los recursos que necesita cualquier algoritmo que resuelva un problema computacional dado. Estas estimaciones resultan ser bastante útiles en la búsqueda de algoritmos eficientes.[1]

2. Problema: minimizar recursos

2.1. Enunciado

Suponga que se planea contruir una nueva cadena de tiendas en una ciudad dada, usted tiene identificado una serie de ubicaciones potenciales en diferentes barrios. Además asuma que la demanda de productos en cada barrio de la ciudad es conocida. Si usted quiere construir exactamente k tiendas, ¿dónde debería localizarlas de forma que minimice la distancia promedio de los clientes? ¿Si en lugar usted dese construir una cantidad variable de tiendas, y el costo de construir una tienda en cada sitio es conocido, ¿dónde debería construir las tiendas de forma que minimice el costo total del construcción y la distancia promedio de los clientes?

2.2. Información Asociada

2.2.1. Programación lineal

La Programación Lineal corresponde a un algoritmo a través del cual se resuelven situaciones reales en las que se pretende identificar y resolver dificultades para aumentar la productividad respecto a los recursos (principalmente los limitados y costosos), aumentando así los beneficios. El objetivo primordial de la Programación Lineal es optimizar, es decir, maximizar o minimizar funciones lineales en varias variables reales con restricciones lineales (sistemas de inecuaciones lineales), optimizando una función objetivo también lineal. [2]

Los resultados y el proceso de optimización se convierten en un respaldo cuantitativo de las decisiones frente a las situaciones planteadas. Decisiones en las que sería importante tener en cuenta diversos criterios administrativos como: Los hechos La experiencia La intuición La autoridad [2]

Figura 1. Solución problema de PL [2]

2.2.2. ¿Como resolver un problema mediante programación lineal?

2.3. Problema Específico

Según la distribución de los barrios construir k tiendas o n tiendas (donde n es variable), de tal manera que se minimice tanto la distancia como el costo de contruccion de las tiendas

2.4. Soluciones Propuestas

2.4.1. Solución:

Entradas: un grafo G(n,m) dirigido donde n es el barrio de origen y m el barrio de destino G representa el grafo de los distintos barrios; las aristas del grafo contienen la distancia entre barrio y k es el numero de tiendas que se desean construir

Salidas: la solucion del problema en n grafos;

Ejemplo: suponga que se ingresan los siguientes datos: k=2 , un grafo con sus respectivas distancias
Procedemos a realizar una matriz de tamaño numero de barrios x numero de barrios, y en su contenido colocamos la distancia minima entre barrios usando algun algoritmo (por ejemplo: floyd warshall o Djkstra)
Luego sumamos las columnas para ver la menor distancia que hay de los demas barrios al de cada columna, y seleccionamos el barrio con esta menor suma para colocar la primera tienda.

De la figura 3 podemos ver que para este caso la primera tienda debe estar ubicada en el barrio E y los demas barrios acceden a el para realizar sus compras. El grafo en esta primera iteracion seria:
Para colocar la segunda tienda tenemos en cuenta la tabla y observamos la segunda menor suma, al ver esa columna

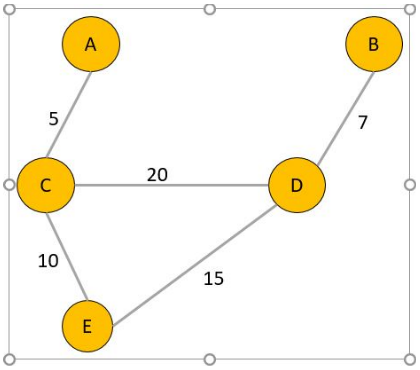

Figura 2. Datos de entrada

	A	B	C	D	E
A	0	32	5	25	15
B	32	0	27	7	22
C	5	27	0	20	10
D	25	7	20	0	15
E	15	22	10	15	0

	A	B	C	D	E
A	0	32	5	25	15
B	32	0	27	7	22
C	5	27	0	20	10
D	25	7	20	0	15
E	15	22	10	15	0
SUMA	77	88	62	67	62

Figura 3. Matriz 1

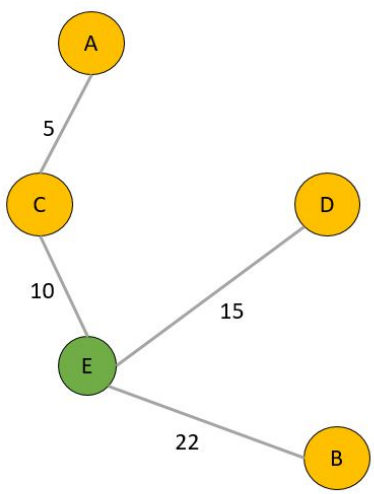

Figura 4. Grafo 1

comparamos si los barrios que puede surtir lo harian con menor distancia desde este barrio o desde el anterior.

	A	B	C	D	E
A	0	32	5	25	15
B	32	0	27	7	22
C	5	27	0	20	10
D	25	7	20	0	15
E	15	22	10	15	0
SUMA	77	88	62	67	62

Figura 5. Matriz 2

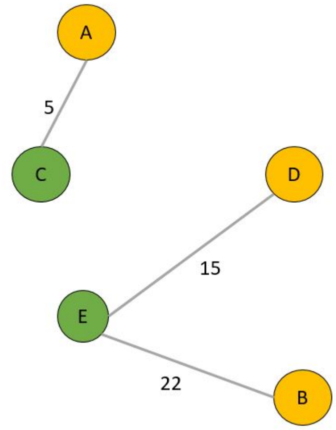

Figura 6. Grafo 2

3. Problema: Árbol De Mínima Expansión

3.1. Enunciado

Dado un grafo G = (V, E) con n vertices y m aristas. (El grafo podría representar una red telefónica). Cada arista es coloreada azul o roja. Tambien esta dado un parámetro k como parte de la entrada. Proponga un algoritmo que encuentre un árbol de expanción sobre G con exactamente k aristas azules, y exactamente n-k-1 aristas rojas. Determine el tiempo de ejecución del algoritmo y muestre que es correcto.

3.2. Información Asociada

3.2.1. Árbol

Un árbol es una estructura de datos ramificada (no lineal) que puede representarse como un conjunto de nodos enlazados entre sí por medio de ramas. La información contenida en un nodo puede ser de cualquier tipo simple o estructura de datos.
Los árboles permiten modelar diversas entidades del mundo real tales como, por ejemplo, el índice de un libro, la clasificación del reino animal, el árbol genealógico de un apellido, etc.[3]

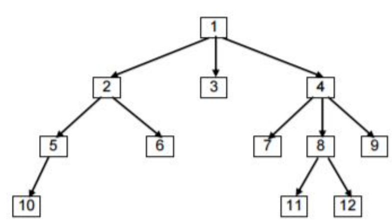

Figura 7. Ejemplo árbol[]

3.2.2. Árbol de Expansión

Dado un grafo conexo, no dirigido G. Un árbol de expansión es un árbol compuesto por todos los vértices y algunas (posiblemente todas) de las aristas de G. Al ser creado un árbol no existirán ciclos, además debe existir una ruta entre cada par de vértices.[4]

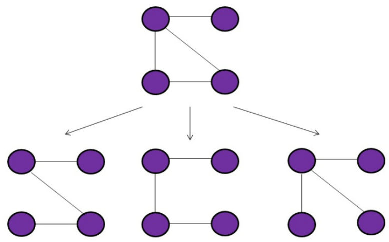

Figura 8. Ejemplo árbol de expansion[]

3.2.3. *Árbol de Expansión Mínima*

Dado un grafo conexo, no dirigido y con pesos en las aristas, un árbol de expansión mínima es un árbol compuesto por todos los vértices y cuya suma de sus aristas es la de menor peso.[4]

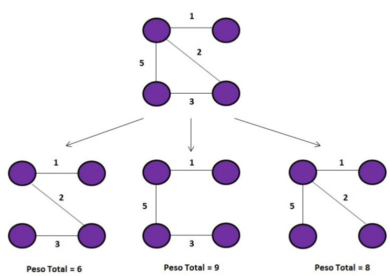

Figura 9. Ejemplo árbol de expansion mínima[4]

3.3. Problema Específico

A partir de un grafo dado, se debe construir un arbol de expansión con K aristas azules y n-k-1 aristas rojas.

3.4. Solucion Propuesta

Entradas: Grafo, numero de vertices (n), numero de aristas(m) y parametro

Salidas: boleano que indica si se puede o no armar el arbol
Ejemplo: suponga que se ingresan los siguientes datos (Ver figura 12):

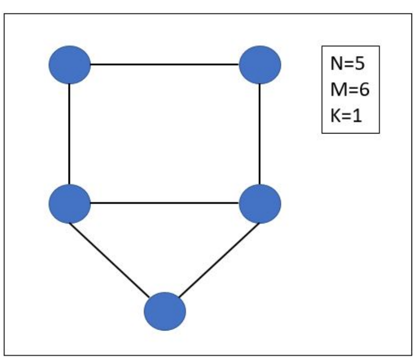

Figura 10. Datos ingresados

Se deberia calcular la cantidad de aristas de cada color. Luego sumar las aristas necesarias de cada color y si esta suma es menor a la cantidad total de aristas del grafo el grafo se puede realizar (Ver figura 13).

Figura 11. Proceso de validación

Finalmente se crea el arbol de expansión (Ver figura 14).

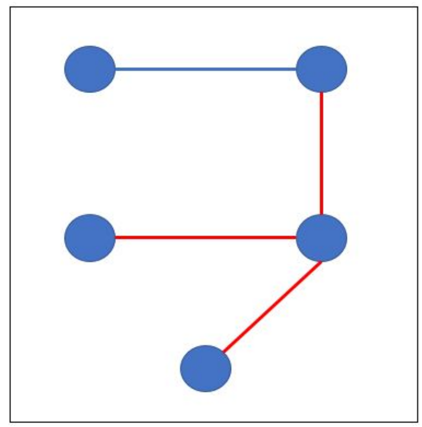

Figura 12. Árbol Dibujado

4. Problema: cliques en grafos

4.1. Enunciado

La distancia de Hamming dist(u,v) entre dos vectores binarios v = $(v_1, ..., v_n)$ and $w = (w_1, ..., w_n)$ es el número de índices k tal que $v_k \neq w_k$. Una pregunta fundamental en la teoría de la codificación es determinar el número

$$A(n,d) = max|S0,1n|dist(u,v)dforalldistinctu,vS|,$$

el máximo número de vectores binarios de longitud n que uno puede encontrar tal que dos vectores diferentes tienen una distancia de Hamming $\geq$ d. Por ejemplo, A(5, 4) = 2.
El grafo de Hamming H(n, d) = (V, E) es el gráfo con 2^n vertices V dados por cadenas binarias de longitud n. Nosotros tenemos (u, v) $\in$ E si y solo si dist(u, v) $\geq$ d. El número A(n,d) coincide con el tamaño de un clique máximo en H(n,d). Encuentre un algoritmo ."eficiente"para calcular el clique máximo en el grafo de Hamming (calcular el clique máximo es un problema NP-difícil).

4.2. Información Asociada

4.2.1. *Clique*

El término proviene de la palabra inglesa clique, que define a un grupo de personas que comparten intereses en común. En esta analogía, las personas serían los vértices y los intereses en común, las aristas. Cuando todas compartan un mismo interés, forman un grafo completo, es decir, forman un clique.[5]

En teoría de grafos, un clique en un grafo no dirigido G es un conjunto de vértices V tal que para todo par de vértices de V, existe una arista que las conecta. En otras palabras, un clique es un subgrafo en que cada vértice está conectado a cada otro vértice del subgrafo, es decir, todos los vértices del subgrafo son adyacentes. Esto equivale a decir que el subgrafo inducido por V es un grafo completo. El tamaño de un clique es el número de vértices que contiene.

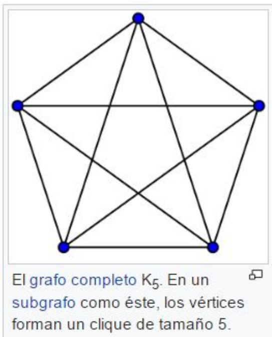

Figura 13. Grafo clique[]

4.2.2. *Aplicaciones al mundo real*

Link Farm

Una granja de enlaces es un grupo de sitios web que enlazan a todos los sitios de un grupo entre sí. Esto es una forma de spamming para los indices de un motor de búsqueda.

Los motores de búsqueda necesitan una forma para encontrar la relevancia de una página. Un método para encontrarla es examinando todos los enlaces provenientes de páginas relevantes.

Segmentación de imágenes y reconocimiento de patrones

Un problema muy común en computación gráca es el de encontrar patrones dentro de una imagen. Hay una gran cantidad de problemas que entran dentro de esta categoría desde reconocimiento de bordes hasta búsqueda de guras geométricas.

Un problema particular que se puede resolver utilizando una variación de Clique Máxima es la de reconocimiento de segmentos o regiones. Una región en una imagen es un conjunto de pixeles que comparten alguna característica.

Cálculo de probabilidades condicionales

El algoritmo general más común en redes bayesianas es el de agrupamiento ”árbol cliques”. El método de agrupamiento consiste en transformar la estructura de la red para obtener un árbol, mediante agrupación de nodos. Para ello, se hace una transformación de la red a un árbol de cliques máximas.

Biología

En el desarrollo evolutivo, una de las maneras de reconocer muchos de los procesos es mirar la expresión de genes en términos de las redes de la expresión de genes, que son los grafos. Los sistemas de regulación a menudo se revela como cliques en la red de expresión.

Al estudiar las estructuras de proteínas, una de las técnicas más importantes es analizar las proteínas similares, y encontrar cliques en común para las redes estructurales basados en grafos de las proteínas[6].

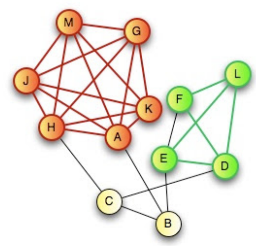

Figura 14. clique[6]

4.2.3. *Códigos de Hamming*

Es un método general propuesto por R. W Hamming usando una distancia mínima m. Con este método, por cada entero m existe un código de hamming de 2m-1 bits que contiene m bits de paridad y 2m-1-m bits de información. En este código, los bits de paridad y los bits de paridad se encuentran entremezclados de la siguiente forma: Si se numeran las posiciones de los bits desde 1 hasta 2m-1, los bits en la posición 2k, donde , son los bits de paridad y los bits restantes son bits de información.

El valor de cada bit de paridad se escoge de modo que el total de unos en un número específico de bits sea par, y estos grupos se escogen de tal forma que ningún bit de información se cubra con la misma combinación de bits de paridad. Es lo anterior lo que proporciona al código su capacidad de corrección.

Para cada bit de paridad en la posición 2k, su grupo de bits de información correspondiente incluye todos esos bits de información correspondiente cuya representación binaria tenga un uno en la posición 2k. La siguiente tabla muestra los grupos de paridad para un código de hamming de 7 bits o sea de la forma 2m-1 con m = 3. En este ejemplo, los bits de información son 4 y los bits de paridad son 3. Los bits de información están en las posiciones 7, 6, 5 ,3. Los bits de paridad están en las posiciones 1, 2, 4. [7]

Es un método general propuesto por R. W Hamming usando una distancia mínima m. Con este método, por cada entero m existe un código de hamming de 2m-1 bits que contiene m bits de paridad y 2m-1-m bits de información. En este código, los bits de paridad se encuentran entremezclados de la siguiente forma: Si se numeran las posiciones de los bits desde 1 hasta 2m-1, los bits en la posición 2k, donde , son los bits de paridad y los bits restantes son bits de información.[8]

4.2.4. *Ejemplos*

• Teniendo en cuenta las siguientes palabras “karolinz ”kerstin”, se puede decir que la distancia de haming es de 3.

• Para los siguientes números 1011101 y 1001001 es 2.
En informática, el código de Hamming es un código detector y corrector de errores.

	p_1	p_2	d_1	p_3	d_2	d_3	d_4	p_4	d_5	d_6	d_7
Palabra de datos (sin paridad):			**0**		**1**	**1**	**0**		**1**	**0**	**1**
p_1	**1**		0		1		0		1		1
p_2		**0**	0			1	0			0	1
p_3				**0**	1	1	0				
p_4								**0**	1	0	1
Palabra de datos (con paridad):	**1**	**0**	0	**0**	1	1	0	**0**	1	0	1

Figura 15. detección de errores [9]

4.3. Problema Específico

A partir de un grafo de Hamming dado y sus subgrafos, se debe hallar el clique máximo.

4.4. Solucion Propuesta

Entradas: Tres enteros n, m y d con los cuales se genera un grafo Hamming H (n, m) y con el entero d se compara las distancias de hamming para que sean menores que este.

Salidas: Para cada grafo H (n, m) se debe imprimir el número de cliques que tiene
Ejemplo:

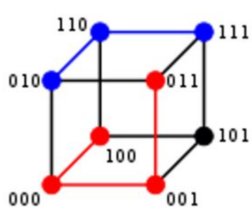

Figura 16.

Se debería calcular la cantidad de cliques que tiene el grafo. Luego de verificar la distancia de hamming entre cada vértice y se identifican los subgrafos posibles y su número de cliques.

5. Referencias

[1] “Análisis de algoritmos,” Wikipedia, la enciclopedia libre. 23-Mar-2015.

[2] “Programación Lineal.” [Online]. Available: http://www.ingenieriaindustrialonline.com/herramientas-para-el-ingeniero-industrial/investigación-de-operaciones/programación-lineal/. [Accessed: 23-Feb-2017].

[3] “Arboles.pdf.” Universidad Politecnica de Madrid, tomado de: http://ocw.upm.es/lenguajes-y-sistemas-informaticos/estructuras-de-datos/contenidos/tema4nuevo/Arboles.pdf.

[4] “ÁRBOL DE EXPANSIÓN MÍNIMA: ALGORITMO DE KRUSKAL,” Algorithms and More, 19-Apr-2012. .

[5] “Clique,” Wikipedia, la enciclopedia libre. 06-Jan-2016.

[6] R. González, “Esteban González: Problema de Clique,” Esteban González, 05-Jul-2011.

[7] “Códigos de Hamming.” [Online]. Available: http://aprendeenlinea.udea.edu.co/boa/contenidos.php/8b077438024 [Accessed: 20-Feb-2017].

[8] J. Simon, Çódigos correctores de errores”[Online] Avaible: http://www.um.es/docencia/jsimon/depmat/2012-2013/CodigosGrado/ApuntesCodigosGrado.pdf

[9] “calculobits.png.” [Online]. Available: http://camp.ucss.edu.pe/ingenium/images/calculobits.png. [Accessed: 23-Feb-2017].