Universidad Rafael Landívar

Facultad De Ingeniería

Matemática Discreta II

Ing. Juan Carlos Soto Santiago

Manual técnico

PROYECTO MATEMATICA DISCRETA II: ISOMORFISMO

José Carlos Mansilla Milian

Carne: 1160916

# INDICE

Contenido

[INDICE 2](#_Toc468375039)

[INTRODUCCION 2](#_Toc468375040)

[MARCO TEORICO 4](#_Toc468375041)

[Teoría de grafos 4](#_Toc468375042)

[Isomorfismo 4](#_Toc468375043)

[PLANTIAMIENTO Y DESARROLLO DEL PROBLEMA 5](#_Toc468375044)

[ANALISIS 5](#_Toc468375045)

[DISEÑO 5](#_Toc468375046)

[CONCLUCIONES 6](#_Toc468375047)

[OBJETIVOS 7](#_Toc468375048)

[GENERALES 7](#_Toc468375049)

[ESPECIFÍCOS 7](#_Toc468375050)

[BIBLIOGRAFIA 8](#_Toc468375051)

# INTRODUCCION

En el siguiente reporte vamos a presentar como fue elaborado la función de isomorfismo, los proceso y las implicaciones que esta misma conlleva para poder dar la respuesta esperada. Se va a presentar el funcionamiento a detalle para lograr comprender cuál fue el proceso para hacer posible el funcionamiento de dicha función.

Para la solución de este problema, fue necesario usar el lenguaje de alto nivel C# y Visual Studio 2013. Se hizo la verificación completa de todos los posibles errores y llegamos a la conclusión que al finalizar nuestro proyecto logramos alcanzar las metas planteadas.

Este reporte introduce también los conceptos básicos sobre los temas que son totalmente necesarios de dominar para poder lograr el desarrollo del proyecto de forma efectiva, alcanzando metas y conocimientos que nos van a permitir desarrollarnos en el ámbito de la ingeniería de sistemas.

Se logró entender cómo funcionan las funciones y las matrices de adyacencia y las permutaciones para dar la función o las funciones que se encuentren a lo largo del desarrollo del proyecto.

También vamos a presentar como es la elaboración del diseño del programa.

# MARCO TEORICO

## Teoría de grafos

Los grafos son una estructura de datos no lineal, la cual se puede usar para modelar diversas aplicaciones. Es una parte importante de la teoría combinatoria de matemáticas discreta, lo cual le da un carácter bastante amplio y complejo.

Los grafos se representan con frecuencia en la vida real, tal es el caso de una red de carreteras que enlace a cierto grupo de ciudades; aquí los nodos de la red o las ciudades representan los vértices del grafo, las carreteras que unen las ciudades representan los arcos o las aristas.

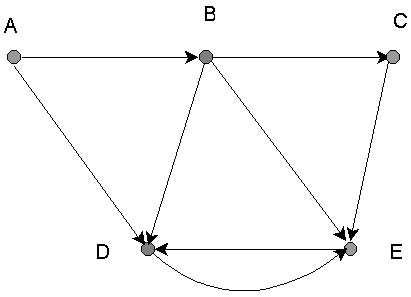


Figura 1: Representación gráfica de un grafo

### Isomorfismo

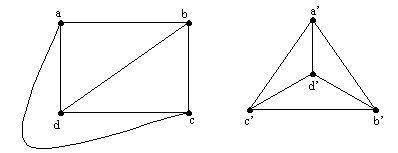
Dos grafos tendrán la misma “forma matemática” cuando la única diferencia entre ambos, en cuanto a su estructura, sea que hemos usado cualquier tipo de representación distinta para sus vértices, mientras que las conexiones entre ellos sean las mismas. En este caso diremos que son isomorfos.

Figura 2: Ejemplo de isomorfismo

# PLANTIAMIENTO Y DESARROLLO DEL PROBLEMA

En el desarrollo del problema nos encontramos con la elaboración de las permutaciones pertinentes, esto nos permite tener más exactitud al momento de desarrollar el proyecto, brevemente se encontrará de qué forma se llevó a cabo este proceso.

Las permutaciones nos permiten no solo encontrar una función de isomorfismo sino varias, llevando a cabo la comparación de todos los vértices de los grafos, con sus respectivos grados y los vértices a los cuales están unidos.

# ANALISIS

Antes de iniciar el desarrollo del proyecto, se llevó a cabo el análisis correspondiente en el que surgieron varias ideas de cómo afrontar el problema y la solución más óptima. Una posible solución fue hacer una función en la cual usando la recursividad íbamos generando una función de isomorfismo ordenando los vértices de los grafos por su respectivo peso, y con los vértices a los que estaba conectado y repetir el proceso hasta que se llegara a la solución, la desventaja de este proceso es que era demasiado pesado y por lo mismo fue más fácil crear las permutaciones de todos los vértices aunque es un poco más tardado, no es tan pesado llegar a la solución de esta forma y fue mucho mejor irnos por este camino y no por el anterior.

# DISEÑO

Para diseñar la solución, usamos las librerías de Windows Forms para desarrollar de una forma más clara a lo que se quería llegar con el proyecto, se adjuntará una imagen al final donde se muestra la solución, la cual se detallará más en el ***manual de usuario.***

# CONCLUCIONES

• El Código Hamming, es un sistema de detección y corrección automática de errores en información electrónica, el cual asocia una serie de bits de validación o paridad a los bits de datos, de tal forma que una alteración en cualquiera de esos bits de datos pueda ser detectada y corregida adecuadamente.

• Para entender de una manera más sencilla la elaboración del código se utilizaron mapas de Karnaugh y tablas de verdad y se relacionaron entre sí para poder lograr relaciones cruzadas y obtener los valores de bits de paridad.

• Los mapas de Karnaugh y la tabla de verdad debían de ser realizados obligatoriamente para poder generar en un proceso demostrado los resultados encontrados de cada una las funciones que el proyecto solicitaba.

• Si el sistema de códigos llega a fallar este debe de generar gráficamente el error en el display de 7 segmentos que contiene los datos del receptor.

# OBJETIVOS

## GENERALES

• Estudiar y comprender los aspectos relacionados con el código Hamming.

• Analizar y desarrollar el planteamiento del problema en el proyecto y proyectarlo en el sistema logisim

•Aprender a utilizar el sistema logisim para la realización del proyecto

## ESPECIFÍCOS

• Conocer las técnicas fundamentales para la detección y la corrección de errores relacionados con el código Hamming.

• Examinar y analizar el método de Hamming para la corrección y su posterior corrección de errores

• Complementar el código Hamming con el display de 7 segmentos para poder tener gráficamente errores que este puede generar

# BIBLIOGRAFIA

• Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). Data Structures and Algorithms. Massachusetts, US: Addison Wesley.

• Feldmeier, D. C. (1995, December). Fast Software Implementation of Error Detection Codes. IEEE/ACM Transactions on Networking, 3 (6), 640-651.

• Jacobsmeyer, J. M. (2004). Introduction to Error-Control Coding (Tech. Rep.). Pericle Communications Company. (URL: http://www.pericle.com/ papers/Error Control Tutorial.pdf)

• Rosquete De Mora, D. H. (2007, Abril). Algoritmos Eficientes para C´odigos Hamming y Estudio de su Aplicaci´on sobre Sistemas Existentes. (Trabajo Especial de Grado, dirigido por A. Mart´ınez y F. Perozo. Universidad de Carabobo, Venezuela.)

• Saturno, M. E. (2003). Software Error Protection Technic for High Density Memory. In 4th International Academy of Astronautics Symposium on Small Satellites for Earth Observation (p. 406-409). Walter de Gruyter, Berlin, DE.