PROYECTO 2: APLICACIÓN DE MATRICES

202001954 – Estuardo Sebastián Valle Bances (Coordinador)

Resumen

El modelado de sistemas se basa en la abstracción de
las ideas basadas en objetos del mundo real para construir, como su nombre lo dice, sistemas que relacionen sus objetos con las acciones específicas requeridas para la realización de una tarea. El lenguaje Unificado de Modelado permite representar estos sistemas en diagramas gráficos para facilitar la comprensión y complejidad de los mismos. Los diagramas se dividen en estructurales, que especifican la estructura computacional de los sistemas para los lenguajes de programación; y los de comportamiento, que grafican el estado de los objetos, sus acciones y consecuencias dentro del sistema sin llegar a la especificación.

Palabras clave
Matrices Ortogonales

Memoria

Pathfinding

Estructuras de Datos

Abstract

Systems modeling is based on the abstraction of object-based ideas from the real world to build, as the
name implies, systems that relate their objects to the specific actions required to perform a task. The Unified Modeling language allows to represent these
systems in graphic diagrams to facilitate their understanding and complexity. Diagrams are divided
into structure diagrams, which specify the computational structure of systems for programming languages; behavior diagrams, that graph the state of the objects, their actions and consequences within the system without reaching the specification.

Keywords
Orthogonal Matrix

Memory

Pathfinding

Data Structures

Introducción

En general, el presente proyecto busca la optimización en todos sentidos. La manera en la cual se ordenan las listas es una forma de optimización. La manera en la cual se guardan las listas es una forma de optimización. La manera en la cual se busca el cambio de un piso a otro es optimización de código en el mejor de todos sus sentidos.

Incluso, auxiliándose del concepto que plantea del enunciado en el siguiente párrafo:

"La empresa Chapín Warriors, S. A. ha desarrollado equipos automatizados para rescatar

civiles y extraer recursos de las ciudades que se encuentran inmersas en conflictos bélicos.

Con el fin de realizar las misiones de rescate y extracción, Chapín Warriors, S. A. ha

construido drones autónomos e invisibles para los radares llamados ChapinEyes. Los

ChapinEyes sobrevuelan las ciudades y construyen un mapa bidimensional de la misma, este

mapa bidimensional consiste en una malla de celdas, donde cada celda es identificada como

un camino, un punto de entrada, una unidad de defensa, una unidad civil, un recurso o una

celda intransitable. La figura 1 muestra la malla de celdas que elabora ChapinEyes luego de

sobrevolar una ciudad en conflicto"

Desarrollo del tema

1. Superioridad en Estructuras de Datos

En referencia a las estructuras de datos, mayormente conocidas cómo TDAs, es importante reconocer que el alocamiento de memoria es dinámico y no estático. Esto quiere decir que cada vez que se realice un cambio dentro de ellas. La forma en la cual se adjuntará un nuevo nodo será a partir de copias en memoria.

Si preferimos los tecnicismos, dentro de muchas estructuras de datos, podemos ampliar lo siguiente:

1. La mayoría de algoritmos que poseen las listas enlazadas buscan la realización de distintas estructuras de datos son presentadas con un gasto de memoria de O(n). Es decir, de manera lineal.
2. Al entender la mayoría de las listas y TDAs es posible notar que todas estas se basan en un algoritmo que crea un punto de partida y/o un punto de llegada.
3. Dentro de las listas enlazadas, las listas circulares y demás, esto puede ser visto cómo el HEAD o la CABEZA de la lista.
4. Todos los algoritmos de búsqueda poseen exactamente la misma estructura con cambios muy pequeños

2. Diseño Orientado a Objetos

El diseño Orientado a objetos es esencial para el desarrollo de distintas actividades incluyendo la forma en la cual se crean las TDAs.

• Desarrollar gráficos de interacción de objetos: Describen las intenciones del diseñador con respecto a cómo los objetos interactuarán en tiempo de ejecución para proporcionar la funcionalidad requerida, por lo que este constituye el primer paso en el desarrollo de un blanco modelo de caja del sistema.

• Desarrollar gráficos de visibilidad: Estos diagramas son utilizado para describir las otras clases a las que una clase determinada necesita hacer referencia, junto con las formas de referencia que se utilizarán y su duración. herencia.

• Desarrollar gráficos de herencia: Este paso considera la herencia como una construcción de diseño. Observa las relaciones entre las clases, y para identificar abstracciones comunes. Identifica nuevas superclases abstractas que pueden generalizar conjuntos de las clases identificadas en el modelo.

El objetivo era integrar todos los procesos más exitosos
de los programas y que se acoplaran a los requerimientos del programa. Se considera un método
de segunda generación. Una característica importante es la existencia del diccionario de datos que incluye el significado de los nombres de procesos, objetos, atributos, etc. durante el proceso. Los pasos para completarlo se dividen en los pasos del análisis y los que se asignan para el diseño

Pasos de Análisis

• Desarrollar el modelado de objetos: En este paso se identifican las clases piloto para el modelado, los atributos de las clases, las relaciones de ambas y también las invariantes (las variables que no cambian porque son requerimientos). Todos se incluyen en el diccionario de datos.

• Determinar la interfaz del sistema: Es una extensión del paso 1, pero con la especificación de cómo se fusionarán los
procesos del paso 1 en el software cuando se vea como un todo. Además de ello se crean escenarios de entrada y salida para agregar procesos al software y manejar todo tipo de problemas.

Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

• Desarrollo del modelo de interfaz: En este paso se estudian detalladamente los escenarios de entrada y salida propuestos en

el anterior y se describen los pasos y procesos que requiere el software para enfrentarlos.

• Evaluar los modelos de análisis: En este paso se analizan los procesos y ciclos descritos en los pasos anteriores y se determina si existe coherencia y relación entre ellos, además se evalúa que esos mismos procesos sean consistentes con el problema inicial y cumplan con los requerimientos iniciales

3. Resolución del Problema

a. Modelación Inicial:

Anterior a la abstracción por tipos de datos, es necesario sostener una idea general sobre las implicaciones del proyecto y cómo se realizará.

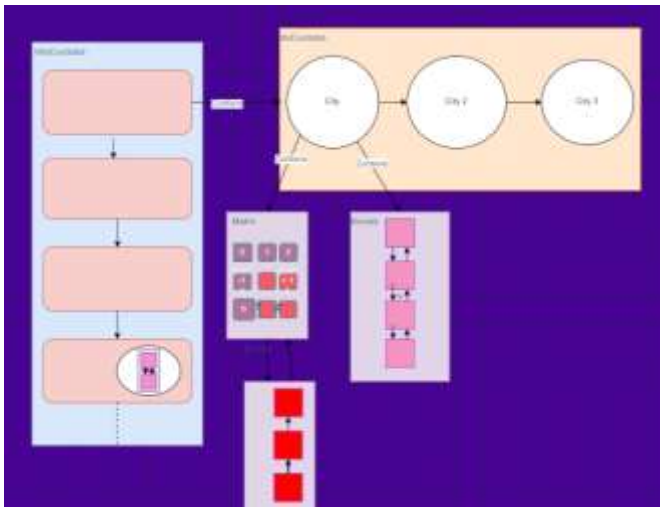

Figura 1. Modelación General del Sistema
Fuente: Elaboración propia

Además, el planteamiento de las matrices fue un diseño líneal que busca representar posiciones a partir de variables distintas

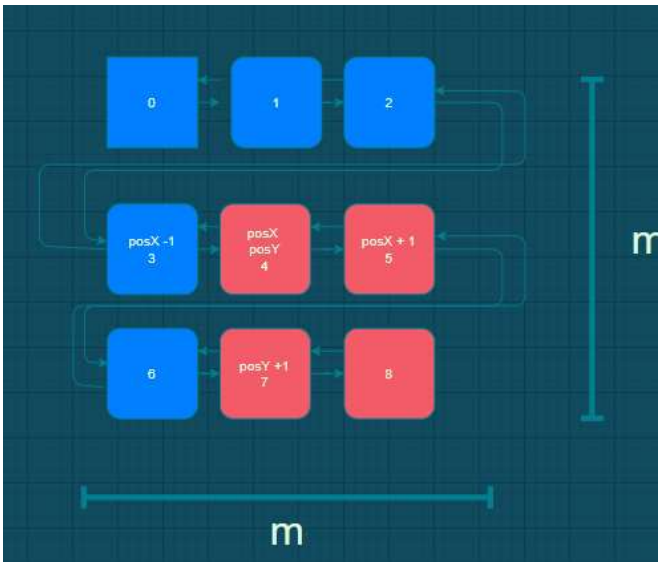

Figura 2. Matriz con encabezados
Fuente: Elaboración Propia

b. La búsqueda exhaustiva

La búsqueda exhaustiva es una técnica general de resolución de problemas. Se realiza una búsqueda exhaustiva y sistemática en el espacio de soluciones. Por ello, suele resultar ineficiente. La búsqueda se suele realizar recorriendo un árbol con el que se representan las posibles soluciones.

Basado en el Algoritmo BFS (Breadth First Search)

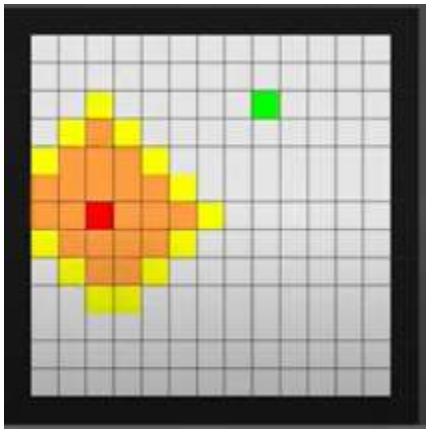

Figura 3. Graficación BFS
Fuente: John Sing

Este algoritmo tiene una expansión que se crea a partir de comparaciones arriba, abajo, izquierda, derecha. Es debido a esto que al encasillarlo, se logra una una forma de diamante.

c. Solución simple: recursividad.
Enfoque: encuentre el índice de origen de la celda en cada matriz y luego busque recursivamente una ruta desde el índice de origen hasta el destino en la matriz. El algoritmo implica encontrar recursivamente todos los caminos hasta que se encuentra un camino final hacia el destino.

Algoritmo:

Atraviese la matriz y encuentre el índice inicial de la matriz.
Cree una función recursiva que tome el índice y la matriz visitada.
Marque la celda actual y verifique si la celda actual es un destino o no. Si la celda actual es el destino, devuelve verdadero.
Llame a la función de recursión para todas las celdas adyacentes vacías y no visitadas.
Si alguna de las funciones recursivas devuelve verdadero, desmarque la celda y devuelva verdadero; de lo contrario, desmarque la celda y devuelva falso.

Conclusiones

El diseño orientado a objetos busca una visualización clara de instancias, clases y la manera en la que estos se relacionan en su ambiente.

2. Los procesos para el desarrollo de software más comunes dentro de la orientación a objetos son el “Unified Process” y el “Fusion”

Referencias bibliográficas

C. J. Date, (1991). *An introduction to Database Systems.* Addison-Wesley Publishing Company, Inc.

D. J. James, (2015). *PathFinding Methods and its comparisons*. John Song: https://youtu.be/GC-nBgi9r0U

Corden, (2020). *Data Structures*. O’Reilly. United States of America.

George T., (2001). *Algorithms in a Nutshell*. John O’Reilly. United States

Kenia (2003). *The Algorithm Design Manual*. John O’Reilly. United States

Figura 4. Diagrama de Objetos
Fuente: Propia
https://app.diagrams.net/#G1XriU50OZgCsKvfQSu8ciCwYfxcw1TjNH

Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2022.