UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Practica 01: Problema de adoquinamiento

ALUMNO

Kevin Antonio Carvajal - `319160242`

PROFESOR

María de Luz Gasca Soto

ASIGNATURA

Análisis de Algoritmos I 7156

13 de septiembre de 2023

1. Introducción:

   El problema de adoquinamiento consiste en cubrir completamente una superficie cuadrada de tamaño n x n con adoquines en forma de L. La cuadrícula inicial comienza con un adoquín especial que ocupa una sola localidad en la matriz y no puede ser tapado por algún adoquín en forma de L. Este informe detalla cómo implementamos una solución para este problema utilizando el lenguaje de programación Python.

   Algoritmo solución:

   a) Se crea una matriz de tamaño n x n inicializada con ceros, la matriz representa la cuadrícula que queremos llenar con adoquines. Inicializarla con ceros nos da una base uniforme para empezar, donde un valor de cero indica que esa posición en la cuadrícula aún no ha sido ocupada por un adoquín.

   b) Se coloca el adoquín especial en una posición aleatoria de la matriz. El valor -1 del adoquin se utilizó para marcar esta ubicación especial en la matriz. La elección del valor -1 como marcador es porque permite identificarlo fácilmente y distinguirlo de los demas adoquines que generalmente se representan con números enteros positivos.

   c) Se utiliza un contador global para llevar un registro de cuántos adoquines hemos colocado en la cuadrícula. Necesitamos una forma de asignar un identificador único a cada adoquín en forma de L que colocamos en la cuadrícula. Al incrementar el contador cada vez que colocamos un adoquín, garantizamos que cada adoquín tenga un número único, lo que facilita la visualización y comprensión de la solución.

   d) La función fillRegion se utiliza para llenar una región específica de la cuadrícula con adoquines en forma de L. Si la región es lo suficientemente pequeña (caso base 2 x 2), se llena directamente con adoquines en forma de L. Si la región es más grande, se divide en cuatro partes y se llama recursivamente a fillRegion para llenar cada una de esas partes. Utilizar una función recursiva para llenar la cuadrícula facilita la solución del problema.

   e) la función adicional llamada neighbors identifica y devuelve los adoquines que rodean al adoquín especial esta función proporciona información adicional que puede ser útil para identificar al adoquin especial.

   Ejecutar el Programa:

   a) Abra una terminal.

   b) Navegue a la carpeta donde se encuentra el archivo main.py que contiene el código fuente.

   c) Ejecute el programa proporcionando un valor entero positivo k como argumento. Este valor k determinará el tamaño de la cuadrícula n x n, donde $n = 2^k$.

   d) Ejemplo de ejecución: python main.py 3

   La implementación exitosa de este programa en Python resuelve el problema de adoquinamiento utilizando adoquines en forma de L. El programa toma como entrada el tamaño de la cuadrícula y genera una representación visual de la solución en la termina.