CETI.

Actividad: Algoritmo de Prim.

Nombre: David Alejandro Cisneros Delgadillo.
Registro: 21110381.
Grupo: 6°E.
Parcial 3.
Fecha: 10/06/2024
Nombre de la asignatura: Inteligencia Artificial.

¿Qué es el Algoritmo de Prim?

El algoritmo de Prim, desarrollado por el matemático y científico de la computación Robert C. Prim en 1957, es una técnica para encontrar un árbol de expansión mínimo en un grafo ponderado. Este árbol recubridor mínimo conecta todos los vértices del grafo con el menor costo posible.

¿Para qué sirve?

El algoritmo de Prim se utiliza para resolver problemas de optimización en redes y sistemas de comunicación. Su objetivo principal es encontrar una red de conexiones que minimice el costo total, como en los siguientes casos:
Redes de Comunicación: En la planificación de redes de datos, como la construcción de redes de fibra óptica o la conexión de routers, Prim ayuda a minimizar los costos de instalación y mantenimiento.
Diseño de Circuitos Electrónicos: En la fabricación de chips y placas electrónicas, Prim se aplica para conectar componentes con la menor longitud de cable posible.
Planificación de Rutas: En logística y transporte, el algoritmo de Prim optimiza las rutas de entrega y minimiza los gastos de combustible y tiempo.

¿Cómo se implementa en el mundo?

El algoritmo de Prim se implementa en sistemas de navegación GPS, en la planificación de redes de comunicación, en la logística de distribución y en la optimización de recursos en general.

¿Cómo lo implementarías en tu vida?

En mi vida cotidiana, podría aplicar el algoritmo de Prim para planificar mis desplazamientos diarios. Por ejemplo, al elegir la ruta más corta para ir al trabajo o al calcular el tiempo estimado de llegada a una cita.

¿Cómo lo implementarías en el diseño de videojuegos?

En el diseño de videojuegos, Prim es útil para crear mapas con caminos interesantes y desafiantes. Puede utilizarse para generar laberintos o para diseñar niveles en los que los jugadores deben explorar áreas conectadas de manera eficiente.

Algoritmo:

```
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // Crear un grafo de ejemplo (representado como una matriz de
adyacencia)
        int[,] graph = {
            { 0, 4, 2, 0, 0 },
            { 4, 0, 1, 5, 0 },
            { 2, 1, 0, 8, 10 },
            { 0, 5, 8, 0, 2 },
            { 0, 0, 10, 2, 0 }
        };

        int startVertex = 0; // Nodo de inicio (puedes cambiarlo según tu
necesidad)

        Prim(graph, startVertex);
    }

    static void Prim(int[,] graph, int start)
    {
        int n = graph.GetLength(0);
        int[] distance = new int[n];
        int[] parent = new int[n];
        bool[] visited = new bool[n];

        for (int i = 0; i < n; i++)
        {
            distance[i] = int.MaxValue;
            parent[i] = -1;
            visited[i] = false;
        }

        distance[start] = 0;

        for (int count = 0; count < n - 1; count++)
        {
            int u = GetMinDistanceVertex(distance, visited);
            visited[u] = true;
```

```
            Console.WriteLine($"Paso {count + 1}: Agregando vértice {u}
al árbol");

            for (int v = 0; v < n; v++)
            {
                if (!visited[v] && graph[u, v] != 0 && graph[u, v] <
distance[v])
                {
                    distance[v] = graph[u, v];
                    parent[v] = u;
                }
            }
        }

        Console.WriteLine("\nÁrbol de expansión mínimo:");
        for (int i = 1; i < n; i++)
        {
            Console.WriteLine($"Arista ({parent[i]}, {i}) con peso
{graph[parent[i], i]}");
        }
    }

    static int GetMinDistanceVertex(int[] distance, bool[] visited)
    {
        int minDist = int.MaxValue;
        int minVertex = -1;

        for (int v = 0; v < distance.Length; v++)
        {
            if (!visited[v] && distance[v] < minDist)
            {
                minDist = distance[v];
                minVertex = v;
            }
        }

        return minVertex;
    }
}
```

Fuentes Bibliográficas:

Prim, R. C. (1957). Shortest connection networks and some generalizations. The Bell System Technical Journal, 36(6), 1389-1401.
Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.

Este ensayo se basa en la obra de Robert C. Prim y en el libro “Introduction to Algorithms” de Cormen, Leiserson, Rivest y Stein.
: Prim, R. C. (1957). Shortest connection networks and some generalizations. The Bell System Technical Journal, 36(6), 1389-1401. : Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.