



Universidad de La Habana
Facultad de Matemática y Computación

Asignatura: Diseño y Análisis de Algoritmos

Conectando la UH

Integrantes

Jabel Resendiz Aguirre
Noel Pérez Calvo
Arianna Camila Palancar Ochando

Carrera: Ciencia de la Computación

Índice

Problema	2
Fase 1: Formalización del Problema	3
Notación	3
Restricciones del Problema	3
Fase 2: Análisis de Complejidad Computacional	4
Fase 3: Diseño de Soluciones Algorítmicas	6
Algoritmo de Fuerza Bruta: Exploración Exhaustiva del Espacio de Árboles Ex- pansivos	6
Algoritmo Heurístico: Reducción del Árbol de Oportunidades	8
Fase 4: Implementación y Análisis Experimental	11

Problema

La Universidad de La Habana, en su constante búsqueda de la excelencia académica y la innovación, se ha embarcado en un proyecto crucial para modernizar y expandir su infraestructura de red. Nuestro objetivo es dotar a todas nuestras facultades, centros de investigación y edificios administrativos con conectividad de fibra óptica de alta velocidad. Para este fin, contamos con el valioso apoyo técnico y logístico de ETECSA (Empresa de Telecomunicaciones de Cuba S.A.).

Nos enfrentamos a un desafío de diseño de red que requiere una solución óptima. Necesitamos interconectar todos los edificios principales de la UH con fibra óptica, creando una red robusta y eficiente. Cada posible conexión de fibra entre dos edificios tiene un costo de instalación asociado, que incluye desde los permisos internos y la mano de obra especializada de ETECSA hasta los materiales y las obras civiles necesarias.

Sin embargo, ETECSA ha establecido una restricción técnica fundamental que debemos respetar:

En cada edificio, la conexión de la fibra óptica se gestionará a través de un equipo de red central (un router o switch principal) que ellos nos proporcionan. Estos equipos tienen una capacidad limitada de puertos. Esto significa que un equipo en un edificio específico solo puede manejar un número máximo de conexiones de fibra óptica directas a otros edificios. Exceder este límite implicaría la necesidad de instalar equipos adicionales mucho más caros y complejos, o la implementación de soluciones de red alternativas que ETECSA no puede garantizar o que dispararían drásticamente el presupuesto del proyecto.

Nuestro objetivo principal es diseñar la red de fibra óptica que conecte todos nuestros edificios principales de la manera más económica posible. Esto implica seleccionar las rutas de fibra de tal forma que:

1. Todos los edificios estén interconectados a la red principal de la universidad, sin crear bucles innecesarios (buscamos una estructura de red eficiente).
2. Ningún equipo de red en ningún edificio exceda su capacidad máxima de conexiones directas (es decir, el número de cables de fibra que llegan o salen de un edificio no puede superar el límite de puertos del equipo de ETECSA).
3. El costo total de instalación de toda la red sea el mínimo posible.

Una planificación subóptima podría resultar en un sobrecoste significativo para la universidad, la necesidad de adquirir hardware de red adicional no previsto, o en una red ineficiente que no cumpla con las especificaciones técnicas y presupuestarias acordadas con ETECSA.

Formalización del Problema

El objetivo es diseñar una red de fibra óptica que conecte todos los edificios principales de la Universidad de La Habana mediante enlaces posibles provistos por ETECSA. Cada enlace tiene un costo de instalación y cada edificio posee un límite máximo de puertos disponibles. Se requiere encontrar una configuración de conexiones que:

- conecte todos los edificios,
- respete los límites de puertos por edificio,
- minimice el costo total de instalación.

Notación

En el lenguaje matemático, podemos definir la estructura del problema como un grafo simple y no dirigido $G = (V, E)$, donde:

- Un conjunto de edificios:

$$V = \{v_1, v_2, \dots, v_n\}.$$

- Un conjunto de posibles enlaces de fibra óptica: $E = (e_1, e_2, \dots, e_m)$

$$E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}.$$

- Un costo de instalación para cada enlace:

$$c : E \rightarrow \mathbb{R}_{>0}.$$

- Un límite de puertos (grado máximo permitido) en cada edificio:

$$d : V \rightarrow \mathbb{Z}_{\geq 1}.$$

Restricciones del Problema

Sea un subgrafo de G como $T = (V^T, E^T)$ debe satisfacer que:

1. Conectividad:

$$T \text{ es conexo} \rightarrow V^T = V$$

2. Estructura de árbol:

$$T \text{ no contiene ciclos} \rightarrow |E^T| = |V| - 1.$$

3. Límite de puertos por edificio:

$$\deg_T(v) \leq d(v), \quad \forall v \in V.$$

donde se define $\deg_T(v)$ como el grado del vértice v (también llamado cardinalidad de su vecindad) en el grafo T. Se desea conseguir el subgrafo T tal que minimiza el costo de la suma de los pesos de sus aristas, es decir:

1. Función objetivo:

$$\min_{T \subseteq E} \sum_{e \in T} c(e)$$

sujeto a las restricciones anteriores.

Análisis de Complejidad Computacional

En esta fase se determina la dificultad computacional del problema formalizado en la Fase 1. Demostraremos que la versión de decisión del problema es NP-completa, y por lo tanto, la versión de optimización es NP-dura.

Versión de Decisión del Problema

Dado un grafo $G = (V, E)$, costos $c(e)$, límites de grado $d(v)$ y un valor K , definimos la siguiente pregunta:

$$\text{¿Existe un subconjunto } T \subseteq E \text{ tal que } \begin{cases} G_T = (V, T) \text{ es un árbol,} \\ \deg_{G_T}(v) \leq d(v) \quad \forall v \in V, \\ \sum_{e \in T} c(e) \leq K? \end{cases}$$

Pertenencia a NP

Dado un conjunto de aristas T , quiere verificarse que se puede comprobar en tiempo polinomial, lo cual se cumplirá si se cumplen las siguientes condiciones:

- G_T es conexo (mediante un recorrido BFS/DFS),
- $|T| = |V| - 1$, es decir el subgrafo G_T forma un árbol,
- $\deg_{G_T}(v) \leq d(v)$ para todo $v \in V$,
- $\sum_{e \in T} c(e) \leq K$.

Por lo tanto, el problema pertenece a NP.

Demostración de NP-completitud

Para demostrar que dicho problema es NP-completo, se presenta una reducción en tiempo polinomial desde el problema HAMILTONIAN PATH en grafos no dirigidos, el cual es NP-completo.

Problema de Partida: Hamiltonian Path

Dado un grafo no dirigido $G = (V, E)$, el problema HAMILTONIAN PATH pregunta si existe un camino que visite todos los vértices exactamente una vez.

Construcción de la Reducción

A partir de una instancia de HAMILTONIAN PATH, construimos una instancia del problema de la siguiente manera:

- Se toma el mismo grafo $G = (V, E)$.
- Para cada arista $e \in E$, se asigna un costo $c(e) = 1$.

- Se fija un límite de grado uniforme:

$$d(v) = 2 \quad \forall v \in V.$$

- Se establece el umbral de costo:

$$K = |V| - 1.$$

La construcción es claramente polinomial.

Correctitud de la Reducción

(\Rightarrow) Si G posee un camino hamiltoniano, dicho camino contiene $|V| - 1$ aristas, es conexo, acíclico y cada vértice tiene grado a lo sumo 2. Por lo tanto, constituye un conjunto T válido para el problema con costo total $|V| - 1 \leq K$.

(\Leftarrow) Si existe un conjunto T que satisface las restricciones del problema, entonces T es un árbol con $|V| - 1$ aristas y $\deg_{G_T}(v) \leq 2$ para todo v . La única estructura de árbol donde todos los grados son a lo sumo 2 es un camino. Por lo tanto, T es un camino hamiltoniano del grafo original.

Conclusión

La existencia de un T válido para DC-MST-DECISION es equivalente a la existencia de un camino hamiltoniano en G . Dado que la reducción es polinomial y HAMILTONIAN PATH es NP-completo, se concluye:

DC-MST-DECISION es NP-completo.

Consecuencia

Dado que la versión de decisión es NP-completa, la versión de optimización (*Degree-Constrained Minimum Spanning Tree*) es NP-dura:

DC-MST es NP-dura.

Esto implica que no se conoce un algoritmo polinomial que resuelva el problema de forma óptima para instancias generales, salvo que P = NP.

Diseño de Soluciones Algorítmicas

El problema de construir un árbol expansivo con restricciones de grado (DC-MST) es NP-difícil, lo que implica que no existe un algoritmo polinomial conocido capaz de garantizar la solución óptima para instancias generales. Por ello, en esta sección se exploran distintas estrategias algorítmicas para abordar el problema, combinando métodos exactos y heurísticos.

Presentaremos los algoritmos propuestos de manera estructurada, incluyendo:

- Una descripción de la idea principal de cada algoritmo.
- El pseudocódigo correspondiente para su implementación.
- El análisis de complejidad computacional.

Esto permitirá evaluar las ventajas y limitaciones de cada enfoque, y proporcionará una base sólida para la implementación experimental y la comparación de resultados en la Fase 4.

Algoritmo de Fuerza Bruta: Exploración Exhaustiva del Espacio de Árboles Expansivos

Dado que el problema de encontrar un árbol expansivo de costo mínimo con restricciones de grado es NP-difícil, una primera aproximación consiste en analizar exhaustivamente todas las posibles soluciones y seleccionar aquella que cumple las restricciones y minimiza el costo total.

Un candidato a solución válida debe ser un **árbol expansivo**, es decir, un subgrafo conexo y acíclico que contiene todos los vértices del grafo y exactamente $|V| - 1$ aristas. El algoritmo de fuerza bruta consiste en enumerar todos los subconjuntos de aristas del grafo y verificar cuáles de ellos cumplen las condiciones necesarias para ser un árbol expansivo factible.

Idea del Algoritmo

Sea $G = (V, E)$ un grafo no dirigido con $|V| = n$ vértices y $|E| = m$ aristas. El procedimiento seguido es el siguiente:

1. Enumerar todos los subconjuntos $S \subseteq E$.
2. Descartar aquellos subconjuntos que no contengan exactamente $n - 1$ aristas.
3. Para cada subconjunto candidato:
 - Verificar que el subgrafo inducido sea conexo y acíclico, utilizando una estructura de datos *Disjoint Set Union* (DSU).
 - Comprobar que el grado de cada vértice no exceda su límite máximo permitido.
 - Calcular el costo total de las aristas seleccionadas.
4. Conservar el subconjunto válido cuyo costo total sea mínimo.

Este algoritmo garantiza encontrar la solución óptima, aunque a costa de un tiempo de ejecución exponencial.

Pseudocódigo

```

1 BruteForce -DCMST(G):
2     bestCost <- +infinity
3     bestTree <- empty
4
5     for each subset S of E:
6         if |S| != |V| - 1:
7             continue
8
9         initialize DSU with |V| elements
10        deg[v] <- 0 for all v in V
11        cost <- 0
12        valid <- true
13
14        for each edge (u,v) in S:
15            if deg[u] + 1 > d(u) or deg[v] + 1 > d(v):
16                valid <- false
17                break
18
19            deg[u] <- deg[u] + 1
20            deg[v] <- deg[v] + 1
21
22            if DSU.find(u) == DSU.find(v):
23                valid <- false
24                break
25
26            DSU.union(u,v)
27            cost <- cost + c(u,v)
28
29        if valid and cost < bestCost:
30            bestCost <- cost
31            bestTree <- S
32
33    return bestCost, bestTree

```

Análisis de Complejidad

Sea $n = |V|$ el número de vértices y $m = |E|$ el número de aristas del grafo.

- El algoritmo enumera todos los subconjuntos posibles de aristas, lo cual implica 2^m iteraciones.
- Para cada subconjunto candidato con $n - 1$ aristas, se realizan:
 - Operaciones de unión y búsqueda en la estructura DSU, con costo $O(n\alpha(n)) \approx O(n)$.
 - Verificación de los límites de grado y cálculo del costo total, ambos en $O(n)$.

Por tanto, la complejidad temporal total del algoritmo es:

$$O(2^m \cdot n)$$

En el peor caso, cuando el grafo es denso y $m = O(n^2)$, la complejidad crece como $O(2^{n^2})$, lo cual hace que este enfoque sea impracticable para instancias de tamaño moderado o grande.

Conclusión

El algoritmo de fuerza bruta permite obtener la solución óptima del problema de diseño de la red de fibra óptica bajo restricciones de grado. Sin embargo, su costo computacional exponencial limita su uso a instancias muy pequeñas. Esto justifica la necesidad de desarrollar algoritmos heurísticos o aproximados, los cuales se abordan en las siguientes secciones.

Algoritmo Heurístico: Reducción del Árbol de Oportunidades

Con el objetivo de disminuir el tamaño del espacio de búsqueda y la dificultad computacional del problema, se introducen una serie de propiedades estructurales del **árbol expansivo mínimo con restricciones de grado** que permiten reducir el grafo original sin perder optimalidad.

Sea T^* un árbol expansivo mínimo con restricciones de grado del grafo $G = (V, E)$.

Propiedades Estructurales

Lema 1. Sea $v \in V$ un vértice hoja, es decir, un vértice con grado uno en el grafo original G . Entonces, la única arista incidente a v debe pertenecer a T^* .

Demostración. Dado que T^* es un grafo conexo que contiene todos los vértices de G , el vértice v debe estar conectado al resto del árbol mediante su única arista incidente. Excluir dicha arista implicaría que v quedaría aislado, contradiciendo la conectividad de T^* . Por tanto, toda arista incidente a un vértice colgante debe incluirse necesariamente en T^* . \square

Lema 2. Sea $V_1 = \{v_i \in V \mid d(v_i) = 1\}$ el conjunto de vértices cuyo grado máximo permitido es uno, y sea

$$E_1 = \{(v_i, v_j) \in E \mid v_i, v_j \in V_1\}.$$

Si $|V| > 2$, entonces ninguna arista de E_1 puede pertenecer a T^* .

Demostración. Supóngase que existe una arista $(v_i, v_j) \in E_1$ incluida en T^* . Como $d(v_i) = d(v_j) = 1$, ninguno de estos vértices puede conectarse con ningún otro vértice adicional en T^* . Esto implica que el subgrafo resultante no puede ser conexo cuando $|V| > 2$, lo cual contradice la definición de árbol expansivo. Por tanto, todas las aristas de E_1 deben ser excluidas de T^* . \square

Lema 3. Sea v_k un vértice de grado dos en G , con vecinos v_i y v_j . Si no existe ningún camino entre v_i y v_j que no pase por v_k , entonces las aristas (v_k, v_i) y (v_k, v_j) deben pertenecer a T^* .

Demostración. Supóngase que alguna de las aristas (v_k, v_i) o (v_k, v_j) no pertenece a T^* . Dado que no existe un camino alternativo entre v_i y v_j que evite el vértice v_k , se deduce que T^* no puede ser conexo, lo cual contradice su definición. Por tanto, ambas aristas deben incluirse necesariamente en T^* . \square

Algoritmo de Reducción

A partir de las propiedades anteriores, se define un procedimiento de reducción que simplifica el grafo original antes de aplicar un algoritmo constructivo.

```
1 Reduction_DCMST(G = (V,E), d):
2     T* <- empty
3     1. Eliminar todas las aristas que satisfacen el Lema 2.
4     2. Identificar todos los vértices colgantes (grado 1),
5         agregar sus aristas incidentes a T* y eliminarlos del grafo.
6     3. Identificar vértices que satisfacen el Lema 3,
7         agregar las aristas correspondientes a T* y eliminarlas del grafo
8
9     return G reducido, T*
```

El costo computacional de este algoritmo de reducción es:

- Paso 1: $O(n^2)$,
- Paso 2: $O(n)$,
- Paso 3: $O(n^3)$.

Por tanto, la complejidad temporal total del algoritmo de reducción es:

$$O(n^3)$$

Construcción del Árbol con Restricciones de Grado

Una vez reducido el grafo, se construye un árbol expansivo utilizando un algoritmo greedy basado en el algoritmo clásico de Kruskal, adaptado para respetar las restricciones de grado.

El algoritmo itera seleccionando aristas de menor costo que conecten componentes distintas y que no violen los límites de grado de los vértices involucrados. Este procedimiento continúa hasta obtener $|V| - 1$ aristas.

```
1 MAIN_DCMST(G = (V, E), w, d):
2     V* <- V
3     E* <- empty
4     T* <- (V*, E*)
5
6     1. Ejecutar REDUCTION_DCMST(G, d)
7
8     2. E1 <- E
9
10    3. while |E*| < |V| - 1 do
11        seleccionar la arista de menor costo
12        emin = (vk, vh) en E1
13
14        eliminar emin de E1
15
16        if vk y vh estan en componentes distintas de T*
17            and deg_T*(vk) < d(vk)
18            and deg_T*(vh) < d(vh) then
```

```

20     E* <- E* U {emin}
21         unir las componentes de vk y vh en T*
22     end if
23 end while
24
25 4. Aplicar tecnicas de intercambio de aristas
26      (1-opt y 2-opt) para mejorar la solucion
27
28 return G* = (V*, E*)

```

Técnicas de Intercambio de Aristas

Para mejorar la solución obtenida, se emplean técnicas de *edge exchange*.

Intercambio 1-opt. Consiste en eliminar una arista del árbol y reemplazarla por una arista externa, siempre que el resultado sea un árbol. Esta operación puede modificar los grados de los vértices y, por tanto, debe verificarse nuevamente la restricción de grado.

Intercambio 2-opt. Consiste en reemplazar dos aristas del árbol por dos aristas externas, de forma que el grado de cada vértice se conserve. Esta operación garantiza que la restricción de grado siga siendo satisfecha y permite mejorar el costo total del árbol.

Complejidad del Algoritmo Heurístico

La complejidad temporal del algoritmo completo es la siguiente:

- Reducción del grafo: $O(n^3)$.
- Construcción inicial del árbol (Kruskal modificado): $O(n)$.
- Intercambio 1-opt: $O(n)$.
- Intercambio 2-opt: $O(n^2)$.

Por tanto, la complejidad temporal total del algoritmo heurístico es:

$$O(n^3)$$

Este enfoque permite obtener soluciones de alta calidad en tiempo polinomial, haciendo viable el tratamiento de instancias de tamaño considerable.

Implementación y Análisis Experimental

El Algoritmo Heurístico: Reducción del Árbol de Oportunidades no garantiza encontrar una solución factible incluso cuando esta existe, debido a decisiones greedy irreversibles que pueden saturar restricciones de grado prematuramente. Además, las técnicas de edge exchange propuestas solo son aplicables cuando ya se ha construido un árbol generador, lo cual no siempre ocurre.

Input donde falla:

```
1 10 18
2 4 7 3
3 8 7 19
4 3 1 19
5 4 1 13
6 6 5 5
7 9 1 2
8 3 0 9
9 2 6 4
10 0 2 5
11 5 2 11
12 8 0 11
13 6 8 6
14 2 7 8
15 7 5 15
16 3 7 17
17 2 1 1
18 1 5 3
19 7 0 7
20 1 2 2 3 3 1 3 3 3 2
```