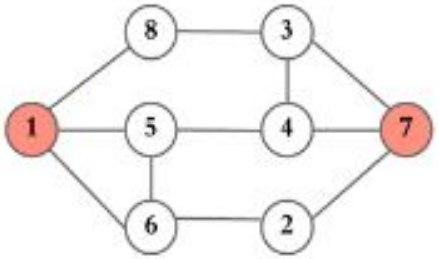
DOMINANT SET

Joaquin Casusol Escalante
Fabián Concha Sifuentes
Paolo Delgado Vidal
Sebastian Postigo Ávalos
Frank Salas Ticona

Introducción

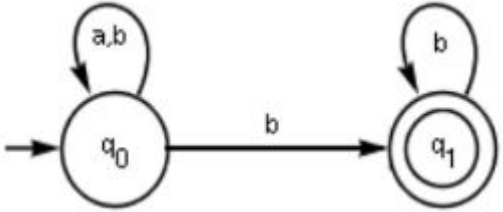
Un conjunto dominante se puede definir como un subconjunto de vértices R del conjunto de vértices V de un grafo G si y sólo si todos los vértices en V-R son adyacentes a por lo menos un vértice en R. Por esto podríamos decir que con el conjunto R de vértices, podemos llegar a todos los vértices del grafo.

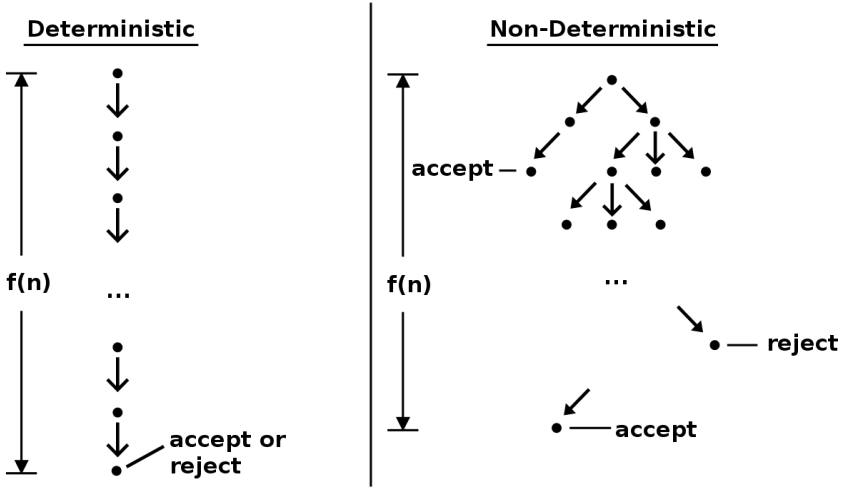


NP - NPC

Grupo NP: Resuelto en tiempo polinomial por un algoritmo no determinista, o en una máquina de turing no determinista.

NPC: Los problemas NP-completos pueden ser clasificados como tales si, estos problemas pueden ser verificables en tiempo polinomial y existe un algoritmo de fuerza bruta el cual encuentre dicha solución.

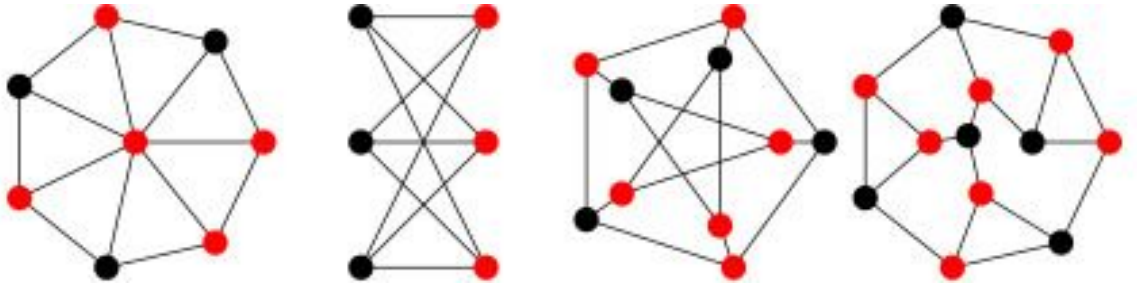




Vertex cover

La cobertura de vértices en un grafo se da cuando se escoge ciertos vértices pertenecientes a un grafo X, los cuales contienen todas las aristas de ese grafo.

Todos los vértices también representan una cobertura.
Si se obtiene el cubrimiento de un vértice el complemento debe de ser un set independiente.



Problemas relacionados a la cobertura

Cobertura mínima de vértices

Buscamos obtener la mínima y más óptima solución para el problema. Es difícil de aproximarse debido a la conjetura del juego único.

Problema de decisión del cubrimiento de vértices

A diferencia del anterior, este problema resuelve la cuestión si existe un cubrimiento de N vértices para un grafo G.

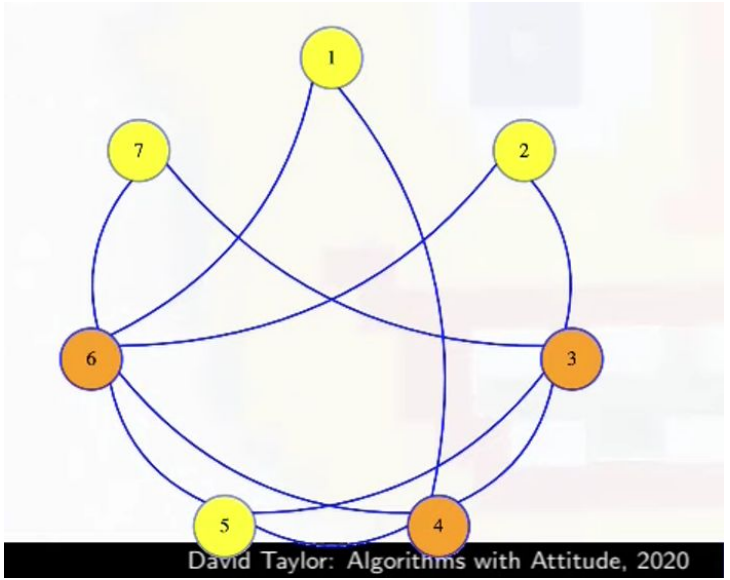
Aplicaciones

Suele ser usado frecuentemente para determinar la repetición de secuencias en el ADN.

Demostración

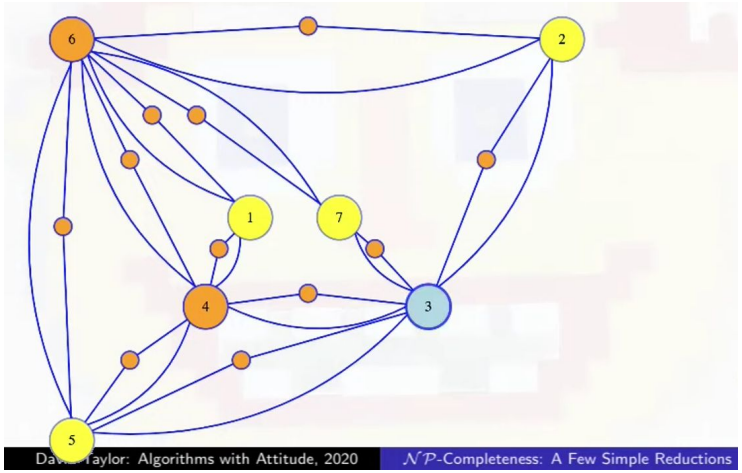
Conjunto Dominante: Dado un número natural k y un grafo G(V,E) donde V es el conjunto de vértices y E es el conjunto de aristas,¿Existe un conjunto dominante de G de tamaño k?

Cobertura de vértices: Dado un número natural k y un grafo G(V,E) donde V es el conjunto de vértices y E el conjunto de aristas, ¿Existe una cobertura de vértices de G de tamaño k?



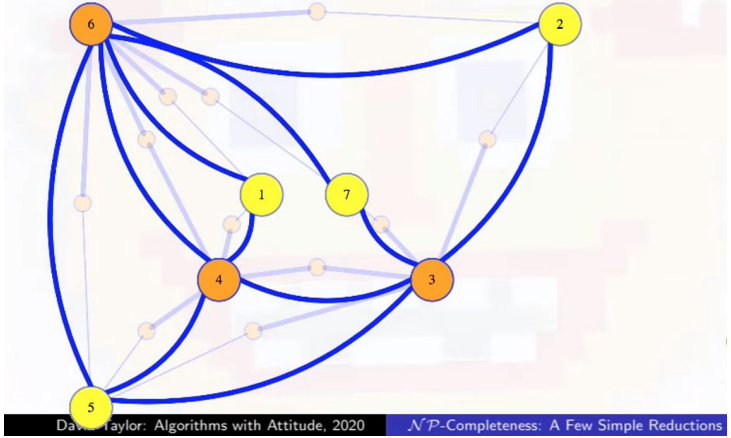
Demostración

- Con este grafo, vamos realizar una transformación en tiempo polinomial de nuestra entrada para que se nos sea más cómodo resolver este problema.
- Cada nuevo vértice añadido, une dos vértices originales y también reemplaza por decirlo así una arista original.
- Aquí entra el concepto de Conjunto dominante, para lo cuál buscaremos 3 vértices que dominen a todos los nuevos vértices.
- Estos vértices también dominan a los vértices originales.

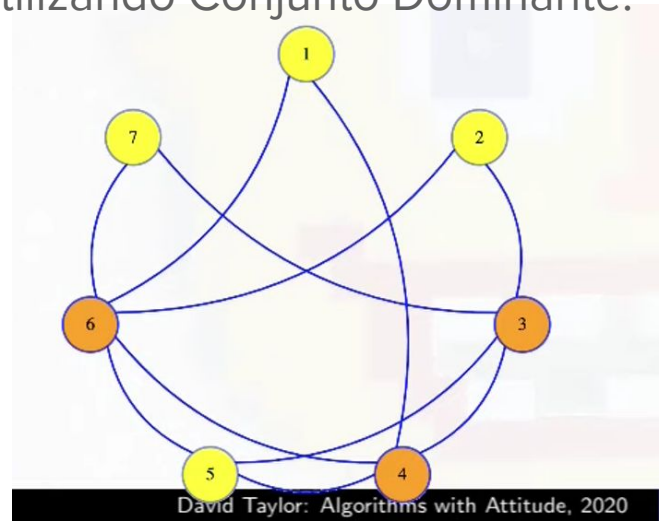


Demostración

- Estos también cumplen la particularidad de estar en el conjunto que cumple una cobertura de vértices.



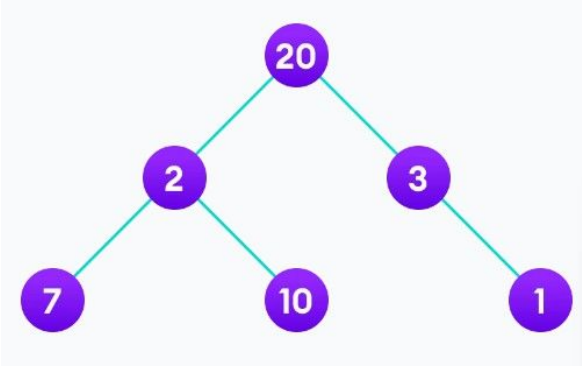
- Transformando de nuevo el grafo a la entrada inicial, damos con la respuesta de Cobertura de Vértices utilizando Conjunto Dominante.



Algoritmo Greedy

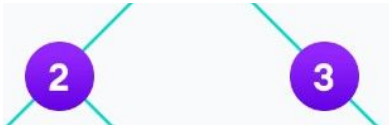
- Permiten resolución de problemas al dividir el problema en subproblemas
- Se tendrá una solución óptima para cada subproblema, bajo la información disponible en ese momento
- El algoritmo nunca revierte la decisión anteriormente tomada, incluso si la elección es incorrecta
    - Funciona en un enfoque de arriba hacia abajo (top-down)
- Al finalizar cada sub-solución, se tendrá un conjunto de soluciones
    - Resultará en la posible solución óptima del problema general

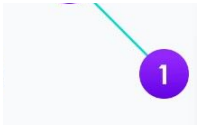
Algoritmo Greedy - Ejemplo



Encontrar el camino más largo:

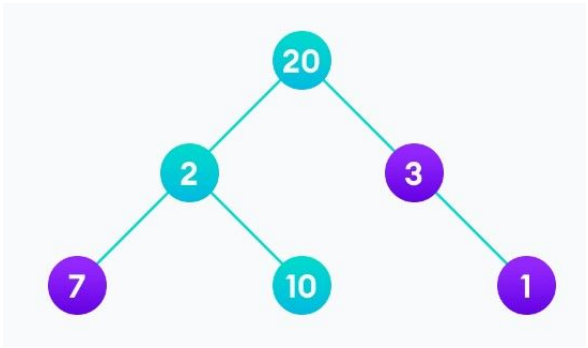
- Mejor solución en el momento:
  - Más pesado: 3





20 + 3 + 1 = 24

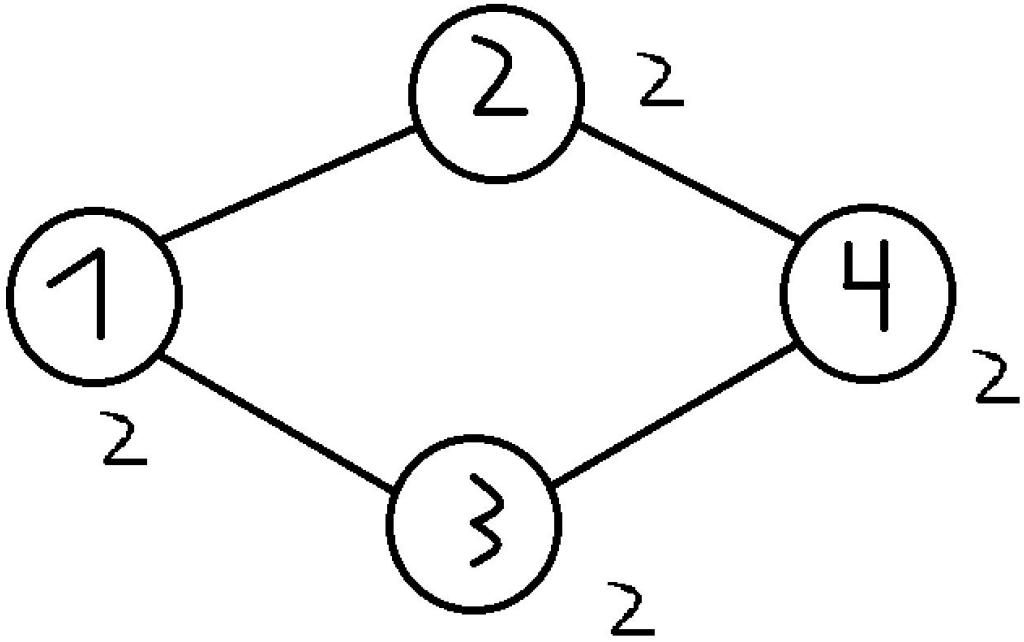
- Solución correcta:



20 + 2 + 10 = 32

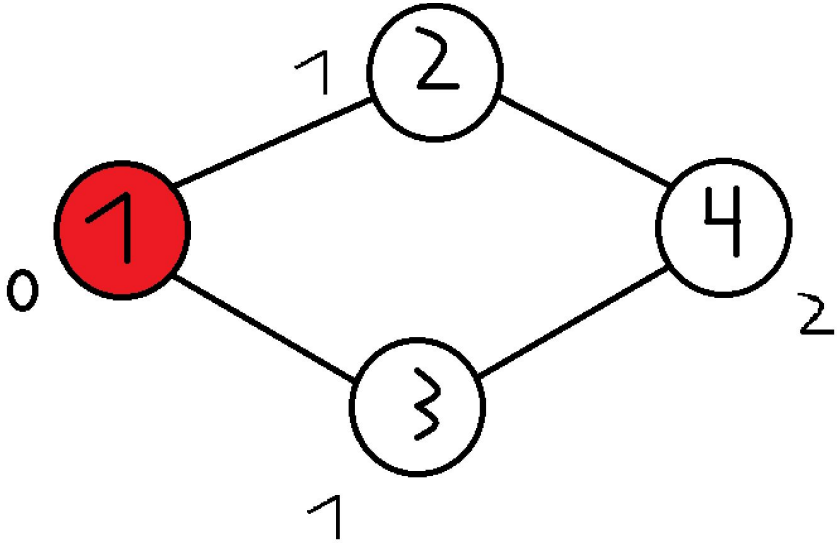
Implementación de heurística.

Empezamos leyendo nuestro grafo G(V,E), y sacamos el grado de cada vértice. Este grado representa el peso de cada vértice



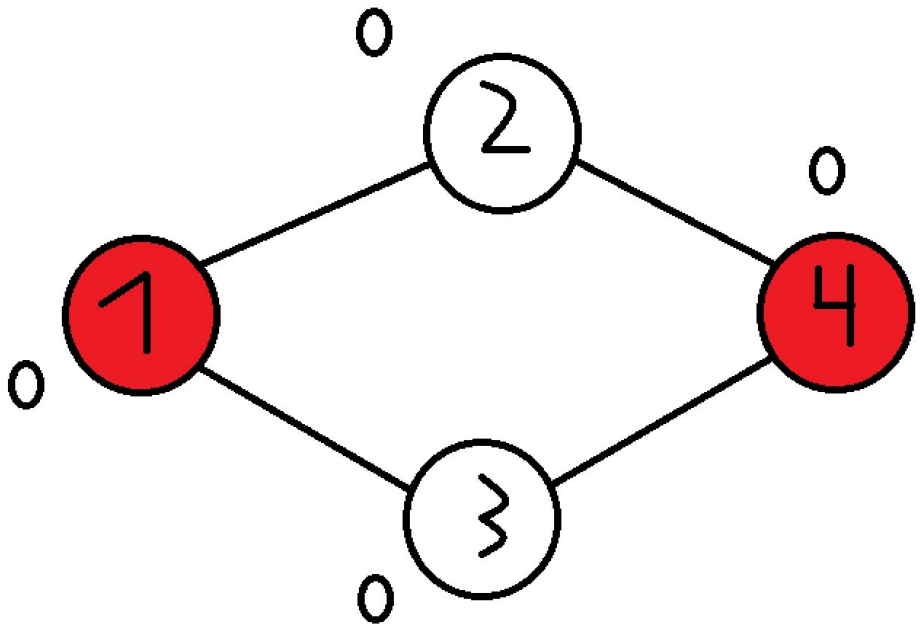
Implementación de heurística.

Seleccionamos al vértice con mayor grado (si hay más de uno escogemos a cualquiera), y restamos el peso de los vértices adyacentes en uno. Ahora el peso del vértice 1 será cero y los vértices 2 y 3 serán uno.



Implementación de heurística.

Volvemos a seleccionar el vértice de mayor grado, esta vez el de mayor grado es el vértice 4 y ahora su peso será cero , como podemos ver el peso de todos los vértices es cero así que ya tenemos nuestro conjunto dominante mínimo.



Prueba

```
1  #include <iostream>
2  #include <fstream>
3  #include "Graph.h"
4  #include "Algorithms.h"
5
6  sing namespace std;
7
8  nt main() {
9
10     grafo* Grafo;
11     ifstream O;
12     O.open("C:/Users/sebpost/Documents/ADA/NPC/g-sample.txt");
13
14     Grafo = Grafo->creatGraph(O);
15     conjunto_vertices solucion = heuristica_greedy(Grafo);
16
17     cout << "numero dominante: "<< solucion.numero_dominante() << endl;
18
19     delete(Grafo);
20     O.close();
21
22     return 0;
23 }
```

g-sample.txt - Notepad

File Edit Format View Help

```
4 8
1 2
2 1
1 3
3 1
2 4
4 2
3 4
4 3
```

Microsoft Visual Studio Debug Console

```
0
3
numero dominante: 2

E:\university\ADA\Dominating
h code 0.
To automatically close the co
le when debugging stops.
Press any key to close this w
```