Análisis de Algoritmos, Sem: 2019-1, 3CV1, Práctica 10, 03/12/2018

Práctica 10: Verificación en Tiempo Polinomial.

Hernández Castellanos César Uriel, Aguilar Garcia Mauricio

Escuela Superior de Cómputo
Instituto Politécnico Nacional, México
uuriel12009u@gmail.com, mauricio.aguilar.garcia,90@gmail.com

Resumen: Se verifica que dado un grafo, éste es un ciclo Hamiltoniano del grafo problema, esta verificación se realiza en tiempo polinomial.

Palabras Clave: Algoritmo, Complejidad, Tiempo, Polinomial, Ciclos, Hamiltoniano, Grafo y Problema

1. Introducción

Un camino hamiltoniano, en el campo matemático de la teoría de grafos, es un camino de un grafo, una sucesión de aristas adyacentes, que visita todos los vértices del grafo una sola vez. Si además el último vértice visitado es adyacente al primero, el camino es un ciclo hamiltoniano.

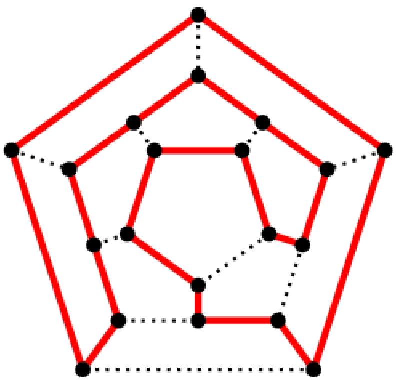


Figura 1: Camino hamiltoniano

El problema de encontrar un ciclo (o camino) hamiltoniano en un grafo arbitrario se sabe que es NP-completo.

Los caminos y ciclos hamiltonianos se llaman así en honor de William Rowan Hamilton, inventor de un juego que consistía en encontrar un ciclo hamiltoniano en las aristas de un grafo de un dodecaedro. Hamilton resolvió este problema usando cuaterniones, aunque su solución no era generalizable a todos los grafos.

Figura 2: Camino hamiltoniano

En teoría de la complejidad computacional, la clase de complejidad NP-completo es el subconjunto de los problemas de decisión en NP tal que todo problema en NP se puede reducir en cada uno de los problemas de NP-completo. Se puede decir que los problemas de NP-completo son los problemas más difíciles de NP y muy probablemente no formen parte de la clase de complejidad P. La razón es que de tenerse una solución polinómica para un problema NP-completo, todos los problemas de NP tendrían también una solución en tiempo polinómico. Si se demostrase que un problema NP-completo, llamémoslo A, no se pudiese resolver en tiempo polinómico, el resto de los problemas NP-completos tampoco se podrían resolver en tiempo polinómico. Esto se debe a que si uno de los problemas NP-completos distintos de A, digamos X, se pudiese resolver en tiempo polinómico, entonces A se podría resolver en tiempo polinómico, por definición de NP-completo. Ahora, pueden existir problemas en NP y que no sean NP-completos para los cuales exista solución polinómica

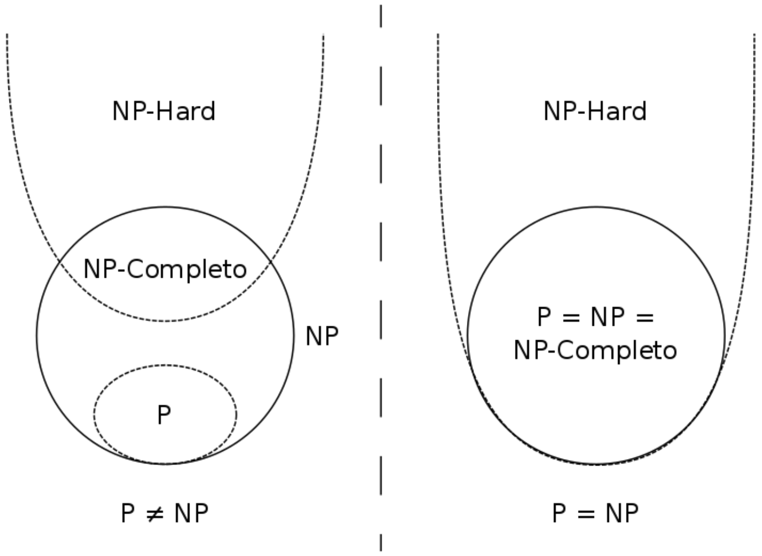


Figura 3: Diagrama de Euler de las familias de problemas y sus complejidades

2. Conceptos Básicos

2.1. Cota ajustada asintótica

En análisis de algoritmos una cota ajustada asintótica es una función que sirve de cota tanto superior como inferior de otra función cuando el argumento tiende a infinito. Usualmente se utiliza la notación $\theta(g(x))$ para referirse a las funciones acotadas por la función $g(x)$.[3]

2.2. Cota inferior asintótica

En análisis de algoritmos una cota inferior asintótica es una función que sirve de cota inferior de otra función cuando el argumento tiende a infinito. Usualmente se utiliza la notación $\Omega(g(x))$ para referirse a las funciones acotadas inferiormente por la función $g(x)$.[3]

2.3. Cota superior asintótica

En análisis de algoritmos una cota superior asintótica es una función que sirve de cota superior de otra función cuando el argumento tiende a infinito. Usualmente se utiliza la notación de Landau: $O(g(x))$, Orden de $g(x)$, coloquialmente llamada Notación O Grande, para referirse a las funciones acotadas superiormente por la función $g(x)$.[3]

3. Experimentación y resultados

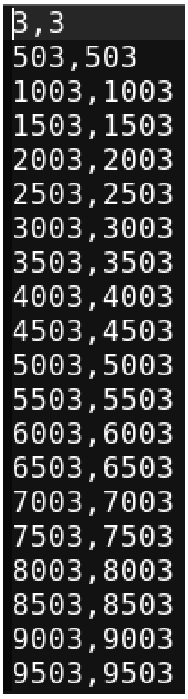
3.1. Verificación de Ciclos Hamiltonianos

(i). Mediante gráficas, muestre la complejidad que el algoritmo VerficarCicloHamiltoniano tiene:
Se tiene el algoritmo implementado del VerficarCicloHamiltoniano:

```
def checarCicloHamiltoniano(Grafo, caminoHam, n):
    contador = 0
    vertices = [ i for i in range(n)]
    for i in range(n-1):
        a = caminoHam[i];
        b = caminoHam[i+1];
        contador += 1
        if Grafo[a][b] == 0:
            return contador
        if a not in vertices:
            return contador
        else:
            vertices.remove(a)
    return contador
```

Figura 4: Función checarCicloHamiltoniano

La salida es la que se muestra en la siguiente figura, con n el tmaño de la entrada y t el contador.



```
3,3
503,503
1003,1003
1503,1503
2003,2003
2503,2503
3003,3003
3503,3503
4003,4003
4503,4503
5003,5003
5503,5503
6003,6003
6503,6503
7003,7003
7503,7503
8003,8003
8503,8503
9003,9003
9503,9503
```

Figura 5: Salida generada

Si procedemos a graficar los valores obtenidos en la salida, obtendremos como resultado lo siguiente:

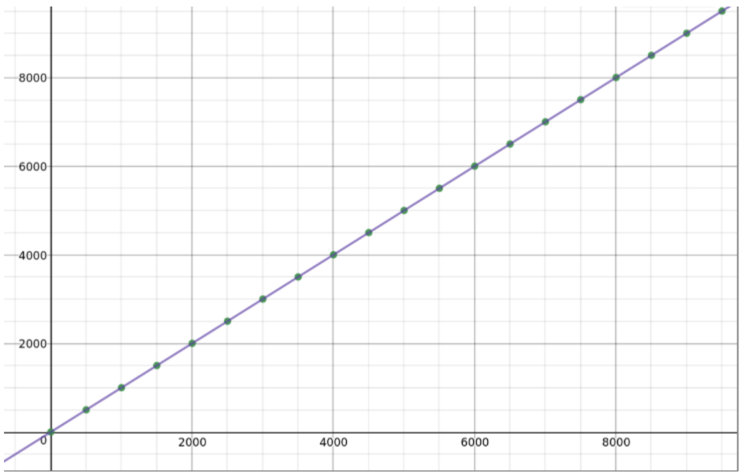


Figura 6: Complejidad del algoritmo planteado

Procedemos a calcular la complejidad del algoritmo que verifica un ciclo hamiltoniano, por medio del método del bloques, por lo que finalmente se obtiene que el algoritmo de verificación de un ciclo hamiltoniano, tiene complejidad lineal.

```
Algoritmo VerficarCicloHamiltoniano(grafo,camino)
  Input: Un arreglo camino[0,1,2,...,n-1,0]
  Output:
  vertices = [0,1,2,...,n-1];          O(1)      O(1)
  n = camino.lenght;                   O(1)
  for i = 0 to i = n - 1:                        O(n)      O(n)
      a = camino[i];                   O(1)
      b = camino[i+1];                 O(1)
      if grafo[a][b] == 0:             O(1)
          return False;
      if a not in vertices:            O(1)
          return False;
      else:                            O(1)
          vertices.remove(a);
  if vertices.empty:                   O(1)      O(1)
      return True;
  else:                                O(1)
      return False;
```

4. Conclusiones

4.1. Conclusión de Hernández Castellanos César Uriel

En la presente práctica fue posible verificar soluciones de problemas en los que no existen algoritmos que lo resuelvan de manera eficiente, otra forma de llamar a los problemas np o np completos. El problema que se abordó en el presente documento es que dado un grafo, saber si tiene un c iclo hamiltoniano de otro grafo, y lo único que se espera como resultado de éste programa es un resultado binario.

4.2. Conclusión Mauricio Aguilar Garcia

En la práctica se empleó una estrategia para comprobar si la solución dada a un problema NP completo es la correcta, y dada la definición de NP completos sabemos que la comprobación de la soluciones a estos problemas se realiza en tiempo polinomial, por lo que es bastante rápido de llevar a un algoritmo.

4.3. Conclusiones generales

Se verificó la solución de un problema que es np completo, para el cual no existe un algoritmo que sea capaz de resolverlo de manera eficiente, en lo particular se trató del problema del camino hamiltoniano, que es precisamente np completo

Referencias

[1] Gardner, M. "Mathematical Games: About the Remarkable Similarity between the Icosian Game and the Towers of Hanoi."Sci. Amer. 196, 150-156, May 1957

[2] Garey, M. and D. Johnson, Computers and Intractability; A Guide to the Theory of NP-Completeness, 1979.

[3] Introduction to Algorithms, Second Edition by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein