Índice general

Proyecto Final Diseño y Análisis de Algoritmos 2
Belsai Arango Hernández Daniela Rodríguez Cepero
1. Problema 2
2. Solución 3
3. Correctitud 4
4. Complejidad Temporal 8

Proyecto Final
Diseño y Análisis de Algoritmos

Belsai Arango Hernández
Daniela Rodríguez Cepero

Universidad de La Habana,
San Lázaro y L. Plaza de la Revolución, La Habana, Cuba
https://github.com/Daroce1012/DAA-problems/tree/Presupuesto-de-la-FEU
http://www.uh.cu

1. Problema

2. Solución

3. Correctitud

Sea X un problema de decision tal que:

1. $XinNP$
2. Existe X' (X' es NP - completo and $X' <= pX$)

Entonces X es un problema NP - completo

Un problema NP es un problema que su tiempo de ejecucion es superpolinomial y ademas cumple que el tiempo de verificacion de la solucion es polinomial.

Se asume que el problema 3-CNF es NP-completo.

Teorema: El problema de hallar el clique de tamanno maximo es NP- completo.
Demostracion:
Se cumple que un clique en un grafo G = ¡V,E¿es un subconjunto de V, tal que los vertices que lo integran son mutuamente adyacentes, osea un clique es un subgrafo completo de G. El problema del clique es hallar el clique de tamanno maximo es decir hallar la cantidad maxima de vertices que son mutuamente adyacentes. El problema es NP:
Demostracion
Un algoritmo para determinar que un grafo $G = (V, E)$ junto V vertices tiene un clique de tamanno k es listar los k subconjuntos de V y verficar por cada uno si forma parte del clique. El tiempo de ejecucion del algoritmo es omega k^2 combinaciones de el numero de vertices en k tamano, el cual es polinomial si k es una constante, pero k es un valor cerca de la cantidad de vertices/2, por tanto en ese caso es un algoritmo con un tiempo superpolinomial. Y el tiempo que toma verificar que la solucion obtenida responde al problema es ponlinomial ya que seria comprobar por cada vertice que sea adyacentes a los demas, es decir n^2 donde n es el numero de vertices que pertenece al clique. Por lo que queda demostrado que el problema es NP El problema es NP-completo: Se conoce que el problema 3-CNF es NP-completo, si se demuestra que 3-CNF¡=Clique, lo cual se cumple si se demuestra que el problema del clique es NP-hard. Se tiene una instancia de 3-CNF-SAT Sea a = C1 and C2 and ... and Ck una formula booleana en 3CNF con k clausulas para r = 1, 2,...,k, cada clausula Cr tiene exactamente 3 literales distintos l1, l2 y l3. Podemos construir un grafo que cumpla que a es satisfacible si y solo si G tiene un clique de tamanno k. EL grafo $G = (V, E)$ se construiria de la siguiente manera: por cada clausula Cr = l1 or l2 or l3 in a, colocamos 3 vetices v1, v2 y v3 en V. Se pondria una arista entre 2 vertices si ambos cumplen las siguientes condiciones:

1. vi y vj representan literales de diferentes clausulas y
2. sus literales correspondientes son consistentes, es decir un literal no es la negacion del otro

Este grafo se puede construir en tiempo polinomial. Para demostrar que esta transformacion de a en G es una reduccion: Primero vamos a suponer que a tiene una asignacion satisfactoria. Entonces cada clausula Cr contiene al menos un literal li el cual tiene asignado 1 y cada literal le corresponde un vertice vi. Tomando de cada clausula un literal que es 1 se produce un conjunto V' de k vertices. Digamos que ese conjunto V' es un clique. Por cada 2 vetices que pertenencen a V' donde sus literales son de clausulas distintas y esos literales son 1 por ser una asignacion satifactoria y esos literales no son complementos por ello por las condiciones descritas anteriormente podemos decir que en la construcion de G la arista entre esos 2 vertices pertenece a E. En cambio suponiendo que en G hay un clique V' de tamanno k. las aristas en G no conectan 2 vertices que representan literales que se encuentran en una misma clausula y por tanto en V' solo se encuentra un vetice por cada clausula. Por lo que se puede asignar 1 a cada literal li ya que vi pertence a V' sin temor a asignar 1 a un literal y su complemento ya que se garantiza en G que no existe aristas entre literales inconsistentes. Cada clausula es satisfactoria y por lo tanto a es satisfactoria.

Teorema:
El problema del conjunto independiente maximo es NP-completo.
Demostracion:
C - problema del conjunto independiente maximo Probar C pertenece a NP Para determinar el conjunto mayor de vertices independientes en un grafo $G = (V, E)$ se forman todas las posibles combinaciones de vertices que cumplan que no tienen ninguna arista en comun utilizando dinamica y teniendo en cuenta la lista de adyacencia. El tiempo de ejecucion de ese algoritmo es exponencial, por tanto no forma parte de los problemas polinomiales. El tiempo de ejecucion para obtener el certificado en este problema es polinomial. El procedimiento seria verificar por cada vertice que no tiene ninguna arista en comun con ninguno de los otros vertices del conjunto, este tiempo es n^2

Lo siguiente es probar que el problema del clique maximo $<= p$ C, lo cual muiestra que el problema del conjunto independiente maximo es NP- hard. El algoritmo de reducion empieza con una instancia del problema del clique maximo Sea un grafo $G = (V, E)$ con un clique c = v1, v2, ..., vk con k vertices. Podemos construir un grafo que cumpla que c un clique maximo si y solo si existe otro grafo el cual tiene un conjunto independientes maximo de tamanno k El grafo se construiria de la siguiente manera: Seria el grafo complemento del grafo G, es decir por cada vertice en el grafo G va existir un vertice en G', el conjunto de aristas en G seria las aristas complemento en G'. Se puede verificar que la construccion del grafo es en tiempo polinomial, seria por cada vertice eliminar sus aristas y formar aristas con aquellos vertices que era independiente en el grafo original.

Para mostrar que la tranformacion de problema de clique maximo a problema del conjunto independiente maximo es una reduccion, primero vamos a suponer que tenemos un clique c que es maximo, por tanto cada vertice que pertenece a ese clique tiene aristas con todos los demas vertices del clique y esto se cumple con todos. Si tenemos un grafo que es el complemento del grafo original, entonces todas aristas que existian entre los vertices que eran mutuamente adyacentes dejan de existir convirtiendo estos vertices sin relacion alguna entre ellos, es decir

conviertiendo estos vertices en un conjunto independiente. Por que este conjunto independiente es maximo?.
Supongamos que no lo es entonces existe un vertice que es independiente con todos los vertices del conjunto en el grafo complemento. Si esto se cumple entonces en el grafo original este vertice tiene una arista con cada uno de los vertices del conjunto, por tanto es mutuamente adyacentes a ello, por lo que tambien forma parte de un clique, un clique donde estan todos los vertices anteriores y ademas este, entonces hemos obtenido un clique de tamanno mayor al clique que teniamos al inicio. Contradicion porque hemos asumido que el clique inicial era maximo. Por lo que queda demostrado que el problema del conjunto independiente maximo es NP-completo

Teorema:
El problema kevin el encargado es NP-completo.
Demostracion:
Para demostrar que el problema de kevin el encargado pertenece a NP primero debemos demostrar que su tiempo en ejecucion es superpolinomial: Un algoritmo para determinar las k propuestas validas de m propuestas que plantea los cursos es hallando las combinaciones en m de tamanno k y verificando que las propuestas seleccionadas no tienen dias en comun entre ellas. El tiempo de ejecucion del algoritmo es superpolinomial. Para que pertenezca a los problemas NP el tiempo de verificacion debe ser polinomial, es decir el tiempo que debe tomar para comprobar que una solucion es valida y satisface el problema debe ser polinomial. La solucion al problema es un conjunto de k propuestas, la forma de comprobar que es valida es verificando por cada propuesta que los dias selecionado para las pruebas son diferentes a los dias de las demas propuestas, esto se haria con una lista de marcas donde cada vez que se obtenga un dia de prueba se marca como ocupado esa casilla en el array. El tiempo de ejecucion de este procedimiento seria la suma de las cantidad de pruebas de cada curso, es decir un tiempo polinomial. Lo siguiente seria probar que el conjunto independiente $<= p$ problema de kevin, lo cual muestra que el problema de kevin es NP-hard. Supongamos que tenemos una instancia del problema del conjunto independiente Sea un grafo $G = (V, E)$, podemos construir k propuestas validas si y solo si en G existe un conjunto independiente de tamanno k. Se construiria el problema de la siguiente forma: por cada vertice vi representaremos una propuesta de un curso en el problema de kevin. Cada arista entre 2 vertices en G representaria en el problema de kevin 2 propuestas que tienen al menos 1 dia en comun. La construccion de estas propuestas seria en tiempo polinomial. Un ejemplo de esta construccion seria: ... Debemos mostrar que esta tranformacion del grafo en propuestas es una reduccion. Primero supongamos que existen n vertices que son el conjunto maximo independiente en el grafo. Cada vertice corresponde con una propuesta de un curso. Podemos decir entonces que la cantidad de propuestas que no tienen ningun dia en comun es n, ya que como se planteo anteriormente una arista entre 2 vertices representaria 2 propuestas que tienen dias en comun, por tanto si son

independientes los vertices las propuestas no tienen ninguna interseccion entre ellas.

Utilizamos una metaheuristica para nuestro problema. La meteheuristica escogida fue el algoritmo genetico. Los resultados obtenidos despues de aplicar la metaheuristicas es que al tener soluciones que puede que un principio no sean validas al tener dias en comun, al mezclarlas entre ellas o mutarlas podemos llegar a obtener la solucion deseada.

4. Complejidad Temporal