Solución de algunos problemas

Marlon Díaz Pérez

December 20, 2024

1 Número Cromático:

Definición del problema: el número cromático de un grafo es el número mínimo de colores necesarios para colorear los vértices del grafo de manera que dos vértices adyacentes no compartan el mismo color. Hallar el número cromático de un grafo.

Solución: Para demostrar que este problema es NP-Completo primero vamos a demostrar que el problema de dado un grafo G y el entero 3, decir si el grafo G es 3 coloreable es NP-Completo. A este problema le llamaremos 3-Coloreable.

Demostremos que $3 - Coloreable \in NP$:

Dado:
G = (V,E)

T: vector que asigna a cada vértice un color

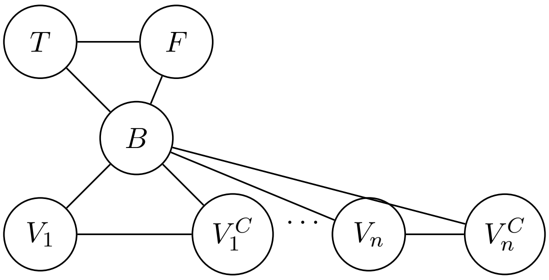
Se puede verificar en tiempo polinomial que la cantidad de colores distintos en T es 3 y recorriendo cada arista del grafo y revisando en T si los vértices de estas tiene colores distintos en T podemos ver si es o no una 3-coloración válida. Como estas operaciones se pueden hacer en tiempo polinomial entonces podemos afirmar que $3 - Coloreable \in NP$

Demostremos ahora que $3 - Coloreable \in NP - Hard$:

Para esta demostración intentaremos reducir 3SAT a un problema de 3-Coloreo en tiempo polinomial
Dada una instancia de 3SAT con k clausulas y n variables crearemos diferentes gadgets que serán subgrafos para colorear:
1 gadget por cada variable
1 gadget por clausula

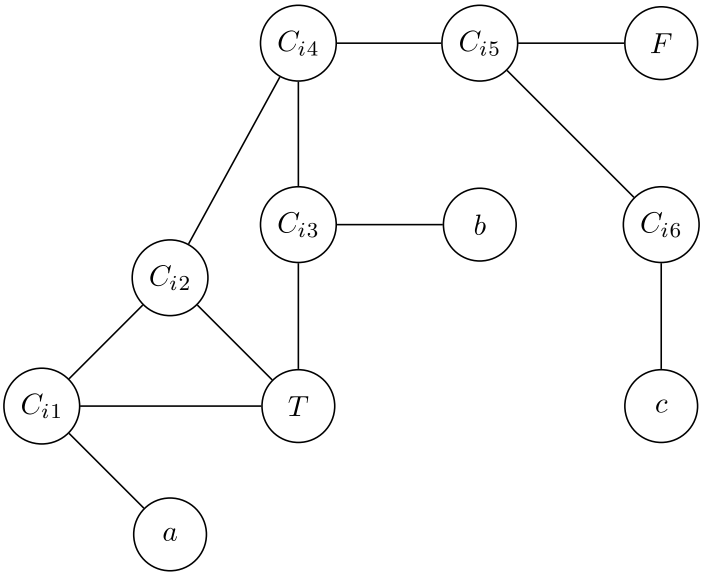
Definimos:
Los nodos V_i y V_i^C, correspondientes a cada variable X_i y su negación.
Los nodos especiales T(True), F(false) y B(base)

Generamos las siguientes aristas:
Entre cada V_i y V_i^C y ambos con B
También vamos a unir entre sí a T,F y B



De aquí podemos ir sacando varias conclusiones como que la variable X_i tendrá el valor True si el nodo V_i tiene el mismo color que el nodo T, además se puede observar que un nodo V_i y V_i^C nunca tendrán el mismo color

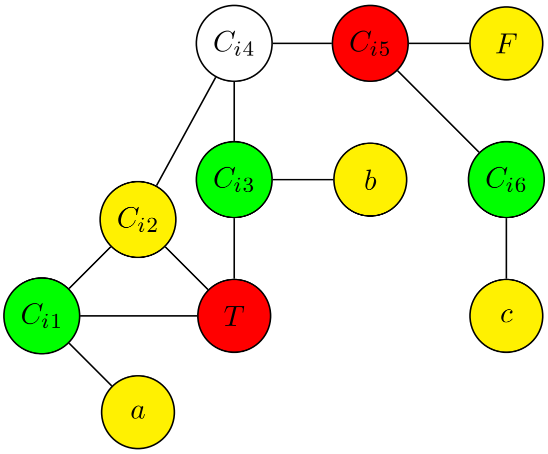
Ahora crearemos un gadget para cada clausula:
Sea la clausula C_i = (a,b,c) con $a, b, c \in X_i$, las variables de la clausula corresponden a los nodos creados para las variables, es decir, si a es la variable X_3 por ejemplo, sería entonces el nodo V_3



Los nodos T y F, son los nodos creados anteriormente Observemos que cada gadget lo que agrega al grafo son los 6 nodos C_{ij}, conectados de la forma que se aprecia en la imagen

Con esto ya tenemos nuestro grafo construido, y vamos a definir K = 3(cantidad de colores).

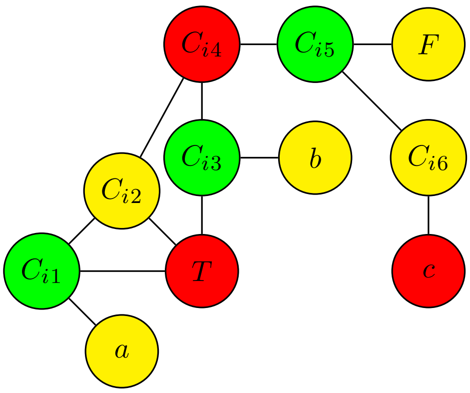
Entonces lo que vamos a decir es que si el grafo se puede pintar con tres colores entonces la expresión dada se puede satisfacer, y los nodos correspondientes a las variables con igual color a T, deben estar en true y si el color es igual a F, entonces esa variable está en false Notemos que por la naturaleza por como está construido cada gadget asociado a cada clausula, si los nodos equivalentes a las varibales que participan en la clausula, están todos del mismo color que el nodo F, pues no será posible colorear ese subgrafo con solo tres colores



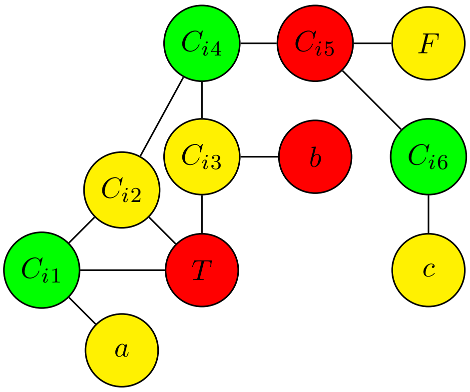
Necesariamente necesitaríamos otro color para colorear el subgrafo,

Ahora veamos que solo basta con que al menos una de las variables de la clausula esté activa para colorear el subgrafo con tres colores. Para ello analicemos cada posible caso.

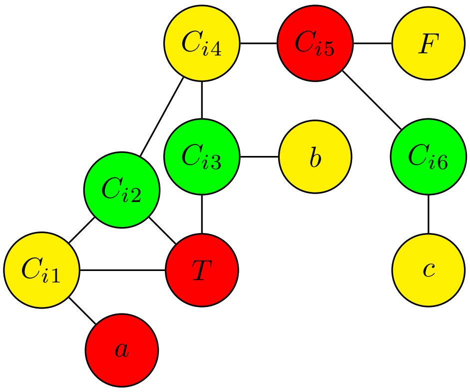
Caso1: 001



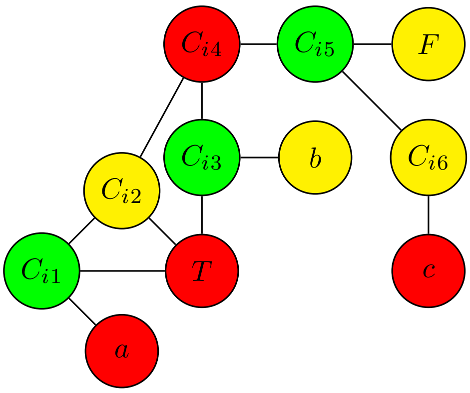
Caso2: 010



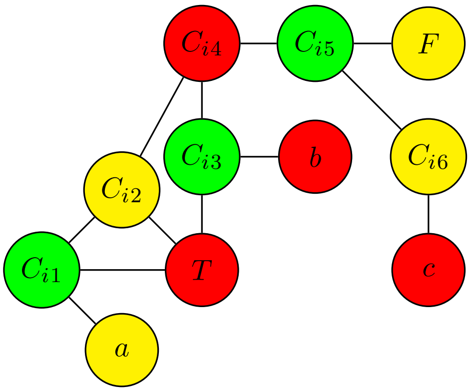
Caso3: 100



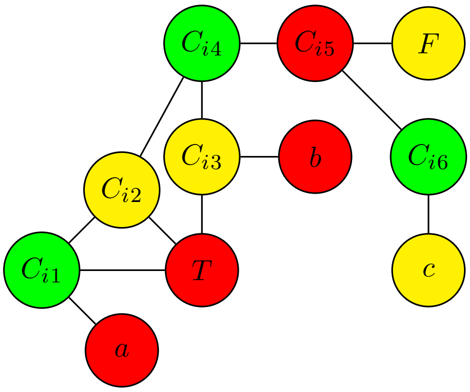
Caso4: 101



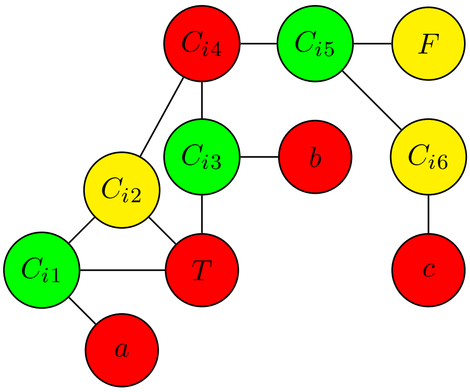
Cas5: 011



Caso6:110



Caso7: 111



Por tanto podemos ver que si el grafo construido es 3-Coloreable entonces la fórmula de 3SAT es satisfacible, y si el grafo no los entonces la fórmula no es satisfacible. Como construir este grafo se puede hacer en tiempo polinomial entonces hemos encontrado una forma de de reudicir 3SAT a 3-Coloreable, y como sabemos que 3SAT es NP-Completo entonces 3-Coloreable es NP-Completo(cabe destacar que hay casos especiales como los grafos bipartitos y los completos en los que la coloración de estos se puede determinar en tiempo polinomial, pero solo en casos como esos, a modo general el problema es NP-Completo).

Cuando $k \geq 3$ se puede reducir 3-Coloreable a K-Coloreable, todo lo que hay que hacer es tomar una instancia de 3-Coloreable y agragarle K-3 nodos, todos conectados entre sí y con todos los nodos de G, el grafo resultante es K-Coloreable si y solo si G es 3-Coloreable. Entonces en general K-Coloreable es NP-Completo Una vez demostrado esto podemos demostrar que el problema de hallar el número cromático es NP-Hard pues podemos reducir K-Coloreable a número cromático, dado que para saber si un grafo es k-coloreable basta hallar su número cromático y si este es menor que k entonces sí lo es y si no es menor entonces no es k-coloreable.

2 Problema 3D Matching

Definición del problema: el problema se basa en encontrar un emparejamiento dentro de un conjunto tridimensional. Supongamos que tienes tres conjuntos disjuntos: X, Y y Z, cada uno de tamaño n. También tienes un conjunto T de ternas de la forma (x,y,z), donde $x \in X$, $y \in Y$ y $z \in Z$. El objetivo es determinar si existe un subconjunto de T de tamaño n, tal que cada elemento de X,Y y Z aparezca exactamente una vez en las ternas seleccionadas.

Solución: Demostremos primero que $3DM \in NP$

Dado:
X,Y,Z conjuntos de n elementos
T = (x,y,z) conjunto de triplas
C, subconjunto de T

Se puede verificar en tiempo polinomial que cardinalidad de C = n y que todo elemento de X,Y y Z se encuentra una y solo una vez en algún C_i.
Entonces podemos afirmar que $3DM \in NP$.

Ahora demostremos que 3DM es NP-Hard. Para ello intentaremos reducir 3SAT a 3DM en tiempo polinomial.
Sea una instancia de 3SAT con:
n variables = { x_1, x_2,...,x_n } y k clausulas = {c_1,...,c_k}

Por cada variable x_i crearemos los siguientes conjuntos:
A_i = {a_{i1}, a_{i2}, ... , a_{i2k} }(2k elementos)
B_i = {b_{i1}, b_{i2}, ... , b_{i2k} }(2k elementos)

T_i = conjunto de triplas de la forma $\{(a_{ij}, a_{ij+1}, b_{ij})\}$, es decir $\{(a_{i1}, a_{i2}, b_{i1}), ...\}$
Aquí diremos que nuestra tripla es par si el subíndice de b correspondiente a j es para e impar si este es impar. Además nótese que para poder seleccionar todos los elementos del conjunto A_i se deben seleccionar todas las triplas cuya b_{ij} sea para el subindice j o todas las que sean impar, no es posible seleccionar una par y alguna impar pues se repetiría algún elemento de A_i.

Por cada clausula C_j crearemos los elementos p_j y p_{j1} y por cada variable i en la clausula C_j crearemos una tripla de la siguiente forma:
Si contiene la variable x_i negada creamos la tripla (p_j, p_{j1}, b_{i2j})
Si contiene la variable x_i creamos la tripla (p_j, p_{j1}, b_{i2j-1})
(en total por cada clausula se crearan 3 triplas)

Si en la solución al problema de 3DM para un índice i se seleccionan las triplas de la forma $(a_{ij}, a_{ij+1}, b_{ij})$ con j par en las b entoces en el problema de 3SAT esa varible estará en true y si se seleccionan las triplas con j impar es porque estará en false, las b_{ij} que no sean seleccionadas en estas triplas pueden ser seleccionadas en las triplas (p_j, p_{j1}, b_{ij}) o en otras que veremos más adelante.
Cada clausula tiene tres triplas asociadas pero solo podemos seleccionar una para activar la clausula o decir que esta se satisface
Notemos que si una clausula l tiene una variable p sin negar y esa variable está en true es porque se seleccionaron todas las triplas de la froma $(a_{pj}, a_{pj+1}, b_{pj})$ con j par en las b, por lo que podría elegirse la tripla (p_l, p_{l1}, b_{p2l-1}) para activar esa clausula En total hay 2*n*k b_{ij}, dado que hay n variables y por cada una se crea 2*k b_{ij}, si hay una solución al problema, notemos que las triplas de las clausulas cubren k b_{ij}, cada clausula tiene 3 b_{ij} pero solo podemos elegir uno. Además las triplas asociadas a cada variable cubren n*k b_{ij}, por lo que faltaría por cubrir (n-1)k b_{ij}, entonces para remediar esto creamos los elementos $\{q_m, q_{m1}\}$ con m desde 1 hasta (n-1)k y agregaremos las triplas $(q_m, q_{m1}$, b) o sea conectaremos estos elementos con todos los b_{ij}.
Para terminar debemos construir los conjuntos disjuntos X,Y,Z
Conjunto X: aquí pondremos todos los a_{ij} con j par(serían nk elementos), también pondremos todos los p_j(k elementos) y los q_m((n-1)k elementos).
Conjunto Y: pondremos todos los a_{ij} con j impar, los p_{j1} y los q_{m1}
Conjunto Z: todos los b_{ij}.

Esto hace que tanto X, como Y y como Z tengan en total 2nk elementos.

El conjunto T serían todas las triplas que hemos definido. De esta forma hemos construido las instancias para el problema 3DM. Entonces si resolvemos este problema tendremos una solución a nuestro problema de 3SAT, bastaría fijarnos en las triplas escogidas, pues como explicamos anteriormente según los b_{ij} podemos saber si la variable está en true o false y la variable que activa cada clausula, pueden haber varias variables que activen la clausula pero basta con saber solo una. Como la construcción de estos conjuntos puede hacerse en tiempo polinomial pues solo es crear y unir elementos hemos encontrado una forma de reducir 3SAT a 3DM en tiempo polinomial y como sabemos que 3SAT es NP-Hard entonces 3DM es también NP-Hard.

Como 3DM es NP y NP-Hard entonces es NP-Completo.

3 Problema del conjunto dominante

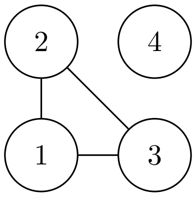
Definición del problema: en un gafo G = (V,E), un conjunto de vértices $D \subseteq V$ es un conjunto dominante si cada vértice de V que no está en D es adyacente a al menos un vértice en D. Una partición de los vértices V en k conjuntos D_1, D_2,...,D_k es una partición domática si cada D_i (para i = 1,2,...,k) es un conjunto dominante. El número dominante es la cardinalidad del menor conjunto dominante de G. Hallar el númeo dominante de G.
Solución: primero vamos a demostrar que el problema de dado un grafo G = (V,E) decir si existe un conjunto dominante de tamaño menor o igual a k es NP-Completo, a este problema le llamaremos K-Dominating.

Demostración de que K-Dominating es NP:
Sea <G, K> la instancia del problema K-Dominating. Sea A la matriz de adyacencia nxn del grafo G de n nodos. Un certificado para K-Dominating consiste en un conjunto C de nodos de G que constituyen un conjunto dominante. Lo podemos representar como un vector de booleanos de largo n, en donde un 1 indica pertenencia al conjunto dominante.
El algorito verificado sería:
Chequear que cardinalidad de C sea menor o igual que K. Esto se hace recorriendo el vector booleano de C en tiempo O(n).Después habría que recorrer todos los nodos del grafo que no están en C y comprobar que cada unotenga al menos un adyacente en C. Esta verificación se puede hacer en tiempo O(n^2) en el peor caso, ya que implica recorrer por filas la matriz de adyacencia A. Se puede ver entonces que K-Dominating es NP.

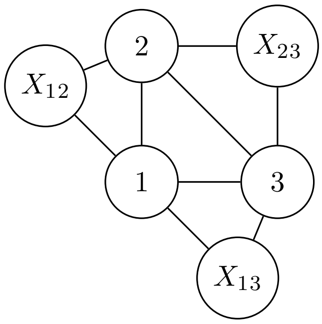
Demostración de que K-Dominating es NP-Hard:
Para esto vamos a intentar reducir el problema de Vertex Cover a K-Dominating en tiempo polinomial.

Sea <G,K> una instancia de Vertex Cover. La reducción propuesta mapea <G,K> en <G_1, K_1> en donde:
G_1 contiene todos los nodos no aislados y todas las aristas de G. Por cada arista (u,v) en G, se crea un nuevo nodo X_{uv} en G_1 y se añaden las aristas (u, X_{uv}) y (v, X_{uv}).
Se toma K_1 = K.

Ver siguiente ejempo: Dado el grafo G:



EL grago G_1 se vería de siguiente forma:



Demostremos que la reducción se puede hace en tiempo polinomial:
Sea <G, K> una instancia de Vertex Cover, donde G tiene n nodos y e aristas.
La instancia correspondiente f(<G, K>) = <G_1, K_1> en K-Dominating se construye recorriendo todas las aristas de G y para cada una de ellas generando la misma arista en G_1, así como el nuevo nodo asociado a la arista y las dos nuevas aristas correspondientes. Si G y G_1 se representan por sus respectivas matrices de adyacencia, el tiempo de construcción de la matriz de G_1 es O(n^2), por lo que la reducción propuesta es polinomial.

Correctitud de la reducción propuesta:
Probaremos que < G, K>pertenece a Vertex Cover si y solo si <G_1, K_1> pertenece a K-Dominating.

H^* en <G, K> es una instancia sí de Vertex Cover.
T^* en <G_1, K_1> es una instancia sí de K-Dominating.
Sea <G, K> que pertenece a Vertex Cover. Por H^*, G tiene un vertex cover H, con cardinalidad de H menor o igual a K.
Sea un nodo cualquiera $v \in G_1$, tal que v no pertenece a H. Hay dos casos:

1- $v \in G$ entonces por definición de Vertex Cover existe una arista entre u y un cierto nodo w de H en G, por definición de f(función que transforma G en G_1) existe una arista entre v y $w \in H$ en G^*.
2- $v \notin G$ entonces por definición de f v es adyacente a dos nodos u y w de G / (u,v) es arista de G, entones por definición de Vertex Cover u o w o ambos perteneces a H, entonces existe arista entre v y un nodo de H en G_1.

De 1 y 2 por definición de K-Dominating H es un K-Dominating de G_1, con cardinalidad de H menor o igual que K.

Ahora lo demostraremos en el otro sentido:
Sea:
H^* en <G_1, K> una instancia sí de K-Dominating.
T^* en <G, K> es una instancia sí de Vertex Cover.

Sea <G_1, K> que pertenece a K-Dominating. Por H^*, G_1 tiene un K-Dominating D con cardinalidad de D menor o igual que k.
Probaremos primero que G_1 tiene un K-Dominating D' con $|D'| \leq$ K y todos los nodos de D' pertenecientes a G.
Sea $x \in D$ y $x \notin G$, por definición de f x es del tipo X_{uv} con (u,v) aristas de G. Distinguimos dos casos:
1- Si $u \in D$ o $v \in D$, considero $D' =$ D - X_{uv}, entonces D' sigue siendo un K-Dominating para G_1, con $|D'| = D$ - 1 $\leq$ K.
2- Si $u \notin D$ y $v \notin D$, considero $D' =$ D - X_{Uv} + u que sigue siendo un K-Dominating para G_1, con $|D'| = |D| \leq$ K.

Dada una arista (u,v) que pertenece a G se cumple que u o v o ambos pertenecen al K-Dominating D' de G_1, pues de lo contrario el nodo X_{uv} de G_1 no sería adyacente a ningún nodo del K-Dominating D' de G_1.
Por definición de Vertex Cover se cumple que D' es un Vertex Cover para G, con $|D'| \leq$ K.

4 Problema Set Cover

Definición del problema: Dado un conjunto X y una colección S de subconjuntos de X, el problema consiste en determinar si existe un subcolector S' subconjunto de S tal que cada elemento de X aparezca exactamente una vez en los subconjuntos de S'.

Demostremos que el problema pertenece a la clase NP: si nos dan una solución(un subcolector S'), podemos verificar en tiempo polinomial si cada elemento de X aparece exactamente una vez en los subconjuntos de S'. Si el conteo es 1 para todos los elementos, la solución es válida.

Demostremos ahora que el problema es NP-Hard: para ello vamos a intentar reducir el problema de 3-Dimensional-Matching(3DM, demostrado anteriormente que es NP-Completo) a nuestro problema en tiempo polinomial.
Para realizar la reducción, construimos una instancia de nuestro a partir de una instancia de 3DM de la siguiente manera:

X' = $X \cup Y \cup Z$, el conjunto X' es la unión de los 3 conjuntos de 3DM.

S = {{x,y,z}| $(x, y, z) \in T\}$: Cada triple en T se convierte en un subconjunto en S.

Ahora debemos demostrar que existe un 3-matching en la instancia de 3DM si y solo si existe un subcolector S' subconjunto de S que cubre X' exactamente una vez.
(=>) Si existe un 3-matching M subconjunto de T que cubre X,Y y Z exactamente una vez, entonces el subcolector S' = {{x,y,z}| $(x, y, z) \in M\}$ cubre X'exactamente una vez. Cada elemento de X', ya sea de X, Y o Z aparecerá exactamente en un subconjunto de S'.

(<=) Si existe un subcolector S' subconjunto que cubre X' exactamente una vez, entonces cada elemento de X, Y y Z debe aparecer exactamente una vez en los subconjuntos de S'. Esto implica que S' define un conjunto de triples M subconjunto de T que forma un 3-Matching, ya que cada triple en S' corresponde a un triple en T, y cada elemento de X, Y y Z está cubierto exactamente una vez.

Esta reducción es polinomial, ya que la construcción de X' y S se puede hacer en tiempo polinomial en el tamaño de la instancia de 3DM. Por lo tanto hemos demostrado que nuestro problema es NP-Hard.
Como es NP y NP-Hard => es NP-Completo.

5 Problema Clique

Definición del problema: Un clique es un subgrafo completo dentro de un grafo. Formalmente, un clique en un grafo G = (V,E) es un subconjunto de vértices $C \subset V$, tal que todos los pares de vértices en C están conectados directamente por una arista. En otras palabras, todos los vértices del clique están mutuamente conectados. Hallar el clique de mayor tamaño en el grafo.

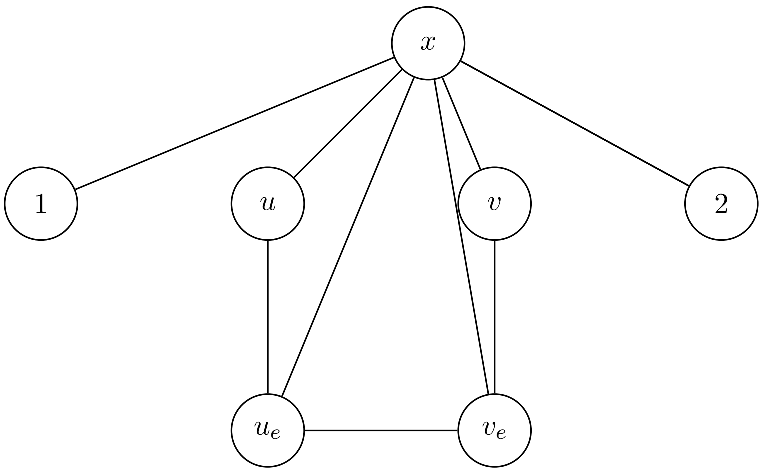
Solución: Demostremos que el problema es NP-Hard.
Ya sabemos que el problema de dado un grafo G y un entero k, saber si existe un clique te tamaño k es NP-Completo(llamemosle a este problema K-clique), entonces intentaremos reducir este problema al nuestro.

Dado una instancia de K-clique con un grafo G y un entero k creamos para nuestro problema un nuevo grafo G' = G, por teoría de grafos sabemos que si existe un clique de tamaño k entonces existe uno de tamaño k-1, k-2 y así hasta 1. Entonces si G tiene clique de tamaño k, en G' hay un clique de tamaño mayor o igual que k y si en G' el clique de mayor tamaño es de longitud k' entonces en G igual. Por lo que si resolvemos nuestro problema bastaría comparar K con la longitud del clique de mayor tamaño, si k es menor o igual entonces existe un clique de tamaño k en G y si k es mayor entonces no.

6 Problema Máximo Corte

Definición del problema: Sea G = (V,E) un grafo con aristas ponderadas. Un corte es una división de los vértices en dos conjuntos T y V-T. El costo de un corte es la suma de los pesos de las aristas que van de T a V-T. El problema trata de encontrar el corte de mayor costo de un grafo.

Solución: Demostremos que el problema es NP-Hard.
Para ello nos apoyaremos en el problema de Conjunto independiente(dado un grafo, saber si en este hay un conjunto independiente de tamaño k) el cual sabemos que es NP-Completo.
La idea es reducir Conjunto independiente a Máximo corte.
Sea G = (V, E) y K una instancia de Conjunto independiente, construiremos un grafo G' = (V', E') de la siguiente forma:
G' contendrá todos los nodos de V y un nodo especial x, el cual estará conectado a todos los nodos de V. Por cada arista e = $(u, v) \in E$, agregaremos dos vértices: u_e, v_e a V' y cinco aristas {u_e, v_e}, {u, u_e}, {v, v_e}, {x, u_e}, {x, v_e} a E', estos dos vértices y cinco aristas diremos que forman el gadget G_e correspondiente a la arista e. Ejemplo:



Todas las aristas del grafo G' tendrán peso uno, de forma que encontrar el corte de mayor peso equivale a encontrar el corte con más aristas que lo atraviesan. Denotaremos el costo de un corte S como c(S)
Primero pobaremos que si G contiene un conjunto independiente $I \subseteq V$, tal que $|I| \geq k$, entonces existe un corte S en G' tal que $c(S) \geq k + 4 * |E|$. Dado un conjunto I, construiremos S de la siguiente forma: empezaremos con S = I, entonces por cada arista $e = \{u, v\} \in E$, haremos lo siguiente:
Si $u \in I$ y $v \notin I$, entonces agregamos v_e a S. De forma similar si $u \notin S$ y $v \in S$, entonces agregamos u_e a S. Notemos que en cualquier caso se cortan 4 de las 5 aristas del gadget G_e.
Si $u, v \notin I$, entonces agregamos u_e y v_e a S. De nuevo 4 de las 5 aristas de G_e se cortan.

El caso donde $u, v \in I$ no es posible porque I es un conjunto independiente y {u, v} es una arista.
Por cada $w \in I$, la arista {x, w} se corta por S. Además por cada arista e = $\{u, v\} \in E$, independientemente de la posición de u y v con respecto a I, exactamente 4 aristas del gadget G_e se cortan por S. Entonces el número de aristas que atraviesan el corte S es k + $4 * |E|$.

Ahora probaremos el sentido contrario:
Dado un corte de mayor costo S en G' tal que $c(S) \geq k + 4 * |E|$, construiremos un conjunto independiente $I \subseteq V$ tal que $|I| \geq k$. Para ello tomaremos uno de los dos conjuntos del corte, sin perder generalidad diremos que el nodo x no está en este subconjuto, de lo contrario se tomaría el otro subconjunto, llamemosle S_1 a este subconjunto tomado. Sea I el subconjunto de S_1 correspondiente a los nodos de G, supongamos que I no es un conjunto independiente. Consideremos entonces cualquier arista $e = \{u, v\} \in E$, si $u, v \in S$ entonces podemos ver que S corta al menos tres aristas del gadget G_e, ya que al menos uno de los nodos u_e o v_e debe estar en el otro conjunto, de lo contrario el corte no sería máximo. En cualquier otro caso(al menos u o v no está en S_1) ya vimos una forma de picar en la que siempre se cortan 4 aristas del gadget. Entonces siendo m(I) el número de aristas entre los nodos del conjunto I, tenemos que:

$$c(S) \leq |I| + 3 * m(I) + 4 * (|E| - m(I)) = |I| + 4 * |E| - m(I)$$

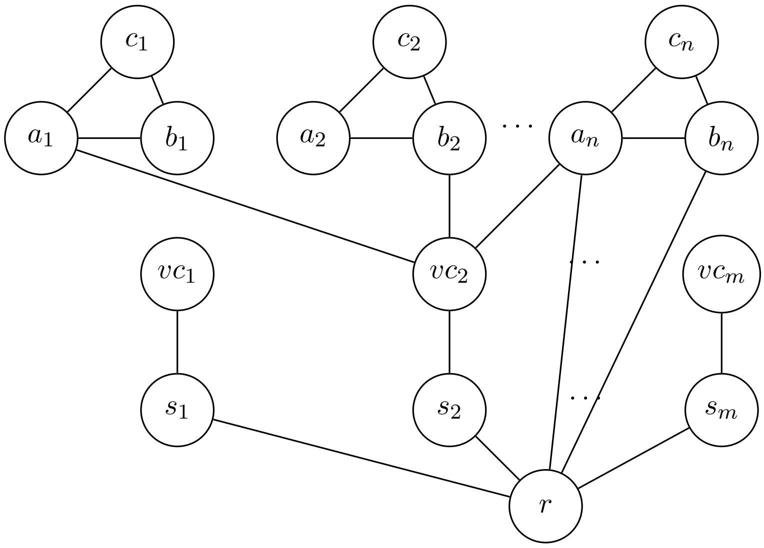
Como $c(S) \geq k + 4 * |E|$, implica que $|I| \geq k + m(I)$. Entonces por cada arista entre I, podemos remover uno de los nodos para decrementar el valor de m(I) en uno. Después de hacer esto al menos m(I) veces , tendremos un conjunto independiente de al menos k.

7 Problema Número Domatic

Definición del problema: el número de domatic de un grafo G, denotado como domatic(G), es el número máximo k tal que los vértices de G pueden dividirse en k conjuntos disjuntos D_1, D_2, ..., D_k donde cada D_i es dominante. Hallar el número de Domatic de un grafo.

Solución: Demostraremos que este problema es NP-Hard, para ello reduciremos en tiempo polinomial el problema de 3-SAT(que sabemos que es NP) a Número domatic.
Dada una instancia I de 3-SAT con m cláusulas C_1, C_2, ..., C_m y n variables x_1, x_2, ..., x_n, crearemos un grafo G = (V, E) de la siguiente forma:
Por cada cláusula C_i crearemos un vértice VC_i y un vértice S_i, ambos conectados.

Por cada variable x_j crearemos tres vértices: a_j, b_j y c_j, conectados en forma de triángulo, además se creará un vértice r conectado a todos los s_i y a todos los a_j, b_j.
El vértice a_j corresponde al valor positivo de la variable x_j, el vértice b_j al valor negativo. También hay una arista entre el vértice VC_i y el vértice a_j si x_j aparece en la cláusula C_i, de forma similar con b_j si aparece negada. Veamos un esbozo del grafo construido:



Podemos ver que como los nodos c_i tienen degree 2 el máximo número domatic que puede tener este tipo de grafos es 3. (=>) Supongamos que I tiene una asignación para la cual se cumple. Sin perder generalidad diremos que las variables x_1, ..., x_k son true y las variables x_{k+1}, ..., x_n son false. Veamos que el grafo tiene como número domatic 3. Los vértices de V se pueden particionar en tres conjuntos:
$V_1 = \{a_1, ..., a_k, b_{k+1}, ..., b_n, \text{r}\}$
$V_2 = \{b_1, ..., b_k, a_{k+1}, ..., a_n, vc_1, ..., vc_m\}$
$V_3 = \{c_1, ...,c_n, s_1, ..., s_m\}$
Se puede ver cada uno de los tres conjuntos son dominantes.

(<=)Ahora supongamos que el grafo G = (V,E) tiene como número domatic 3. Sea entonces V_1, V_2, $V_3 \subset V$, una particón de V tal que cada V_i es dominante. Sin perder generalidad, digamos que el vértice $r \in V_1$. Entonces hay solo dos posibilidades para $i \in \{1, ..., n\}$:
(1) $vc_i \in V_2$ y $s_i \in V_3$

(2) $vc_i \in V_3$ y $s_i \in V_2$

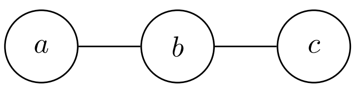
En otro caso s_i no puede ser dominado por uno de los conjuntos V_2 o V_3. Sin perder generalidad consideremos un vértice vc_i en particular y digamos que pertenece a V_2. Entonces $s_i \in V_3$. ahora notemos que vc_i debe ser dominado por algún vértice en V_1, por lo que al menos uno de los vértices a_j/b_j que conecta a vc_i debe estar en V_1. Entonces el conjunto $V_1 \cap \{a_1, ..., a_n, b_1, ..., b_n\}$ es una forma válida de satisfacer I, notemos que para cualquier j a_j y b_j no pueden estar en V_1 de lo contrario c_j no puede ser dominado por V_2 y V_3 a la misma vez. Con esto completamos la demostración.

8 Problema Retroalimentación de Vértices

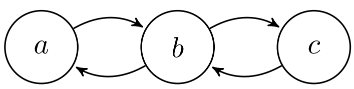
Definición del problema: Dado un grafo G = (V, E), un conjunto de retroalimentación de vértices es un subconjunto de vértices $F \subseteq V$ tal que al eliminar todos los vértices de F(y sus aristas incidentes), el grafo resultante no contiene ciclos(es un grafo acíclico o un bosque, si es no dirigido). El objetivo del problema es encontrar el conjunto de retroalimentación de vértices de tamaño mínimo.

Solución: Vamos a demostrar que el problema es Np-Hard, para ello vamos a demostrar primero que el problema de retroalimentación de vértices de tamaño k es NP-Completo:

Demostración de que es NP: Dada una instancia de retroalimentación de vértices y un conjunto solución D, bastaría verificar que el tamaño de D es menor o igual que k y que al eliminar esos vértices del grafo este es acíclico, lo cual se puede hacer haciendo un recorrido por el grafo.çç Demostración de que es NP-Hard: para esta parte intentaremos reducir el problema de Vertex Cover(el cual sabemos que es NP-Completo) a Retroalimentación de Vértices. La reducción se creará de la siguiente forma: dada una instancia de Vertex Cover con G = (V,E) crearemos un grafo dirigido G' = (V', E') con V' = V y por cada arista $(a, b) \in E$ crearemos las aristas (a, b) y (b, a) en E', generando un ciclo en G'. Ejemplo dado la siguiente instancia de Vertex Cover:



Crearemos el siguiente grafo:



Relación entre ambos problemas
Para Vertex Cover necesitamos un conjunto mínimo tal que cada arista es cubierta, para

retroalimentación de vértices necesitamos un conjunto de vértices mínimo tal que si lo eliminamos del grafo, resulta en un grafo sin ciclos.
Ahora dado que cada arista en G de Vertex Cover se transforma en dos aristas dirigidas en G' formando un ciclo, entonces el conjunto seleccionado para satisfacer retroalimentación de vértices, dígase F, debe incluir al menos uno de los dos vértices para romper ese ciclo. Si tenemos entonces F en G , entonces F debe cubrir cada arista en G, eso significa que para cada ciclo dirigido correspondiente en G', F debe contener al menos uno de los vértices de ese ciclo para romperlo. Al eliminar F en G' todos los ciclos dirigidos se eliminan lo que hace que G' sea acíclico. Por tanto si en Retroalimentación de Vértices hay una solución de tamaño k, entonces también la hay en Vertex Cover.

Habiendo demostrado que este problema es NP-Completo podemos reducir el mismo al de hallar el mínimo conjunto que satisfaga retroalimentación de vértices, pues si el mínimo es menor que k entonces hay una solución de longitud k y si no lo es pues no la hay. Por tanto el problema de hallar el mínimo conjunto que satisfaga Retroalimentación de vértices es NP-Hard.

9 Problema Retroalimentación de Arcos

Definición del problema: Dado un grafo G = (V, E), un conjunto de retroalimentación de arcos es un subconjunto de arcos $F \subseteq E$ tal que al eliminar todos los arcos en F, el grafo resultante no contiene ciclos. El objetivo del problema es encontrar el conjunto de retroalimentación de arcos te tamaño mínimo.

Solución: Al igual que en el problema anterior, demostraremos primero que el problema de encontrar un conjunto de arcos de longitud k tal que al quitarlos no queden ciclos en G es NP-Completo. Para ello intentaremos reducir el problema de Retroalimentación de Vértices a este en tiempo polinomial.
Dada una instancia (G, k) de Retroalimentación de Vértices con G = (V, E) crearemos una instancia (G', k') de Retroalimentación de arcos de la siguiente forma:
Por cada nodo $v \in V$ creamos dos nodos v_1 y v_2 en G' y el arco (v_1, v_2), a estos arcos le llamaremos arcos internos. Luego por cada arco $(u, v) \in E$ crearemos en E' el arco (u_2, v_1), estos serán los arcos externos.

Cada vértice en G se duplica en G', creando dos copias de vértices y las aristas del grafo original se dividen en aristas internas(entre las copias de un mismo vértice) y aristas externas(entre copias de vértices diferentes).

Al resolver RA(Retroalimentación de arcos) en G' buscamos un conjunto de aristas que rompa todos los ciclos en G'. La clave está en que todos los ciclos que involucran aristas externas también pasan por aristas internas, por lo que eliminar solo aristas internas(que corresponden a vértices en G) es suficiente para romper los ciclos.
Por lo tanto, cualquier conjunto de aristas internas que forme una solución para RA en G' corresponde a un conjunto vértices que forman una solución para RV(Retroalimentación

de Vértices) en G. De esta forma resolver RA en G' es equivalente a resolver RV en G, ya que los ciclos que rompes en G' equivalen a los ciclos que rompes en G.
Y análogamente como en el problema anterior demostramos que hallar el mínimo conjunto es NP-Hard pues en este problema también sirve para demostrar que hallar el mínimo conjunto de aristas es NP-Hard.