Demostraciones Formales de Problemas NP-Hard y NP-Completos

Miguel Alejandro Yáñez Martínes

10-12-2024

Introducción

Este documento contiene demostraciones detalladas de que varios problemas clásicos de la teoría de algoritmos son NP-Hard o NP-Completo. Todas las reducciones asumieron como base los siguientes problemas que ya han sido demostrados como NP-Completos:

- SAT
- 3-SAT
- Clique
- Conjunto Independiente
- Vertex Cover
- Subset Sum
- Mochila
- Hamilton (Dirigido)
- Viajante (Traveling Salesperson Problem)

Demostración de que el problema Exact Cover es NP-Completo

Definición del problema El problema **Exact Cover** se define de la siguiente manera: Dado un conjunto X y una colección $\mathcal{S}$ de subconjuntos de X, el problema consiste en determinar si existe un subcolector $\mathcal{S}' \subseteq \mathcal{S}$ tal que cada elemento de X aparezca exactamente una vez en los subconjuntos de $\mathcal{S}'$.

Paso 1: Exact Cover pertenece a NP

Para demostrar que Exact Cover está en NP, mostramos que, dada una solución candidata, podemos verificar en tiempo polinomial si esta solución es válida.Para verificar si una solución es válida, primero calculamos la unión de los subconjuntos en $\mathcal{C}$ y comprobamos si $\bigcup_{S \in \mathcal{C}} S = U$. Este procedimiento requiere recorrer los elementos de U y verificar si cada uno está cubierto por algún subconjunto en $\mathcal{C}$ y tambíen hay que comprobar que no se encuentre en má de uno, lo cual puede realizarse en tiempo $O(nm)$, donde n es el tamaño de U y m es el número de subconjuntos en $\mathcal{S}$. Dado que esta verificación puede realizarse en tiempo polinomial, concluimos que Exact Cover pertenece a NP.

Paso 2: Exact Cover es NP-Completo

Para demostrar que Exact Cover es NP-Completo, reducimos el problema **Vertex Cover**, conocido por ser NP-Completo, a Exact Cover en tiempo polinomial. Recordemos que en Vertex Cover se nos da un grafo no dirigido $G = (V, E)$, y el objetivo es determinar si existe un conjunto de vértices $V' \subseteq V$ tal que toda arista $e \in E$ es incidente a al menos un vértice de V'.

Dada una instancia de Vertex Cover, construimos una instancia equivalente de Exact Cover. Definimos el conjunto base de Exact Cover como $U = E$, es decir, el conjunto de todas las aristas del grafo. Para cada vértice $v \in V$, definimos un subconjunto $S_v \subseteq U$, donde S_v contiene todas las aristas de E que son incidentes a v. La colección de subconjuntos es entonces $\mathcal{S} = \{S_v : v \in V\}$. Por lo tanto, seleccionar un vértice en Vertex Cover equivale a seleccionar el subconjunto correspondiente en Exact Cover.

La equivalencia entre las soluciones se establece de la siguiente manera. Si existe un Vertex Cover, entonces los $k \geq 0$ vértices seleccionados corresponden a k subconjuntos de $\mathcal{S}$ cuya unión cubre U. Inversamente, si en Exact Cover podemos seleccionar k subconjuntos cuya unión sea U, entonces los vértices correspondientes forman un Vertex Cover en G.

La construcción de esta instancia de Exact Cover a partir de una instancia de Vertex Cover puede realizarse en tiempo polinomial, ya que simplemente recorremos los vértices y las aristas del grafo para construir U y $\mathcal{S}$. Por lo tanto, hemos reducido Vertex Cover a Exact Cover en tiempo polinomial, demostrando que Exact Cover es al menos tan difícil como Vertex Cover.

Demostración de que el problema *Max-Clique* es NP-completo

Definición del problema El problema consiste en hallar el *clique* de mayor tamaño en un grafo dado G, es decir, encontrar un subconjunto $C_{\max} \subseteq V$ tal que $|C_{\max}|$ sea máximo y $C_{\max}$ forme un subgrafo completo..

Paso 1: Max-Clique $\in$ NP Para verificar una solución al problema, basta con recibir un subconjunto V' y que todos los pares de vértices en V' están conectados por aristas en G. La comprobación puede realizarse en tiempo $O(|V'|^2)$, verificando las aristas incidentes para cada par de vértices en V'. Esto asegura que el problema *Max-Clique* pertenece a la clase NP, ya que existe un algoritmo que puede verificar en tiempo polinomial si una solución propuesta es válida.

Paso 2: Reducción polinomial desde 3-SAT Para demostrar que *Max-Clique* es NP-completo, realizamos una reducción polinomial desde el problema 3-SAT. Recordemos que en 3-SAT se nos da una fórmula booleana ϕ en forma normal conjuntiva (CNF), con m cláusulas donde cada cláusula contiene exactamente 3 literales. El objetivo es determinar si existe una asignación de valores true o false para las variables de ϕ que haga que la fórmula sea verdadera.

Para construir la reducción, transformamos una instancia de 3-SAT en un grafo $G = (V, E)$. Para cada literal presente en cada cláusula C_i, creamos un vértice en V. Así, el grafo contiene $3m$ vértices en total, uno por cada literal de cada cláusula. Luego, añadimos una arista entre dos vértices si y solo si los literales correspondientes pertenecen a cláusulas diferentes ($C_i \neq C_j$) y no son mutuamente excluyentes (es decir, no pueden ser ambos verdaderos al mismo tiempo, como ocurre con x y $\neg x$).

En este grafo, cualquier clique de tamaño m corresponde a una asignación que satisface la fórmula ϕ. Esto es porque un clique de tamaño m selecciona exactamente un literal de cada cláusula (ya que los vértices correspondientes a los literales de una misma cláusula no forman un clique). Además, las aristas entre los vértices seleccionados garantizan que no hay conflictos entre los literales seleccionados, asegurando así que estos representan una asignación válida que satisface todas las cláusulas de ϕ.

El grafo G puede construirse en tiempo polinomial con respecto al tamaño de la fórmula ϕ. Contiene $O(m)$ vértices y $O(m^2)$ posibles pares de aristas. Tanto la construcción de los vértices como la determinación de las aristas se puede realizar eficientemente.

Demostración de que el problema *Número Cromático* es NP-Hard

Definición del problema El problema del *Número Cromático* consiste en determinar, dado un grafo $G = (V, E)$, si es posible colorear los vértices de G utilizando como máximo k colores siendo este su número cromático, de manera que no haya dos vértices adyacentes que compartan el mismo color. Formalmente, se trata de decidir si existe una función de coloreado $c : V \to \{1, 2, \ldots, k\}$ tal que $c(u) \neq c(v)$ para toda arista $(u, v) \in E$.

Reducción polinomial desde *Conjunto Independiente* Para demostrar que *Número Cromático* es NP-Hard, realizamos una reducción polinomial desde el problema *Conjunto Independiente*. Recordemos que en *Conjunto Independiente*, se nos da un grafo $G = (V, E)$ y debemos determinar si existe un subconjunto $V' \subseteq V$ tal que ningún par de vértices en V' está conectado por una arista en G.

Dada una instancia del problema *Conjunto Independiente*, construimos una instancia equivalente del problema *Número Cromático*. Sea $G = (V, E)$ un grafo dado,construimos un grafo complementario $\overline{G}$, donde $\overline{G}$ tiene los mismos vértices que G, pero sus aristas se definen de manera opuesta: hay una arista (u, v) en $\overline{G}$ si y solo si $(u, v) \notin E$ en G.

La relación clave entre estos problemas es que encontrar un conjunto independiente G es equivalente a colorear $\overline{G}$ con al menos $|V| - k$ colores. Esto se debe a que un conjunto independiente en G corresponde a un clique en $\overline{G}$, y un conjunto de vértices en el mismo color en $\overline{G}$ no puede contener cliques.

Por lo tanto, para resolver el problema de *Número Cromático*, verificamos si el número cromático $\chi(\overline{G}) \leq |V| - k$. Si esto es cierto, entonces la instancia original de *Conjunto Independiente* en G tiene una solución con número cromático k.

La construcción del complemento de un grafo puede realizarse en tiempo polinomial, ya que implica revisar cada par de vértices u, v en G y determinar si están conectados o no. Esta transformación es eficiente y asegura una equivalencia entre las soluciones de *Conjunto Independiente* en G y *Número Cromático* en $\overline{G}$.

Demostración de que el problema *Conjunto Dominante* es NP-Hard

Definición del problema En un grafo $G = (V, E)$, 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 $D_1, D_2, \ldots, D_k$ es una *partición domática* si cada D_i (para $i = 1, 2, \ldots, k$) es un conjunto dominante. El *número dominante* es la cardinalidad del menor conjunto dominante de G.

El problema consiste en hallar el *número dominante* de G.

Reducción polinomial desde *Cobertura de Vértices* Para demostrar que *Conjunto Dominante* es NP-Hard, realizamos una reducción polinomial desde el problema *Cobertura de Vértices*.

Dada una instancia del problema *Cobertura de Vértices*, transformamos el grafo $G = (V, E)$ en un nuevo grafo $G' = (V', E')$ para resolver el problema de *Conjunto Dominante*. La transformación se realiza de la siguiente manera: Para cada vértice $v \in V$ en G, creamos un vértice $v \in V'$ en G'. Para cada arista $(u, v) \in E$ en G, creamos un nuevo vértice $w_{uv} \in V'$ y añadimos las aristas (w_{uv}, u) y (w_{uv}, v) en G'.

En G', los vértices originales V dominan a los nuevos vértices w_{uv}. Por lo tanto, resolver el problema de *Cobertura de Vértices* en G con un subconjunto C es equivalente a encontrar un conjunto dominante D en G' que incluya C. Esto se debe a que los vértices de C dominan todas las aristas en G, lo que significa que también dominan todos los vértices w_{uv} en G'.

La construcción del grafo G' se realiza en tiempo polinomial con respecto al tamaño del grafo original G, ya que para cada arista en G se añade un nuevo vértice y dos nuevas aristas en G'. Además, una solución al problema de *Cobertura de Vértices* en G se traduce directamente en una solución al problema de *Conjunto Dominante* en G' con el mismo número de vértices.

Demostración de que el problema *Número Domatic* es NP-Hard

Definición del problema El *número domático* 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 $D_1, D_2, \ldots, D_k$, donde cada D_i es un conjunto dominante.

El problema consiste en hallar el *número domático* de un grafo G.

Reducción polinomial desde *3-SAT* Para demostrar que el problema del *Número Domatic* es NP-Hard, realizamos una reducción polinomial desde el problema *3-SAT*.

Dada una instancia del problema *3-SAT*, construimos un grafo $G = (V, E)$ como sigue: Para cada cláusula C_i, añadimos un triángulo completo (un subgrafo K_3) con 3 vértices correspondientes a los literales en C_i. Denotamos estos vértices como x_{i1}, x_{i2}, x_{i3}, representando los literales de C_i. Para cada par de literales ℓ y $\neg\ell$ (un literal y su negación) en cláusulas distintas, añadimos una arista entre sus vértices correspondientes, asegurando que ambos no puedan pertenecer a conjuntos dominantes disjuntos.

El objetivo es particionar los vértices del grafo G en m conjuntos dominantes disjuntos. Intuitivamente, cada conjunto dominante debe seleccionar exactamente un literal de cada cláusula C_i, asegurando que una asignación válida de ϕ satisface todas las cláusulas.

El grafo G tiene la propiedad de que cualquier partición en m conjuntos dominantes corresponde a una asignación que satisface ϕ. Esto se debe a que: Cada conjunto dominante debe incluir exactamente un vértice del triángulo asociado a cada cláusula C_i, lo que corresponde a la elección de un literal verdadero en esa cláusula. Las aristas entre los literales opuestos aseguran que no se seleccionen ℓ y $\neg\ell$ en conjuntos dominantes diferentes, preservando la coherencia de la asignación.

La construcción del grafo G requiere $O(m)$ triángulos (uno por cláusula) y $O(n^2)$ aristas adicionales para garantizar la consistencia entre los literales. Esto puede realizarse en tiempo polinomial respecto al tamaño de la fórmula ϕ.

Demostración de que el problema *Ancho de Banda* es NP-Hard

Definición del problema Dado un grafo $G = (V, E)$ y una disposición lineal de sus vértices representada como una función $f : V \to \{1, 2, \ldots, |V|\}$, el *ancho de banda* de G para esa disposición es:

$$\max\{|f(u) - f(v)| : (u, v) \in E\}.$$

El problema consiste en encontrar una disposición f que minimice este valor, es decir, reducir al mínimo la distancia máxima en la disposición lineal entre los extremos de las aristas del grafo.

Reducción polinomial desde el problema *Hamiltoniano Dirigido* Para demostrar que el problema del *Ancho de Banda* es NP-Hard, realizamos una reducción polinomial desde el problema *Hamiltoniano Dirigido* (Directed Hamiltonian Path). Recordemos que en *Hamiltoniano Dirigido*, se nos da un grafo dirigido $G = (V, E)$, y debemos determinar si existe un camino dirigido que pase exactamente una vez por cada vértice de G.

Dada una instancia del problema *Hamiltoniano Dirigido*, transformamos $G = (V, E)$ en una instancia del problema del *Ancho de Banda* como sigue: Construimos un grafo $G' = (V', E')$ que tiene los mismos vértices $V = V'$ que G. Para cada arista dirigida $(u, v) \in E$, añadimos una arista no dirigida $\{u, v\}$ en G'. Definimos el ancho de banda objetivo $k = 1$.

La idea principal es que un camino hamiltoniano dirigido en G corresponde exactamente a una disposición de los vértices de G' tal que el ancho de banda sea $k = 1$. Esto ocurre porque: En un camino hamiltoniano, cada vértice u está directamente conectado con su sucesor v en el camino. Por lo tanto, en la disposición lineal de G', los vértices adyacentes estarán colocados consecutivamente, cumpliendo $|\pi(u) - \pi(v)| = 1$.

Si G tiene un camino hamiltoniano dirigido, entonces G' tiene una disposición lineal con ancho de banda $k = 1$. Por el contrario, si G' tiene una disposición con ancho de banda $k = 1$, esta disposición induce un camino hamiltoniano dirigido en G.

La transformación de G en G' se realiza en tiempo polinomial, ya que se mantienen los mismos vértices y se transforman las aristas dirigidas en aristas no dirigidas. Además, el ancho de banda objetivo $k = 1$ asegura que solo las disposiciones lineales que respeten el orden de un camino hamiltoniano son válidas.

Demostración de que el problema *Retroalimentación de Vértices* es NP-completo

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 en 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.

Paso 1: *Retroalimentación de Vértices* pertenece a NP Dado un grafo $G = (V, E)$, una certificación válida para este problema sería un subconjunto de vértices $S \subseteq V$. Para verificar esta certificación, basta con eliminar los vértices de S del grafo G y comprobar que el grafo resultante no contiene ciclos. Esto puede realizarse usando un algoritmo de detección de ciclos en tiempo $O(|V|+|E|)$, lo que garantiza que la verificación se realiza en tiempo polinomial respecto al tamaño de G. Por lo tanto, el problema de *Retroalimentación de Vértices* pertenece a la clase NP.

Paso 2: Reducción polinomial desde *Cobertura de Vértices* Para demostrar que el problema de *Retroalimentación de Vértices* es NP-completo, realizamos una reducción polinomial desde el problema de *Cobertura de Vértices* (Vertex Cover), que es NP-completo.

Dada una instancia del problema *Cobertura de Vértices*, transformamos $G = (V, E)$ en un nuevo grafo $G' = (V', E')$ para resolver el problema de *Retroalimentación de Vértices*: 1. Construimos el mismo conjunto de vértices $V' = V$ y las mismas aristas $E' = E$. 2. Añadimos un ciclo adicional en G' que conecta todos los vértices V en un ciclo de longitud $|V|$, es decir, añadimos las aristas (v_i, v_{i+1}) y $(v_{|V|}, v_1)$ para $v_1, v_2, \ldots, v_{|V|} \in V$.

En este nuevo grafo G', la tarea de encontrar un conjunto de retroalimentación de tamaño k corresponde exactamente a encontrar una cobertura de vértices en G. Esto se debe a que: En G', las nuevas aristas añadidas forman un ciclo de $|V|$ vértices. Para romper este ciclo y garantizar que G' sea un bosque, se deben eliminar al menos $|V| - 1$ vértices, lo que obliga a cubrir todas las aristas del ciclo y las aristas originales de G.

Por lo tanto, resolver *Cobertura de Vértices* en G es equivalente a resolver *Retroalimentación de Vértices* en G'.

La construcción del grafo G' puede realizarse en tiempo polinomial, ya que implica añadir $|V|$ aristas adicionales al grafo G. Además, cualquier solución para *Retroalimentación de Vértices* en G' se traduce directamente en una solución para *Cobertura de Vértices* en G, y viceversa.

Demostración de que el problema *Retroalimentación de Arcos* es NP-completo

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 (es un grafo acíclico o un bosque, si es no dirigido).

El objetivo del problema es encontrar el *conjunto de retroalimentación de arcos* de tamaño mínimo.

Paso 1: *Retroalimentación de Arcos* pertenece a NP Dado un grafo dirigido $G = (V, E)$, una certificación válida para este problema sería un subconjunto de arcos $F \subseteq E$. Para verificar esta certificación, basta con eliminar los arcos de F del grafo G y comprobar que el grafo resultante no contiene ciclos. Esto puede realizarse usando un algoritmo de detección de ciclos en tiempo $O(|V|+|E|)$, lo que garantiza que la verificación se realiza en tiempo polinomial respecto al tamaño de G. Por lo tanto, el problema de *Retroalimentación de Arcos* pertenece a la clase NP.

Paso 2: Reducción polinomial desde *Hamiltoniano Dirigido* Para demostrar que el problema de *Retroalimentación de Arcos* es NP-completo, realizamos una reducción polinomial desde el problema *Hamiltoniano Dirigido*.

Dada una instancia del problema *Hamiltoniano Dirigido*, transformamos G en una instancia del problema *Retroalimentación de Arcos* de la siguiente manera: Sea $G = (V, E)$ el grafo original dado. Definimos el mismo grafo $G' = (V, E)$ con los mismos vértices y arcos. Definimos el entero $k = |E| - |V|$.

La idea principal es que un ciclo hamiltoniano en G es equivalente a un grafo en el que, tras eliminar k arcos, el grafo resultante se vuelve acíclico. Esto se basa en la propiedad de que: Si G contiene un ciclo hamiltoniano, este ciclo pasa por todos los vértices exactamente una vez. Los $|V|$ arcos que forman el ciclo deben eliminarse para romper todos los ciclos en G. Por el contrario, si es posible eliminar $|E|-|V|$ arcos y dejar el grafo acíclico, los arcos restantes forman un árbol dirigido que no puede contener ciclos hamiltonianos adicionales.

Por lo tanto, determinar si existe un ciclo hamiltoniano en G es equivalente a determinar si se puede encontrar un conjunto de retroalimentación de arcos F con $|F| = k$.

La construcción del grafo G' no requiere modificaciones, ya que se utiliza el mismo grafo original G. El valor de k se calcula directamente a partir del número de vértices y arcos del grafo. La reducción asegura que resolver el problema *Retroalimentación de Arcos* en G' responde directamente si G tiene un ciclo hamiltoniano.

La construcción del grafo G' y el cálculo del valor k se realizan en tiempo polinomial respecto al tamaño del grafo G. Por lo tanto, esta reducción es computacionalmente eficiente.

Demostración de que el problema *3D Matching* es NP-completo

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 (es decir, n ternas) tal que cada elemento de X, Y, y Z aparezca exactamente una vez en las ternas seleccionadas.

Paso 1: *3D Matching* pertenece a NP Dado un conjunto X, Y, Z, un conjunto de tríos T, y un subconjunto candidato $M \subseteq T$, podemos verificar en tiempo polinomial si M es una solución válida. Esto se hace comprobando que: M contiene exactamente $|n|$ elementos. Cada elemento de X, Y, Z aparece en exactamente un trío de M. Estas verificaciones pueden realizarse recorriendo los conjuntos X, Y, Z y ver si ya apareció, lo cual requiere un tiempo $O(|M| + |X| + |Y| + |Z|)$. Por lo tanto, el problema de *3D Matching* pertenece a la clase NP.

Paso 2: Reducción polinomial desde *3-SAT* Para demostrar que el problema de *3D Matching* es NP-completo, realizamos una reducción polinomial desde el problema *3-SAT*.

Dada una instancia de *3-SAT*, construimos una instancia del problema *3D Matching* como sigue:

Para cada cláusula C_i, definimos un conjunto $Z = \{z_1, z_2, \ldots, z_m\}$, donde cada z_i corresponde a una cláusula. Para cada variable x_j en ϕ, definimos un conjunto $X = \{x_1, x_2, \ldots, x_n\}$, donde n es el número de variables. Para las asignaciones de los valores de verdad a los literales, definimos un conjunto Y que incluye elementos correspondientes a las literales positivas y negativas $(y_{x_1}, y_{\neg x_1}, y_{x_2}, y_{\neg x_2}, \ldots)$.

Luego, construimos el conjunto de tríos $T \subseteq X \times Y \times Z$ como sigue: Para cada cláusula C_i que contiene los literales ℓ_1, ℓ_2, ℓ_3, añadimos los tríos:

$$(x_j, y_{\ell_1}, z_i), \quad (x_k, y_{\ell_2}, z_i), \quad (x_l, y_{\ell_3}, z_i),$$

donde x_j, x_k, x_l corresponden a las variables de los literales ℓ_1, ℓ_2, ℓ_3.

La idea es que una solución válida para el problema *3D Matching* seleccionará un subconjunto de T donde cada cláusula esté cubierta por exactamente un literal, y cada variable tenga una asignación coherente (es decir, no puede seleccionar tanto y_{x_j} como $y_{\neg x_j}$).

La correspondencia entre las soluciones de *3D Matching* y *3-SAT* se establece como sigue: Si existe una asignación que satisface ϕ, podemos construir un conjunto de tríos M seleccionando un literal para cada cláusula C_i y asignando su valor de verdad correspondiente. Este conjunto de tríos cubre exactamente los elementos de X, Y, Z. Por el contrario, si existe un conjunto de tríos M que es una solución para *3D Matching*, podemos deducir una asignación para las variables de ϕ seleccionando los literales correspondientes a los tríos elegidos para cada cláusula.

La construcción del conjunto de tríos T requiere tiempo proporcional al número de cláusulas y literales en ϕ. Esto implica que el tamaño de T es $O(m \cdot 3) = O(m)$, donde m es el número de cláusulas. Por lo tanto, la construcción puede realizarse en tiempo polinomial.

Demostración de que el problema *Subgrafo Máximo Bipartito* es NP-Hard

Definición del problema El problema consiste en encontrar, dado un grafo $G = (V, E)$, el subgrafo $G' = (V', E')$ con $V' \subseteq V$ y $E' \subseteq E$ de forma que G' sea bipartito y $|E'|$ sea máximo.

Reducción polinomial desde *3-SAT* Para demostrar que el problema de *Subgrafo Máximo Bipartito* es NP-Hard, realizamos una reducción polinomial desde el problema *3-SAT*. Dada una instancia del problema *3-SAT*, construimos un grafo $G = (V, E)$ como sigue: Para cada cláusula C_i, añadimos tres vértices c_{i1}, c_{i2}, c_{i3} que representan los literales de C_i. Conectamos estos tres vértices mediante un triángulo completo, añadiendo las aristas (c_{i1}, c_{i2}), (c_{i2}, c_{i3}), y (c_{i1}, c_{i3}). Para cada par de literales ℓ y $\neg\ell$ en cláusulas distintas, añadimos una arista $(v_\ell, v_{\neg\ell})$ que conecta los vértices correspondientes.

El objetivo es determinar si existe un subgrafo bipartito de tamaño k, donde $k = 3m$, que incluya exactamente dos aristas de cada triángulo y todas las aristas adicionales entre literales opuestos. Este subgrafo bipartito corresponde a una asignación de valores que satisface ϕ.

La correspondencia entre las soluciones de *3-SAT* y *Subgrafo Máximo Bipartito* se establece como sigue: Si existe una asignación que satisface ϕ, podemos construir un subgrafo bipartito seleccionando: Para cada cláusula C_i, dos aristas del triángulo c_{i1}, c_{i2}, c_{i3} que correspondan al literal verdadero. Todas las aristas adicionales entre literales opuestos. Esto asegura que el subgrafo resultante sea bipartito y contenga exactamente $3m$ aristas. Por el contrario, si existe un subgrafo bipartito de tamaño $3m$, entonces: De cada triángulo c_{i1}, c_{i2}, c_{i3}, se deben seleccionar exactamente dos aristas (pues un triángulo completo no puede ser bipartito). La selección de las aristas corresponde a la asignación de un literal verdadero para cada cláusula C_i, y la bipartición asegura que no se seleccionen literales contradictorios (ℓ y $\neg\ell$).

La construcción del grafo G requiere tiempo proporcional al número de cláusulas y literales en ϕ. El número de vértices es $3m$, y el número de aristas incluye $3m$ aristas para los triángulos más un número adicional para las conexiones entre literales opuestos. Por lo tanto, la construcción puede realizarse en tiempo polinomial.

Demostraciónde que el problema *Máximo Corte* es NP-Hard

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 $V \setminus T$. El *costo de un corte* es la suma de los pesos de las aristas que van de T a $V \setminus T$.

El problema trata de encontrar el *corte de mayor costo* de un grafo.

Reducción polinomial desde *3-SAT* Para demostrar que el problema de *Máximo Corte* es NP-Hard, realizamos una reducción polinomial desde el problema *3-SAT*. Dada una instancia del problema *3-SAT*, construimos un grafo $G = (V, E)$ como sigue: Para cada variable x_i en ϕ, añadimos dos vértices v_{x_i} y $v_{\neg x_i}$ que representan $x_i = \text{true}$ y $x_i = \text{false}$, respectivamente. Conectamos estos dos vértices mediante una arista con peso $w = 1$ para garantizar que v_{x_i} y $v_{\neg x_i}$ estén en lados opuestos de la partición. Para cada cláusula $C_j = (\ell_1 \vee \ell_2 \vee \ell_3)$, añadimos un vértice c_j que representa la cláusula C_j. Este vértice se conecta a los tres literales correspondientes $(v_{\ell_1}, v_{\ell_2}, v_{\ell_3})$ mediante aristas con peso $w = 1$.

El objetivo es encontrar una partición de los vértices V en A y B tal que el número de aristas cruzadas sea máximo. En este contexto: Cada cláusula C_j estará "satisfecha" si al menos uno de sus literales está en el lado opuesto de la partición respecto a c_j. Para maximizar el corte, las aristas entre v_{x_i} y $v_{\neg x_i}$ deben cruzar la partición, lo que asegura que cada variable tenga una asignación consistente.

La correspondencia entre las soluciones de *3-SAT* y *Máximo Corte* se establece como sigue: Si existe una asignación que satisface ϕ, podemos construir una partición de los vértices V en A y B colocando: Los vértices v_{x_i} y $v_{\neg x_i}$ en lados opuestos según la asignación de x_i. Cada vértice c_j del lado opuesto a al menos uno de los literales que satisfacen C_j. En esta partición, el número de aristas cruzadas corresponde al número de cláusulas satisfechas más el número de variables. Por el contrario, si existe un corte con al menos $k = m + n$ aristas cruzadas (donde n es el número de variables y m es el número de cláusulas), podemos derivar una asignación para ϕ colocando v_{x_i} o $v_{\neg x_i}$ en lados opuestos, según su posición en la partición.

La construcción del grafo G requiere: $2n$ vértices para las variables y m vértices para las cláusulas, dando un total de $O(n+m)$ vértices. n aristas para conectar v_{x_i} y $v_{\neg x_i}$, y $3m$ aristas para conectar cada cláusula c_j con sus literales. Esto da un total de $O(n + 3m)$ aristas. Por lo tanto, la construcción de G puede realizarse en tiempo polinomial respecto al tamaño de ϕ.