Algunos problemas NP-completos

Horst H. von Brand
vonbrand@inf.utfsm.cl

Departamento de Informática
Universidad Técnica Federico Santa María

Contenido

Definiciones previas

Consideremos subconjuntos $\mathscr{I} \subseteq \mathscr{U}$, donde $\mathscr{I}$ cumple con alguna condición particular. Se dice que un conjunto $\mathscr{I}$ es *maximal* si no se pueden agregar elementos a $\mathscr{I}$ sin que deje de cumplir la condición, se llama *máximo* si no hay conjuntos mayores. Similarmente se definen *minimal* y *mínimo*.

3SAT restringidos

El problema 3SAT sigue siendo duro incluso bajo la condición de que ninguna variable aparezca en más de tres cláusulas. Como este es un caso especial de 3SAT, está en NP. Reducimos 3SAT a esta variante para demostrar que es NP-duro.

3SAT restringido

Supongamos que x aparece en $k > 3$ cláusulas. Reemplace sus apariciones por $x_1, x_2, \ldots, x_k$, una nueva variable por cada vez que aparece. Añada cláusulas:

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge \cdots \wedge (\overline{x}_k \vee x_1)$$

Esta expresión es verdadera si y solo si todas las variables x_j tienen el mismo valor. Cada literal aparece a lo más dos veces (una vez en la fórmula original y una vez en las cláusulas nuevas). El largo de las cláusulas extra es proporcional al número de variables repetidas, menor que el largo de la expresión original. El largo de la expresión resultante (y el trabajo necesario para construirla) está acotada por un polinomio, como requerimos.

1-in-3 3SAT

El problema 3SAT sigue siendo duro bajo la condición de que cada cláusula contiene exactamente un literal verdadero. Esta variante es útil para reducciones. Un certificado es valores de las variables que cumplen la restricción. Reducimos 3SAT a esta variante para demostrar que es NP-duro.

1-in-3 3SAT

Supongamos una construcción $R(u, v, w)$ que es verdadera si y solo si exactamente una de u, v, w es verdadera. Queremos reducir una cláusula $x \vee y \vee z$ a una conjunción de $R(\cdot)$.
La fórmula:

$$R(\overline{x}, a, b) \wedge R(b, y, c) \wedge R(c, d, \overline{z})$$

donde a, b, c, d son nuevas variables a usar solo para esta cláusula, es cierta si y solo si al menos una de x, y, z es cierta. Una fórmula con m cláusulas y n variables da una fórmula con $3m$ cláusulas y $4m + n$ variables.

El problema Clique

Una *clique* de un grafo es un subgrafo completo (un K_n, n vértices conectados entre sí). El problema Clique toma un grafo y un entero n, y pregunta si el grafo tiene una *clique* de tamaño n.

Teorema

Clique *es* NP*-completo.*

El problema Clique

Demostración.

Un certificado que demuestra que Clique está en NP es un conjunto de n vértices conectados entre sí.

Para demostrar que Clique es NP-duro, reducimos 3SAT $\leq_p$ Clique.

El problema Clique

Para $\phi(x_1, \ldots, x_n)$ con m cláusulas, construimos $G = (V, E)$ como sigue:

- Para cada literal ℓ_k^r, donde k identifica a la cláusula y r numera al literal en la cláusula, crear un vértice v_k^r (acá $1 \leq r \leq 3$).
- Arcos son $v_k^r v_j^s$ tales que:
  - $k \neq j$ (literales de cláusulas diferentes)
  - $\ell_k^r \neq \neg \ell_j^s$ (literales no son opuestos)

  O sea, hay un arco entre literales independientes de cláusulas distintas.

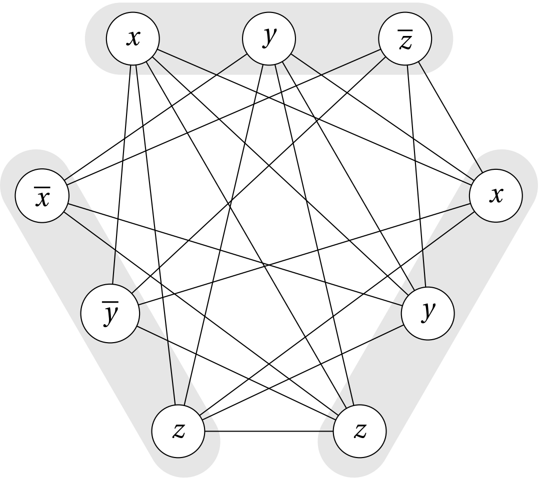
La instancia de Clique es (G, m).

El problema Clique

Es claro que si G tiene una clique de tamaño m podemos asignar el valor verdadero a los literales que corresponden a sus vértices, haciendo que cada cláusula contenga un literal verdadero. Al revés, si ϕ es satisfacible, podemos elegir m literales (uno por cláusula) que son verdaderos, y ellos dan los vértices de una clique. $\square$

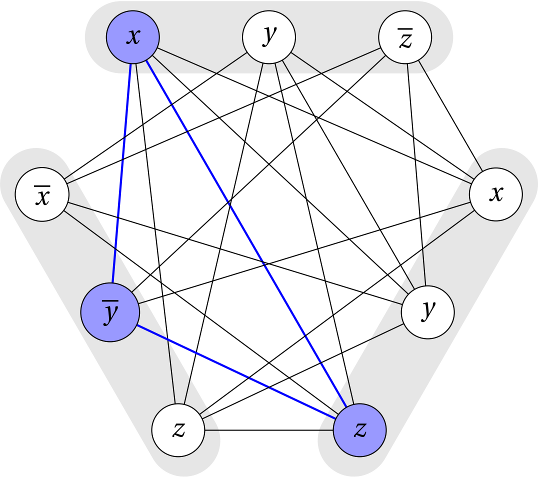
Clique contra 3SAT

Grafo para $(x \vee y \vee \overline{z}) \wedge (\overline{x} \vee \overline{y} \vee z) \wedge (x \vee y \vee z)$



Clique contra 3SAT

Un K_3: $x, \overline{y}, z$ verdaderos.



El problema IS (*Independent Set*)

Un *conjunto independiente* de un grafo es un conjunto de vértices entre los que no hay arcos. El problema IS (en inglés *Independent Set*) da un grafo y un entero n, y pregunta si el grafo tiene un conjunto independiente de tamaño n.

Teorema

IS *es* NP-*completo.*

El problema IS

Demostración.

El grafo G tiene un conjunto independiente de tamaño n si y solo si el grafo complemento $\overline{G}$ tiene una *clique* de tamaño n. □

El *grafo complemento* de $G = (V, E)$ es el grafo $\overline{G} = (V, \overline{E})$, conecta exactamente los vértices no conectados en G.

El problema Vertex Cover

Un *vertex cover* de un grafo es un conjunto de vértices que participan en todos los arcos. El problema Vertex Cover pone un grafo y un número natural k, pregunta si el grafo tiene un *vertex cover* de tamaño k.

Teorema

Vertex Cover *es* NP-*completo.*

El problema Vertex Cover

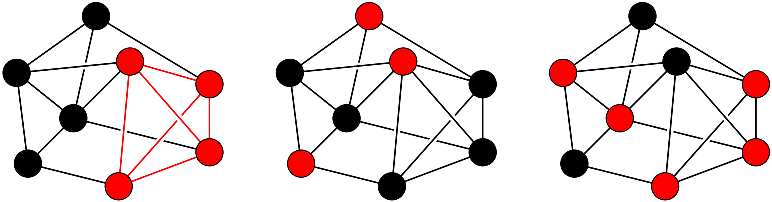
Demostración.

Un certificado es un conjunto de vértices en todos los arcos, está en NP.

Para reducir IS $\leq_p$ Vertex Cover, note que el complemento de un conjunto independiente es un *vertex cover*. □

Clique, Vertex Cover e Independent Set

Se distinguen en rojo los vértices de los conjuntos de interés.



(a) Una *clique* (b) Un *independent set* (c) Un *vertex cover*

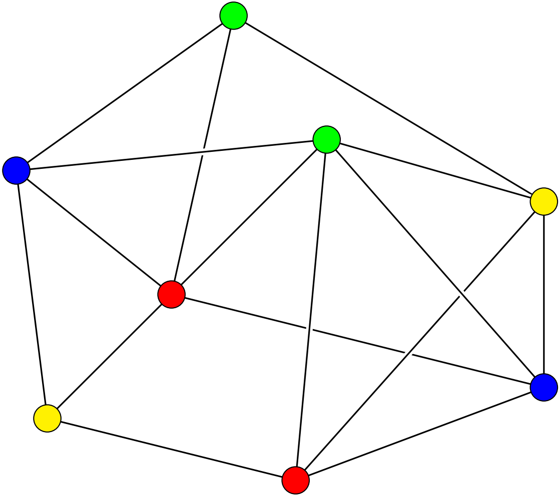
Coloreo de grafos

Un *coloreo* con k colores del grafo $G = (V, E)$ es una función $c \colon V \to [1, k]$ tal que si $uv \in E$ entonces $c(u) \neq c(v)$. El problema de coloreo de grafos es determinar si el grafo $G = (V, E)$ tiene un coloreo con k colores.

Determinar si un grafo tiene un coloreo con dos colores (es bipartito) es simple: Pinte un vértice de rojo, sus vecinos de azul, los vecinos de los azules de rojo, ...

Un coloreo

Un coloreo con cuatro colores.



3Color

El problema 3Color pone un grafo y pregunta si puede colorearse con tres colores.

Teorema

3Color *es* NP*-completo.*

3Color

Demostración.

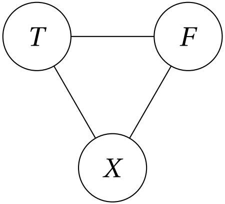
Un certificado que demuestra que está en NP es un coloreo del grafo con 3 colores.

Para demostrar que es NP-duro reducimos 3SAT $\leq_p$ 3Color.

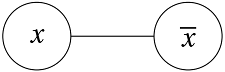
3Color

Sea la instancia ϕ de 3SAT, sin pérdida de generalidad todas sus cláusulas tienen exactamente tres literales. Partimos con K_3, que fuerza al menos tres colores. Uno de sus vértices representa *verdadero* (T), otro *falso* (F), y el tercero *indeciso* (X). Asignamos un vértice a cada literal de ϕ, los conectamos con el vértice indeciso (así deben tomar los colores *verdadero*, o *falso* en un coloreo con tres colores). Para asegurar valores consistentes, conectamos todas las instancias de una variable con sus negaciones.

3Color



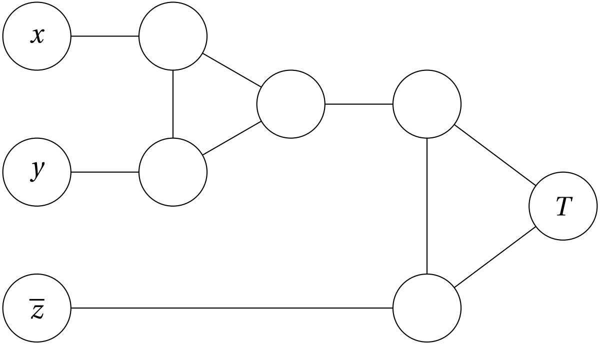
(d) K_3 base



(e) La variable x

3Color

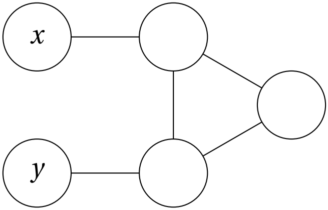
Para cada cláusula (en el ejemplo la cláusula $x \vee y \vee \overline{z}$) construimos un aparato como el de la figura.



El vértice marcado T es del K_3 con el que iniciamos la construcción.

3Color

Consideremos el subaparato de los dos primeros literales:



Viendo las distintas posibilidades de usar los colores T, F en los literales, si ambos literales toman el mismo color el vértice más a la derecha debe tomar ese color; si toman colores distintos su color es arbitrario. Al combinar para formar el aparato completo vemos que al menos uno de los tres literales debe tomar el color T para poder colorear con tres colores, en caso de tomar los tres literales el color F se requieren cuatro colores.

3D-Match

Este problema pone tríos de elementos, uno de cada uno de tres conjuntos (para precisar, sean niña, niño y mascota), de entre los cuales buscamos un subconjunto de tríos que contienen a cada niña, niño y mascota exactamente una vez.

Teorema

3D-Match *es* NP-*completo.*

3D-Match

Demostración.

Un certificado que demuestra que 3D-Match está en NP es un conjunto de tríos que cumplen las condiciones.
Reducimos polinomialmente nuestra variante de 3SAT que menciona cada variable tres veces y cada literal a lo más dos veces a 3D-Match para demostrar que es NP-duro.

3D-Match

Debemos hallar una manera de representar variables y cláusulas mediante tríos y sus participantes. Para representar la variable x, usamos los tríos (las letras son por *boy*, *girl* y *pet*):

$$
\begin{aligned}
T_{x0} &= \{b_{x0}, g_{x0}, p_{x1}\} \\
T_{x1} &= \{b_{x0}, g_{x1}, p_{x0}\} \\
T_{x2} &= \{b_{x1}, g_{x0}, p_{x2}\} \\
T_{x3} &= \{b_{x1}, g_{x1}, p_{x3}\}
\end{aligned}
$$

donde b_{xi} y g_{xi} no participan en otros tríos. Una solución incluye ya sea el par T_{x0} y T_{x3} o el par T_{x1} y T_{x2}. Usamos estas alternativas para representar «verdadero» y «falso». Note que quedan sin amos las mascotas p_{x0} y p_{x2} o p_{x1} y p_{x3} en estos dos casos, respectivamente.

3D-Match

Para una cláusula, introducimos tríos con nuevos niños y niñas. La elección de mascota del trío indicará si la variable representada está o no negada en una cláusula. Concretamente, para la cláusula $c = (x \vee \overline{y} \vee z)$, agregamos b_c y g_c. Si c es verdadera, es porque x es verdadera, y es falsa o z es verdadera. Para la variable x, si aparece en c por primera vez en ϕ, queremos encontrar hogar para p_{x0}, agregamos $\{b_c, g_c, p_{x0}\}$. Si es la segunda vez que aparece, el trío es $\{b_c, g_c, p_{x2}\}$ (recuerde que x aparece a lo más tres veces, a lo más dos veces sin negar, en nuestra variante de 3SAT). Para y falso, incluimos el trío $\{b_c, g_c, p_{y1}\}$ o $\{b_c, g_c, p_{y3}\}$, según si es la primera o segunda vez que se menciona la variable y.

3D-Match

Quedan mascotas huérfanas. Si hay n variables y m cláusulas, hay $2n+m$ niños (y el mismo número de niñas), mientras hay $4n$ mascotas. Siempre hay exactamente $4n-(2n+m)=2n-m$ mascotas en esta triste situación. Pero esto es fácil de corregir, creando $2n-m$ parejas de niñas y niños acogedores, que estarían felices con cualquier mascota.
El tamaño del problema resultante es proporcional al largo de la expresión, y el trabajo requerido para armarlo también. O sea, la reducción se efectúa en tiempo polinomial. $\square$

Ecuaciones cero-uno (ZOE)

Una instancia de este problema, abreviado ZOE (por *Zero One Equations*), es una matriz $\mathbf{A}$ de $m \times n$ de entradas 0 o 1, buscamos determinar si hay un vector $\mathbf{x}$ de 0 y 1 tal que:

$$\mathbf{A}\mathbf{x} = \mathbf{1}$$

Teorema

ZOE *es* NP-*completo.*

ZOE

Demostración.

Un certificado para ZOE es los valores de las variables, está en NP.
Reducimos 3D-Match $\leq_p$ ZOE, con lo que ZOE es NP-duro.

ZOE

Consideremos dados m niñas, m niños y m mascotas. Sean dados n tríos conformados de una niña, un niño y una mascota elegidos de entre los anteriores. Usamos variables cero-uno $x_1, \ldots, x_n$, donde $x_i = 1$ significa que el trío i se elige. Si los tríos que contienen a una niña específica son $j_1, \ldots, j_k$, el que esta niña aparece exactamente en uno de los tríos elegidos se traduce en:

$$x_{j_1} + x_{j_2} + \cdots + x_{j_k} = 1$$

La misma idea es aplicable a niños y mascotas. Esto da $3m$ ecuaciones cero-uno en n variables, que tienen solución si y solo si la instancia de 3D-Match tiene solución.
Queda de ejercicio ver que la reducción es polinomial. $\square$

Programación lineal entera (ILP)

El problema programación lineal entera (ILP, por *integer linear program*) consiste en variables $x_1, \ldots, x_n$ que deben ser enteras, sujetas a restricciones lineales de la forma:

$$\sum_j a_{ij} x_j \le b_i$$

donde $a_{ij}, b_i \in \mathbb{Z}$. La pregunta es ¿hay valores enteros de x_i tales que las restricciones se cumplen?

Teorema

ILP *es* NP*-completo.*

Programación lineal entera (ILP)

Demostración.

Un certificado que demuestra que está en NP es un vector de valores que satisfacen las restricciones.
Cada ecuación de ZOE se traduce en dos restricciones (una vez mayor o igual, una vez menor o igual), con lo que la reducción ZOE $\leq_p$ ILP es trivial. □

Programación lineal entera (ILP)

Alternativamente, podemos reducir Vertex Cover $\leq_p$ ILP.
Dado el grafo $G = (V, E)$ definimos variables $x_v \in \{0, 1\}$ para $v \in V$ (el vértice pertenece o no al *vertex cover*), ponemos restricciones $x_u + x_v \geq 1$ para $uv \in E$ (todo arco tiene al menos un representante) y $\sum_{v \in V} x_v = k$ (el conjunto tiene tamaño k).

Subset Sum

El problema Subset Sum presenta un multiconjunto de naturales $\mathscr{A}$ y un natural S, ¿hay un multisubconjunto de $\mathscr{A}$ que suma S?

Teorema

Subset Sum *es* NP-*completo.*

Subset Sum

Demostración.

Un certificado para Subset Sum es un multisubconjunto de $\mathscr{A}$ que suma S, está en NP.

Reducimos 3SAT $\leq_p$ Subset Sum.

Subset Sum

Sea ϕ una instancia de 3SAT, de m variables y n cláusulas. La idea de la reducción es construir números que representan las variables, que consideramos divididos en rangos de dígitos con ciertos significados. Para cada x_i construimos números v_i y f_i de $m+2n$ dígitos, Incluir v_i indica que x_i es verdadera, incluir f_i corresponde a x_i falso.

Subset Sum

- El dígito i de v_i y f_i es 1, los demás de los primeros m dígitos son 0. Exigiendo que en las posiciones 1 a m la suma pedida sea 1, estamos forzando a elegir solo uno de v_i o f_i para cada variable x_i.
- El dígito $m+k$ de v_i es 1 si x_i aparece en la cláusula k, el dígito $m+k$ de f_i es 1 si $\overline{x}_i$ aparece en la cláusula k; todos los demás en el rango $m+1$ a $m+2n$ son 0. Si la posición $m+k$ de la suma es al menos 1 la cláusula k se satisface.

Subset Sum

Tenemos cláusulas con 1, 2 o 3 literales verdaderos si se satisface ϕ, para poder «rellenar» una suma de tres para cada cláusula k agregamos valores:

- z_k (por *zero*) tiene dígito 1 en la posición $m+n+k$, cero en todos los demás. A usar si la cláusula k tiene tres literales verdaderos, completamos con 0.
- u_k (por *unity*) dígito 1 en la posición $m+k$, se usa en caso de que hayan dos literales verdaderos en la cláusula k, y dígito 1 en la posición $m+n+k$.
- d_k (por *duo*) tiene dígito 2 en la posición $m+k$, a usar en el caso que haya un literal verdadero en la cláusula k, y dígito 1 en la posición $m+n+k$.

Subset Sum

Con la suma anterior estos tres valores permiten completar los dígitos $m+1$ a $m+n$ a 3. Los dígitos en la tercera parte de z_k, u_k y d_k permiten usar solo uno de los tres exigiendo que esas posiciones sean 1.

El valor de la suma buscada tiene 1 en los primeros m dígitos (todas las variables tienen valores, cada una verdadera o falsa), 3 en los dígitos $m+1$ a $m+n$) (todas las cláusulas con a lo menos un literal verdadero, completando a 3 incluyendo el relleno necesario), y 1 en las posiciones $m+n+1$ a $m+2n$ (rellenar con z_k, u_k o d_k según necesidad, pero permitir solo uno de z_k, u_k o d_k).

Es claro que esta instancia de Subset Sum tiene solución si y solo si la instancia correspondiente de 3SAT tiene solución. $\square$

Resumen

- Dimos una pequeña lista de problemas NP-completos, ilustrando algunas técnicas de demostración en el proceso.
- Se aprecia en nuestros ejemplos que (como es común) demostrar que el problema está en NP es muy simple, lo difícil es hallar de dónde reducir y construir la reducción. Debemos traducir los objetos y sus relaciones del problema desde el cual reducimos en objetos y relaciones del problema entre manos, y asegurar que la representación es posible en tiempo polinomial.