Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2213 - Lógica para Ciencias de la Computación
Profesor: Miguel Romero
Ayudantes: Caetano Borges, Valentina Juri

Ayudantía 11
Problemas NP-completos

19 de junio de 2026

Ejercicio 1 Sea α una fórmula proposicional cualquiera. Escribimos $\alpha = \beta$ para decir que α y β son la misma fórmula. Considere el siguiente lenguaje:

$$\text{SAT}{+}\alpha = \{\langle\beta\rangle \mid \beta \text{ es una fórmula proposicional que es satisfacible o } \beta = \alpha\}.$$

Demuestre que SAT+α es NP-completo.

Solución

Para demostrar que SAT+α es NP-completo, debemos probar que pertenece a la clase NP y que es NP-hard.

1. **SAT**+$\alpha \in$ **NP**: Sea $\langle\beta\rangle$ una instancia del lenguaje. Un verificador polinomial puede operar de la siguiente forma:

   - Primero, comprueba si la cadena representa sintácticamente de forma exacta a la fórmula α (es decir, si $\beta = \alpha$). Dado que α es una constante fija del lenguaje, esta verificación toma tiempo constante $O(1)$ con respecto al tamaño de β. Si se cumple, el verificador acepta inmediatamente.
   - Si $\beta \neq \alpha$, el verificador utiliza como certificado una asignación de variables para β. Luego, evalúa si dicha asignación satisface a β, lo cual se puede realizar en tiempo polinomial respecto al tamaño de la fórmula.

   Por lo tanto, existe un verificador en tiempo polinomial, lo que implica que SAT+$\alpha \in$ NP.

2. **SAT**+α **es NP-hard**: Realizaremos una reducción desde el problema SAT clásico, el cual ya sabemos que es NP-completo. Queremos definir una función computable en tiempo polinomial f tal que $\beta \in \text{SAT} \iff f(\beta) \in \text{SAT} + \alpha$.

   Para cualquier fórmula β, definimos $f(\beta) = \beta'$, donde β' se construye de la siguiente manera: Escogemos una variable proposicional totalmente nueva x que no aparezca en la fórmula fija α. Luego, definimos:

   $$\beta' = \beta \wedge (x \vee \neg x)$$

Esta modificación sintáctica trivial exactamente el estado de satisfacibilidad de β y se puede construir en tiempo polinomial. Demostramos su correctitud:

- ($\Rightarrow$) Supongamos que $\beta \in \text{SAT}$. Entonces existe una asignación que la satisface. Como la cláusula $(x \vee \neg x)$ siempre es una tautología, cualquier extensión de dicha asignación que incluya a x también satisfará a β'. Por lo tanto, β' es satisfacible, lo que por definición implica que $\beta' \in \text{SAT} + \alpha$.
- ($\Leftarrow$) Supongamos que $\beta' \in \text{SAT} + \alpha$. Por definición del lenguaje, esto significa que β' es satisfacible o bien $\beta' = \alpha$. Sin embargo, por construcción, β' contiene la variable x, la cual explícitamente se escogió de modo que no perteneciera a α. Por ende, estructuralmente es imposible que $\beta' = \alpha$. Esto nos obliga a concluir que β' debe ser satisfacible. Como $\beta' \equiv \beta$, se deduce directamente que β es satisfacible, es decir, $\beta \in \text{SAT}$.

Al ser un problema en NP y NP-hard, concluimos que SAT+α es NP-completo.

Ejercicio 2 Sea $\mathcal{S}$ una colección de conjuntos finitos. Un packing de $\mathcal{S}$ de tamaño k es una subcolección $\mathcal{C} \subseteq \mathcal{S}$ de k conjuntos de $\mathcal{S}$ tal que $C \cap C' = \emptyset$, para todo $C \neq C'$ en $\mathcal{C}$. Es decir, todos los conjuntos de $\mathcal{C}$ son disjuntos entre sí. Considere el siguiente lenguaje:

$$\text{Set-Packing} = \{\langle \mathcal{S}, k\rangle \mid \mathcal{S} \text{ es una colección de conjuntos finitos con un packing de tamaño } k\}.$$

Demuestre que Set-Packing es NP-completo.

Solución

1. **Set-Packing $\in$ NP**: Dada una instancia $\langle \mathcal{S}, k\rangle$, definimos nuestro certificado como una subcolección $\mathcal{C}$. El verificador realiza los siguientes pasos en tiempo polinomial:
   - Comprueba que la cardinalidad de la subcolección sea exactamente k ($|\mathcal{C}| = k$).
   - Para cada par de conjuntos distintos $C, C' \in \mathcal{C}$, calcula su intersección y verifica que $C \cap C' = \emptyset$. Como hay $\binom{k}{2}$ pares y los conjuntos son finitos, este proceso toma tiempo polinomial en el tamaño de la entrada.

   Dado que el certificado es de tamaño polinomial y el proceso de verificación es eficiente, Set-Packing $\in$ NP.

2. **Set-Packing es NP-hard**: Reduciremos el problema de Clique, conocido por ser NP-completo, a Set-Packing.

   Sea $\langle G, k\rangle$ una instancia de Clique, donde $G = (V, E)$ es un grafo no dirigido y queremos determinar si existe un subconjunto de k vértices completamente conectados entre sí. Construimos una instancia $\langle \mathcal{S}, k'\rangle$ de Set-Packing de la siguiente forma:

   - Consideremos el grafo complemento $\overline{G} = (V, \overline{E})$, donde las aristas son exactamente aquellas que no existen en G. Es decir, $\overline{E} = \{\{u, v\} \mid u, v \in V, u \neq v \text{ y } \{u, v\} \notin$

$E\}$.

- Para cada vértice $v \in V$, definimos un conjunto S_v compuesto por todas las aristas en el grafo complemento que inciden en v. Formalmente, $S_v = \{e \in \overline{E} \mid v \in e\}$.
- La colección final será $\mathcal{S} = \{S_v \mid v \in V\}$.
- Fijamos el tamaño del packing requerido como $k' = k$.

Esta transformación toma tiempo polinomial, ya que calcular el grafo complemento y construir los conjuntos requiere a lo más iterar sobre todos los pares posibles de vértices, lo cual toma tiempo $O(|V|^2)$. Demostramos la correctitud de la reducción:

- ($\Rightarrow$) Supongamos que G tiene un clique $V' \subseteq V$ de tamaño k. Por definición de clique, para todo par de vértices distintos $u, v \in V'$, existe la arista $\{u, v\} \in E$. Esto implica que $\{u, v\} \notin \overline{E}$, por lo que en el grafo complemento no existe ninguna arista que conecte a u y v. Por lo tanto, los conjuntos S_u y S_v no comparten ninguna arista incidente en común, es decir, $S_u \cap S_v = \emptyset$. Así, la subcolección $\mathcal{C} = \{S_v \mid v \in V'\}$ forma un packing de tamaño k en $\mathcal{S}$.
- ($\Leftarrow$) Supongamos que $\mathcal{S}$ tiene un packing $\mathcal{C} \subseteq \mathcal{S}$ de tamaño k. Cada elemento en $\mathcal{C}$ corresponde a un conjunto S_v asociado a algún vértice. Definimos el conjunto de vértices $V' = \{v \in V \mid S_v \in \mathcal{C}\}$, por lo que $|V'| = k$. Como los elementos de $\mathcal{C}$ son mutuamente disjuntos, se cumple que para todo par $u, v \in V'$ distintos, $S_u \cap S_v = \emptyset$. Esto significa que u y v no comparten ninguna arista incidente en el grafo complemento $\overline{G}$, por lo que $\{u, v\} \notin \overline{E}$. En consecuencia, la arista sí debe existir en el grafo original, es decir, $\{u, v\} \in E$. Como esto se cumple para todos los pares de vértices en V', concluimos que V' es un clique de tamaño k en G.

Por lo tanto, Set-Packing es NP-completo.

Ejercicio 3 Sea $G = (V, E)$ un grafo (no dirigido). Un k-coloreo de G es una función $c\colon V \to \{1, \ldots, k\}$ tal que $c(u) \neq c(v)$ para toda arista $\{u, v\} \in E$. Considere el siguiente lenguaje:

$$\text{3-COL} = \{\langle G \rangle \mid G \text{ es un grafo que tiene un 3-coloreo}\}.$$

Demuestre que 3-COL es NP-completo.

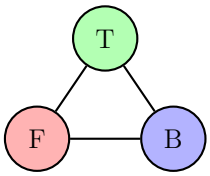
Solución

1. **3-COL $\in$ NP**: Dada una instancia $\langle G \rangle$, el certificado es una función de asignación de colores para los vértices $c\colon V \to \{1, 2, 3\}$. El verificador simplemente recorre la lista de aristas E del grafo y comprueba para cada una de ellas si sus extremos tienen asignados colores diferentes ($c(u) \neq c(v)$). Esta verificación toma tiempo lineal $O(|E|)$ con respecto al tamaño del grafo, por lo que el problema está en NP.

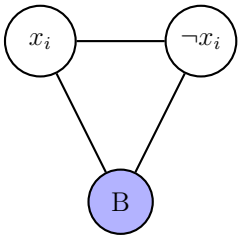
2. **3-COL es NP-hard**: Procederemos mediante una reducción desde el problema 3-SAT. Sea φ una fórmula booleana en formato 3-CNF con variables $x_1, \ldots, x_n$ y cláusulas $C_1, \ldots, C_m$. Construiremos un grafo $G = (V, E)$ que admita un 3-coloreo válido si y solo si φ es satisfacible.

   Asociaremos los 3 colores disponibles con las etiquetas T (*True*), F (*False*) y B (*Base*). La reducción se compone estructuralmente de tres partes:

   - **El triángulo de control:** Creamos tres nodos especiales interconectados en un triángulo elemental, denotados como T, F y B. Al estar mutuamente conectados, cualquier coloración válida del grafo los forzará a tener tres colores completamente diferentes.
   - **Gadgets de variables:** Para cada variable independiente $x_i \in \varphi$, creamos un par de nodos correspondientes a los literales x_i y $\neg x_i$. Conectamos estos dos nodos entre sí mediante una arista, y además los conectamos al nodo de control B.

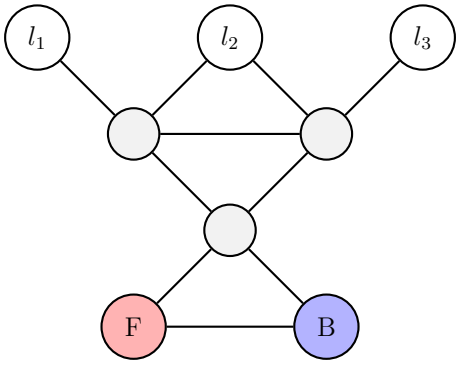


Triángulo de Control



Gadget de Variable

   - **Gadgets de cláusulas:** Para cada cláusula $C_j = (l_1 \vee l_2 \vee l_3)$, introducimos un subgrafo auxiliar interconectado entre los literales l_1, l_2, l_3 y los nodos globales F y B. Las conexiones internas de este subgrafo están diseñadas para que la estructura sea 3-coloreable si y solo si al menos uno de los tres literales de entrada posee el color de T.



Esquema Gadget de Cláusula

Esta construcción requiere agregar un número acotado de vértices y aristas por cada variable y cláusula, realizándose de forma evidente en tiempo polinomial.

Correctitud de la reducción:

- ($\Rightarrow$) Si $\varphi \in$ 3-SAT, existe una asignación de verdad que satisface la fórmula. Coloreamos el triángulo base asignándole los tres colores correspondientes. Evaluamos los literales de acuerdo a la asignación: si un literal es verdadero, toma el color de T; si es falso, toma el color de F. Al estar cada cláusula satisfecha, al menos un literal de entrada en cada gadget de cláusula tendrá el color T, lo que permite extender el 3-coloreo de manera consistente a los nodos intermedios del subgrafo sin generar conflictos con F ni B.
- ($\Leftarrow$) Si G admite un 3-coloreo, fijamos la semántica de los colores mediante los nodos T, F y B. La restricción sobre los pares de nodos x_i y $\neg x_i$ garantiza que cada variable reciba una asignación booleana consistente (es decir, una es T y la otra es F, ya que ambas están conectadas a B). Por el diseño del gadget de cláusula, si los tres literales tuvieran el color F, forzarían a los nodos internos a un conflicto de colores al conectarse al nodo global B y al nodo global F. Por ende, la coloración sólo es realizable si al menos uno de los literales de la cláusula está pintado con el color de T. La asignación derivada satisface todas las cláusulas de φ, demostrando que $\varphi \in$ 3-SAT.

Al cumplir con ambas condiciones, queda demostrado que el problema 3-COL es NP-completo.