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 12

NP-completitud

26 de junio de 2026

Problema 1. Definimos el siguiente lenguaje:

$$\text{SET-COVER} = \{\langle S, k\rangle \mid S \text{ colección de conjuntos finitos con un set cover de tamaño k}\}$$

Demuestre que SET-COVER es NP-completo.

Solución: Primero veamos que SET-COVER está en NP. Para esto consideramos una MTN M con el siguiente pseudocódigo:

1. Adivinar un subconjunto $\mathcal{C} \subseteq S$ de tamaño k.
2. Calcular $\bigcup_{s\in\mathcal{S}} s$.
3. Calcular $\bigcup_{c\in\mathcal{C}} c$.
4. Verificar que $\bigcup_{s\in\mathcal{S}} s = \bigcup_{c\in\mathcal{C}} c$. De ser así, **aceptar**, en caso contrario, **rechazar**.

Véase que la MTN descrita toma tiempo polinomial en $|S|$. En primer lugar, como la colección de conjuntos $\mathcal{C}$, cumple con $|\mathcal{C}| \leq |S|$, el paso (1) toma tiempo polinomial. Por otro lado, las uniones realizadas en los pasos (2) y (3) son polinomiales en la cantidad de conjuntos de cada conjunto ($\mathcal{C}$ y S respectivamente), puesto que solo requieren recorrer el arreglo e ir revisando que cada valor a insertar no este repetido, lo que es polinomial en $|S|$. Finalmente, la comparación entre las uniones es polinomial en $|S|$, dado en que consiste en una comparación elemento a elemento entre ambos conjuntos (si los conjuntos están ordenados, puede incluso ser lineal).

Ahora veamos que SET-COVER es NP-hard. Para esto construiremos una reducción polinomial desde VERTEX-COVER, es decir, probaremos que VERTEX-COVER $\leq_p$ SET-COVER. Sea un par $\langle G, k\rangle$ con $G = (V, E)$, nuestra reducción computa el par $\langle S_G, k\rangle$, de la siguiente forma:

1. Para cada nodo $v \in V$, definimos el conjunto S_v como: $S_v = \{e \in E \mid e \text{ es incidente a } v\}$
2. Si $V = \{v_1, ..., v_n\}$, definimos $S_G = \{S_{v_1}, ..., S_{v_n}\}$.

Véase que esta reducción se puede computar en tiempo polinomial en $n = |V|$ que es nuestro parámetro de referencia. Así, recorremos, como cota superior, todas las artistas de G ($|E| \approx n^2$), por cada nodo del grafo, para determinar el conjunto S_v, por lo que este paso es polinomial en n.

Demostremos la correctitud de la reducción, es decir, que $\langle G, k\rangle \in$ VERTEX-COVER $\iff$ $\langle S_G, k\rangle \in$ SET-COVER. En primer lugar, si $\langle G, k\rangle \in$ VERTEX-COVER con $G = (V, E)$, existe un vertex cover $D = \{z_1, ..., z_k\} \subseteq V$ con k nodos. Veamos que $\mathcal{C} = \{S_{z_1}, ..., S_{z_k}\}$ es un set cover para $S_G = \{S_{v_1}, ..., S_{v_n}\}$. Primero nótese que por construcción $\bigcup_{j=1}^{n} S_{v_n}$ o el universo de elementos, siempre será un subconjunto de aristas de G. Es decir, para que $\mathcal{C}$ sea un set cover, se debe cumplir que cada arista $e \in E$ esté contenida en $\bigcup_{j=1}^{k} S_{z_j}$. Sea $e \in E$ arbitraria. Como $D = \{z_1, ..., z_k\}$ es un vertex cover, existe $j = \{1, ..., k\}$ tal que la arista es incidente a $z_j \in D$. Así, por definición de S_{z_j} (aristas incidentes al nodo) se tiene que $e \in S_{z_j}$ y luego $e \in \bigcup_{j=1}^{k} S_{z_j}$. Concluimos que existe un set cover de tamaño k para S_G, por lo que $\langle S_G, k\rangle \in$ SET-COVER.

Para la otra dirección, supongamos que $\langle S_G, k\rangle \in$ SET-COVER, así, existe un set cover $\mathcal{C} \subseteq S_G = \{S_{v_1}, ..., S_{v_n}\}$ de tamaño k, con $G = (V, E)$ y $V = \{v_1, ..., v_n\}$. Tenemos que $\mathcal{C} = \{S_{z_1}, ..., S_{z_k}\}$ para cierto subconjunto de nodos $D = \{z_1, ..., z_k\} \subseteq V$. Veamos que D es un vertex cover para G. Sea $e \in E$ una arista arbitraria. Como $\mathcal{C} = \{S_{z_1}, ..., S_{z_k}\}$ es un set cover, se cumple que $e \in \bigcup_{j=1}^{k} S_{z_j}$, luego, existe $j \in \{1, ..., k\}$ tal que $e \in S_{z_j}$. Por definición de S_{z_j}, tenemos que e es incidente a z_j, de modo que toda arista es incidente a algún nodo de D. Concluimos que $\langle G, k\rangle \in$ VERTEX-COVER.

Dada la correctitud de la reducción, tenemos que como SET-COVER pertenece a la clase de problemas NP y es NP-hard, por definición es NP-completo.

Problema 2. Definimos el siguiente lenguaje:

$$\text{Partition } = \{\langle A\rangle \mid A \text{ multi-conjunto de números naturales, existe } S \subseteq A \text{ tal que } \sum_{s\in S} s = \frac{1}{2}\sum_{a\in A} a\}$$

Demuestre que Partition es NP-completo.

Solución: Primero veamos que Partition está en NP. Para esto consideramos una MTN M con el siguiente pseudocódigo:

1. Adivinar un multiconjunto $S \subseteq A$.
2. Calcular $\sum_{a\in A} a$.
3. Calcular $\sum_{s\in S} s$.
4. Verificar que $\sum_{s\in S} s = \frac{1}{2}\sum_{a\in A} a$. De ser así, **aceptar**, en caso contrario, **rechazar**.

Véase que la MTN descrita toma tiempo polinomial en $|A|$. En primer lugar, como el multiconjunto S cumple que $|S| \leq |A|$, el paso (1) toma tiempo polinomial. Por otro lado, las sumas realizadas en los pasos (2) y (3) son lineales en el largo de los arreglos sumados (A y S, respectivamente), por lo que ambas son polinomiales en $|A|$. Finalmente, la verificación puede considerarse constante.

Nota: En estricto rigor, el parámetro de referencia más correcto para este ejercicio sería $|A| + |\log_2(\max(A))|$, porque la cantidad de operaciones realizada por la MTN al sumar y comparar depende del largo de la codificación de los números que componen el arreglo. Por ejemplo, en estricto rigor, la comparación es lineal en $|\log_2(\max(A))|$ no constante, por simplicidad se considera $|A|$, pero el algoritmo es polinomial en $|A| + |\log_2(\max(A))|$.

Ahora veamos que Partition es NP-hard. Para esto construiremos una reducción polinomial desde SUBSET-SUM, es decir, probaremos que SUBSET-SUM $\leq_p$ Partition. La reducción, dada una instancia $\langle A, t\rangle$, con A un multiconjunto de naturales y t un natural, realiza lo siguiente:.

1. Calcula $T = \sum_{a \in A} a$.
2. Si $t > T$, retorna una instancia inválida de Partition (eg: $A' = \{1, 2\}$).
3. Si $t \leq T$, retorna el arreglo $A' = \{a_1, ..., a_n, 2T - t, T + t\}$.

Nótese que esta reducción es polinomial, puesto que: calcular la suma de todos los elementos de A en el paso (1) es lineal en $|A|$, la comparación entre t y T puede considerarse constante. Adicionalmente, retornar una instancia inválida es constante, puesto que esta puede quedar 'hardcodeada' en la máquina y agregar dos elementos al arreglo en (3) es lineal en $|A|$, porque debe llegarse hasta el final del arreglo antes de agregar los nuevos valores.

Demostremos la correctitud de la reducción, es decir, que $\langle A, t\rangle \in$ SUBSET-SUM $\iff$ $\langle A'\rangle \in$ Partition. En primera instancia, véase que en el paso (2) si $t > T$, esto significa que la suma objetivo es inalcanzable, puesto que estamos trabajando con naturales y el arreglo completo suma T. Así, $\langle A, t\rangle \notin$ SUBSET-SUM por lo que para respetar la pertenencia de lenguajes, se debe cumplir que $\langle A'\rangle \notin$ Partition. En este caso la correctitud es evidente. Por otro lado, en (3) si $t \leq T$ la suma podría ser alcanzable, y se agregan al arreglo los números 2T-t y T+t. Nótese además que la suma de los elementos del arreglo retornado es $\sum_{a' \in A'} a' = \sum_{a \in A} a + (2T - t) + (T + t) = T + 3T = 4T$, por lo que la mitad de esta es $2T$. Demostraremos la correctitud en este caso.

En primer lugar, si $\langle A, t\rangle \in$ SUBSET-SUM, se tiene que existe un subconjunto $S \subseteq A$ tal que $\sum_{s \in S} s = t$. Así, el conjunto $I = S \cup \{2T - t\}$ (subconjunto de A′) cumple con que $\sum_{i \in I} i = 2T$, siendo este valor la mitad de $\sum_{a' \in A'} a'$. Dado esto, concluimos que $\langle A'\rangle \in$ Partition.

Para la otra dirección, supongamos que $\langle A'\rangle \in$ Partition, esto significa que existe un subconjunto de $I \subseteq A'$ tal que $\sum_{i \in I} i = 2T$. Véase primero, que $\sum_{a \in A} a = T$, por la definición de T, de modo que no es posible que un subconjunto de A sume $2T$, por lo que I debe considerar al menos un elemento de $A'/A = \{2T - t, T + t\}$ para llegar al valor pedido de la suma. Además, tenemos que $2T - t + T + t = 3T$, por lo que I no puede considerar ambos elementos de A'/A, por que al sumar se pasaría de $2T$. Esto nos lleva a concluir que I contiene a $2T - t$ o (exclusivo) a $T + t$. Sea $S \subseteq A$ el conjunto de elementos pertenecientes a A en I, evaluemos cada caso.

- Si $I/S = \{2T - t\}$: Tenemos que $\sum_{i \in I} i = \sum_{s \in S} s + 2T - t = 2T$, de modo que $\sum_{s \in S} s = t$. Es decir, S es un subconjunto de A tal que la suma de sus elementos es t, por lo que $\langle A, t\rangle \in$ SUBSET-SUM.
- Si $I/S = \{T + t\}$: Tenemos que $\sum_{i \in I} i = \sum_{s \in S} s + T + t = 2T$, de modo que $\sum_{s \in S} s = T - t$. Ahora bien, como $\sum_{a \in A} a = T$, tenemos que $\sum_{s \in S} s + \sum_{s' \in A/S} s' = T$, por lo que $\sum_{s' \in A/S} s' = t$. Siendo A/S el complemento de S respecto a A y corresponde a un subconjunto de A tal que al sumar sus elementos se obtiene t, por lo que $\langle A, t\rangle \in$ SUBSET-SUM.

Dada la correctitud de la reducción, tenemos que como Partition pertenece a la clase de problemas NP y es NP-hard, por definición es NP-completo.

Problema 3. Decimos que una fórmula proposicional φ está en CNF_3 si está en CNF y cada variable aparece

a lo más 3 veces en φ. Por ejemplo la fórmula:
$\varphi_1 = (\neg x_1 \vee x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5 \vee \neg x_6) \vee (x_6 \vee x_7)$ está en CNF_3 ya que cada variable aparece a lo más 3 veces, pero:
$\varphi_2 = (\neg x_1 \vee x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee x_5 \vee \neg x_6) \vee (x_6 \vee x_3)$ no está en CNF_3 ya que la variable x_3 aparece 4 veces.

Definamos el siguiente lenguaje:

$$\text{CNF}_3\text{-SAT} = \{\langle\varphi\rangle \mid \varphi \text{ es una fórmula satisfacible en CNF}_3\}$$

Demuestre que CNF_3-SAT es NP-completo.

Solución: Primero veamos que CNF_3-SAT está en NP. Para esto consideramos una MTN M con el siguiente pseudocódigo:

1. Verificar que φ esté en CNF_3.
2. Adivinar una valuación σ para φ.
3. Calcular $\sigma(\varphi)$, si $\sigma(\varphi) = 1$, **aceptar**, en caso contrario, **rechazar**.

Véase que el paso (1) toma tiempo polinomial en $|\varphi|$, puesto que para verificar que φ esté en CNF_3, se debe revisar que la fórmula esté en CNF (lineal) y que cada variable no aparezca más de 3 veces en la fórmula (para cada variable recorro a lo más el largo completo de φ, polinomial). Similarmente, el paso (2) también toma tiempo polinomial, ya que el tamaño de las valuaciones de σ es polinomial (a lo más $|\varphi|$) y el paso (3) también es polinomial, cómo se discutió en clases (uso de árboles sintácticos lo hacen lineal).

Ahora veamos que CNF_3-SAT es NP-hard. Para esto construiremos una reducción polinomial desde CNF-SAT, es decir, probaremos que CNF-SAT $\leq_p$ CNF_3-SAT. La reducción computa dada una fórmula $\varphi = C_1 \wedge ... \wedge C_m$ en CNF una fórmula α_φ, inicialmente idéntica a φ, pero si una variable x ocurre ℓ veces con $\ell \geq 4$, entonces las ℓ ocurrencias de x se reemplazan por ℓ variables nuevas $x_1, ..., x_\ell$. Adicionalmente, para cada grupo de variables nuevas agregado agregamos un ciclo de implicancias sobre $x_1, ..., x_\ell$, es decir:

$$(x_1 \to x_2) \wedge (x_2 \to x_3) \wedge .... \wedge (x_{\ell-1} \to x_\ell) \wedge (x_\ell \to x_1)$$

Nótese que la fórmula $(x \to x')$ es equivalente a la cláusula $(\neg x \vee x)$. Veamos que α_φ está en CNF_3. Cada variable x que aparece más de 3 veces en φ, es reemplazada por variables distintas $x_1, ..., x_\ell$, así, cada una de estas variables x_i aparece una vez en las cláusulas de la fórmula original y dos veces en el ciclo de implicancias (una cuando apunta a la siguiente variable y otra cuando es apuntada). En total, cada variable en α_φ aparece a lo más 3 veces. Por otro lado, veamos que esta reducción se puede calcular en tiempo polinomial, así, para cada variable que aparece más de 3 veces en φ, se deben reemplazar sus apariciones por las variables $x_1, ..., x_\ell$ y agregar el ciclo de implicancias correspondiente con $\ell \leq |\varphi|$ cláusulas adicionales. De este modo, la reducción agrega a lo más $|\varphi|^2$ cláusulas.

Finalmente, demostremos la correctitud de la reducción, es decir, que $\langle\varphi\rangle \in$ CNF-SAT $\iff \langle\alpha_\varphi\rangle \in$ CNF_3-SAT. En primer lugar, si $\langle\varphi\rangle \in$ CNF-SAT, tenemos que existe una valuación σ para φ, tal que σ satisface a φ. Definimos la siguiente valuación τ para α_φ: si x aparece a lo más 3 veces en φ, conservamos su valor de

verdad, es decir, $\tau(x) = \sigma(x)$, si x aparece 4 o más veces, todas las variables $x_1, ..., x_\ell$ tienen el mismo valor de verdad que x, es decir, $\tau(x_1) = \tau(x_2) = ... = \tau(x_\ell) = \sigma(x)$. La valuación τ satisface a α_φ ya que las cláusulas originales de φ tienen el mismo valor de verdad que antes y todos los ciclos de implicancias son verdaderos, ya que o todas las variables son verdaderas o todas son falsas. Concluimos que $\langle \alpha_\varphi \rangle \in \text{CNF}_3\text{-SAT}$.

Nota: una implicancia es verdadera si tanto el antecedente como el consecuente son verdaderos (1) o si el antecedente es falso (0), es decir, $1 \to 1$, $0 \to 0$ o $0 \to 1$ con 0 y 1 los valores de verdad respectivos. Así, si una variable en el ciclo de implicancias es verdadera todas deben serlo (si no la implicancia en que el consecuente es falso es inmediatamente falsa). Por otro lado, si una variable en el ciclo es falsa, todas deben serlo, porque si en algún momento se tiene una implicancia de la forma $0 \to 1$ para un $x_i = 0$, por más que esta implicancia sea verdadera, dado lo dicho anteriormente todas las variables hacia delante deben ser verdaderas, de modo que una vez recorrido todo el ciclo, obtenemos que x_i también debe ser verdadera lo que es una contradicción.

Para la otra dirección, supongamos que $\langle \alpha_\varphi \rangle \in \text{CNF}_3\text{-SAT}$, así existe una valuación τ que satisface a α_φ. Si una variable x aparece $\ell \geq 4$ veces en φ, se debe cumplir que $\tau(x_1) = ... = \tau(x_\ell)$, debido al ciclo de implicancias presente en α_φ. Definimos la siguiente valuación σ para φ como sigue: si x aparece a lo más 3 veces en φ, entonces $\sigma(x) = \tau(x)$, si x aparece más de 3 veces, entonces $\sigma(x) = \tau(x_1) = ... = \tau(x_\ell)$. Las valuaciones σ y τ le asignan el mismo valor de verdad a cada cláusula de φ, luego, como τ satisface a α_φ, se tiene que σ satisface a φ. Concluimos que $\langle \varphi \rangle \in \text{CNF-SAT}$.

Dada la correctitud de la reducción, tenemos que como $\text{CNF}_3\text{-SAT}$ pertenece a la clase de problemas NP y es NP-hard, por definición es NP-completo.