Práctica 6: Complejidad de Problemas

Compilado: 9 de noviembre de 2022

En esta guía los ejercicios están organizados por ejes temáticos. Aquellos ejercicios marcados con $\star$ forman el conjunto **mínimo** que recomendamos resolver para cubrir todo el temario que será evaluado en los exámenes parciales. Los ejercicios no marcados complementan el temario cubriendo distintas variantes la misma temática. Finalmente, los ejercicios marcados con † son de resolución **opcional** y profundizan en todos los conceptos vistos en la materia.

Certificados y autenticación

1. Para cada uno de los siguientes problemas, proponer un certificado de tamaño polinomial de cada instancia positiva que se pueda verificar en tiempo polinomial. En caso que el problema sea polinomial, el certificado debe ser distinto a la instancia. Describir claramente cómo funciona el verificador correspondiente.

   a) FACTORIZATION: dados naturales m y n, ¿m tiene un factor k con $1 < k < n$?

   b) MIN: dado un conjunto A de naturales y $k \in \mathbb{N}$, ¿mín$\{A\} \leq k$?

   c) $\star$ MAXFLOW: dada una red N con fuente s y sumidero t y un natural f, ¿N tiene un flujo de valor mayor o igual a f?

   d) $\star$ SHORTEST PATH (SP): dado un digrafo pesado G, dos vértices s y t, y un natural k ¿G tiene un recorrido de s a t de peso menor o igual a k?

   e) ELEMENTARY SP: dado un digrafo pesado G, dos vértices s y t, y un natural k, ¿G tiene un camino de s a t de peso menor o igual a k?

   f) $\star$ HAMPATH: dado un digrafo G y dos vértices s y t ¿G tiene un camino de s a t que pase por todos los vértices de G?

   g) TSP (Traveling salesman problem): dado un digrafo completo y pesado G y un natural k, ¿G tiene un ciclo de peso menor o igual a k que pase por todos los vértices de G?

   h) $\star$ k-CLIQUE: dado un grafo G, ¿G tiene un subgrafo completo de tamaño mayor o igual a k? (Nota: son infinitos problemas, uno por cada k.)

   i) $\star$ CLIQUE: dado un grafo G y un natural k, ¿G tiene un completo de tamaño mayor o igual a k?

   j) k-COLORING: dado un grafo G, ¿se puede particionar $V(G)$ en a lo sumo k conjuntos disjuntos de forma tal que toda arista de G incida en vértices de particiones diferentes? (Nota: son infinitos problemas, uno por cada k.)

   k) ISOMORPHISM: dados dos grafos G y H, ¿son G y H isomorfos?

   l) SUBGRAPH ISOMORPHISM: dados dos grafos G y H, ¿es H isomorfo a un subgrafo inducido de G?

   m) ¬SAT: dada una formula proposicional (en forma normal disyuntiva) ϕ, ¿existe una valuación que satisface $\neg\phi$?

2. Para cada uno de los siguientes problemas, proponer un certificado de tamaño polinomial de cada instancia negativa que se pueda verificar en tiempo polinomial. En caso que el problema sea polinomial, el certificado debe ser distinto a la instancia. Describir claramente cómo funciona el verificador correspondiente.

a) PRIME: dado un natural n, ¿es n primo?

b) SORTED: dada una secuencia A de naturales, ¿está A ordenada?

c) GIRTH: dado un grafo G y un valor k, ¿todos los ciclos de G tienen k o menos vértices?

d) 2-COLORING (i.e., ¿es G bipartito?).

e) $\star$ k-CLIQUE: ver ejercicio anterior.

f) $\star$ TAUTOLOGY: dada una formula proposicional (en forma normal disyuntiva) ϕ, ¿es ϕ una tautología?

g) † CLIQUE TRANSVERSAL: dado un grafo G y un conjunto de vértices T, ¿toda clique maximal de G tiene intersección no vacía con T?

h) † FACTORIZATION. **Ayuda:** utilizar que PRIME es polinomial y que el tamaño de la lista de factores primos de un número se puede codificar en espacio polinomial. No hace falta demostrar estos hechos.

3. † Proponer certificados **simples e intuitivos que no necesariamente tengan tamaño polinomial** para las instancias positivas de TAUTOLOGY y CLIQUE TRANSVERSAL. Describir claramente cómo funciona el verificador correspondiente. ¿Los certificados obtenidos tienen tamaño polinomial? ¿el verificador es polinomial en función del tamaño de la entrada del problema original?

4. Sea x una instancia de un problema de decisión Π y sea A un algoritmo para Π que tiene complejidad temporal $C\colon \mathbb{N} \to \mathbb{N}$ en peor caso. Proponer un verificador que en tiempo $O(C)$ demuestre que x es un certificado de sí misma, independientemente de si x es positiva o negativa. Al certificado x se lo llama certificado trivial. ¿Qué se puede concluir del problema de certificar instancias cuándo Π es polinomial? Considerar un problema Π que tiene un algoritmo polinomial A que es difícil de programar, ¿el certificador propuesto en este ejercicio es útil en la práctica o es solo un concepto teórico?

5. † Dado un texto A que describe un algoritmo sobre strings en lenguaje Python, una función total $f\colon \mathbb{N} \to \mathbb{N}$ codificada en Python, y un natural k, el problema COMPLEXITY (para Python) consiste en determinar si $A(x)$ termina en $kf(|x|)$ pasos elementales de Python para todo string x. En este problema se supone que tenemos un interprete de Python que cuenta la cantidad de pasos elementales que se realizan. Proponer un certificado para las instancias negativas de COMPLEXITY y un autenticador para el mismo. ¿El certificado tiene tamaño polinomial en función al tamaño de la entrada de COMPLEXITY? ¿El autenticador es polinomial en función del tamaño del certificado de COMPLEXITY?

Concepto de reducción polinomial

6. Considerar los siguientes problemas:

   - PATH: dado un digrafo G, dos vértices s y t y un natural k, ¿hay un recorrido en G de s a t cuya longitud sea menor o igual a k?
   - EVEN PATH: dado un digrafo G, dos vértices s y t y un valor k, ¿hay un recorrido en G de s a t cuya longitud sea par y menor o igual a k?

Dado un digrafo G, definimos G' como el digrafo que tiene dos copias (v, p) y (v, i) de cada vértice $v \in V(G)$ donde $(v, x) \to (w, y)$ es una arista de G' si y solo si $v \to w \in E(G)$ y $x \neq y$.

a) Demostrar que $\langle G, s, t, k\rangle$ es una instancia positiva de EVEN-PATH si y solo si $\langle G', (s, p), (t, p), k\rangle$ es una instancia positiva de PATH.

b) Mostrar que la reducción de EVEN-PATH a PATH implicada por el punto anterior es polinomial.

7. ⋆ Considerar los siguientes problemas:

   - 2-PARTITION: dado un conjunto de naturales X con $\text{mín}\{X\} > 2$, ¿existe $S \subseteq X$ tal que $\sum S = \sum X/2$?
   - 3-PARTITION: dado un conjunto de naturales $X = \{x_1, \ldots, x_{3n}\}$ con $\sum X = nt, 0 < \text{mín}\, X$ y $\text{máx}\, X < \lfloor t/2 \rfloor$, ¿se puede particionar X en n triplas que sumen t?
   - RECTANGLE PACKING: dado un rectángulo grande r y una familia de rectángulos pequeños $R = \{r_1, \ldots, r_n\}$, ¿se pueden ubicar todos los r_i dentro de r (permitiendo rotaciones) sin que se solapen? (Nota: un rectángulo se codifica en la computadora usando un par de números.)

   a) Dada una instancia X de 2-PARTITION, se define la instancia $\langle r, R\rangle$ de RECTANGLE PACKING donde r tiene base $\sum X/2$ y altura 2 y R tiene un rectángulo de base x y altura 1 para todo $x \in X$. Demostrar que X es una instancia positiva de 2-PARTITION si y solo si $\langle r, R\rangle$ es una instancia positiva de RECTANGLE PACKING.

   b) Dada una instancia X de 3-PARTITION, se define la instancia $\langle r, R\rangle$ de RECTANGLE PACKING donde r tiene base n y altura $t + 3n$ y R tiene un rectángulo de base 1 y altura $n + x$ para cada $x \in X$. Demostrar que X es una instancia positiva de 3-PARTITION si y solo si $\langle r, R\rangle$ es una instancia positiva de RECTANGLE PACKING.

   c) Mostrar que las reducciones implicadas por los puntos anteriores son polinomiales en función de los tamaños de las entradas.

8. Considerar el siguiente problema:

   - ISOMORPHISM (restringido): dados dos grafos G y H tales que G y H tienen la misma cantidad de vértices y aristas y las secuencias ordenadas de sus grados son iguales, ¿son G y H isomorfos?

   Proponer una reducción polinomial de ISOMORPHISM a su versión restringida. ¿Qué se puede concluir de ambas versiones si una de ellas es polinomial?

9. † Supongamos que contamos con un algoritmo polinomial A para resolver ¬SAT. Demostrar que el siguiente algoritmo resuelve TAUTOLOGY en tiempo polinomial: "dado una formula ϕ en DNF (forma normal disyuntiva), retornar verdadero si y solo si $A(\phi)$ retorna falso". Explicar por qué $\phi \to \phi$ no es una reducción polinomial de ¬SAT a TAUTOLOGY. En general, explicar por qué la identidad no es una reducción polinomial de un problema Π a su complemento a pesar de que "retornar verdadero si y solo si $A(x)$ retorna falso" es un algoritmo polinomial que resuelve $\overline{\Pi}$ para todo algoritmo polinomial A de Π. ¿Tiene sentido esta diferenciación?

10. Considerar el siguiente problema:

- CONNECTED: dado un digrafo G y dos vértices s y t, ¿G tiene un recorrido de s a t?

Para un digrafo G, sea H el digrafo que tiene un vértice (S, v) para cada $S \subseteq V(G)$ y cada $v \in V(G)$, donde $(S, v) \to (R, w)$ es una arista de H si y solo si $w \notin S$, $R = S \cup \{w\}$ y $v \to w$ es una arista de G.

a) Demostrar que la instancia $\langle G, s, t\rangle$ de HAMPATH es positiva si y solo si la instancia $\langle H, (\{s\}, s), (V(G), t)\rangle$ de CONNECTED es positiva.

b) Mostrar que la reducción de HAMPATH a CONNECTED implicada por el punto anterior **no** es polinomial.

11. † Dado un texto A que describe un algoritmo en Python y un string x que representa un input de A, el problema de HALTING (para Python) consiste en determinar si $A(x)$ termina.

a) Sea A el código Python de un algoritmo para resolver un problema Π. Escribir el algoritmo A^+ (resp. A^-) que dada una instancia x de Π termine si y solo si x es positiva (resp. negativa). ¿Cuál es el tamaño de A^+ y A^-?

b) Concluir que x es una instancia positiva (resp. negativa) de Π si y solo si $\langle A^+, x\rangle$ (resp. $\langle A^-, x\rangle$) es una instancia positiva de HALTING.

c) Describir las reducciones de Π y su problema complemento a HALTING. ¿Estas reducciones son polinomiales?

Relaciones entre clases de complejidad

12. ⋆ Determinar cuáles de las siguientes afirmaciones son verdaderas y cuáles falsas. Demostrar aquellas que son verdaderas y dar contraejemplos para aquellas que son falsas.

a) $\mathsf{P} \subseteq \mathsf{NP}$ y $\mathsf{P} \subseteq \mathsf{coNP}$.

b) Si $\mathsf{P} = \mathsf{NP}$, entonces $\mathsf{coNP} = \mathsf{NP}$.

c) Si $\mathsf{P} = \mathsf{NP}$, entonces todos los problemas computacionales pertenecen a P.

d) Si $\mathsf{coNP} = \mathsf{NP}$, entonces $\mathsf{SAT} \in \mathsf{coNP}$.

e) Si $\mathsf{coNP} \subseteq \mathsf{NP}$, entonces $\mathsf{NP} = \mathsf{coNP}$.

13. ⋆ ¿Es cierto que si dos problemas Π y Γ pertenecen a NPC entonces $\Pi \leq_p \Gamma$, y también $\Gamma \leq_p \Pi$? Justificar.

14. ⋆ Sean Π y Γ dos problemas de decisión tales que $\Pi \leq_p \Gamma$. ¿Qué se puede inferir?

a) Si $\Pi \in \mathsf{P}$ entonces $\Gamma \in \mathsf{P}$.

b) Si $\Gamma \in \mathsf{P}$ entonces $\Pi \in \mathsf{P}$.

c) Si $\Gamma \in \mathsf{NPC}$ entonces $\Pi \in \mathsf{NPC}$.

d) Si $\Pi \in \mathsf{NPC}$ entonces $\Gamma \in \mathsf{NPC}$.

e) Si $\Gamma \in \mathsf{NPC}$ y $\Pi \in \mathsf{NP}$ entonces $\Pi \in \mathsf{NPC}$.

f) Si $\Pi \in \mathsf{NPC}$ y $\Gamma \in \mathsf{NP}$ entonces $\Gamma \in \mathsf{NPC}$.

g) Π y Γ no pueden pertenecer ambos a NPC.

15. Decir si las siguientes afirmaciones son verdaderas o falsas:

    a) ⋆ Si P = NP, entonces todo problema NP-completo es polinomial.

    b) ⋆ Si P = NP, entonces todo problema NP-hard es polinomial.

    c) ⋆ Si las clases NP-completo y coNP-completo son disjuntas entonces P ≠ NP.

    d) † HALTING es NP-hard y coNP-hard.

    e) † Si HALTING está en P, entonces P = NP.

    f) † Si HALTING está en P, entonces todo problema computable es polinomial.

16. Suponiendo que P = NP, diseñar un algoritmo polinomial que dado un grafo G retorne un completo de tamaño máximo de G.

Problemas NP-completos

17. ⋆ Sabiendo que CLIQUE es NP-completo, demostrar que SUBGRAPH ISOMORPHISM es NP-completo.

18. ⋆ Sabiendo que HAMPATH es NP-completo, demostrar que ELEMENTARY SHORTEST PATH es NP-completo.

19. ⋆ El problema DOUBLE-SAT consiste en deteminar si una formula proposicional ϕ tiene al menos dos valuaciones que la satisfacen. Demostrar que DOUBLE-SAT es NP-completo.

20. ⋆ Una biclique B de un grafo G es un conjunto de vértices que induce un subgrafo bipartito completo de G. Dados dos grafos G y H, el grafo $G + H$ se obtiene agregando todas las aristas entre $V(G)$ y $V(H)$ en $G \cup H$. Dado un grafo G y un natural k, el problema BICLIQUE consiste en determinar si G tiene una biclique de cardinal mayor o igual a k.

    a) Demostrar que un grafo G tiene una clique de tamaño k si y solo si $\overline{G}+\overline{G}$ tiene una biclique de tamaño $2k$.

    b) Demostrar que BICLIQUE es NP-completo sabiendo que CLIQUE es NP-completo.

21. El problema HALF-CLIQUE consiste en determinar si un grafo G de tamaño n tiene un completo de tamaño $n/2$. Sabiendo que CLIQUE es NP-completo, demostrar que HALF-CLIQUE es NP-completo. ¿Por qué este resultado no contradice el hecho de que k-CLIQUE es polinomial para todo k?

22. Un *conjunto independiente* de un grafo G es un subconjunto S de vértices tal que $vw \notin E(G)$ para todo $v, w \in S$. Un *cubrimiento por vértices* de G es un subconjunto C de vértices tal que toda arista de G incide en algún vértice de C. Un *conjunto dominante* de G es un conjunto de vértices D tal que todo vértice de G pertenece a D o es adyacente a un vértice de D. Considerar los siguientes problemas:

    - INDEPENDENT SET: dado un grafo G y un natural k, ¿G tiene un conjunto independiente de tamaño mayor o igual a k?
    - VERTEX COVER: dado un grafo G con al menos una arista y un natural k, ¿G tiene un cubrimiento por vértices de tamaño menor o igual a k?

- DOMINATING SET: dado un grafo G y un natural k, ¿G tiene un conjunto dominante de tamaño menor o igual a k?

Sabiendo que CLIQUE es NP-completo, demostrar que todos los problemas anteriores son NP-completos. Para ello, puede utilizar las siguientes ayudas una vez demostradas.

- Demostrar que K es un completo de G si y solo si K es un conjunto independiente de $\overline{G}$.
- Demostrar que S es un conjunto independiente de G si y solo si $V(G) \setminus S$ es un cubrimiento por vértices de G.
- Sea H el grafo que se obtiene de G en dos pasos. Primero, se eliminan todos los vértices aislados de G. Luego, se agrega un vértice z_{vw} adyacente a v y a w para toda arista vw de G. Demostrar que (G, k) es una instancia positiva de VERTEX COVER si y solo si (H, k) es una instancia positiva de conjunto dominante.

23. † Demostrar que TAUTOLOGY es coNP-completo. Ayuda: reducir SAT, recordando que para demostrar que Π es coNP-hard, alcanza con demostrar que su complemento es NP-hard.

Resolución de problemas NP-completos con Backtracking (Opcional)

24. ⋆ Proponer un algoritmo de backtracking para SAT que incluya una poda por factibilidad y esté basado en la siguiente estrategia:

- Las variables de ϕ tienen un orden prefijado $v_1, \ldots, v_n$.
- Las soluciones válidas son las valuaciones que satisfacen ϕ.
- Las soluciones parciales son las valuaciones que asignan un prefijo $v_1, \ldots, v_i$ de las variables.
- Proponer una poda por factibilidad.

25. ⋆ Proponer un algoritmo de backtracking para CLIQUE que incluya una poda por optimalidad y esté basado en la siguiente estrategia:

- Las soluciones válidas son las cliques maximales de G.
- Las soluciones parciales son pares (K, V) donde K es un completo de G y $V \subseteq V(G) \setminus K$ es el conjunto de vértices adyacentes a todos los vértices de K.

26. ⋆ Proponer un algoritmo de backtracking para ISOMORPHISM que esté basado en la siguiente estrategia:

- Los vértices de H tienen un orden prefijado $w_1, \ldots, w_n$.
- Las soluciones válidas son las permutaciones $v_1, \ldots, v_n$ de $V(G)$ tales que $v_i \to w_i$ es un isomorfismo entre G y H.
- Las soluciones parciales son los ordenes $v_1, \ldots, v_k$ de los subconjuntos de $V(G)$ tales que $v_i \to w_i$ es un isomorfismo del subgrafo de G inducido por $v_1, \ldots, v_k$ al subgrafo de H inducido por $w_1, \ldots, w_k$.
- † Incluir una poda basada en la idea de que la cantidad de aristas que cruzan de $\{v_1, \ldots, v_k\}$ a $\{v_{k+1}, \ldots, v_n\}$ es la misma que las que cruzan de $\{w_1, \ldots, w_k\}$ a $\{w_{k+1}, \ldots, w_n\}$.

27. Proponer un algoritmo de backtracking para TSP que esté basado en la siguiente estrategia:

- El ciclo empieza y termina un vértice prefijado llamado v.
- Las soluciones válidas son todos los recorridos que empiezan y terminan en v y pasan por cada $w \in V \setminus \{v\}$ exactamente una vez.
- Las soluciones parciales son todos los recorridos que empiezan en el v, pasan por cada $w \in V \setminus \{v\}$ a lo sumo una vez, y visitan nuevamente v sólo si ya pasaron por todos los vértices.
- † Proponer una poda basada en la siguiente idea: si un camino C de longitud k que finaliza en un vértice w y visita un subconjunto de vértices W tiene un costo mayor que otro camino de longitud k que finaliza en w y visita W, entonces se pueden descartar todos los caminos que contengan a C como subcamino. (El algoritmo resultante es un algoritmo de programación dinámica.)

28. † Proponer un algoritmo de backtracking para ELEMENTARY SHORTEST PATH que esté basado en la siguiente estrategia:

    - Las soluciones válidas son todos los caminos que empiezan en s, terminan en t, y visitan cada vértice a lo sumo una vez.
    - Las soluciones parciales son todos los caminos que empiezan en s, visitan cada vértice a lo sumo una vez y no contienen a t en su interior.
    - Proponer una poda basada en la siguiente idea: si un camino C que finaliza en un vértice w y visita un subconjunto de vértices W tiene un costo mayor que otro camino que finaliza en w y visita $W' \subseteq W$, entonces se pueden descartar todos los caminos que contengan a C como subcamino. (El algoritmo resultante es un algoritmo de programación dinámica.)

29. † Proponer un algoritmo de backtracking para SAT que esté basado en la siguiente estrategia. Compararlo con el algoritmo del Ejercicio 24.

    - Decimos que un conjunto de literales X es *factible* cuando $\overline{x} \notin X$ para todo $x \in X$.
    - La X-simplificación de ϕ, para X factible, se obtiene en dos pasos. Primero, se eliminan todas las cláusulas de ϕ que contengan algún $x \in X$. Luego, se elimina a $\overline{x}$ de todas las cláusulas restantes, para todo $x \in X$.
    - Decimos que X *satisface* ϕ cuando la X-simplificación de ϕ es vacía (i.e., no tiene cláusulas) y que X *contradice* ϕ cuando la X-simplificación de ϕ tiene alguna cláusula vacía.
    - Decimos que X es *maximal* cuando ninguna cláusula de la X-simplificación de ϕ tiene exactamente un literal.
    - Las soluciones válidas del algoritmo son los conjuntos de literales factibles que satisfacen ϕ.
    - Las soluciones parciales son los conjuntos maximales de literales factibles que no contradicen a ϕ.