Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2213 — Lógica para ciencia de la computación — 1' 2023

TAREA 4

Publicación:	Martes 9 de mayo.
Entrega:	**Lunes 22 de mayo hasta las 23:59 horas**.

Indicaciones

- Cada pregunta tiene 6 puntos (+1 base) y la nota de la tarea es el promedio de las preguntas.
- La solución debe estar escrita en LaTeX. No se aceptarán tareas escritas de otra forma.
- La tarea es individual, pudiendo discutirla con sus pares. Toda referencia externa debe citarse.

Objetivos

- Aplicar el concepto de NP-completitud.
- Modelar propiedades con fórmulas en lógica de primer orden.
- Construir $\mathcal{L}$-estructuras con restricciones.

Pregunta 1: Problemas NP-completos

(a) Demuestre que si existe un lenguaje finito no vacío que es NP-completo, entonces P=NP.

(b) ¿Es NP-completo el siguiente lenguaje? Demuestre su respuesta.

$$\text{NEQ} = \{(\varphi_1, \varphi_2) \mid \varphi_1 \text{ y } \varphi_2 \text{ son fórmulas proposicionales no equivalentes}\}$$

Solución

Parte (a)

Supongamos que tenemos un lenguaje finito $L = \{w_1, \ldots, w_n\}$, entonces podemos construir una máquina de Turing $\mathcal{M}$ que revise cualquier input w n veces de tal manera que en la $j-$ésima revisión decida si el input es igual a w_j. Esto claramente se puede realizar en $n|w|$ pasos, es decir, L es un lenguaje en P. Luego, si L fuera NP-completo, tenemos que un lenguaje lineal es NP-completo, por lo que todo lenguaje NP es polinomial, obteniendo N=NP.

2 pts. por argumentar por qué el lenguaje está en NP y 1 pt. por concluir correctamente lo pedido.
La máquina no tiene por qué estar construida tan formalmente.

Parte (b)

Este lenguaje sí es NP-completo, para probarlo mostraremos que es NP-hard y que está en NP.

Definimos la función $f : A^* \to A^*$ como la función tal que $f(w) = 0$ si w no es una fórmula proposicional y $f(w) = (w, \varphi)$ con φ una contradicción si w es una fórmula proposicional. Probemos que esto es una reducción de SAT a NEQ.

1. f es computable: Notemos que para computar f solo hay que revisar el formato y luego borrar el input y poner 0 o concatenar el input con φ, por lo tanto es computable en tiempo polinomial.

2. $w \in$ SAT $\Rightarrow f(w) \in$ NEQ: Si tenemos $w \in$SAT, entonces sabemos que existe una valuación σ tal que $\sigma(w) = 1$. Como φ es una contradicción, sabemos que $\sigma(varphi) = 0$, por lo tanto w y φ no son equivalentes y $(w, \varphi) \in$ NEQ.

3. $w \notin$ SAT $\Rightarrow f(w) \notin$ NEQ: Si w no es una fórmula proposicional, esto se tiene trivialmente. Si w es una fórmula porposicional no satisfacible, entonces para toda valuación σ, se tiene que $\sigma(w) = 0 = \sigma(\varphi)$, entonces tenemos que w y φ son equivalentes y $(w, \varphi) \notin$ NEQ.

Probamos así que SAT $\leq_f$ NEQ, entonces NEQ es NP-hard.

Para probar que es NP, podríamos construir una máquina que probara valuaciones, pero para repasar nuevamente reducciones, se presenta una reducción que también demuestra lo pedido. Definimos la función $g : A^* \to A^*$ tal que $g(w) = 0$ si w no está en el formato adecuado y $g((\varphi_1, \varphi_2)) = \varphi_1 \wedge \neg\varphi_2$. Probemos que es reducción:

1. g es computable: Análogo a argumento para f.

2. $w \in$ NEQ $\Rightarrow g(w) \in$ SAT: Si $w \in$ NEQ, entonces existe una valuación tal que $\sigma(\varphi_1) = 1$ y $\sigma(\varphi_2) = 0$, ya que no son equivalentes, entonces $\sigma(\varphi_1 \wedge \neg\varphi_2) = 1$, por lo tanto $g(w) \in$ SAT.

3. $w \notin$ NEQ $\Rightarrow g(w) \notin$ SAT: Si w no es una fórmula proposicional, esto se tiene trivialmente. Si $w = (\varphi_1, \varphi_2) \notin$ NEQ entonces para toda valuacuón σ, tenemos $\sigma(\varphi_1) = \sigma(\varphi_2)$, por lo tanto $\sigma(\varphi_1 \wedge \neg\varphi_2) = 0$, entonces $g(w) \notin$ SAT.

Tenemos así que NEQ $\leq_g$ SAT, entonces NEQ está en NP.

1,5 pts. por probar NP-hard y 1,5 pts. por probar que el lenguaje es NP. Para esto probablemente la mayoría construya la máquina no determinista, cualquiera de las dos opciones de resolución obtienen puntaje completo. Además no importa si eligen otro lenguaje mientras que la reducción sea correcta.

Pregunta 2: Fórmulas en lógica de primer orden

En un grafo dirigido $G = (N, A)$, un camino dirigido de n nodos es una secuencia $u_0, \ldots, u_n$ tal que

- $u_i \in N$, para todo $0 \leq i \leq n$
- $(u_i, u_{i+1}) \in A$, para todo $0 \leq i < n$

Un ciclo simple dirigido de n nodos es un camino dirigido tal que $u_0 = u_n$ y todos los demás nodos deben ser distintos.

Sea el vocabulario $\mathcal{L} = \{E\}$ con símbolo de relación binaria E. Construya $\mathcal{L}$-oraciones en lógica de primer orden que representen las siguientes propiedades.

(a) El grafo es un clique. Utilice para esto la noción de clique dirigido en que deben existir todas las aristas dirigidas posibles.

(b) El grafo contiene un ciclo simple dirigido de 3 nodos.

Observe que cada oración deben ser satisfecha por una $\mathcal{L}$-estructura $\mathfrak{A}$ si, y solo si, el grafo representado por $\mathfrak{A}$ cumple la propiedad modelada con dicha oración.

Solución

Parte (a)

Una oración que cumple lo pedido sería

$$\varphi = \forall u. \forall v.\ E(u, v)$$

Claramente si se satisface φ, entonces todas las combinaciones posibles de nodos forman aristas. Es decir, el grafo es un clique.

> 3 pts. por formar la oración. La justificación no es necesaria, pero en caso que la oración esté incorrecta, pero haya una justificación razonable igual se otorgará puntaje.

Parte (b)

Una oración que representa la propiedad pedida es

$$\varphi = \exists u. \exists v. \exists w.\ E(u, v) \land E(v, w) \land E(w, u) \land \neg(u = v) \land \neg(v = w) \land \neg(w = u)$$

Si se satisface φ, entonces tenemos que existen tres nodos tales que existen aristas dirigidas entre ellos formando un ciclo y además los nodos son distintos.

> 3 pts. por formar la oración correctamente (2 pts. si olvidan mencionar que los nodos deben ser diferentes). Si la oración es incorrecta, tratar igual que en (a).

Pregunta 3: Estructuras en lógica de primer orden

Sea $\mathcal{L} = \{c, f\}$ un vocabulario con símbolo de constante c y símbolo de función unaria f. Se definen las siguientes $\mathcal{L}$-oraciones

$$\begin{aligned}
\varphi_1 &= \forall x \forall y\ (f(x) = f(y) \rightarrow x = y) \\
\varphi_2 &= \forall x\ (f(x) \neq c) \\
\varphi_3 &= \forall x\ (x \neq c \rightarrow \exists y\ f(y) = x)
\end{aligned}$$

Construya $\mathcal{L}$-estructuras $\mathfrak{A}_1$, $\mathfrak{A}_2$ y $\mathfrak{A}_3$ tales que

(a) $\mathfrak{A}_1 \models \varphi_1 \wedge \varphi_2 \wedge \varphi_3$

(b) $\mathfrak{A}_2 \models (\neg\varphi_1) \wedge \varphi_2 \wedge \varphi_3$

(c) $\mathfrak{A}_3 \models \varphi_1 \wedge \varphi_2 \wedge (\neg\varphi_3)$

y justifique por qué satisfacen las fórmulas especificadas. No necesita entregar una demostración formal de que cada estructura satisface cada fórmula.

Solución

Parte (a)

Definimos la $\mathcal{L}$-estructura $\mathfrak{A}_1 = \langle \mathbb{N}_{>0}, c^{\mathfrak{A}_1}, f^{\mathfrak{A}_1} \rangle$ donde la constante es el 1 y la función unaria es la de sucesor. Notemos que cumple la oración ya que la función de sucesor es inyectiva (cumple φ_1), no existe un antecesor del 1 (cumple φ_2) y todo número distinto de 1 tiene un antecesor (cumple φ_3). Es decir, $\mathfrak{A}_1$ satisface $\varphi_1 \wedge \varphi_2 \wedge \varphi_3$.

Parte (b)

Definimos la $\mathcal{L}$-estructura $\mathfrak{A}_2 = \langle \{\bigstar, \heartsuit\}, c^{\mathfrak{A}_2}, f^{\mathfrak{A}_2} \rangle$ donde la constante es $\bigstar$ y la función unaria es la función constante $\heartsuit$. Notemos que esto no satisface φ_1 ya que $f(\bigstar) = f(\heartsuit) = \heartsuit$, pero $\bigstar \neq \heartsuit$. Además, $f(\bigstar) = f(\heartsuit) \neq \bigstar$, por lo tanto se satisface φ_2 y como 1 es el único elemento no 0, se satisface φ_3. Es decir $\mathfrak{A}_2$ satisface lo pedido

Parte (c)

Definimos la $\mathcal{L}$-estructura $\mathfrak{A}_3 = \langle \{☺\}^*, c^{\mathfrak{A}_3}, f^{\mathfrak{A}_3} \rangle$ donde la constante es la palabra ☺, y la función es concatenar la palabra consigo misma. Esta función es inyectiva, por lo que se satisface φ_1. Claramente ☺ no tiene preimagen, ya que es de largo impar y al concatenar una palabra consigo misma siempre resulta una palabra de largo par, por lo que se satisface φ_2. Además por el mismo argumento tenemos que no existe preimagen para ☺☺☺ y ☺ $\neq$ ☺☺☺, por lo tanto no se satisface φ_3. En conclusión $\mathfrak{A}_3 \models \varphi_1 \wedge \varphi_2 \wedge (\neg\varphi_3)$.

En cada ítem se da 1 pt. por mostrar una estructura que satisface lo pedido y 1 pt. por justificar bien por qué es así. No es necesario que sea muy formal, pero sí que se mencionen las tres fórmulas.