Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC1253 - Matemáticas Discretas

Examen

27 de Noviembre de 2015
Profesores: Gabriel Diéguez - Fernando Suárez

Pregunta 1

a) Sea Σ un conjunto de fórmulas en $L(P)$. Demuestre que Σ es inconsistente si y sólo si $\Sigma \models \Box$.

b) Considere el siguiente predicado:

$$R(x, y) : \text{“}\textit{x se relaciona con y}\text{”}$$

Para cada una de las siguientes afirmaciones encuentre una fórmula que utilice sólo el predicado $R(\cdot, \cdot)$, tal que la fórmula sea satisfacible sólo por una estructura que cumpla la afirmación.

(i) R es una relación de equivalencia.

(ii) R es un orden total

(iii) R es una función biyectiva

Solución

a) ($\Rightarrow$) Dado que Σ es inconsistente, debemos demostrar que $\Sigma \models \Box$. Como Σ es inconsistente, sabemos que toda valuación σ es tal que $\sigma(\Sigma) = 0$, y luego se cumple trivialmente que $\Sigma \models \Box$.

($\Leftarrow$) Dado que $\Sigma \models \Box$, debemos demostrar que Σ es inconsistente. Por contradicción, supongamos que Σ es satisfacible. Luego, existe una valuación σ tal que $\sigma(\Sigma) = 1$. Como $\Box$ es una contradicción, tenemos que $\sigma(\Box) = 0$, y por lo tanto obtenemos que $\sigma(\Sigma) = 1$ pero $\sigma(\Box) = 0$, lo que contradice que $\Sigma \models \Box$. $\square$

Pauta

- 1.5 pts. por ($\Rightarrow$)
- 1.5 pts. por ($\Leftarrow$)
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

b) Consideremos las siguientes fórmulas:

- $\varphi_{\text{refleja}} = \forall x(R(x,x))$
- $\varphi_{\text{simétrica}} = \forall x \forall y(R(x,y) \rightarrow R(y,x))$
- $\varphi_{\text{transitiva}} = \forall x \forall y \forall z(R(x,y) \wedge R(y,z) \rightarrow R(x,z))$
- $\varphi_{\text{antisimétrica}} = \forall x \forall y(R(x,y) \wedge R(y,x) \rightarrow (x=y))$
- $\varphi_{\text{conexa}} = \forall x \forall y(R(x,y) \vee R(y,x))$
- $\varphi_{\text{función}} = \forall x \forall y \forall z(R(x,y) \wedge R(x,z) \rightarrow (y=z))$
- $\varphi_{\text{total}} = \forall x \exists y(R(x,y))$
- $\varphi_{\text{inyectiva}} = \forall x \forall y \forall z(R(x,z) \wedge R(y,z) \rightarrow (x=y))$
- $\varphi_{\text{sobreyectiva}} = \forall y \exists x(R(x,y))$

Luego, para cada una de las propiedades definimos las fórmulas:

- $\varphi_{\text{equivalencia}} = \varphi_{\text{refleja}} \wedge \varphi_{\text{simétrica}} \wedge \varphi_{\text{transitiva}}$
- $\varphi_{\text{orden total}} = \varphi_{\text{refleja}} \wedge \varphi_{\text{antisimétrica}} \wedge \varphi_{\text{transitiva}} \wedge \varphi_{\text{conexa}}$
- $\varphi_{\text{función biyectiva}} = \varphi_{\text{función}} \wedge \varphi_{\text{total}} \wedge \varphi_{\text{inyectiva}} \wedge \varphi_{\text{sobreyectiva}}$

Pauta

- 1 pto. por $\varphi_{\text{equivalencia}}$
- 1 pto. por $\varphi_{\text{orden total}}$
- 1 pto. por $\varphi_{\text{función biyectiva}}$
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

Pregunta 2

Para cada una de las siguientes relaciones, demuestre que es de equivalencia y encuentre el índice[1] de la relación:

a) Sea $\mathbb{W} = \{(x,y) \in \mathbb{R}^2 \mid x \neq 0 \wedge y \neq 0\}$. Definimos la relación $\sim \subseteq \mathbb{W} \times \mathbb{W}$:

$$(a_1, a_2) \sim (b_1, b_2) \Leftrightarrow a_1 b_1 > 0 \wedge a_2 b_2 > 0$$

b) Sea $\mathbb{C}$ el conjunto de los números complejos. Definimos la relación $\sim \subseteq \mathbb{C} \times \mathbb{C}$:

$$z_1 \sim z_2 \Leftrightarrow ||z_1|| = ||z_2||$$

[1] Cardinalidad del conjunto cuociente inducido por la relación.

c) Sea $\mathbb{Z}$ el conjunto de los números enteros. Definimos la relación $\sim \subseteq \mathbb{Z} \times \mathbb{Z}$:

$$n_1 \sim n_2 \Leftrightarrow 2015 \mid (n_2 - n_1)$$

Pregunta 3

a) Sean $f(n) = \sum_{i=0}^{n} i!$ y $g(n) = (n+1)!$. ¿Es cierto que $f \in \mathcal{O}(g)$? Demuestre.

b) Sean $a, m, n \in \mathbb{Z}$. Demuestre que $a^m \bmod n = (a \bmod n)^m \bmod n$.

Solución

a) Basta notar que

$$\sum_{i=0}^{n} i! \leq \sum_{i=0}^{n} n!$$

Desarrollando este último resultado:

$$\sum_{i=0}^{n} n! = (n+1) \cdot n! = (n+1)!$$

Luego, tomando $c = 1$ y $n_0 = 0$, se cumple que $\forall n \geq n_0$, $f(n) \leq c \cdot g(n)$, y por lo tanto $f \in \mathcal{O}(g)$.$\square$

Pauta

- 1 pto. por acotar.
- 1 pto. por desarrollar.
- 1 pto. por concluir.
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

b) Sabemos que:

$$a \equiv_n b \text{ si y solo si } a \bmod n = b \bmod n \tag{1}$$
$$a \equiv_n a \bmod n \tag{2}$$
$$\text{Si } a \equiv_n b \text{ y } c \equiv_n d, \text{ entonces } a \cdot c \equiv_n b \cdot d \tag{3}$$

Si aplicamos (3) usando (2) dos veces, obtenemos que $a^2 \equiv_n (a \bmod n)^2$. Si hacemos esto m veces, tenemos que

$$a^m \equiv_n (a \bmod n)^m \tag{4}$$

Finalmente, aplicando (1) sobre (4), obtenemos que $a^m \bmod n = (a \bmod n)^m \bmod n$, que era lo que queríamos demostrar.$\square$

Pauta

- 1 pto. por aplicar (2) y (3).
- 1 pto. por generalizar lo anterior a m.
- 1 pto. por aplicar (1).
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

Pregunta 4

Sea G un grafo. Definimos el *diámetro* de G como el más largo de los caminos más cortos entre dos vértices de G.

Demuestre que no puede ser que G y $\overline{G}$ tengan ambos diámetro mayor que 3.

Solución

Denotaremos como $D(G)$ al diámetro del grafo G.
Si $D(G) \leq 3$, entonces el resultado se cumple. Supongamos entonces que $D(G) \geq 4$, y debemos mostrar que $D(\overline{G}) \leq 3$; es decir, que dos vértices cualquiera en $\overline{G}$ están conectados por un camino de largo ≤ 3. Dividiremos la demostración en dos casos:

- G no es conexo: en este caso, entre todos los vértices que están en componentes conexas distintas en G habrá una arista en $\overline{G}$ (y por lo tanto un camino de largo 1), mientras que los vértices que están en la misma componente conexa en G estarán ambos conectados en $\overline{G}$ a través de un vértice de otra componente conexa (y por lo tanto habrá un camino de largo 2 entre ellos). Luego, se cumple que $D(\overline{G}) \leq 3$, y más aún, se cumple que $D(\overline{G}) \leq 2$.
- G es conexo: si G es conexo y $D(G) \geq 4$, entonces existen dos vértices u y v en G conectados por un camino de largo ≥ 4, y tales que no existe un camino de largo < 4 entre ellos. Luego, es claro que en $\overline{G}$ habrá una arista entre u y v. Por otro lado, cualquier vértice distinto a u y v en G será adyacente a al menos uno de ellos en $\overline{G}$, pues no existe un camino de largo 2 entre u y v en G. Sean x e y dos vértices cualquiera de G distintos de u y v. Si en $\overline{G}$ ellos comparten a u o v como vecino, entonces xuy o xvy será un camino entre ellos en $\overline{G}$ (de largo 2). Si no, $xuvy$ o $xvuy$ será un camino entre ellos en $\overline{G}$ (de largo 3). En cualquier caso, dos vértices cualquiera de $\overline{G}$ están conectados por un camino de largo a lo más 3.$\square$

Pauta

- 0.5 pts. por el caso en que $D(G) \leq 3$.
- 2.5 pts. por el caso en que $D(G) \geq 4$ y G no es conexo.
- 3 pts. por el caso en que $D(G) \geq 4$ y G es conexo.
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

Pregunta 5

Considere el siguiente problema:

$$3\text{-SAT} = \{\varphi \in L(P)_{3\text{-CNF}} \mid \varphi \text{ es satisfacible}\}$$

En otras palabras, las instancias $I_{3\text{-SAT}}$ son todas las fórmulas en 3-CNF y el lenguaje $L_{3\text{-SAT}}$ son todas las fórmulas en 3-CNF sastisfacibles. Demuestre que 3-SAT es NP-completo.

Solución

En primer lugar, notemos que el problema 3-SAT $\in NP$. Basta con considerar la valuación como certificado(polinomial en literales) y luego evaluar la fórmula cláusula a cláusula. Esto puede realizarse en tiempo polinomial, ya que dada una fórmula ψ en FNC de m cláusulas y n proposiciones, podemos verificar si la valuación la satisface en $(\mathcal{O}(nm))$.

En segundo lugar, debemos mostrar que 3-SAT es NP-hard, reduciremos desde SAT. Dada una instancia de SAT con un conjunto $C = \{c_1, c_2, ..., c_m\}$ de cláusulas con literales $L = \{l_{1,1}, ..., l_{m,k_m}\}$. Consideremos la función $A : I_{\text{SAT}} \to I_{3\text{-SAT}}$:

$A(C)$

for $c_i \in C$ **do**
  if $|c_i| == 1$ **then**
    Caso 1
  end if
  if $|c_i| == 2$ **then**
    Caso 2
  end if
  if $|c_i| == 3$ **then**
    Caso 3
  end if
  if $|c_i| > 3$ **then**
    Caso 4
  end if
end for
return $C' = c'_1 \wedge ... \wedge c'_p$

Caso 1 Sea $c_i = \{l_{i,1}\}$. Usaremos dos variables adicionales $\{y_{i,1}, y_{i,2}\}$. Luego formamos el conjunto $c'_i = \{\{l_{i,1}, y_{i,1}, y_{i,2}\}, \{l_{i,1}, y_{i,1}, \bar{y}_{i,2}\}, \{l_{i,1}, \bar{y}_{i,1}, y_{i,2}\}, \{l_{i,1}, \bar{y}_{i,1}, \bar{y}_{i,2}\}\}$.

Caso 2 Sea $c_i = \{l_{i,1}, l_{i,2}\}$. Usaremos una variable adicional $\{y_{i,1}\}$. Luego formamos el conjunto $c'_i = \{\{l_{i,1}, l_{i,2}, y_{i,1}\}, \{l_{i,1}, l_{i,2}, \bar{y}_{i,1}\}\}$.

Caso 3 Sea $c_i = \{l_{i,1}, l_{i,2}, l_{i,3}\}$. No usaremos variables adicionales. Luego $c'_i = c_i$.

Caso 4 Sea $c_i = \{l_{i,1}, l_{i,2}, ..., l_{i,m_i}\}$ con $m_i > 3$. Usaremos las variables adicionales $\{y_{i,1}, y_{i,2}, ..., y_{i,m_i-3}\}$. Luego formamos el conjunto:

$$c'_i = \{\{l_{i,1}, l_{i,2}, y_{i,1}\}, \{\bar{y}_{i,1}, l_{i,3}, y_{i,2}\}, \{\bar{y}_{i,2}, l_{i,4}, y_{i,3}\}, \{\bar{y}_{i,3}, l_{i,5}, \bar{y}_{i,4}\}, .., \{\bar{y}_{i,m_i-3}, l_{i,m_i-1}, l_{i,m_i}\}\}.$$

Es claro que la transformación está acotada en el peor caso por $\mathcal{O}(nm)$, luego es polinomial. Sin embargo, debemos demostrar que la reducción es correcta, es decir

$$\varphi \in L_{\text{SAT}} \Leftrightarrow A(\varphi) \in L_{3\text{-SAT}}$$

($\Rightarrow$) Dado que φ es satisfacible, podemos limitarnos a analizar cada cláusula c_i. En primer lugar, notemos que $A(\varphi)$ es trivialmente satisfacible para los casos en que $|c_i| \leq 3$(podemos asignar cualquier valor a las variables auxiliares). Mientras que para los casos en que $|c_i| > 3$ se tiene que:

- Si $l_{i,1}$ o $l_{i,2}$ es verdadero, basta con asignar un valor de verdad negativo a todos los $y_{i,k}$. De esta manera el primer literal de todas las cláusulas es verdadero.
- Si l_{i,m_i-1} o l_{i,m_i} es verdadero, basta con asignar un valor de verdad positivo a todos los $y_{i,k}$. De esta manera el tercer literal de todas las cláusulas es verdadero.
- Si $l_{i,s}$ es verdadero, asignamos un valor de verdad positivo a cada yi, j con $j \leq s - 2$ y un valor de verdad negativo a cada yi, j con $j \geq s + 1$. De esta forma, el tercer literal de todas las cláusulas a la izquierda de $l_{i,s}$ serán verdaderos y el primer literal de todos los de la derecha también lo serán.

Finalmente, como c_i es arbitrario, todas las cláusulas son satisfechas.

($\Leftarrow$) En este caso ,dado que $A(\varphi)$ es satisfacible, podemos tomar la misma valuación pero acotada a las variables originales de φ para satisfacerla. Luego, φ es satisfacible.

□

Pauta

- 0.5 pts. dar certificado polinomial.
- 0.5 pts. por argumentar cómo evaluar el certificado en tiempo polinomial.
- 1 pts. por caso en que la cláusula tiene 1 literales. (**Caso 1**)
- 1 pts. por caso en que la cláusula tiene 2 literales.(**Caso 2**)
- 1 pts. por caso en que la cláusula tiene más de 3 literales. (**Caso 4**)
- 0.5 pts. por argumentar que la transformación es polinomial.
- 1 pts. por demostrar ($\Rightarrow$).
- 0.5 pts. por demostrar ($\Leftarrow$).
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

Pregunta 6

Considere el siguiente problema:

INDEPENDENT= $\{G \mid G$ es un grafo que contiene un conjunto independiente de tamaño $k\}$

En otras palabras, las instancias $I_{\text{INDEPENDENT}}$ son todos los grafos y el lenguaje $L_{\text{INDEPENDENT}}$ son todos los grafos que tienen un conjunto independiente de vértices de tamaño k. Demuestre que INDEPENDENT es NP-completo.

Solución

En primer lugar demostraremos que el problema esta en NP. Sea $G(V, E) \in L_{\text{INDEPENDENT}}$, es claro que si G está en el lenguaje, entonces G debe contener un conjunto independiente V'. Luego, podemos tomar el conjunto como certificado polinomial dado que $|V'|$ está acotado por $|V|$. Consideremos el siguiente algoritmo para verificar que V' es conjunto independiente en G:

```
checkINDEPENDENT(G(V,E), V')
for v ∈ V' do
    for u ∈ V' do
        if (u,v) ∈ E then
            return FALSE
        end if
    end for
end for
return TRUE
```

Es claro que el algoritmo corre es polinomial en $\mathcal{O}(|V'|^2)$, luego INDEPENDENT está en NP. Ahora sólo resta demostrar que INDEPENDENT es NP-hard, reduciremos desde CLIQUE. Consideremos la siguiente función $A : I_{\text{CLIQUE}} \to I_{\text{INDEPENDENT}}$:

```
A(G(V,E))
Sea G'(V,E')
for v ∈ V do
    for u ∈ V do
        if (u,v) ∉ E then
            E'.add((u,v))
        end if
    end for
end for
return G'
```

La función A complementa el grafo $G(V, E)$ en tiempo polinomial acotado por $\mathcal{O}(|V|^2)$. Finalmente, por teorema visto en clases sabemos que:

$$G(V, E) \text{ tiene un clique} \Leftrightarrow \bar{G}(V, \bar{E}) \text{ tiene un conjunto independiente}$$

Luego la reducción es correcta e INDEPENDENT es NP-hard. $\square$

Pauta

- 0.5 pts. por dar certificado polinomial.
- 1 pts. por dar el algoritmo para verificar el certificado en tiempo polinomial.
- 2.5 pts. por dar la reducción
- 0.5 pts. por argumentar que la reducción es polinomial.
- 1.5 pts. por demostrar que la reducción es correcta y concluir.
- Puntajes intermedios y demostraciones alternativas a criterio del corrector.

Pregunta Bonus (5 décimas)

Considere el siguiente problema:

$$\text{ALL-HALT} = \{p \mid p \text{ es un programa que se detiene ante cualquier input}\}$$

En otras palabras, las instancias $I_{\text{ALL-HALT}}$ son todos los programas y el lenguaje $L_{\text{ALL-HALT}}$ son todos los programas que terminan su ejecución (no entran en un loop infinito) ante todos los posibles inputs. Demuestre que no puede existir un algoritmo que decida a ALL-HALT.

Solución

Para demostrar que ALL-HALT es indecidible, basta con reducir desde HALTING. Sea $p \in I_{HALT}$ nuestra reducción A será generar el siguiente programa:

$M(x)$

```
if x == p then
    RUN p;
else
    Halt;
end if
```

Es claro que si p termina entonces M termina en todos los inputs. Además, si M termina en todos los inputs, en particular debe terminar cuando su input es p, luego p termina. Concluímos que

$$p \in L_{\text{HALT}} \Leftrightarrow A(p) \in L_{\text{ALL-HALT}}$$

Pauta

- 0.5 ptos en el examen por reducir y argumentar por qué la reducción es correcta.