Complejidad Computacional

Santiago Figueira

Departamento de Computación - FCEN - UBA

clase 11

Clase 11

Circuitos booleanos

La clase $\mathbf{P}_{/\mathbf{poly}}$

Tamaño y profundidad de circuitos

Circuitos booleanos

Circuitos booleanos

Definición

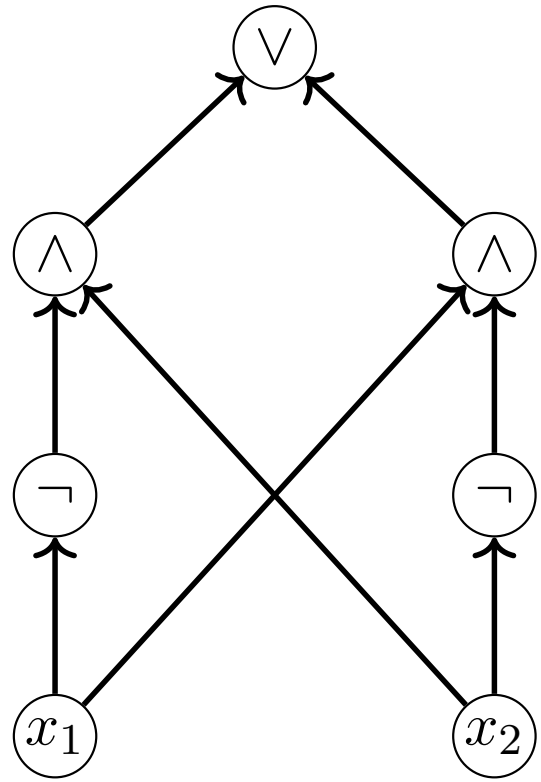
Un **circuito booleano** de n entradas es un grafo acíclico dirigido tal que

- tiene n **entradas** (*sources*) = nodos sin flechas de entrada
- tiene 1 nodo destino o **salida** (*sink*, sumidero) = nodo sin flecha de salida

El resto de los nodos se llaman compuertas (*gates*) y están etiquetados con

- $\neg$: *fan-in* =1, *fan-out* =1
- $\wedge$: *fan-in* =2, *fan-out* =1
- $\vee$: *fan-in* =2, *fan-out* =1

El **tamaño** $|C|$ de un circuito booleano C es la cantidad de vértices de C.



Entrada: x_1, x_2
Tamaño: 7

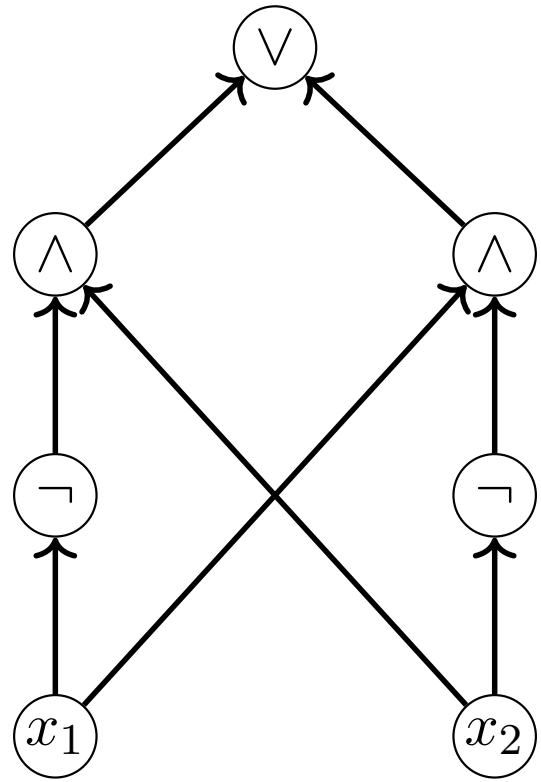
Semántica de circuitos booleanos

Definición

Dado un circuito booleano $C = (V, E)$ con entradas $X = \{x_1, \ldots, x_n\}$ y $x \in \{0,1\}^n$, definimos recursivamente $v : V \to \{0,1\}$ de esta forma:

- $v(x_i) = x(i)$
- si en C tenemos $u_1 \to u$ y u tiene *fan-in* $=1$ entonces $v(u) = \neg v(u_2)$ (u tenía etiqueta $\neg$)
- si en C tenemos $u_1 \to u \leftarrow u_2$, y
  - u tiene etiqueta $\wedge$, entonces $v(u) = v(u_1) \wedge v(u_2)$
  - u tiene etiqueta $\vee$, entonces $v(u) = v(u_1) \vee v(u_2)$

$C(x) = v(t)$, donde t es nodo de salida de C. El circuito C calcula la función $x \mapsto C(x)$.



Circuito C

Entrada: x_1, x_2

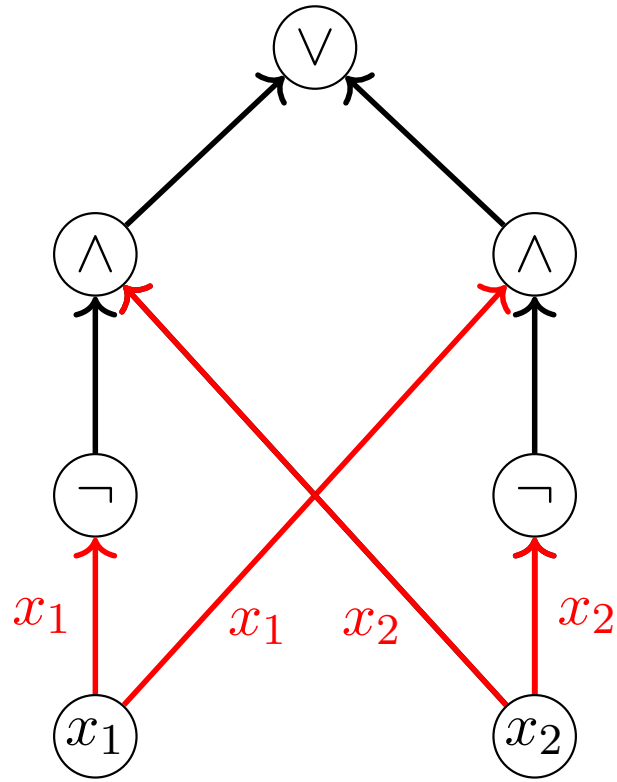
Semántica de circuitos booleanos

Definición

Dado un circuito booleano $C = (V, E)$ con entradas $X = \{x_1, \ldots, x_n\}$ y $x \in \{0,1\}^n$, definimos recursivamente $v : V \to \{0,1\}$ de esta forma:

- $v(x_i) = x(i)$
- si en C tenemos $u_1 \to u$ y u tiene *fan-in* $=1$ entonces $v(u) = \neg v(u_2)$ (u tenía etiqueta $\neg$)
- si en C tenemos $u_1 \to u \leftarrow u_2$, y
  - u tiene etiqueta $\wedge$, entonces $v(u) = v(u_1) \wedge v(u_2)$
  - u tiene etiqueta $\vee$, entonces $v(u) = v(u_1) \vee v(u_2)$

$C(x) = v(t)$, donde t es nodo de salida de C. El circuito C calcula la función $x \mapsto C(x)$.



Circuito C

Entrada: x_1, x_2

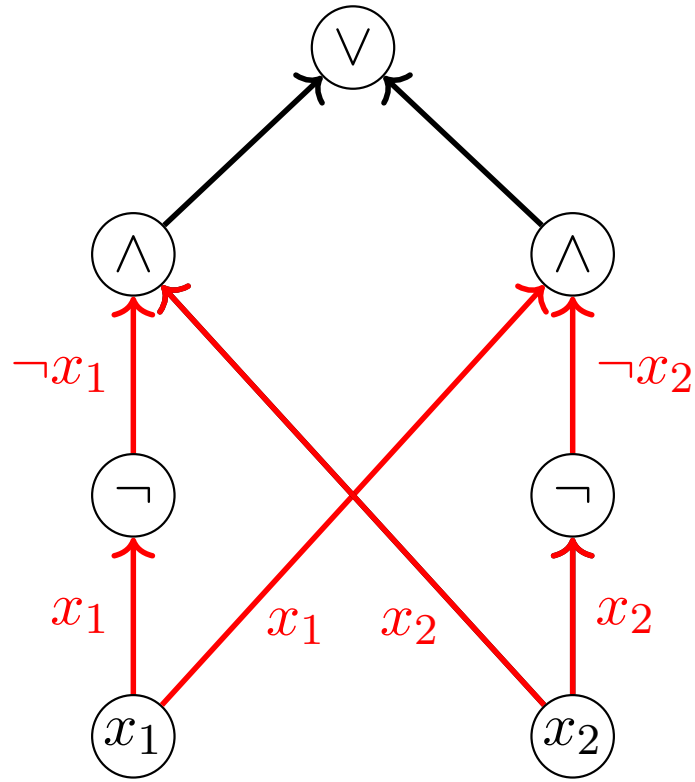
Semántica de circuitos booleanos

Definición

Dado un circuito booleano $C = (V, E)$ con entradas $X = \{x_1, \ldots, x_n\}$ y $x \in \{0,1\}^n$, definimos recursivamente $v : V \to \{0,1\}$ de esta forma:

- $v(x_i) = x(i)$
- si en C tenemos $u_1 \to u$ y u tiene *fan-in* $=1$ entonces $v(u) = \neg v(u_2)$ (u tenía etiqueta $\neg$)
- si en C tenemos $u_1 \to u \leftarrow u_2$, y
  - u tiene etiqueta $\wedge$, entonces $v(u) = v(u_1) \wedge v(u_2)$
  - u tiene etiqueta $\vee$, entonces $v(u) = v(u_1) \vee v(u_2)$

$C(x) = v(t)$, donde t es nodo de salida de C. El circuito C calcula la función $x \mapsto C(x)$.



Circuito C

Entrada: x_1, x_2

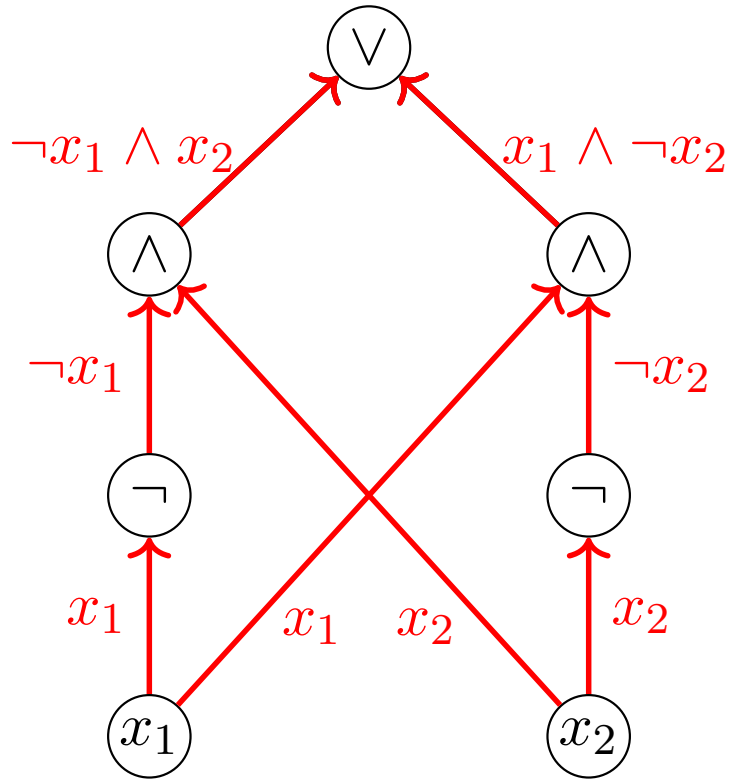
Semántica de circuitos booleanos

Definición

Dado un circuito booleano $C = (V, E)$ con entradas $X = \{x_1, \ldots, x_n\}$ y $x \in \{0,1\}^n$, definimos recursivamente $v : V \to \{0,1\}$ de esta forma:

- $v(x_i) = x(i)$
- si en C tenemos $u_1 \to u$ y u tiene *fan-in* $=1$ entonces $v(u) = \neg v(u_2)$ (u tenía etiqueta $\neg$)
- si en C tenemos $u_1 \to u \leftarrow u_2$, y
  - u tiene etiqueta $\wedge$, entonces $v(u) = v(u_1) \wedge v(u_2)$
  - u tiene etiqueta $\vee$, entonces $v(u) = v(u_1) \vee v(u_2)$

$C(x) = v(t)$, donde t es nodo de salida de C. El circuito C calcula la función $x \mapsto C(x)$.



Circuito C

Entrada: x_1, x_2

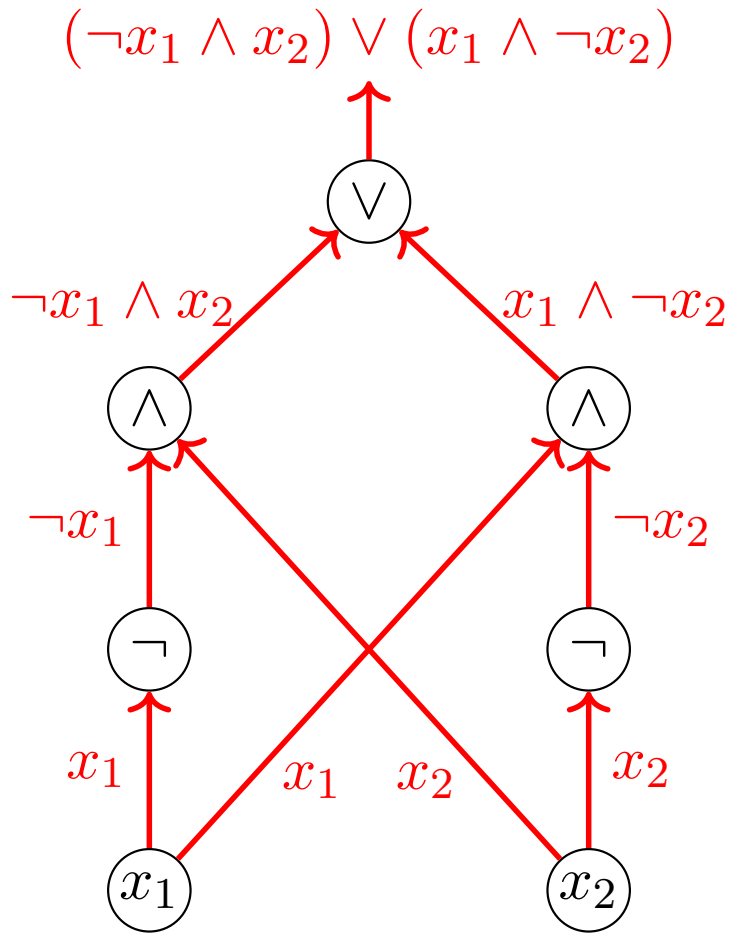
Semántica de circuitos booleanos

Definición

Dado un circuito booleano $C = (V, E)$ con entradas $X = \{x_1, \ldots, x_n\}$ y $x \in \{0,1\}^n$, definimos recursivamente $v : V \to \{0,1\}$ de esta forma:

- $v(x_i) = x(i)$
- si en C tenemos $u_1 \to u$ y u tiene *fan-in* $=1$ entonces $v(u) = \neg v(u_2)$ (u tenía etiqueta $\neg$)
- si en C tenemos $u_1 \to u \leftarrow u_2$, y
  - u tiene etiqueta $\wedge$, entonces $v(u) = v(u_1) \wedge v(u_2)$
  - u tiene etiqueta $\vee$, entonces $v(u) = v(u_1) \vee v(u_2)$

$C(x) = v(t)$, donde t es nodo de salida de C. El circuito C calcula la función $x \mapsto C(x)$.



Circuito C
Entrada: x_1, x_2
Salida: $C(x_1, x_2) =$
$x_1 \oplus x_2 =$
$(\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2)$

La clase $\textsc{Size}(S(n))$

Definición

Sea $S : \mathbb{N} \to \mathbb{N}$. Una **familia de circuitos de tamaño** $S(n)$ es una secuencia $(C_n)_{n\in\mathbb{N}}$ tal que

- C_n es un circuito con n entradas
- $|C_n| \leq S(n)$

La clase $\textsc{Size}(S(n))$

Definición

Sea $S : \mathbb{N} \to \mathbb{N}$. Una **familia de circuitos de tamaño** $S(n)$ es una secuencia $(C_n)_{n\in\mathbb{N}}$ tal que

- C_n es un circuito con n entradas
- $|C_n| \leq S(n)$

Definición

Una familia de circuitos $(C_n)_{n\in\mathbb{N}}$ [de tamaño $S(n)$] **decide** un lenguaje $\mathcal{L}$ si para todo $n \geq 1$ y todo $x \in \{0,1\}^n$

$$x \in \mathcal{L} \qquad \text{sii} \qquad C_n(x) = 1$$

Identificamos la tupla $(x_1, \ldots, x_n) \in \{0,1\}^n$ con la cadena $x_1 \ldots x_n$.

La clase $\textsc{Size}(S(n))$

Definición

Sea $S : \mathbb{N} \to \mathbb{N}$. Una **familia de circuitos de tamaño** $S(n)$ es una secuencia $(C_n)_{n\in\mathbb{N}}$ tal que

- C_n es un circuito con n entradas
- $|C_n| \leq S(n)$

Definición

Una familia de circuitos $(C_n)_{n\in\mathbb{N}}$ [de tamaño $S(n)$] **decide** un lenguaje $\mathcal{L}$ si para todo $n \geq 1$ y todo $x \in \{0,1\}^n$

$$x \in \mathcal{L} \qquad \text{sii} \qquad C_n(x) = 1$$

Identificamos la tupla $(x_1, \ldots, x_n) \in \{0,1\}^n$ con la cadena $x_1 \ldots x_n$.

- Observar que alcanza con que exista la familia de circuitos $(C_n)_{n\in\mathbb{N}}$.
- No pedimos (por ahora) que sea **uniforme**, es decir que exista una máquina determinística M tal que $M(n) = \langle C_n \rangle$.

La clase $\textbf{SIZE}(S(n))$

Clase de complejidad: $\textbf{SIZE}(S(n))$

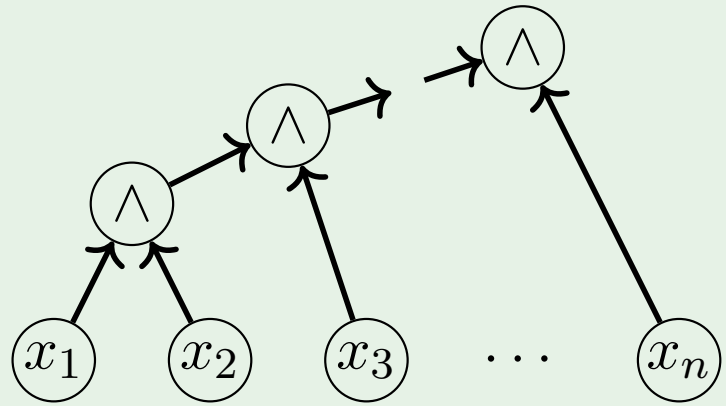
$\textbf{SIZE}(S(n))$ es la clase de lenguajes $\mathcal{L}$ ($\epsilon \notin \mathcal{L}$) tal que existe un familia de circuitos $(C_n)_{n\in\mathbb{N}}$ de tamaño $S(n)$ que decide $\mathcal{L}$.

Ejemplo: cualquier lenguaje unario

$$\mathcal{L} \subseteq \{1^n : \ \geq 1\}$$

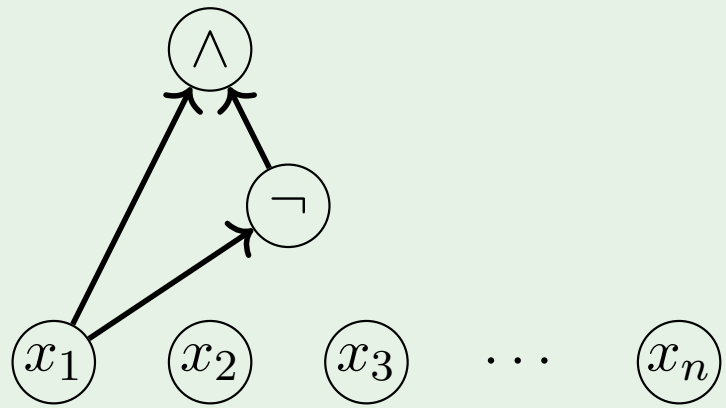
Si $x = x_1 \dots x_n \in \mathcal{L}$ definimos

$C_n =$



si no,

$C_n =$



$\mathcal{L} \in \textsc{Size}(S(n))$ para S lineal.

Ejemplo: suma bit a bit

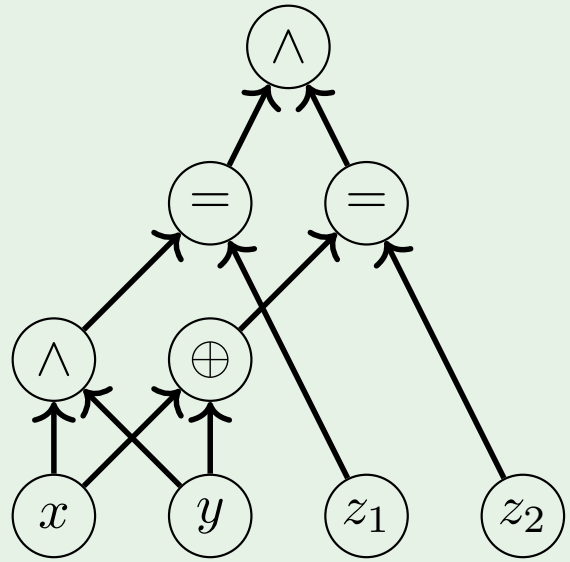
$$\mathcal{L} = \{xyz : |x| = |y| = n, |z| = n+1, x + y = z, n \geq 1\}$$

$0000, 0101, 1001, 1110 \in \mathcal{L}$
El resto de las palabras de largo 4 no están en $\mathcal{L}$
En general,

$$x_n \dots x_1 \; y_n \dots y_1 \; z_{n+1} \dots z_1 \in \mathcal{L}$$

sii

- $c_1 = 0$
- para $j = 1 \dots n$:
  $z_j = x_j \oplus y_j \oplus c_j$
  $c_{j+1} = (c_j \wedge (x_j \vee y_j)) \vee (x_j \wedge y_j))$
- $z_{n+1} = c_{n+1}$



C_4

- $\oplus$ representa el circuito para xor (o exclusivo)
- $=$ representa el circuito para el sii

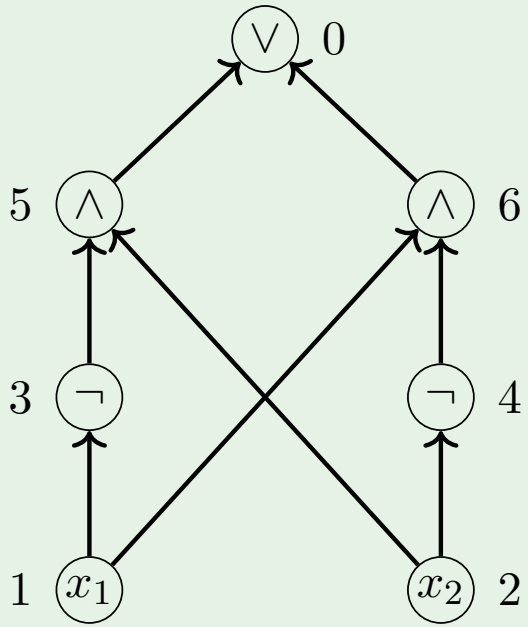
$\mathcal{L} \in \textsc{Size}(S(n))$ para S lineal.

Codificación de un circuito de tamaño S

Representamos C en binario:

- Supongamos nodos de C: $u_0, \ldots, u_{S-1}$ de modo tal que $u_1, \ldots u_n$ corresponden a la entrada y u_0 a la salida
- codificamos cada nodo u con $[u]$, es decir, con $\lceil \log S \rceil$ bits
- cada nodo de C tiene a lo sumo dos hijos (porque las operaciones son unarias o binarias)
- codificamos C con $\tau_0\sigma_0 \ \ldots \ \tau_{S-1}\sigma_{S-1}$, donde
  - si u_i es una entrada y por lo tanto no tiene hijos
    $\tau_i = 00$ y $\sigma_i = \varepsilon$
    (en realidad alcanza con codificar la cantidad de entradas)
  - si u_i tiene etiqueta $\neg$ (tiene solo 1 hijo v_i),
    $\tau_i = 01$ y $\sigma_i = [v_i]$
  - si u_i tiene etiqueta $\wedge$ y tiene hijos v_i, w_i,
    $\tau_i = 10$ y $\sigma_i = [v_i][w_i]$
  - si u_i tiene etiqueta $\vee$ y tiene hijos v_i, w_i,
    $\tau_i = 11$ y $\sigma_i = [v_i][w_i]$
- codificamos C con $\leq S \cdot (2 \cdot \lceil \log S \rceil + 2) \leq 4 \cdot S \cdot \log S$ bits para $S > 1$.

Ejemplo: codificación de un circuito



nodo u_i	tipo τ_i	hijos σ_i	
0	$\vee$	5	6
1	entrada		
2	entrada		
3	$\neg$	1	
4	$\neg$	2	
5	$\wedge$	3	2
6	$\wedge$	1	4

tipo τ_i	hijos σ_i	
11	101	110
00		
00		
01	001	
01	010	
10	100	010
10	001	010

11 101 110 00 00 01 001 01 010 10 100 010 10 001 010

(los espacios no forman parte de la codificación)

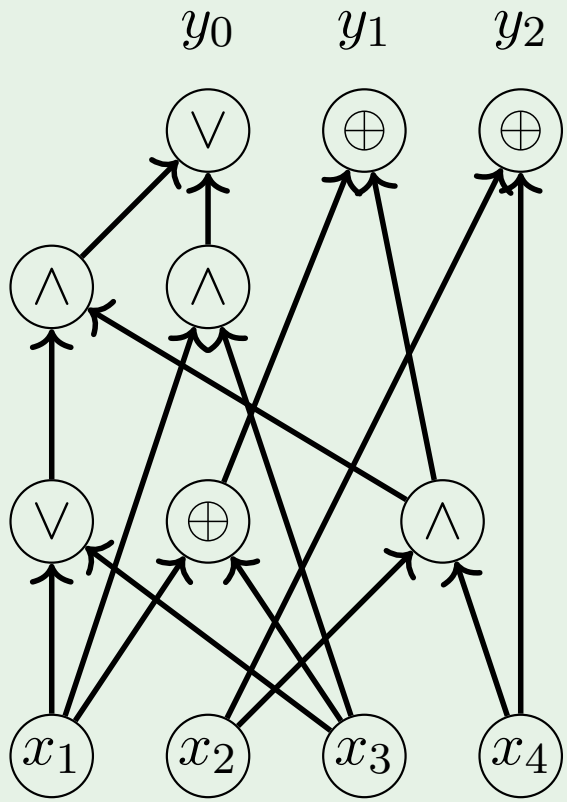
La clase $\mathbf{P}_{/\mathbf{poly}}$

Circuitos con salidas m-arias

Lo mismo que ya vimos sirve para representar funciones $\{0,1\}^n \to \{0,1\}^m$.

- hay n nodos de entrada (con *fan-in* $=0$)
- hay m nodos de salida (con *fan-out* $=0$)
- las representaciones son análogas a las que vimos solo que representamos todos los nodos de salida

Ejemplo



Calcula $x_1x_2x_3x_4 \mapsto y_0y_1y_2$, donde $y_0y_1y_2 = x_1x_2 + x_3x_4$ (suma en binario)

Repaso de la idea del teorema de Cook-Levin (una vez más)

Supongamos una máquina determinística M tal que

- M corre en tiempo $t(n)$
- M es *oblivious* y sin cinta de salida
- $x \in \mathcal{L}$ sii existe una secuencia de mini-configuraciones $z_0, \ldots, z_{t(|x|)}$ de M con entrada x tal que
  - z_0 es inicial para x
  - z_i evoluciona en un paso en z_{i+1}
  - $z_{t(|x|)}$ es final

Repaso de la idea del teorema de Cook-Levin (una vez más)

Supongamos una máquina determinística M tal que

- M corre en tiempo $t(n)$
- M es *oblivious* y sin cinta de salida
- $x \in \mathcal{L}$ sii existe una secuencia de mini-configuraciones $z_0, \ldots, z_{t(|x|)}$ de M con entrada x tal que
  - z_0 es inicial para x
  - z_i evoluciona en un paso en z_{i+1}
  - $z_{t(|x|)}$ es final
- cada mini-configuración tiene información sobre el estado, el bit leído de la cinta de trabajo y el bit leído de la cinta de entrada

Repaso de la idea del teorema de Cook-Levin (una vez más)

Supongamos una máquina determinística M tal que

- M corre en tiempo $t(n)$
- M es *oblivious* y sin cinta de salida
- $x \in \mathcal{L}$ sii existe una secuencia de mini-configuraciones $z_0, \ldots, z_{t(|x|)}$ de M con entrada x tal que
  - z_0 es inicial para x
  - z_i evoluciona en un paso en z_{i+1}
  - $z_{t(|x|)}$ es final
- cada mini-configuración tiene información sobre el estado, el bit leído de la cinta de trabajo y el bit leído de la cinta de entrada
- Recordar que, para $n = |x|$:
  - $|z_i| = k$, k depende solo depende de M
  - z_0 solo depende de x
  - z_{i+1} solo depende de 1 bit de x, z_i y $z_{prev(i,n)}$, donde $prev(i,n) = \text{máx}\{j < i : t(j,n) = t(i,n)\} \cup \{1\}$
    ($t(n,j)$ es la posición de la cabeza de trabajo al paso j)

La clase $\mathbf{P}_{/\mathbf{poly}}$

Clase de complejidad: $\mathbf{P}_{/\mathbf{poly}}$

$$\mathbf{P}_{/\mathbf{poly}} = \bigcup_{c} \textbf{SIZE}(c \cdot n^c)$$

Teorema

$\mathbf{P} \subseteq \mathbf{P}_{/\mathbf{poly}}$.

Teorema

$\mathbf{P} \subseteq \mathbf{P}_{/\mathbf{poly}}$.

Demostración

Sea $\mathcal{L} \in \mathbf{P}$ y sea M una máquina determinística tal que

- M corre en tiempo $t(n)$ con t un polinomio
- M es *oblivious*, con 1 cinta de trabajo, sin cinta de salida
- $x \in \mathcal{L}$ sii existe una secuencia de mini-configuraciones $z_0, \ldots, z_{t(|x|)}$ de M con entrada x tal que
  - z_0 es inicial para x
  - z_i evoluciona en z_{i+1}
  - $z_{t(|x|)}$ es final

Teorema

$\mathbf{P} \subseteq \mathbf{P}_{/\mathbf{poly}}$.

Demostración

Sea $\mathcal{L} \in \mathbf{P}$ y sea M una máquina determinística tal que

- M corre en tiempo $t(n)$ con t un polinomio
- M es *oblivious*, con 1 cinta de trabajo, sin cinta de salida
- $x \in \mathcal{L}$ sii existe una secuencia de mini-configuraciones $z_0, \ldots, z_{t(|x|)}$ de M con entrada x tal que
  - z_0 es inicial para x
  - z_i evoluciona en z_{i+1}
  - $z_{t(|x|)}$ es final

Representamos z_i con la cadena ets, donde:

- $e \in \{0,1\}^2$ codifica el símbolo en la cinta de entrada,
- $t \in \{0,1\}^2$ codifica el símbolo en la cinta de trabajo,
- $s \in \{0,1\}^c$ codifica el estado de M (c es constante)

Entonces $|z_i| = k = 4 + c$ (constante).

Para n fijo, construimos un circuito C_n tal que $M(x) = C_n(x)$ para todo $x \in \{0,1\}^n$.
Sea $x \in \{0,1\}^n$ y $z_i = ets$, $|z_i| = 4 + c$.
z_{i+1} solo depende de

- el símbolo leído por la cabeza de entrada (codificado con 2 bits),
- z_i
- $z_{prev(i,n)}$

Para n fijo, construimos un circuito C_n tal que $M(x) = C_n(x)$ para todo $x \in \{0,1\}^n$.
Sea $x \in \{0,1\}^n$ y $z_i = ets$, $|z_i| = 4 + c$.
z_{i+1} solo depende de

- el símbolo leído por la cabeza de entrada (codificado con 2 bits),
- z_i
- $z_{prev(i,n)}$

Sea $F_n : \{0,1\}^{2k+2} \to \{0,1\}^k$ tal que

$$F_n(e\ z_i\ z_{prev(i,n)}) = z_{i+1}$$

F_n se representa con un circuito de tamaño constante.

Para n fijo, construimos un circuito C_n tal que $M(x) = C_n(x)$ para todo $x \in \{0,1\}^n$.
Sea $x \in \{0,1\}^n$ y $z_i = ets$, $|z_i| = 4 + c$.
z_{i+1} solo depende de

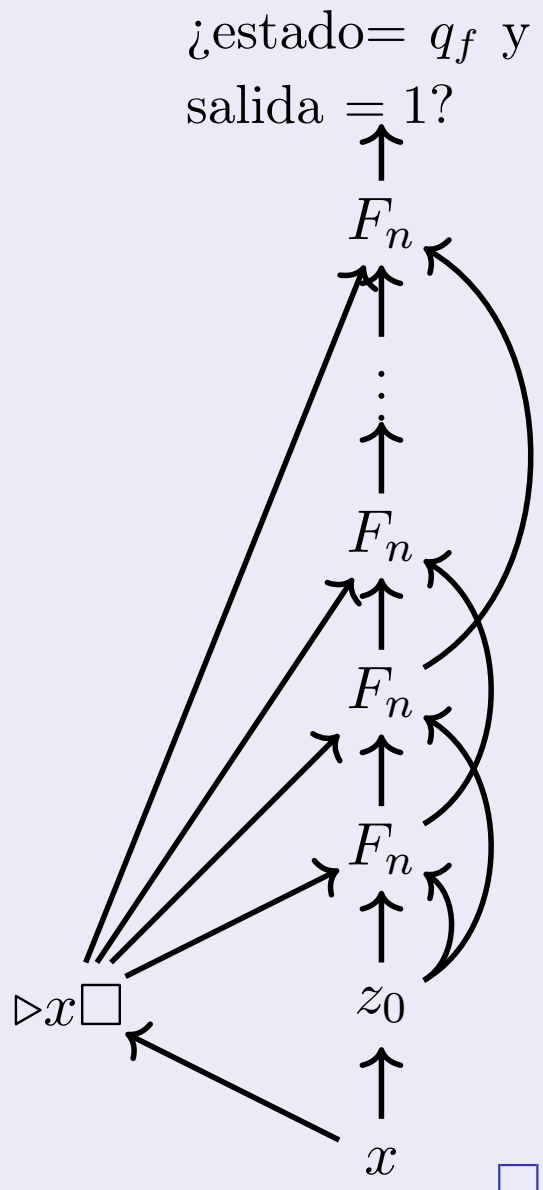
- el símbolo leído por la cabeza de entrada (codificado con 2 bits),
- z_i
- $z_{prev(i,n)}$

Sea $F_n : \{0,1\}^{2k+2} \to \{0,1\}^k$ tal que

$$F_n(e\ z_i\ z_{prev(i,n)}) = z_{i+1}$$

F_n se representa con un circuito de tamaño constante.
Entonces todo el circuito C_n cumple

$$|C_n| = O(t(n)) \quad \text{y} \quad C_n(x) = M(x)$$



Teorema

$\mathbf{P} \not\supseteq \mathbf{P}_{/\mathbf{poly}}$.

Teorema

$\mathbf{P} \not\supseteq \mathbf{P}_{/\mathbf{poly}}$.

Demostración.

Recordemos que si $\mathcal{L} \subseteq \{1^n : n \in \mathbb{N}\}$ entonces $\mathcal{L} \in \mathbf{P}_{/\mathbf{poly}}$.
Podemos tomar un $\mathcal{L} \subseteq \{1^n : n \in \mathbb{N}\}$ indecidible, por ejemplo

$$\mathcal{H} = \{1^n : x \text{ es la } n\text{-ésima cadena y } halt(x) = 1\}$$

(numeramos las cadenas $\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots$)
$\mathcal{H} \in \mathbf{P}_{/\mathbf{poly}} \setminus \mathbf{P}$. □

Notación para reemplazo de variables por constantes

Notación: Variables de una fórmula booleana

Si φ es una fórmula booleana con variables entre $x_1, \ldots, x_n$ (pueden ser menos), la notamos

$$\varphi(x_1, \ldots, x_n).$$

Notación: Reemplazo de variable por constante

Sea $\varphi(x_1, \ldots, x_n)$ una fórmula booleana. Representamos

$$\varphi(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_n)$$

a la fórmula booleana que resulta de reemplazar todas las apariciones de x_i por $b \in \{0, 1\}$.
Notar que

$$\underbrace{b_1 \ldots b_n}_{\substack{\text{valuación} \\ v \in \{0,1\}^n}} \models \varphi(\underbrace{x_1, \ldots, x_n}_{\text{variables}}) \quad \text{sii} \quad \models \varphi(\underbrace{b_1, \ldots, b_n}_{\substack{\text{constantes} \\ v \in \{0,1\}^n}})$$

Ejemplo

$$\varphi(x_1, x_2, x_3) = (x_1 \wedge \neg x_2) \vee (x_3 \wedge x_2)$$
$$\varphi(x_1, 1, x_3) = (x_1 \wedge \neg 1) \vee (x_3 \wedge 1)$$
$$\varphi(x_1, 0, x_3) = (x_1 \wedge \neg 0) \vee (x_3 \wedge 0)$$

Ejemplo

$$\varphi(x_1, x_2, x_3) = (x_1 \wedge \neg x_2) \vee (x_3 \wedge x_2)$$
$$\varphi(x_1, 1, x_3) = (x_1 \wedge \neg 1) \vee (x_3 \wedge 1)$$
$$\varphi(x_1, 0, x_3) = (x_1 \wedge \neg 0) \vee (x_3 \wedge 0)$$

Ejemplo

$$\varphi(x_1, x_2, x_3) = (x_1 \wedge \neg x_2) \vee (x_3 \wedge x_2)$$
$$\varphi(x_1, 1, x_3) = (x_1 \wedge \neg 1) \vee (x_3 \wedge 1)$$
$$\varphi(x_1, 0, x_3) = (x_1 \wedge \neg 0) \vee (x_3 \wedge 0)$$

$$110 \models \varphi(x_1, x_2, x_3) \text{ sii } \models \varphi(1, 1, 0) \text{ sii } \models (1 \wedge \neg 1) \vee (0 \wedge 1)$$

Encontrar valuaciones booleanas

Supongamos $\varphi(x_1, \ldots, x_n)$ satisfacible. Entonces

$$\varphi(1, x_2, \ldots, x_n) \in \mathsf{SAT} \implies \exists v \in \{0,1\}^{n-1}\ v \models \varphi(1, x_2, \ldots, x_n)$$
$$\varphi(1, x_2, \ldots, x_n) \notin \mathsf{SAT} \implies \exists v \in \{0,1\}^{n-1}\ v \models \varphi(0, x_2, \ldots, x_n)$$

Encontrar valuaciones booleanas

Supongamos $\varphi(x_1, \ldots, x_n)$ satisfacible. Entonces

$$\varphi(1, x_2, \ldots, x_n) \in \mathsf{SAT} \implies \exists v \in \{0,1\}^{n-1}\ v \models \varphi(1, x_2, \ldots, x_n)$$
$$\varphi(1, x_2, \ldots, x_n) \notin \mathsf{SAT} \implies \exists v \in \{0,1\}^{n-1}\ v \models \varphi(0, x_2, \ldots, x_n)$$

Es decir,

$$\varphi(\overbrace{\chi_{\mathsf{SAT}(\varphi(1,x_2,\ldots,x_n))}}^{\in\{0,1\}}, x_2, \ldots, x_n)$$

es satisfacible.

Encontrar valuaciones booleanas

Supongamos $\varphi(x_1, \ldots, x_n)$ satisfacible. Entonces

$$\varphi(1, x_2, \ldots, x_n) \in \mathsf{SAT} \implies \exists v \in \{0,1\}^{n-1}\ v \models \varphi(1, x_2, \ldots, x_n)$$
$$\varphi(1, x_2, \ldots, x_n) \notin \mathsf{SAT} \implies \exists v \in \{0,1\}^{n-1}\ v \models \varphi(0, x_2, \ldots, x_n)$$

Es decir,

$$\varphi(\overbrace{\chi_{\mathsf{SAT}(\varphi(1,x_2,\ldots,x_n))}}^{\in\{0,1\}}, x_2, \ldots, x_n)$$

es satisfacible.
Podemos seguir adelante con el mismo razonamiento:

$$\varphi(\overbrace{\chi_{\mathsf{SAT}(\varphi(1,x_2,\ldots,x_n))}}^{b\in\{0,1\}}, \overbrace{\chi_{\mathsf{SAT}(\varphi(b,1,x_3\ldots,x_n))}}^{\in\{0,1\}}, x_3, \ldots, x_n)$$

es satisfacible. Y así sucesivamente.

Circuito para encontrar valuaciones booleanas

- Podemos codificar fórmulas booleanas (con constantes 0 y 1) en binario.
- Cada fórmula de tamaño n tiene a lo sumo n variables y las suponemos numeradas $x_1, \ldots, x_i$, $i \leq n$.
- Consideramos circuitos booleanos con n entradas y m salidas, que representan funciones $\{0,1\}^n \to \{0,1\}^m$.

Circuito para encontrar valuaciones booleanas

- Podemos codificar fórmulas booleanas (con constantes 0 y 1) en binario.
- Cada fórmula de tamaño n tiene a lo sumo n variables y las suponemos numeradas $x_1, \ldots, x_i$, $i \leq n$.
- Consideramos circuitos booleanos con n entradas y m salidas, que representan funciones $\{0,1\}^n \to \{0,1\}^m$.

Proposición

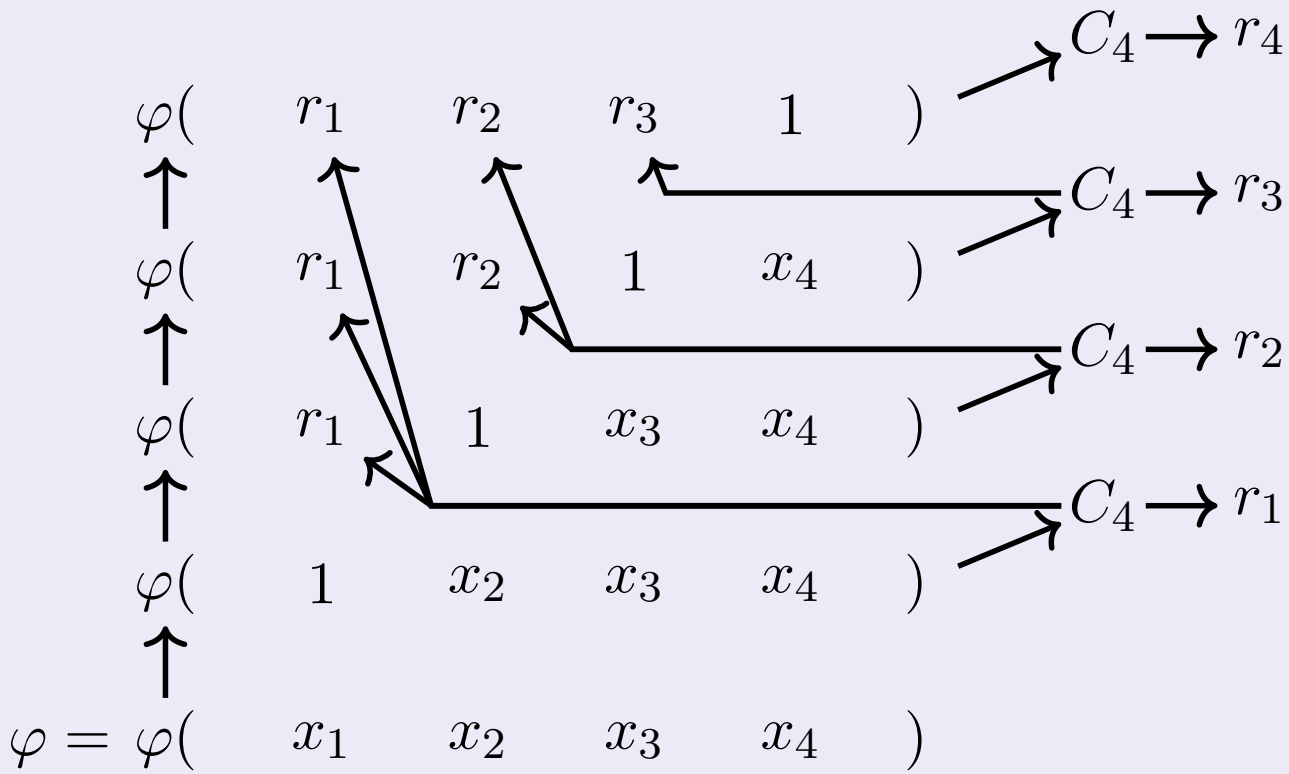
Sea $(C_n)_{n\in\mathbb{N}}$ una familia de circuitos de tamaño $S(n)$ tal que $C_n(\varphi) = \chi_{\mathsf{SAT}}(\varphi)$ para toda fórmula booleana $\varphi = \varphi(x_1, \ldots, x_n)$ con $|\varphi| = n$. Entonces existe una familia de circuitos $(C'_n)_{n\in\mathbb{N}}$ de tamaño polinomial en $S(n)$ tal que para todo n: C'_n tiene n salidas y para toda fórmula booleana $\varphi = \varphi(x_1, \ldots, x_n)$ con $|\varphi| = n$, tenemos

$$\varphi(C'_n(\varphi)) = \chi_{\mathsf{SAT}}(\varphi).$$

(notamos $C_n(\varphi)$ en vez de $C_n(\langle\varphi\rangle)$ y lo mismo para C'_n)

Demostración.

Dada $\varphi = \varphi(x_1, \ldots, x_n)$ de tamaño n y el circuito C_n tal que $C_n(\varphi) = \chi_{\mathsf{SAT}}(\varphi)$ definimos el circuito C'_n con entrada φ y salidas $r_1, \ldots, r_n$ de esta forma: (ejemplo para $n = 4$):



$\varphi \in \mathsf{SAT}$ sii $r_1 \ldots r_n \models \varphi$ sii $C'_n(\varphi) \models \varphi$ □

Problemas $\mathbf{\Pi_i^p}$-completos

Ya probamos que para todo $i > 0$, $\Sigma_i\mathsf{SAT} \in \mathbf{\Sigma_i^p}$-**completo**.
Análogamente,

Problema: Satisfacibilidad de QBF acotada

$$\Pi_i\mathsf{SAT} = \{\langle\varphi\rangle : \begin{array}{l}\varphi \text{ es una QBF de la forma} \\ \forall\bar{y}_1\exists\bar{y}_2 \ldots Q_i\bar{y}_i\psi(\bar{y}_1,\ldots,\bar{y}_i) \text{ donde las } \bar{y}_i \text{ son} \\ \text{tuplas de variables booleanas, } \psi \text{ es una fór-} \\ \text{mula booleana, los cuantificadores se alternan} \\ \text{y } \models\varphi\end{array}\}$$

Proposición

Para todo $i > 0$, $\Pi_i\mathsf{SAT} \in \mathbf{\Pi_i^p}$-**completo**.

Teorema de Karp-Lipton

Teorema (Karp-Lipton)

Si $\mathbf{NP} \subseteq \mathbf{P_{/poly}}$, entonces $\mathbf{PH} = \mathbf{\Sigma_2^p}$.

Como se cree que **PH** no colapsa, es un indicio de que $\mathbf{NP} \not\subseteq \mathbf{P_{/poly}}$.

Notación: Reemplazo de variables por constantes

Para la fórmula booleana

$$\varphi = \varphi(x_1, \ldots, x_i, y_1, \ldots, y_j)$$

con variables $x_1, \ldots, x_i, y_1, \ldots, y_j$, y $\bar{u} = u_1, \ldots, u_i \in \{0,1\}^*$ notamos

$$\varphi_{\bar{u}} = \varphi_{\bar{u}}(y_1, \ldots, y_j) = \varphi(u_1, \ldots, u_i, y_1, \ldots, y_j)$$

Demostración

Alcanza con probar $\mathbf{\Pi_2^p} \subseteq \mathbf{\Sigma_2^p}$, o $\Pi_2\mathsf{SAT} \in \mathbf{\Sigma_2^p}$ porque $\Pi_2\mathsf{SAT}$ es $\mathbf{\Pi_2^p}$-completo.

Demostración

Alcanza con probar $\mathbf{\Pi_2^p} \subseteq \mathbf{\Sigma_2^p}$, o $\Pi_2\mathsf{SAT} \in \mathbf{\Sigma_2^p}$ porque $\Pi_2\mathsf{SAT}$ es $\mathbf{\Pi_2^p}$-completo.

Supongamos $\mathbf{NP} \subseteq \mathbf{P_{/poly}}$. Existe un polinomio p y una familia de circuitos $(C_n)_{n\in\mathbb{N}}$ de tamaño $p(n)$ tal que para toda fórmula booleana φ de tamaño n

$$\begin{array}{lll} \varphi \in \mathsf{SAT} & \text{sii} & \exists \bar{v} \in \{0,1\}^n \ \bar{v} \models \varphi \\ & \text{sii} & \exists \bar{v} \in \{0,1\}^n \ \models \varphi(\bar{v}) \\ & \text{sii} & C_n(\varphi) = 1 \end{array}$$

Demostración

Alcanza con probar $\mathbf{\Pi_2^p} \subseteq \mathbf{\Sigma_2^p}$, o $\Pi_2\mathsf{SAT} \in \mathbf{\Sigma_2^p}$ porque $\Pi_2\mathsf{SAT}$ es $\mathbf{\Pi_2^p}$-completo.

Supongamos $\mathbf{NP} \subseteq \mathbf{P_{/poly}}$. Existe un polinomio p y una familia de circuitos $(C_n)_{n\in\mathbb{N}}$ de tamaño $p(n)$ tal que para toda fórmula booleana φ de tamaño n

$$\begin{array}{lll} \varphi \in \mathsf{SAT} & \text{sii} & \exists \bar{v} \in \{0,1\}^n \ \bar{v} \models \varphi \\ & \text{sii} & \exists \bar{v} \in \{0,1\}^n \ \models \varphi(\bar{v}) \\ & \text{sii} & C_n(\varphi) = 1 \end{array}$$

Supongamos $\varphi = \varphi(\underbrace{x_1, \ldots, x_i}_{\bar{x}}, \underbrace{y_1, \ldots, y_j}_{\bar{y}})$ de tamaño n:

$$\forall \bar{u} \in \{0,1\}^i \left(\underbrace{\exists \bar{v} \in \{0,1\}^j \ \models \varphi_{\bar{u}}(\bar{v})}_{\varphi_{\bar{u}} = \varphi_{\bar{u}}(\bar{y}) \in \mathsf{SAT}} \iff C_n(\varphi_{\bar{u}}) = 1 \right)$$

$\forall \bar{x} \exists \bar{y}\ \varphi(\bar{x}, \bar{y}) \in \Pi_2\mathsf{SAT}$ sii

$$\forall \bar{x} \exists \bar{y}\ \varphi(\bar{x}, \bar{y}) \in \Pi_2\mathsf{SAT} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i \overbrace{\exists \bar{v} \in \{0,1\}^j \ \models \varphi(\bar{u}, \bar{v})}^{\varphi_{\bar{u}} = \varphi_{\bar{u}}(y) \in \mathsf{SAT}} \qquad \text{sii}$$

$$\forall \bar{x} \exists \bar{y}\ \varphi(\bar{x}, \bar{y}) \in \Pi_2\mathsf{SAT} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i \overbrace{\exists \bar{v} \in \{0,1\}^j \ \models \varphi(\bar{u}, \bar{v})}^{\varphi_{\bar{u}} = \varphi_{\bar{u}}(y) \in \mathsf{SAT}} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i\ C_n(\varphi_{\bar{u}}) = 1 \qquad \text{sii}$$

- $|\varphi(\bar{x}, \bar{y})| = |\varphi_{\bar{u}}(\bar{y})| = n$

$$\forall \bar{x} \exists \bar{y}\ \varphi(\bar{x}, \bar{y}) \in \Pi_2\mathsf{SAT} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i \overbrace{\exists \bar{v} \in \{0,1\}^j\ \models \varphi(\bar{u}, \bar{v})}^{\varphi_{\bar{u}} = \varphi_{\bar{u}}(y) \in \mathsf{SAT}} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i\ C_n(\varphi_{\bar{u}}) = 1 \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i\ \models \varphi_{\bar{u}}(C'_n(\varphi_{\bar{u}})) \qquad \text{sii}$$

- $|\varphi(\bar{x}, \bar{y})| = |\varphi_{\bar{u}}(\bar{y})| = n$
- C'_n es tiene una codificación de tamaño $q(n)$ con q un polinomio tal que $\psi(C'_n(\psi)) = \chi_{\mathsf{SAT}}(\psi)$ para toda ψ de tamaño n

$$\forall \bar{x} \exists \bar{y}\ \varphi(\bar{x}, \bar{y}) \in \Pi_2\mathsf{SAT} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i \overbrace{\exists \bar{v} \in \{0,1\}^j\ \models \varphi(\bar{u}, \bar{v})}^{\varphi_{\bar{u}} = \varphi_{\bar{u}}(y) \in \mathsf{SAT}} \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i\ C_n(\varphi_{\bar{u}}) = 1 \qquad \text{sii}$$

$$\forall \bar{u} \in \{0,1\}^i\ \models \varphi_{\bar{u}}(C'_n(\varphi_{\bar{u}})) \qquad \text{sii}$$

$$\underbrace{\exists c \in \{0,1\}^{q(n)} \forall \bar{u} \in \{0,1\}^i\ \models \varphi_{\bar{u}}(R_c(\varphi_{\bar{u}}))}_{\exists c \in \{0,1\}^{q(n)} \forall \bar{u} \in \{0,1\}^i\ [\text{Polinomial}]\ \in\ \mathbf{\Sigma_2^P}}$$

- $|\varphi(\bar{x}, \bar{y})| = |\varphi_{\bar{u}}(\bar{y})| = n$
- C'_n es tiene una codificación de tamaño $q(n)$ con q un polinomio tal que $\psi(C'_n(\psi)) = \chi_{\mathsf{SAT}}(\psi)$ para toda ψ de tamaño n
- R_c es el circuito representado por $c \in \{0,1\}^*$
- Decidimos $\varphi_{\bar{u}}(R_c(\varphi_u))$ en tiempo polinomial

Tamaño y profundidad de circuitos

Cotas al tamaño mínimo de circuitos

Definición

Sea $f : \{0,1\}^n \to \{0,1\}$. Definimos

$$\text{minSize}(f) = \text{mín}\{|C| : C \text{ tiene } n \text{ entradas y } \forall \bar{x}\ C(\bar{x}) = f(\bar{x})\}$$

Cotas al tamaño mínimo de circuitos

Definición

Sea $f : \{0,1\}^n \to \{0,1\}$. Definimos

$$\text{minSize}(f) = \text{mín}\{|C| : C \text{ tiene } n \text{ entradas y } \forall \bar{x}\ C(\bar{x}) = f(\bar{x})\}$$

Proposición

$\text{minSize}(f) = O(n2^n)$ para toda $f : \{0,1\}^n \to \{0,1\}$.

Cotas al tamaño mínimo de circuitos

Definición

Sea $f : \{0,1\}^n \to \{0,1\}$. Definimos

$$\text{minSize}(f) = \text{mín}\{|C| : C \text{ tiene } n \text{ entradas y } \forall \bar{x}\ C(\bar{x}) = f(\bar{x})\}$$

Proposición

$\text{minSize}(f) = O(n2^n)$ para toda $f : \{0,1\}^n \to \{0,1\}$.

Demostración.

Ya lo vimos para CNF. Más directo es verlo para DNF (forma normal disyuntiva):

$$f(x_1, \ldots, x_n) = \bigvee_{\substack{(y_1,\ldots,y_n) \in \{0,1\}^n : \\ f(y_1,\ldots,y_n) = 1}} \bigwedge_{1 \le i \le n} x_i = y_i$$

La subfórmula $x_i = y_i$ es x_i si $y_i = 1$ y $\neg x_i$ si no.
La fórmula se representa con un circuito de tamaño $O(n2^n)$. □

Teorema

Para todo $n > 1$, existe $f : \{0,1\}^n \to \{0,1\}$ tal que $\text{minSize}(f) > 2^n/4n$.

Teorema

Para todo $n > 1$, existe $f : \{0,1\}^n \to \{0,1\}$ tal que $\text{minSize}(f) > 2^n/4n$.

Demostración.

Recordemos que codificamos u circuito de tamaño S con $\leq 4 \cdot S \cdot \log S$ bits. Para una función $S : \mathbb{N} \to \mathbb{N}$ cualquiera, definimos

$$K_{n,S} = \{\text{circuitos con } n \text{ entradas de tamaño} \leq S(n)\}$$
$$\#K_{n,S} \leq 2^{4S(n)\log S(n)}$$

Teorema

Para todo $n > 1$, existe $f : \{0,1\}^n \to \{0,1\}$ tal que $\text{minSize}(f) > 2^n/4n$.

Demostración.

Recordemos que codificamos u circuito de tamaño S con $\leq 4 \cdot S \cdot \log S$ bits. Para una función $S : \mathbb{N} \to \mathbb{N}$ cualquiera, definimos

$$K_{n,S} = \{\text{circuitos con } n \text{ entradas de tamaño} \leq S(n)\}$$
$$\#K_{n,S} \leq 2^{4S(n)\log S(n)}$$

Tomemos $S(n) = 2^n/4n$. Codificamos $C \in K_{n,S}$ con a lo sumo esta cantidad de bits:

$$\begin{aligned} 4S(n)\log S(n) &= 4\frac{2^n}{4n}\log\frac{2^n}{4n} \\ &< \frac{2^n}{n}\log 2^n = 2^n \end{aligned}$$

Teorema

Para todo $n > 1$, existe $f : \{0,1\}^n \to \{0,1\}$ tal que $\text{minSize}(f) > 2^n/4n$.

Demostración.

Recordemos que codificamos u circuito de tamaño S con $\leq 4 \cdot S \cdot \log S$ bits. Para una función $S : \mathbb{N} \to \mathbb{N}$ cualquiera, definimos

$$K_{n,S} = \{\text{circuitos con } n \text{ entradas de tamaño} \leq S(n)\}$$
$$\#K_{n,S} \leq 2^{4S(n)\log S(n)}$$

Tomemos $S(n) = 2^n/4n$. Codificamos $C \in K_{n,S}$ con a lo sumo esta cantidad de bits:

$$\begin{aligned} 4S(n)\log S(n) &= 4\frac{2^n}{4n}\log\frac{2^n}{4n} \\ &< \frac{2^n}{n}\log 2^n = 2^n \end{aligned}$$

Entonces $\#K_{n,S} < 2^{2^n}$.

Teorema

Para todo $n > 1$, existe $f : \{0,1\}^n \to \{0,1\}$ tal que $\text{minSize}(f) > 2^n/4n$.

Demostración.

Recordemos que codificamos u circuito de tamaño S con $\leq 4 \cdot S \cdot \log S$ bits. Para una función $S : \mathbb{N} \to \mathbb{N}$ cualquiera, definimos

$$K_{n,S} = \{\text{circuitos con } n \text{ entradas de tamaño} \leq S(n)\}$$
$$\#K_{n,S} \leq 2^{4S(n)\log S(n)}$$

Tomemos $S(n) = 2^n/4n$. Codificamos $C \in K_{n,S}$ con a lo sumo esta cantidad de bits:

$$\begin{aligned} 4S(n)\log S(n) &= 4\frac{2^n}{4n}\log\frac{2^n}{4n} \\ &< \frac{2^n}{n}\log 2^n = 2^n \end{aligned}$$

Entonces $\#K_{n,S} < 2^{2^n}$.
La cantidad de funciones $\{0,1\}^n \to \{0,1\}$ es 2^{2^n}. Por lo tanto existe una función $f : \{0,1\}^n \to \{0,1\}$ tal que $\text{minSize}(f) > 2^n/4n$. □

Profundidad de un circuito

Definición

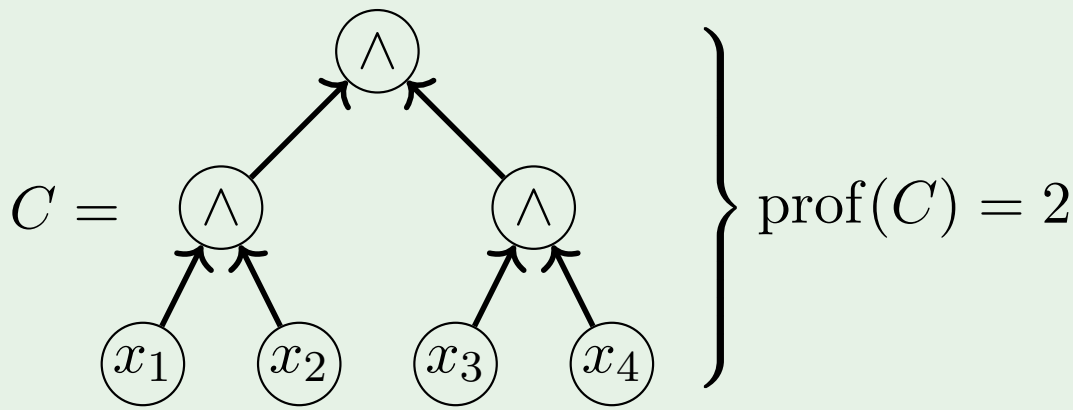
La **profundidad** de un circuito C, notado $\text{prof}(C)$, es la longitud del camino más largo desde alguna entrada hasta alguna salida.

Profundidad de un circuito

Definición

La **profundidad** de un circuito C, notado $\text{prof}(C)$, es la longitud del camino más largo desde alguna entrada hasta alguna salida.

Profundidad de un circuito

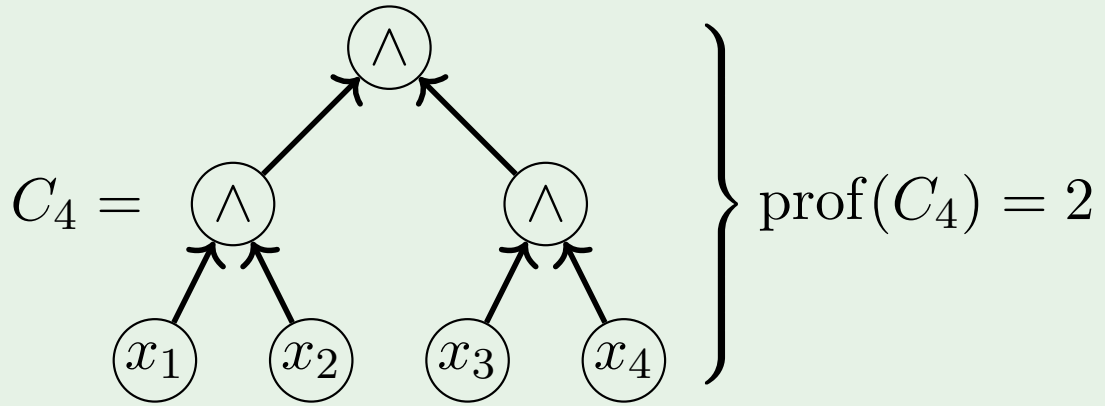


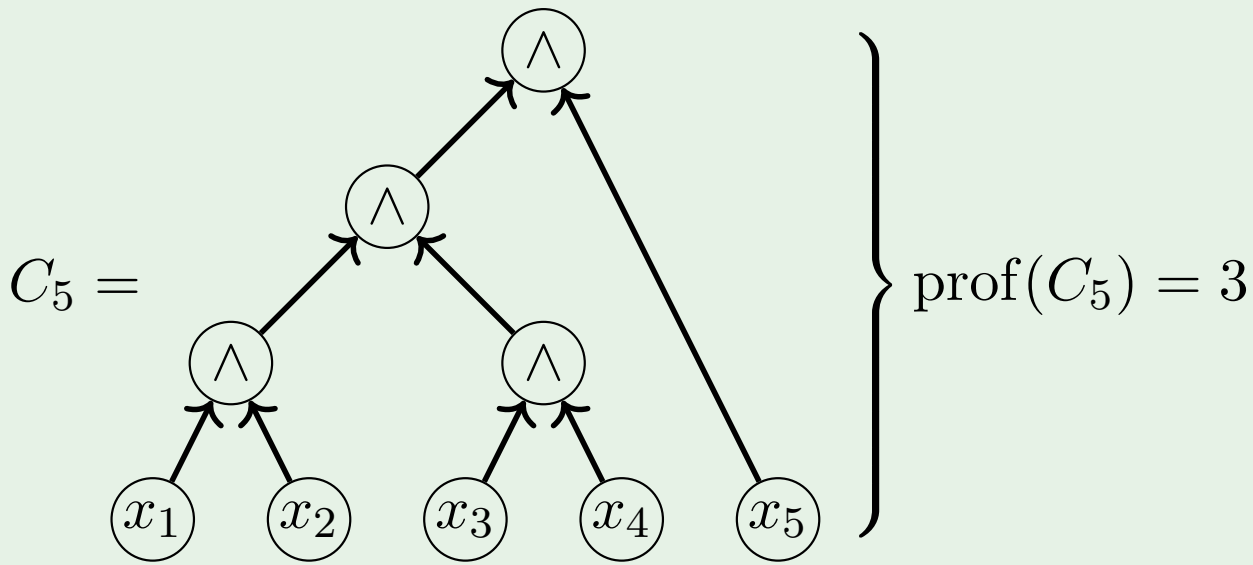
Definición

Una familia de circuitos $(C_n)_{n\in\mathbb{N}}$ tiene profundidad $d(n)$ si $\text{prof}(C_n) \leq d(n)$ para todo n.

Profundidad logarítmica

$\{1^n : n \geq 1\}$ es decidible por una familia de circuitos de profundidad $O(\log n)$





Conjunciones y disyunciones de aridad arbitraria

A veces vamos a considerar circuitos con $\wedge$ y $\vee$ con *fan-in* arbitrario. Esto permite profundidades más bajas.

Profundidad constante

Existe una familia de circuitos con $\wedge$ de *fan-in* arbitrario de profundidad $O(1)$ que acepta $\{1^n : n \geq 1\}$.